Computer Algebra Independent Integration Tests

Summer 2023 edition

4-Trig-functions/4.2-Cosine/97-4.2.9-trig^m-a+b-cos^n+c-cos^-2-n-^p

Nasser M. Abbasi

September 5, 2023 Compiled on September 5, 2023 at 11:10pm

Contents

CHAPTER 1

INTRODUCTION

This report gives the result of running the computer algebra independent integration test.The download section in on the main webpage contains links to download the problems in plain text format used for all CAS systems. The number of integrals in this report is [20]. This is test number [97].

1.1 Listing of CAS systems tested

The following are the CAS systems tested:

1. Mathematica 13.3.1 (August 16, 2023) on windows 10.
2. Rubi 4.16.1 (Dec 19, 2018) on Mathematica 13.3 on windows 10
3. Maple 2023.1 (July, 12, 2023) on windows 10.
4. Maxima 5.47 (June 1, 2023) using Lisp SBCL 2.3.0 on Linux via sagemath 10.1 (Aug 20, 2023).
5. FriCAS 1.3.9 (July 8, 2023) based on sbcl 2.3.0 on Linux via sagemath 10.1 (Aug 20, 2023).
6. Giac/Xcas 1.9.0-57 (June 26, 2023) on Linux via sagemath 10.1 (Aug 20, 2023).
7. Sympy 1.12 (May 10, 2023) Using Python 3.11.3 on Linux.
8. Mupad using Matlab 2021a with Symbolic Math Toolbox Version 8.7 on windows 10.

Maxima and Fricas and Giac are called using Sagemath. This was done using Sagemath `integrate` command by changing the name of the algorithm to use the different CAS systems.

Sympy was run directly in Python not via sagemath.

1.2 Results

Important note: A number of problems in this test suite have no antiderivative in closed form. This means the antiderivative of these integrals can not be expressed in terms of elementary, special functions or `Hypergeometric2F1` functions. `RootSum` and `RootOf` are not allowed. If a CAS returns the above integral unevaluated within the time limit, then the result is counted as passed and assigned an A grade.

However, if CAS times out, then it is assigned an F grade even if the integral is not integrable, as this implies CAS could not determine that the integral is not integrable in the time limit.

If a CAS returns an antiderivative to such an integral, it is assigned an A grade automatically and this special result is listed in the introduction section of each individual test report to make it easy to identify as this can be important result to investigate.

The results given in in the table below reflects the above.

System	% solved	% Failed
Rubi	100.00 (20)	0.00 (0)
Mathematica	100.00 (20)	0.00 (0)
Maple	100.00 (20)	0.00 (0)
Mupad	100.00 (20)	0.00 (0)
Giac	100.00 (20)	0.00 (0)
Fricas	95.00 (19)	5.00 (1)
Sympy	25.00 (5)	75.00 (15)
Maxima	20.00 (4)	80.00 (16)

Table 1.1: Percentage solved for each CAS

The table below gives additional break down of the grading of quality of the antiderivatives generated by each CAS. The grading is given using the letters A,B,C and F with A being the best quality. The grading is accomplished by comparing the antiderivative generated with the optimal antiderivatives included in the test suite. The following table describes the meaning of these grades.

grade	description
A	Integral was solved and antiderivative is optimal in quality and leaf size.
B	Integral was solved and antiderivative is optimal in quality but leaf size is larger than twice the optimal antiderivatives leaf size.
C	Integral was solved and antiderivative is non-optimal in quality. This can be due to one or more of the following reasons 1. antiderivative contains a hypergeometric function and the optimal antiderivative does not. 2. antiderivative contains a special function and the optimal antiderivative does not. 3. antiderivative contains the imaginary unit and the optimal antiderivative does not.
F	Integral was not solved. Either the integral was returned unevaluated within the time limit, or it timed out, or CAS hanged or crashed or an exception was raised.

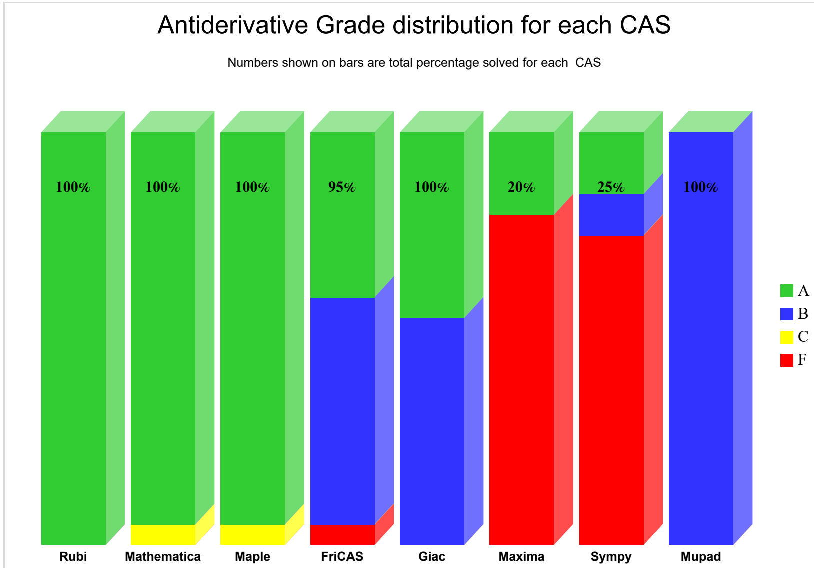
Table 1.2: Description of grading applied to integration result

Grading is implemented for all CAS systems. Based on the above, the following table summarizes the grading for this test suite.

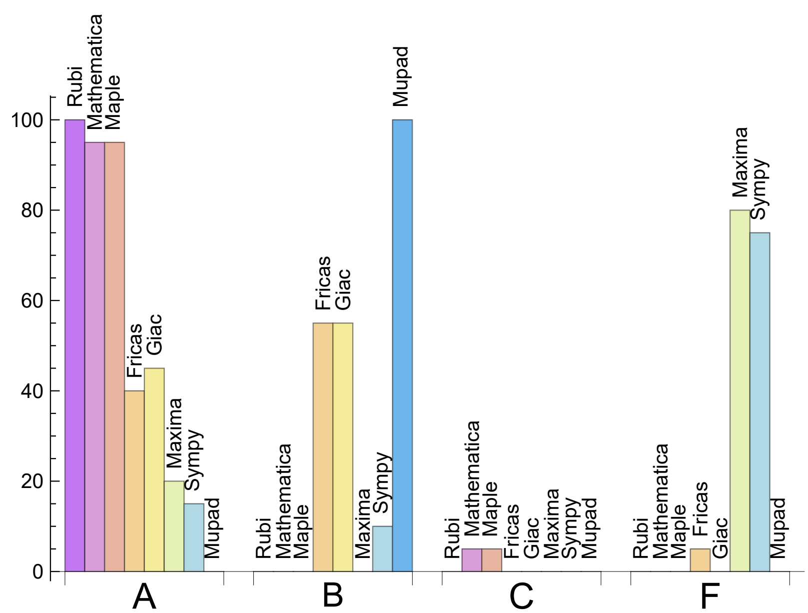
System	% A grade	% B grade	% C grade	% F grade
Rubi	100.000	0.000	0.000	0.000
Mathematica	95.000	0.000	5.000	0.000
Maple	95.000	0.000	5.000	0.000
Giac	45.000	55.000	0.000	0.000
Fricas	40.000	55.000	0.000	5.000
Maxima	20.000	0.000	0.000	80.000
Sympy	15.000	10.000	0.000	75.000
Mupad	0.000	100.000	0.000	0.000

Table 1.3: Antiderivative Grade distribution of each CAS

The following is a Bar chart illustration of the data in the above table.



The figure below compares the grades of the CAS systems.



The following table shows the distribution of the different types of failures for each CAS. There are 3 types failures. The first is when CAS returns the input within the time limit, which means it could not solve it. This is the typical failure and given as **F**.

The second failure is due to time out. CAS could not solve the integral within the 3 minutes time limit which is assigned. This is assigned **F(-1)**.

The third is due to an exception generated, indicated as **F(-2)**. This most likely indicates an interface problem between sagemath and the CAS (applicable only to FriCAS, Maxima and Giac) or it could be an indication of an internal error in the CAS itself. This type of error requires more investigation to determine the cause.

System	Number failed	Percentage normal failure	Percentage timeout failure	Percentage exception failure
Rubi	0	0.00	0.00	0.00
Mathematica	0	0.00	0.00	0.00
Maple	0	0.00	0.00	0.00
Mupad	0	0.00	0.00	0.00
Giac	0	0.00	0.00	0.00
Fricas	1	0.00	100.00	0.00
Sympy	15	40.00	60.00	0.00
Maxima	16	68.75	0.00	31.25

Table 1.4: Failure statistics for each CAS

1.3 Time and leaf size Performance

The table below summarizes the performance of each CAS system in terms of time used and leaf size of results.

Mean size is the average leaf size produced by the CAS (before any normalization). The Normalized mean is relative to the mean size of the optimal anti-derivative given in the input files.

For example, if CAS has **Normalized mean** of 3, then the mean size of its leaf size is 3 times as large as the mean size of the optimal leaf size.

Median size is value of leaf size where half the values are larger than this and half are smaller (before any normalization). i.e. The Middle value.

Similarly the **Normalized median** is relative to the median leaf size of the optimal.

For example, if a CAS has Normalized median of 1.2, then its median is 1.2 as large as the median leaf size of the optimal.

System	Mean time (sec)
Maxima	0.40
Sympy	0.42
Mathematica	0.76
Giac	1.54
Rubi	1.95
Maple	2.74
Mupad	10.07
Fricas	18.75

Table 1.5: Time performance for each CAS

System	Mean size	Normalized mean	Median size	Normalized median
Maxima	18.25	0.73	15.00	0.75
Sympy	54.20	1.76	26.00	1.37
Rubi	191.95	1.00	226.50	1.00
Maple	212.45	1.04	206.00	1.04
Mathematica	220.15	1.16	238.50	1.03
Fricas	3421.79	12.61	1991.00	9.71
Giac	4964.45	16.93	2983.50	13.18
Mupad	15358.45	51.12	5501.00	24.29

Table 1.6: Leaf size performance for each CAS

1.4 Performance based on number of rules Rubi used

This section shows how each CAS performed based on the number of rules Rubi needed to solve the same integral. One diagram is given for each CAS.

On the y axis is the percentage solved which Rubi itself needed the number of rules given the x axis. These plots show that as more rules are needed then most CAS system percentage of solving decreases which indicates the integral is becoming more complicated to solve.

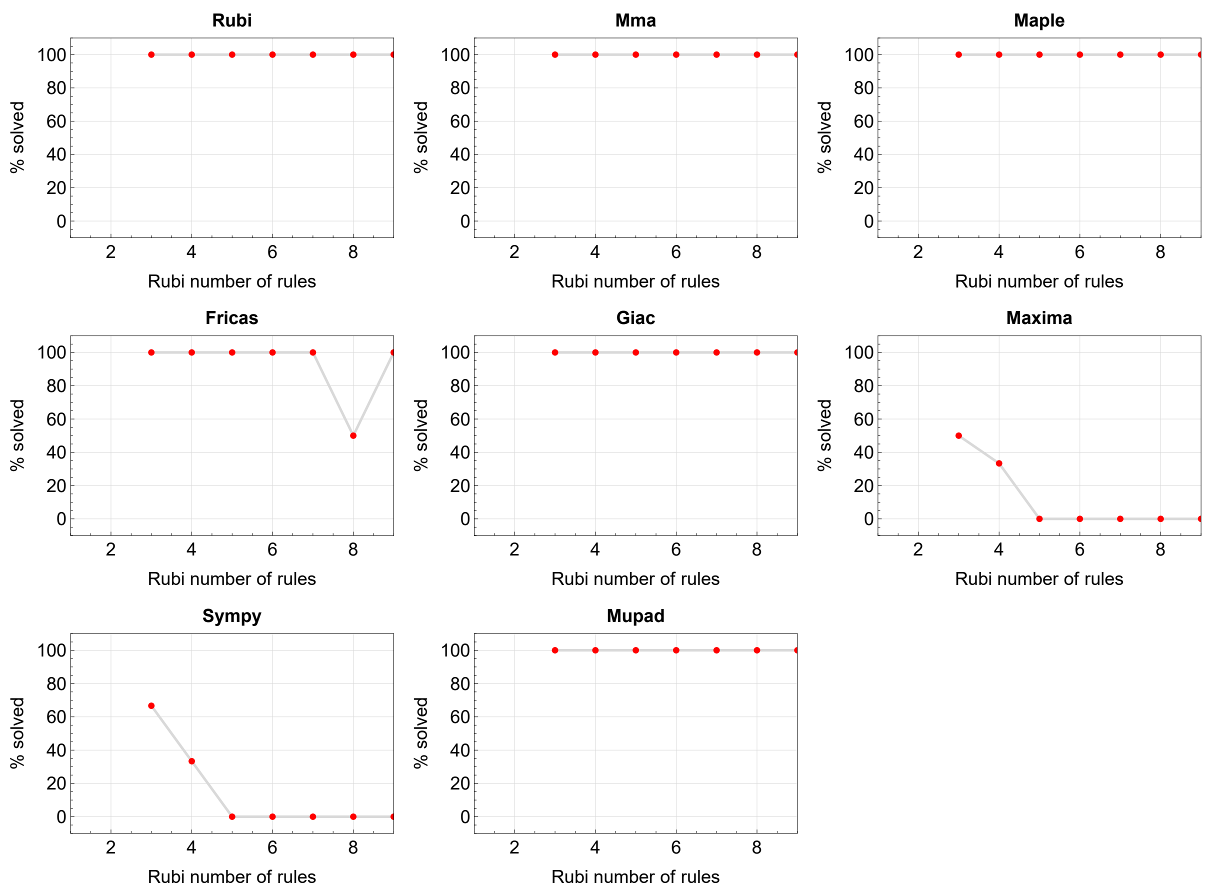


Figure 1.1: Solving statistics per number of Rubi rules used

1.5 Performance based on number of steps Rubi used

This section shows how each CAS performed based on the number of steps Rubi needed to solve the same integral. Note that the number of steps Rubi needed can be much higher than the number of rules, as the same rule could be used more than once.

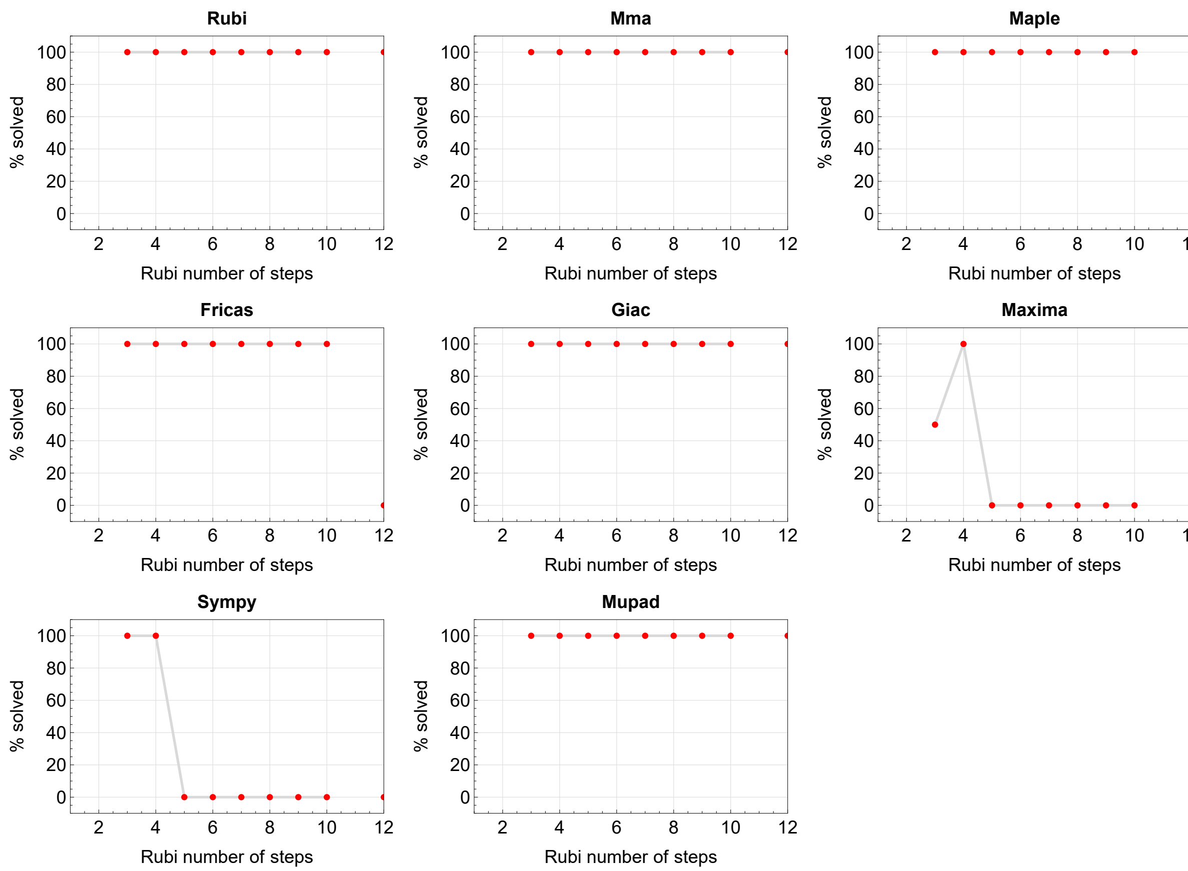


Figure 1.2: Solving statistics per number of Rubi steps used

The above diagram show that the precentage of solved intergals decreases for most CAS systems as the number of steps increases. As expected, for integrals that required less steps by Rubi, CAS systems had more success which indicates the integral was not as hard to solve. As Rubi needed more steps to solve the integral, the solved percentage decreased for most CAS systems which indicates the integral is becoming harder to solve.

1.6 Solved integrals histogram based on leaf size of result

The following shows the distribution of solved integrals for each CAS system based on leaf size of the antiderivatives produced by each CAS. It shows that most integrals solved produced leaf size less than about 100 to 150. The bin size used is 40.

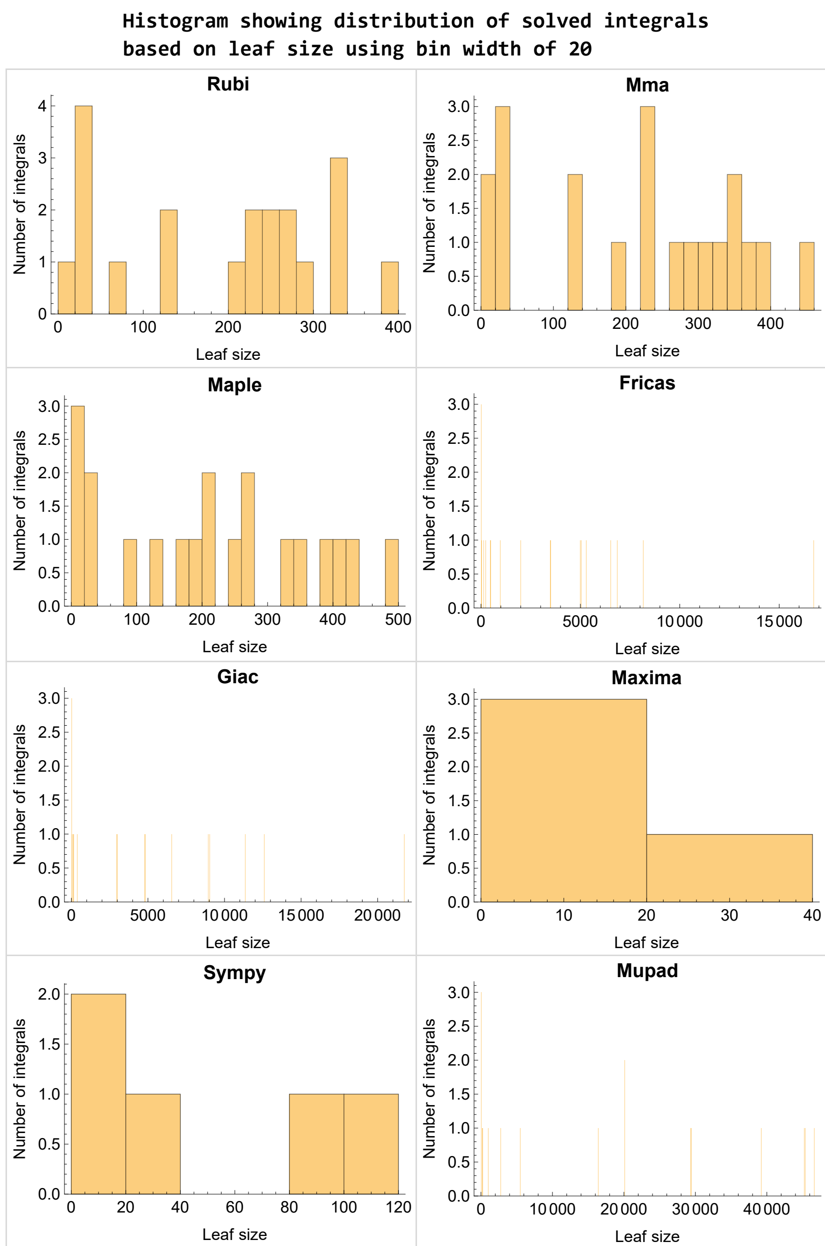


Figure 1.3: Solved integrals based on leaf size distribution

1.7 Solved integrals histogram based on CPU time used

The following shows the distribution of solved integrals for each CAS system based on CPU time used in seconds. The bin size used is 0.1 second.

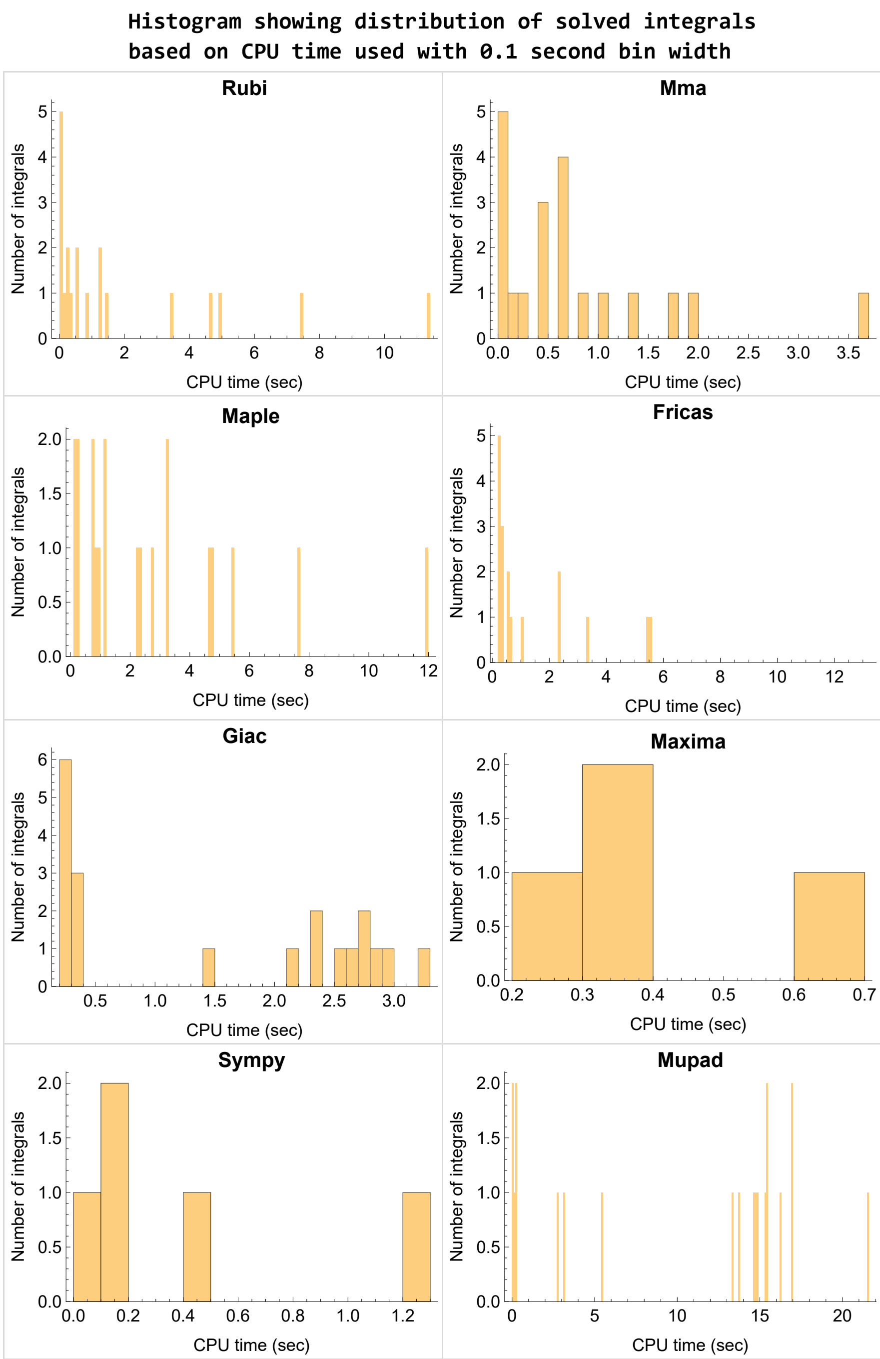


Figure 1.4: Solved integrals histogram based on CPU time used

1.8 Leaf size vs. CPU time used

The following shows the relation between the CPU time used to solve an integral and the leaf size of the antiderivative.

The result for Fricas, Maxima and Giac is shifted more to the right than the other CAS system due to the use of sagemath to call them, which causes an initial slight delay in the timing to start the integration due to overhead of starting a new process each time. This should also be taken into account when looking at the timing of these three CAS systems. Direct calls not using sagemath would result in faster timings, but current implementation uses sagemath as this makes testing much easier to do.

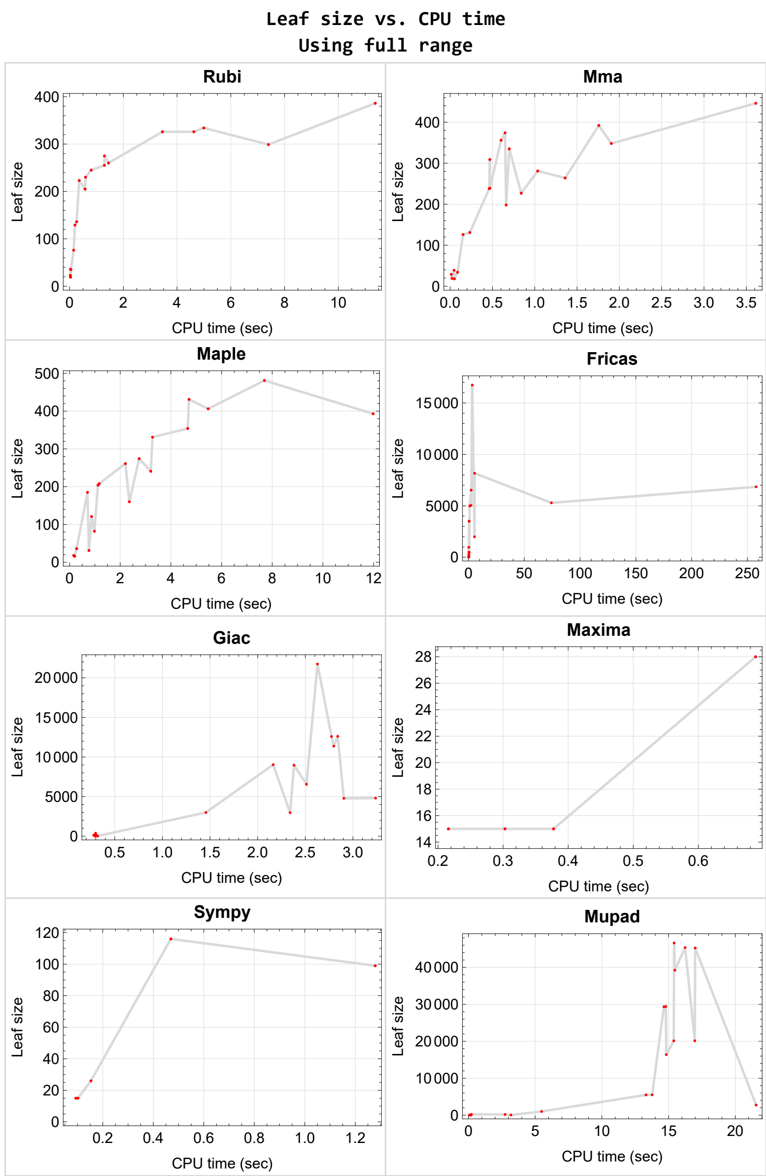


Figure 1.5: Leaf size vs. CPU time. Full range

1.9 list of integrals with no known antiderivative

{}

1.10 List of integrals solved by CAS but has no known antiderivative

Rubi {}

Mathematica {}

Maple {}

Maxima {}

Fricas {}

Sympy {}

Giac {}

Mupad {}

1.11 list of integrals solved by CAS but failed verification

The following are integrals solved by CAS but the verification phase failed to verify the anti-derivative produced is correct. This does not necessarily mean that the anti-derivative is wrong as additional methods of verification might be needed, or more time is needed (3 minutes time limit was used). These integrals are listed here to make it possible to do further investigation to determine why the result could not be verified.

Rubi {}

Mathematica {}

Maple {}

Maxima Verification phase not currently implemented.

Fricas Verification phase not currently implemented.

Sympy Verification phase not currently implemented.

Giac Verification phase not currently implemented.

Mupad Verification phase not currently implemented.

1.12 Timing

The command AbsoluteTiming[] was used in Mathematica to obtain the elapsed time for each integrate call. In Maple, the command Usage was used as in the following example

```
cpu_time := Usage(assign ('result_of_int',int(expr,x)),output='realtime'
```

For all other CAS systems, the elapsed time to complete each integral was found by taking the difference between the time after the call completed from the time before the call was made. This was done using Python's time.time() call.

All elapsed times shown are in seconds. A time limit of 3 CPU minutes was used for each integral. If the integrate command did not complete within this time limit, the integral was aborted and considered to have failed and assigned an F grade. The time used by failed integrals due to time out was not counted in the final statistics.

1.13 Verification

A verification phase was applied on the result of integration for Rubi and Mathematica.

Future version of this report will implement verification for the other CAS systems. For the integrals whose result was not run through a verification phase, it is assumed that the antiderivative was correct.

Verification phase also had 3 minutes time out. An integral whose result was not verified could still be correct, but further investigation is needed on those integrals. These integrals were marked in the summary table below and also in each integral separate section so they are easy to identify and locate.

1.14 Important notes about some of the results

Important note about Maxima results

Since tests were run in a batch mode, and using an automated script, then any integral where Maxima needed an interactive response from the user to answer a question during the evaluation of the integral will fail.

The exception raised is ValueError. Therefore Maxima results is lower than what would result if Maxima was run directly and each question was answered correctly.

The percentage of such failures were not counted for each test file, but for an example, for the Timofeev test file, there were about 14 such integrals out of total 705, or about 2 percent. This percentage can be higher or lower depending on the specific input test file.

Such integrals can be identified by looking at the output of the integration in each section for Maxima. The exception message will indicate the cause of error.

Maxima integrate was run using SageMath with the following settings set by default

```
'besselexpand : true'
'display2d : false'
'domain : complex'
'keepfloat : true'
'load(to_poly_solve)'
'load(simplify_sum)'
'load(abs_integrate)' 'load(diag)'
```

SageMath automatic loading of Maxima `abs_integrate` was found to cause some problems. So the following code was added to disable this effect.

```
from sage.interfaces.maxima_lib import maxima_lib
maxima_lib.set('extra_definite_integration_methods', '[]')
maxima_lib.set('extra_integration_methods', '[]')
```

See `https://ask.sagemath.org/question/43088/integrate-results-that-are-different-from-using-maxima/` for reference.

Important note about FriCAS result

There were few integrals which failed due to SageMath interface and not because FriCAS system could not do the integration.

These will fail With error `Exception raised: NotImplementedError`.

The number of such cases seems to be very small. About 1 or 2 percent of all integrals. These can be identified by looking at the exception message given in the result.

Important note about finding leaf size of antiderivative

For Mathematica, Rubi, and Maple, the builtin system function `LeafSize` was used to find the leaf size of each antiderivative.

The other CAS systems (SageMath and Sympy) do not have special builtin function for this purpose at this time. Therefore the leaf size for Fricas and Sympy antiderivative was determined using the following function, thanks to user `slelievre` at `https://ask.sagemath.org/question/57123/could-we-have-a-leaf_count-function-in-base-sagemath/`

```
def tree_size(expr):
    r"""
    Return the tree size of this expression.
    """
    if expr not in SR:
        # deal with lists, tuples, vectors
        return 1 + sum(tree_size(a) for a in expr)
    expr = SR(expr)
```

```
  x, aa = expr.operator(), expr.operands()
  if x is None:
    return 1
  else:
    return 1 + sum(tree_size(a) for a in aa)
```

For Sympy, which was called directly from Python, the following code was used to obtain the leafsize of its result

```
try:
  # 1.7 is a fudge factor since it is low side from actual leaf count
  leafCount = round(1.7*count_ops(anti))

  except Exception as ee:
       leafCount =1
```

Important note about Mupad results

Matlab's symbolic toolbox does not have a leaf count function to measure the size of the antiderivative. Maple was used to determine the leaf size of Mupad output by post processing Mupad result.

Currently no grading of the antiderivative for Mupad is implemented. If it can integrate the problem, it was assigned a B grade automatically as a placeholder. In the future, when grading function is implemented for Mupad, the tests will be rerun again.

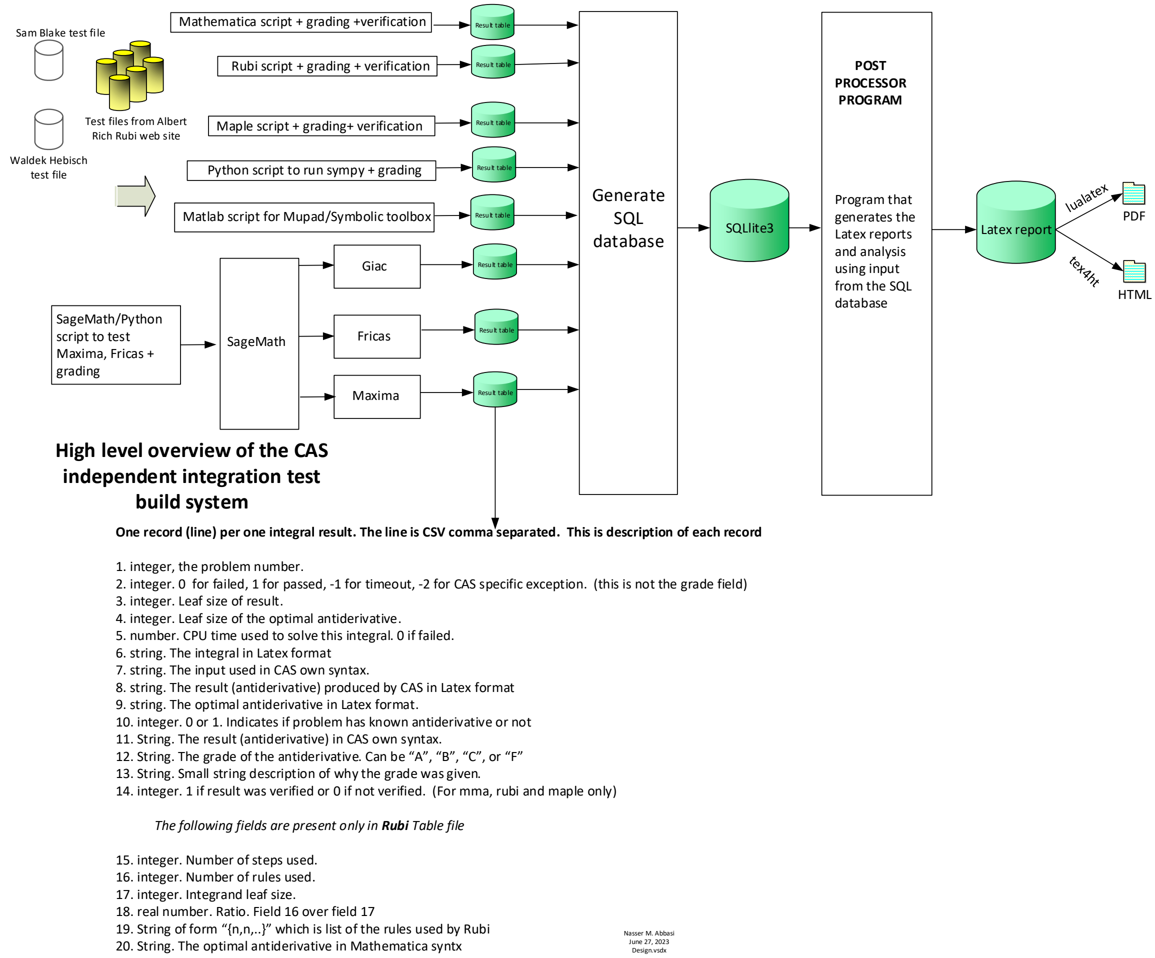
The following is an example of using Matlab's symbolic toolbox (Mupad) to solve an integral

```
integrand = evalin(symengine,'cos(x)*sin(x)')
the_variable = evalin(symengine,'x')
anti = int(integrand,the_variable)
```

Which gives `sin(x)^2/2`

1.15 Design of the test system

The following diagram gives a high level view of the current test build system.



CHAPTER 2

DETAILED SUMMARY TABLES OF RESULTS

2.1 List of integrals sorted by grade for each CAS

Rubi

A grade { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 }

B grade { }

C grade { }

F normal fail { }

F(-1) timedout fail { }

F(-2) exception fail { }

Mma

A grade { 1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 }

B grade { }

C grade { 5 }

F normal fail { }

F(-1) timedout fail { }

F(-2) exception fail { }

Maple

A grade { 1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 }

B grade { }

C grade { 7 }

F normal fail { }

F(-1) timedout fail { }

F(-2) exception fail { }

Fricas

A grade { 1, 2, 3, 4, 9, 10, 11, 12 }

B grade { 5, 6, 7, 8, 13, 14, 15, 16, 17, 18, 19 }

C grade { }

F normal fail { }

F(-1) timedout fail { 20 }

F(-2) exception fail { }

Maxima

A grade { 9, 10, 11, 12 }

B grade { }

C grade { }

F normal fail { 6, 7, 8, 13, 14, 15, 16, 17, 18, 19, 20 }

F(-1) timedout fail { }

F(-2) exception fail { 1, 2, 3, 4, 5 }

Giac

A grade { 1, 2, 3, 4, 5, 9, 10, 11, 12 }

B grade { 6, 7, 8, 13, 14, 15, 16, 17, 18, 19, 20 }

C grade { }

F normal fail { }

F(-1) timedout fail { }

F(-2) exception fail { }

Mupad

A grade $\{\ \}$

B grade $\{\ 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20\ \}$

C grade $\{\ \}$

F normal fail $\{\ \}$

F(-1) timedout fail $\{\ \}$

F(-2) exception fail $\{\ \}$

Sympy

A grade $\{\ 9,10,11\ \}$

B grade $\{\ 3,12\ \}$

C grade $\{\ \}$

F normal fail $\{\ 4,5,8,18,19,20\ \}$

F(-1) timedout fail $\{\ 1,2,6,7,13,14,15,16,17\ \}$

F(-2) exception fail $\{\ \}$

2.2 Detailed conclusion table per each integral for all CAS systems

Detailed conclusion table per each integral is given by the table below. The elapsed time is in seconds. For failed result it is given as **F(-1)** if the failure was due to timeout. It is given as **F(-2)** if the failure was due to an exception being raised, which could indicate a bug in the system. If the failure was due to integral not being evaluated within the time limit, then it is given as **F**.

In this table, the column `N.S.` means **normalized size** and is defined as $\frac{\text{antiderivative leaf size}}{\text{optimal antiderivative leaf size}}$. To make the table fit the page, the name `Mathematica` was abbreviated to `MMA`.

Problem 1	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F(-2)	A	F(-1)	A	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	136	136	239	160	0	491	0	153	197
N.S.	1	1.00	1.76	1.18	0.00	3.61	0.00	1.12	1.45
time (sec)	N/A	0.263	0.470	2.360	0.000	0.356	0.000	0.303	2.743

Problem 2	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F(-2)	A	F(-1)	A	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	76	76	131	82	0	258	0	76	226
N.S.	1	1.00	1.72	1.08	0.00	3.39	0.00	1.00	2.97
time (sec)	N/A	0.152	0.233	0.982	0.000	0.308	0.000	0.303	0.219

Problem 3	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F(-2)	A	B	A	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	35	35	39	36	0	126	99	35	47
N.S.	1	1.00	1.11	1.03	0.00	3.60	2.83	1.00	1.34
time (sec)	N/A	0.049	0.046	0.279	0.000	0.274	1.279	0.298	3.183

Problem 4	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F(-2)	A	F	A	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	129	129	126	121	0	470	0	130	1003
N.S.	1	1.00	0.98	0.94	0.00	3.64	0.00	1.01	7.78
time (sec)	N/A	0.201	0.153	0.869	0.000	0.681	0.000	0.280	5.482

Problem 5	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	C	A	F(-2)	B	F	A	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	205	205	392	241	0	1991	0	378	2742
N.S.	1	1.00	1.91	1.18	0.00	9.71	0.00	1.84	13.38
time (sec)	N/A	0.574	1.757	3.206	0.000	5.428	0.000	0.299	21.564

Problem 6	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F	B	F(-1)	B	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	388	386	374	406	0	5045	0	11375	46613
N.S.	1	0.99	0.96	1.05	0.00	13.00	0.00	29.32	120.14
time (sec)	N/A	11.373	0.649	5.471	0.000	2.397	0.000	2.799	15.415

Problem 7	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	C	F	B	F(-1)	B	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	260	260	238	185	0	971	0	6566	16390
N.S.	1	1.00	0.92	0.71	0.00	3.73	0.00	25.25	63.04
time (sec)	N/A	1.448	0.462	0.709	0.000	0.343	0.000	2.512	14.827

Problem 8	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F	B	F	B	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	326	326	335	393	0	16741	0	21750	39229
N.S.	1	1.00	1.03	1.21	0.00	51.35	0.00	66.72	120.33
time (sec)	N/A	3.457	0.699	11.984	0.000	3.380	0.000	2.628	15.471

Problem 9	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	A	A	A	A	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	21	21	19	16	15	17	15	17	9
N.S.	1	1.00	0.90	0.76	0.71	0.81	0.71	0.81	0.43
time (sec)	N/A	0.028	0.022	0.195	0.216	0.246	0.093	0.299	0.200

Problem 10	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	A	A	A	A	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	23	23	29	16	15	19	15	19	9
N.S.	1	1.00	1.26	0.70	0.65	0.83	0.65	0.83	0.39
time (sec)	N/A	0.028	0.014	0.205	0.303	0.244	0.102	0.291	0.143

Problem 11	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	A	A	A	A	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	19	19	18	18	15	19	26	15	15
N.S.	1	1.00	0.95	0.95	0.79	1.00	1.37	0.79	0.79
time (sec)	N/A	0.038	0.050	0.165	0.378	0.244	0.153	0.292	0.068

Problem 12	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	A	A	B	A	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	36	36	34	31	28	38	116	28	30
N.S.	1	1.00	0.94	0.86	0.78	1.06	3.22	0.78	0.83
time (sec)	N/A	0.036	0.087	0.765	0.688	0.254	0.470	0.320	0.075

Problem 13	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F	B	F(-1)	B	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	326	326	356	431	0	8167	0	12587	45364
N.S.	1	1.00	1.09	1.32	0.00	25.05	0.00	38.61	139.15
time (sec)	N/A	4.624	0.602	4.716	0.000	5.516	0.000	2.775	16.240

Problem 14	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F	B	F(-1)	B	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	299	299	309	331	0	6529	0	4810	29362
N.S.	1	1.00	1.03	1.11	0.00	21.84	0.00	16.09	98.20
time (sec)	N/A	7.401	0.469	3.276	0.000	2.349	0.000	3.237	14.653

Problem 15	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F	B	F(-1)	B	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	255	255	264	261	0	4983	0	9030	20133
N.S.	1	1.00	1.04	1.02	0.00	19.54	0.00	35.41	78.95
time (sec)	N/A	1.296	1.359	2.203	0.000	1.063	0.000	2.162	16.964

Problem 16	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F	B	F(-1)	B	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	230	230	227	208	0	3513	0	2973	5488
N.S.	1	1.00	0.99	0.90	0.00	15.27	0.00	12.93	23.86
time (sec)	N/A	0.594	0.840	1.173	0.000	0.568	0.000	2.341	13.304

Problem 17	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F	B	F(-1)	B	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	223	223	198	204	0	3493	0	2994	5514
N.S.	1	1.00	0.89	0.91	0.00	15.66	0.00	13.43	24.73
time (sec)	N/A	0.359	0.662	1.121	0.000	0.517	0.000	1.456	13.765

Problem 18	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F	B	F	B	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	245	245	281	274	0	5292	0	8954	20126
N.S.	1	1.00	1.15	1.12	0.00	21.60	0.00	36.55	82.15
time (sec)	N/A	0.805	1.035	2.746	0.000	74.287	0.000	2.382	15.389

Problem 19	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F	B	F	B	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	275	275	348	354	0	6851	0	4784	29417
N.S.	1	1.00	1.27	1.29	0.00	24.91	0.00	17.40	106.97
time (sec)	N/A	1.298	1.905	4.664	0.000	257.772	0.000	2.905	14.792

Problem 20	Optimal	Rubi	MMA	Maple	Maxima	Fricas	Sympy	Giac	Mupad
grade	N/A	A	A	A	F	F(-1)	F	B	B
verified	N/A	Yes	Yes	Yes	TBD	TBD	TBD	TBD	TBD
size	334	334	446	481	0	0	0	12615	45255
N.S.	1	1.00	1.34	1.44	0.00	0.00	0.00	37.77	135.49
time (sec)	N/A	4.994	3.612	7.691	0.000	0.000	0.000	2.840	16.991

2.3 Detailed conclusion table specific for Rubi results

The following table is specific to Rubi only. It gives additional statistics for each integral. the column **steps** is the number of steps used by Rubi to obtain the antiderivative. The **rules** column is the number of unique rules used. The **integrand size** column is the leaf size of the integrand. Finally the ratio $\frac{\text{number of rules}}{\text{integrand size}}$ is also given. The larger this ratio is, the harder the integral is to solve. In this test file, problem number [5] had the largest ratio of [.4737000000000000010]

Table 2.1: Rubi specific breakdown of results for each integral

#	grade	number of steps used	number of unique rules	normalized antiderivative leaf size	integrand leaf size	$\frac{\text{number of rules}}{\text{integrand leaf size}}$
1	A	7	6	1.00	19	0.316
2	A	7	6	1.00	19	0.316
3	A	3	3	1.00	17	0.176
4	A	9	8	1.00	17	0.471
5	A	10	9	1.00	19	0.474
6	A	10	7	0.99	19	0.368
7	A	7	4	1.00	19	0.210
8	A	9	5	1.00	19	0.263
9	A	4	3	1.00	13	0.231
10	A	4	3	1.00	15	0.200
11	A	3	3	1.00	15	0.200
12	A	4	4	1.00	15	0.267
13	A	10	7	1.00	19	0.368
14	A	8	5	1.00	19	0.263
15	A	7	4	1.00	19	0.210
16	A	6	3	1.00	17	0.176
17	A	5	3	1.00	14	0.214
18	A	8	5	1.00	17	0.294
19	A	10	7	1.00	19	0.368
20	A	12	8	1.00	19	0.421

CHAPTER 3

LISTING OF INTEGRALS

3.1 $\int \frac{\sin^5(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 19, antiderivative size = 136

$$\int \frac{\sin^5(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{(b^4+2c^2(a+c)^2-2b^2c(2a+c))\operatorname{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{c^4\sqrt{b^2-4ac}} - \frac{(b^2-c(a+2c))\cos(x)}{c^3} + \frac{b\cos^2(x)}{2c^2} - \frac{\cos^3(x)}{3c} + \frac{b(b^2-2c(a+c))\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2c^4}$$

```
[Out] -(b^2-c*(a+2*c))*cos(x)/c^3+1/2*b*cos(x)^2/c^2-1/3*cos(x)^3/c+1/2*b*(b^2-2*
c*(a+c))*ln(a+b*cos(x)+c*cos(x)^2)/c^4+(b^4+2*c^2*(a+c)^2-2*b^2*c*(2*a+c))*
arctanh((b+2*c*cos(x))/(-4*a*c+b^2)^(1/2))/c^4/(-4*a*c+b^2)^(1/2)
```

Rubi [A] (verified)

Time = 0.26 (sec) , antiderivative size = 136, normalized size of antiderivative = 1.00, number of steps used = 7, number of rules used = 6, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.316, Rules used = {3340, 1671, 648, 632, 212, 642}

$$\int \frac{\sin^5(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{(-2b^2c(2a+c)+2c^2(a+c)^2+b^4)\operatorname{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{c^4\sqrt{b^2-4ac}} + \frac{b(b^2-2c(a+c))\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2c^4} - \frac{\cos(x)\left(b^2-c(a+2c)\right)}{c^3} + \frac{b\cos^2(x)}{2c^2} - \frac{\cos^3(x)}{3c}$$

```
[In] Int[Sin[x]^5/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] ((b^4 + 2*c^2*(a + c)^2 - 2*b^2*c*(2*a + c))*ArcTanh[(b + 2*c*Cos[x])/Sqrt[
b^2 - 4*a*c]])/(c^4*Sqrt[b^2 - 4*a*c]) - ((b^2 - c*(a + 2*c))*Cos[x])/c^3 +
 (b*Cos[x]^2)/(2*c^2) - Cos[x]^3/(3*c) + (b*(b^2 - 2*c*(a + c))*Log[a + b*C
os[x] + c*Cos[x]^2])/(2*c^4)
```

Rule 212

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(1/(Rt[a, 2]*Rt[-b, 2]))*
ArcTanh[Rt[-b, 2]*(x/Rt[a, 2])], x] /; FreeQ[{a, b}, x] && NegQ[a/b] && (Gt
Q[a, 0] || LtQ[b, 0])
```

Rule 632

```
Int[((a_.) + (b_.)*(x_) + (c_.)*(x_)^2)^(-1), x_Symbol] :> Dist[-2, Subst[I
nt[1/Simp[b^2 - 4*a*c - x^2, x], x], x, b + 2*c*x], x] /; FreeQ[{a, b, c},
x] && NeQ[b^2 - 4*a*c, 0]
```

Rule 642

```
Int[((d_) + (e_.)*(x_))/((a_.) + (b_.)*(x_) + (c_.)*(x_)^2), x_Symbol] :> S
imp[d*(Log[RemoveContent[a + b*x + c*x^2, x]]/b), x] /; FreeQ[{a, b, c, d,
e}, x] && EqQ[2*c*d - b*e, 0]
```

Rule 648

```
Int[((d_.) + (e_.)*(x_))/((a_) + (b_.)*(x_) + (c_.)*(x_)^2), x_Symbol] :> D
ist[(2*c*d - b*e)/(2*c), Int[1/(a + b*x + c*x^2), x], x] + Dist[e/(2*c), In
t[(b + 2*c*x)/(a + b*x + c*x^2), x], x] /; FreeQ[{a, b, c, d, e}, x] && NeQ
[2*c*d - b*e, 0] && NeQ[b^2 - 4*a*c, 0] &&  !NiceSqrtQ[b^2 - 4*a*c]
```

Rule 1671

```
Int[(Pq_)*((a_) + (b_.)*(x_) + (c_.)*(x_)^2)^(p_.), x_Symbol] :> Int[Expand
Integrand[Pq*(a + b*x + c*x^2)^p, x], x] /; FreeQ[{a, b, c}, x] && PolyQ[Pq
, x] && IGtQ[p, -2]
```

Rule 3340

```
Int[((a_.) + (b_.)*(cos[(d_.) + (e_.)*(x_)]*(f_.))^(n_.) + (c_.)*(cos[(d_.)
 + (e_.)*(x_)]*(f_.))^(n2_.))^(p_.)*sin[(d_.) + (e_.)*(x_)]^(m_.), x_Symbol
] :> Module[{g = FreeFactors[Cos[d + e*x], x]}, Dist[-g/e, Subst[Int[(1 - g
^2*x^2)^((m - 1)/2)*(a + b*(f*g*x)^n + c*(f*g*x)^(2*n))^p, x], x, Cos[d + e
*x]/g], x]] /; FreeQ[{a, b, c, d, e, f, n, p}, x] && EqQ[n2, 2*n] && Intege
rQ[(m - 1)/2]
```

Rubi steps

$$
\begin{aligned}
\text{integral} &= -\text{Subst}\left(\int \frac{(1-x^2)^2}{a+bx+cx^2}\,dx, x, \cos(x)\right) \\
&= -\text{Subst}\left(\int \left(\frac{b^2-c(a+2c)}{c^3} - \frac{bx}{c^2} + \frac{x^2}{c} - \frac{-a^2c-c^3+a(b^2-2c^2)+b(b^2-2c(a+c))\,x}{c^3\,(a+bx+cx^2)}\right) dx, x, \cos(x)\right) \\
&= -\frac{(b^2-c(a+2c))\cos(x)}{c^3} + \frac{b\cos^2(x)}{2c^2} - \frac{\cos^3(x)}{3c} + \frac{\text{Subst}\left(\int \frac{-a^2c-c^3+a(b^2-2c^2)+b(b^2-2c(a+c))x}{a+bx+cx^2}\,dx, x, \cos(x)\right)}{c^3} \\
&= -\frac{(b^2-c(a+2c))\cos(x)}{c^3} + \frac{b\cos^2(x)}{2c^2} - \frac{\cos^3(x)}{3c} + \frac{(b(b^2-2c(a+c)))\,\text{Subst}\left(\int \frac{b+2cx}{a+bx+cx^2}\,dx, x, \cos(x)\right)}{2c^4} \\
&\quad - \frac{(b^4+2c^2(a+c)^2-2b^2c(2a+c))\,\text{Subst}\left(\int \frac{1}{a+bx+cx^2}\,dx, x, \cos(x)\right)}{2c^4} \\
&= -\frac{(b^2-c(a+2c))\cos(x)}{c^3} + \frac{b\cos^2(x)}{2c^2} - \frac{\cos^3(x)}{3c} + \frac{b(b^2-2c(a+c))\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2c^4} \\
&\quad + \frac{(b^4+2c^2(a+c)^2-2b^2c(2a+c))\,\text{Subst}\left(\int \frac{1}{b^2-4ac-x^2}\,dx, x, b+2c\cos(x)\right)}{c^4} \\
&= \frac{(b^4+2c^2(a+c)^2-2b^2c(2a+c))\,\text{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{c^4\sqrt{b^2-4ac}} - \frac{(b^2-c(a+2c))\cos(x)}{c^3} \\
&\quad + \frac{b\cos^2(x)}{2c^2} - \frac{\cos^3(x)}{3c} + \frac{b(b^2-2c(a+c))\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2c^4}
\end{aligned}
$$

Mathematica [A] (verified)

Time = 0.47 (sec) , antiderivative size = 239, normalized size of antiderivative = 1.76

$$
\int \frac{\sin^5(x)}{a+b\cos(x)+c\cos^2(x)}\,dx
$$

$$
= \frac{3c(-4b^2+c(4a+7c))\cos(x)+3bc^2\cos(2x)-c^3\cos(3x)+\frac{6\left(-b^4-2c^2(a+c)^2+2b^2c(2a+c)+b^3\sqrt{b^2-4ac}-2bc(a+c)\sqrt{b^2-4}}{\sqrt{b^2-4ac}}}{12c^4}
$$

```
[In] Integrate[Sin[x]^5/(a + b*Cos[x] + c*Cos[x]^2),x]
```

[Out] (3*c*(-4*b^2 + c*(4*a + 7*c))*Cos[x] + 3*b*c^2*Cos[2*x] - c^3*Cos[3*x] + (6*(-b^4 - 2*c^2*(a + c)^2 + 2*b^2*c*(2*a + c) + b^3*Sqrt[b^2 - 4*a*c] - 2*b*c*(a + c)*Sqrt[b^2 - 4*a*c])*Log[-b + Sqrt[b^2 - 4*a*c] - 2*c*Cos[x]])/Sqrt[b^2 - 4*a*c] + (6*(b^4 + 2*c^2*(a + c)^2 - 2*b^2*c*(2*a + c) + b^3*Sqrt[b^2 - 4*a*c] - 2*b*c*(a + c)*Sqrt[b^2 - 4*a*c])*Log[b + Sqrt[b^2 - 4*a*c] + 2*c*Cos[x]])/Sqrt[b^2 - 4*a*c])/(12*c^4)

Maple [A] (verified)

Time = 2.36 (sec) , antiderivative size = 160, normalized size of antiderivative = 1.18

method	result
derivativedivides	$\frac{-\frac{(\cos^3(x))c^2}{3}+\frac{(\cos^2(x))bc}{2}+\cos(x)ac-b^2\cos(x)+2c^2\cos(x)}{c^3}+\frac{\frac{(-2cab+b^3-2bc^2)\ln(a+\cos(x)b+c(\cos^2(x)))}{2c}+\frac{2(-a^2c+\cdots}{\cdots}}{c\cdots}$
default	$\frac{-\frac{(\cos^3(x))c^2}{3}+\frac{(\cos^2(x))bc}{2}+\cos(x)ac-b^2\cos(x)+2c^2\cos(x)}{c^3}+\frac{\frac{(-2cab+b^3-2bc^2)\ln(a+\cos(x)b+c(\cos^2(x)))}{2c}+\frac{2(-a^2c+\cdots}{\cdots}}{c\cdots}$
risch	Expression too large to display

[In] int(sin(x)^5/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)

[Out] 1/c^3*(-1/3*cos(x)^3*c^2+1/2*cos(x)^2*b*c+cos(x)*a*c-b^2*cos(x)+2*c^2*cos(x))+1/c^3*(1/2*(-2*a*b*c+b^3-2*b*c^2)/c*ln(a+cos(x)*b+c*cos(x)^2)+2*(-a^2*c+a*b^2-2*a*c^2-c^3-1/2*(-2*a*b*c+b^3-2*b*c^2)*b/c)/(4*a*c-b^2)^(1/2)*arctan((b+2*c*cos(x))/(4*a*c-b^2)^(1/2)))

Fricas [A] (verification not implemented)

none

Time = 0.36 (sec) , antiderivative size = 491, normalized size of antiderivative = 3.61

$$\int \frac{\sin^5(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$$
$$=\left[-\frac{2\,(b^2c^3-4\,ac^4)\cos(x)^3-3\,(b^3c^2-4\,abc^3)\cos(x)^2-3\,(b^4-4\,ab^2c+4\,ac^3+2\,c^4+2\,(a^2-b^2)c^2)\sqrt{b^2\cdots}}{\cdots}\right.$$
$$-\frac{2\,(b^2c^3-4\,ac^4)\cos(x)^3-3\,(b^3c^2-4\,abc^3)\cos(x)^2-6\,(b^4-4\,ab^2c+4\,ac^3+2\,c^4+2\,(a^2-b^2)c^2)\sqrt{-b^2\cdots}}{\cdots}$$

[In] integrate(sin(x)^5/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")

[Out] [-1/6*(2*(b^2*c^3 - 4*a*c^4)*cos(x)^3 - 3*(b^3*c^2 - 4*a*b*c^3)*cos(x)^2 - 3*(b^4 - 4*a*b^2*c + 4*a*c^3 + 2*c^4 + 2*(a^2 - b^2)*c^2)*sqrt(b^2 - 4*a*c)

```
*log(-(2*c^2*cos(x)^2 + 2*b*c*cos(x) + b^2 - 2*a*c + sqrt(b^2 - 4*a*c)*(2*c
*cos(x) + b))/(c*cos(x)^2 + b*cos(x) + a)) + 6*(b^4*c - 5*a*b^2*c^2 + 8*a*c
^4 + 2*(2*a^2 - b^2)*c^3)*cos(x) - 3*(b^5 - 6*a*b^3*c + 8*a*b*c^3 + 2*(4*a^
2*b - b^3)*c^2)*log(c*cos(x)^2 + b*cos(x) + a))/(b^2*c^4 - 4*a*c^5), -1/6*(
2*(b^2*c^3 - 4*a*c^4)*cos(x)^3 - 3*(b^3*c^2 - 4*a*b*c^3)*cos(x)^2 - 6*(b^4
- 4*a*b^2*c + 4*a*c^3 + 2*c^4 + 2*(a^2 - b^2)*c^2)*sqrt(-b^2 + 4*a*c)*arcta
n(-sqrt(-b^2 + 4*a*c)*(2*c*cos(x) + b)/(b^2 - 4*a*c)) + 6*(b^4*c - 5*a*b^2*
c^2 + 8*a*c^4 + 2*(2*a^2 - b^2)*c^3)*cos(x) - 3*(b^5 - 6*a*b^3*c + 8*a*b*c^
3 + 2*(4*a^2*b - b^3)*c^2)*log(c*cos(x)^2 + b*cos(x) + a))/(b^2*c^4 - 4*a*c
^5)]
```

Sympy [F(-1)]

Timed out.

$$\int \frac{\sin^5(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Timed out}$$

```
[In] integrate(sin(x)**5/(a+b*cos(x)+c*cos(x)**2),x)
```

```
[Out] Timed out
```

Maxima [F(-2)]

Exception generated.

$$\int \frac{\sin^5(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Exception raised: ValueError}$$

```
[In] integrate(sin(x)^5/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
```

```
[Out] Exception raised: ValueError >> Computation failed since Maxima requested a
dditional constraints; using the 'assume' command before evaluation *may* h
elp (example of legal syntax is 'assume(4*a*c-b^2>0)', see `assume?` for mo
re deta
```

Giac [A] (verification not implemented)

none

Time = 0.30 (sec) , antiderivative size = 153, normalized size of antiderivative = 1.12

$$\int \frac{\sin^5(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{2\,c^2\cos{(x)}^3-3\,bc\cos{(x)}^2+6\,b^2\cos{(x)}-6\,ac\cos{(x)}-12\,c^2\cos{(x)}}{6\,c^3} + \frac{(b^3-2\,abc-2\,bc^2)\log{\left(c\cos{(x)}^2+b\cos{(x)}+a\right)}}{2\,c^4} - \frac{(b^4-4\,ab^2c+2\,a^2c^2-2\,b^2c^2+4\,ac^3+2\,c^4)\arctan\left(\frac{2\,c\cos(x)+b}{\sqrt{-b^2+4\,ac}}\right)}{\sqrt{-b^2+4\,ac}c^4}$$

```
[In] integrate(sin(x)^5/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] -1/6*(2*c^2*cos(x)^3 - 3*b*c*cos(x)^2 + 6*b^2*cos(x) - 6*a*c*cos(x) - 12*c^
2*cos(x))/c^3 + 1/2*(b^3 - 2*a*b*c - 2*b*c^2)*log(c*cos(x)^2 + b*cos(x) + a
)/c^4 - (b^4 - 4*a*b^2*c + 2*a^2*c^2 - 2*b^2*c^2 + 4*a*c^3 + 2*c^4)*arctan(
(2*c*cos(x) + b)/sqrt(-b^2 + 4*a*c))/(sqrt(-b^2 + 4*a*c)*c^4)
```

Mupad [B] (verification not implemented)

Time = 2.74 (sec) , antiderivative size = 197, normalized size of antiderivative = 1.45

$$\int \frac{\sin^5(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \cos{(x)}\,\left(\frac{a}{c^2}+\frac{2}{c}-\frac{b^2}{c^3}\right)-\frac{\cos{(x)}^3}{3\,c} - \frac{\ln{\left(c\cos{(x)}^2+b\,\cos{(x)}+a\right)}\,\left(8\,a^2\,b\,c^2-6\,a\,b^3\,c+8\,a\,b\,c^3+b^5-2\,b^3\,c^2\right)}{2\,\left(4\,a\,c^5-b^2\,c^4\right)}+\frac{b\cos{(x)}^2}{2\,c^2} - \frac{\mathrm{atan}\left(\frac{b}{\sqrt{4\,a\,c-b^2}}+\frac{2\,c\,\cos(x)}{\sqrt{4\,a\,c-b^2}}\right)\,\left(2\,a^2\,c^2-4\,a\,b^2\,c+4\,a\,c^3+b^4-2\,b^2\,c^2+2\,c^4\right)}{c^4\,\sqrt{4\,a\,c-b^2}}$$

```
[In] int(sin(x)^5/(a + b*cos(x) + c*cos(x)^2),x)
```

```
[Out] cos(x)*(a/c^2 + 2/c - b^2/c^3) - cos(x)^3/(3*c) - (log(a + b*cos(x) + c*cos
(x)^2)*(b^5 - 2*b^3*c^2 + 8*a^2*b*c^2 + 8*a*b*c^3 - 6*a*b^3*c))/(2*(4*a*c^5
 - b^2*c^4)) + (b*cos(x)^2)/(2*c^2) - (atan(b/(4*a*c - b^2)^(1/2) + (2*c*co
s(x))/(4*a*c - b^2)^(1/2))*(4*a*c^3 + b^4 + 2*c^4 + 2*a^2*c^2 - 2*b^2*c^2 -
 4*a*b^2*c))/(c^4*(4*a*c - b^2)^(1/2))
```

3.2 $\int \frac{\sin^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 19, antiderivative size = 76

$$\int \frac{\sin^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{(b^2-2c(a+c))\,\text{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{c^2\sqrt{b^2-4ac}} + \frac{\cos(x)}{c} - \frac{b\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2c^2}$$

```
[Out] cos(x)/c-1/2*b*ln(a+b*cos(x)+c*cos(x)^2)/c^2-(b^2-2*c*(a+c))*arctanh((b+2*c
*cos(x))/(-4*a*c+b^2)^(1/2))/c^2/(-4*a*c+b^2)^(1/2)
```

Rubi [A] (verified)

Time = 0.15 (sec) , antiderivative size = 76, normalized size of antiderivative = 1.00, number of steps used = 7, number of rules used = 6, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.316, Rules used = {3340, 1671, 648, 632, 212, 642}

$$\int \frac{\sin^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{(b^2-2c(a+c))\,\text{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{c^2\sqrt{b^2-4ac}} - \frac{b\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2c^2} + \frac{\cos(x)}{c}$$

```
[In] Int[Sin[x]^3/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] -(((b^2 - 2*c*(a + c))*ArcTanh[(b + 2*c*Cos[x])/Sqrt[b^2 - 4*a*c]])/(c^2*Sq
rt[b^2 - 4*a*c])) + Cos[x]/c - (b*Log[a + b*Cos[x] + c*Cos[x]^2])/(2*c^2)
```

Rule 212

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(1/(Rt[a, 2]*Rt[-b, 2]))*
ArcTanh[Rt[-b, 2]*(x/Rt[a, 2])], x] /; FreeQ[{a, b}, x] && NegQ[a/b] && (Gt
Q[a, 0] || LtQ[b, 0])
```

Rule 632

```
Int[((a_.) + (b_.)*(x_) + (c_.)*(x_)^2)^(-1), x_Symbol] :> Dist[-2, Subst[I
nt[1/Simp[b^2 - 4*a*c - x^2, x], x], x, b + 2*c*x], x] /; FreeQ[{a, b, c},
x] && NeQ[b^2 - 4*a*c, 0]
```

Rule 642

```
Int[((d_) + (e_.)*(x_))/((a_.) + (b_.)*(x_) + (c_.)*(x_)^2), x_Symbol] :> S
imp[d*(Log[RemoveContent[a + b*x + c*x^2, x]]/b), x] /; FreeQ[{a, b, c, d,
e}, x] && EqQ[2*c*d - b*e, 0]
```

Rule 648

```
Int[((d_.) + (e_.)*(x_))/((a_) + (b_.)*(x_) + (c_.)*(x_)^2), x_Symbol] :> D
ist[(2*c*d - b*e)/(2*c), Int[1/(a + b*x + c*x^2), x], x] + Dist[e/(2*c), In
t[(b + 2*c*x)/(a + b*x + c*x^2), x], x] /; FreeQ[{a, b, c, d, e}, x] && NeQ
[2*c*d - b*e, 0] && NeQ[b^2 - 4*a*c, 0] &&  !NiceSqrtQ[b^2 - 4*a*c]
```

Rule 1671

```
Int[(Pq_)*((a_) + (b_.)*(x_) + (c_.)*(x_)^2)^(p_.), x_Symbol] :> Int[Expand
Integrand[Pq*(a + b*x + c*x^2)^p, x], x] /; FreeQ[{a, b, c}, x] && PolyQ[Pq
, x] && IGtQ[p, -2]
```

Rule 3340

```
Int[((a_.) + (b_.)*(cos[(d_.) + (e_.)*(x_)]*(f_.))^(n_.) + (c_.)*(cos[(d_.)
 + (e_.)*(x_)]*(f_.))^(n2_.))^(p_.)*sin[(d_.) + (e_.)*(x_)]^(m_.), x_Symbol
] :> Module[{g = FreeFactors[Cos[d + e*x], x]}, Dist[-g/e, Subst[Int[(1 - g
^2*x^2)^((m - 1)/2)*(a + b*(f*g*x)^n + c*(f*g*x)^(2*n))^p, x], x, Cos[d + e
*x]/g], x]] /; FreeQ[{a, b, c, d, e, f, n, p}, x] && EqQ[n2, 2*n] && Intege
rQ[(m - 1)/2]
```

Rubi steps

$$\text{integral} = -\text{Subst}\left(\int \frac{1-x^2}{a+bx+cx^2}\,dx, x, \cos(x)\right)$$

$$= -\text{Subst}\left(\int \left(-\frac{1}{c}+\frac{a+c+bx}{c\,(a+bx+cx^2)}\right)\,dx, x, \cos(x)\right)$$

$$= \frac{\cos(x)}{c} - \frac{\text{Subst}\left(\int \frac{a+c+bx}{a+bx+cx^2}\,dx, x, \cos(x)\right)}{c}$$

$$= \frac{\cos(x)}{c} - \frac{b\text{Subst}\left(\int \frac{b+2cx}{a+bx+cx^2}\,dx, x, \cos(x)\right)}{2c^2} + \frac{(b^2-2c(a+c))\,\text{Subst}\left(\int \frac{1}{a+bx+cx^2}\,dx, x, \cos(x)\right)}{2c^2}$$

$$= \frac{\cos(x)}{c} - \frac{b\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2c^2} - \frac{(b^2-2c(a+c))\,\text{Subst}\left(\int \frac{1}{b^2-4ac-x^2}\,dx, x, b+2c\cos(x)\right)}{c^2}$$

$$= -\frac{(b^2-2c(a+c))\,\text{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{c^2\sqrt{b^2-4ac}} + \frac{\cos(x)}{c} - \frac{b\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2c^2}$$

Mathematica [A] (verified)

Time = 0.23 (sec) , antiderivative size = 131, normalized size of antiderivative = 1.72

$$\int \frac{\sin^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$$

$$= \frac{2c\sqrt{b^2-4ac}\cos(x) + \left(b^2-2c(a+c)-b\sqrt{b^2-4ac}\right)\log\left(-b+\sqrt{b^2-4ac}-2c\cos(x)\right) - \left(b^2-2c(a+c)+\right.}{2c^2\sqrt{b^2-4ac}}$$

```
[In] Integrate[Sin[x]^3/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (2*c*Sqrt[b^2 - 4*a*c]*Cos[x] + (b^2 - 2*c*(a + c) - b*Sqrt[b^2 - 4*a*c])*L
og[-b + Sqrt[b^2 - 4*a*c] - 2*c*Cos[x]] - (b^2 - 2*c*(a + c) + b*Sqrt[b^2 -
 4*a*c])*Log[b + Sqrt[b^2 - 4*a*c] + 2*c*Cos[x]])/(2*c^2*Sqrt[b^2 - 4*a*c])
```

Maple [A] (verified)

Time = 0.98 (sec) , antiderivative size = 82, normalized size of antiderivative = 1.08

method	result	size
derivativedivides	$\frac{\cos(x)}{c}+\frac{-\frac{b\ln\left(a+\cos(x)b+c\left(\cos^2(x)\right)\right)}{2c}+\frac{2\left(-a-c+\frac{b^2}{2c}\right)\arctan\left(\frac{b+2c\cos(x)}{\sqrt{4ac-b^2}}\right)}{\sqrt{4ac-b^2}}}{c}$	82
default	$\frac{\cos(x)}{c}+\frac{-\frac{b\ln\left(a+\cos(x)b+c\left(\cos^2(x)\right)\right)}{2c}+\frac{2\left(-a-c+\frac{b^2}{2c}\right)\arctan\left(\frac{b+2c\cos(x)}{\sqrt{4ac-b^2}}\right)}{\sqrt{4ac-b^2}}}{c}$	82
risch	Expression too large to display	1094

```
[In] int(sin(x)^3/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] cos(x)/c+1/c*(-1/2*b/c*ln(a+cos(x)*b+c*cos(x)^2)+2*(-a-c+1/2*b^2/c)/(4*a*c-
b^2)^(1/2)*arctan((b+2*c*cos(x))/(4*a*c-b^2)^(1/2)))
```

Fricas [A] (verification not implemented)

none

Time = 0.31 (sec) , antiderivative size = 258, normalized size of antiderivative = 3.39

$$\int \frac{\sin^3(x)}{a + b\cos(x) + c\cos^2(x)}\,dx$$
$$= \left[-\frac{(b^2 - 2\,ac - 2\,c^2)\sqrt{b^2 - 4\,ac}\log\left(-\frac{2\,c^2\cos(x)^2 + 2\,bc\cos(x) + b^2 - 2\,ac + \sqrt{b^2 - 4\,ac}(2\,c\cos(x) + b)}{c\cos(x)^2 + b\cos(x) + a}\right) - 2\,(b^2c - 4\,ac^2)\cos\left(\right.}{2\,(b^2c^2 - 4\,ac^3)}\right.$$
$$-\frac{2\,(b^2 - 2\,ac - 2\,c^2)\sqrt{-b^2 + 4\,ac}\arctan\left(-\frac{\sqrt{-b^2 + 4\,ac}(2\,c\cos(x) + b)}{b^2 - 4\,ac}\right) - 2\,(b^2c - 4\,ac^2)\cos(x) + (b^3 - 4\,abc)\,\mathrm{l}}{2\,(b^2c^2 - 4\,ac^3)}$$

```
[In] integrate(sin(x)^3/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
```

```
[Out] [-1/2*((b^2 - 2*a*c - 2*c^2)*sqrt(b^2 - 4*a*c)*log(-(2*c^2*cos(x)^2 + 2*b*c
*cos(x) + b^2 - 2*a*c + sqrt(b^2 - 4*a*c)*(2*c*cos(x) + b))/(c*cos(x)^2 + b
*cos(x) + a)) - 2*(b^2*c - 4*a*c^2)*cos(x) + (b^3 - 4*a*b*c)*log(c*cos(x)^2
 + b*cos(x) + a))/(b^2*c^2 - 4*a*c^3), -1/2*(2*(b^2 - 2*a*c - 2*c^2)*sqrt(-
b^2 + 4*a*c)*arctan(-sqrt(-b^2 + 4*a*c)*(2*c*cos(x) + b)/(b^2 - 4*a*c)) - 2
*(b^2*c - 4*a*c^2)*cos(x) + (b^3 - 4*a*b*c)*log(c*cos(x)^2 + b*cos(x) + a))
/(b^2*c^2 - 4*a*c^3)]
```

Sympy [F(-1)]

Timed out.

$$\int \frac{\sin^3(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Timed out}$$

```
[In] integrate(sin(x)**3/(a+b*cos(x)+c*cos(x)**2),x)
```

```
[Out] Timed out
```

Maxima [F(-2)]

Exception generated.

$$\int \frac{\sin^3(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Exception raised: ValueError}$$

```
[In] integrate(sin(x)^3/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
```

```
[Out] Exception raised: ValueError >> Computation failed since Maxima requested a
dditional constraints; using the 'assume' command before evaluation *may* h
elp (example of legal syntax is 'assume(4*a*c-b^2>0)', see `assume?` for mo
re deta
```

Giac [A] (verification not implemented)

none

Time = 0.30 (sec) , antiderivative size = 76, normalized size of antiderivative = 1.00

$$\int \frac{\sin^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{\cos(x)}{c} - \frac{b\log\left(c\cos(x)^2+b\cos(x)+a\right)}{2\,c^2} + \frac{(b^2-2\,ac-2\,c^2)\arctan\left(\frac{2\,c\cos(x)+b}{\sqrt{-b^2+4\,ac}}\right)}{\sqrt{-b^2+4\,ac}c^2}$$

```
[In] integrate(sin(x)^3/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] cos(x)/c - 1/2*b*log(c*cos(x)^2 + b*cos(x) + a)/c^2 + (b^2 - 2*a*c - 2*c^2)
*arctan((2*c*cos(x) + b)/sqrt(-b^2 + 4*a*c))/(sqrt(-b^2 + 4*a*c)*c^2)
```

Mupad [B] (verification not implemented)

Time = 0.22 (sec) , antiderivative size = 226, normalized size of antiderivative = 2.97

$$\int \frac{\sin^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{\cos(x)}{c} - \frac{2\,\mathrm{atan}\left(\frac{b}{\sqrt{4\,a\,c-b^2}}+\frac{2\,c\,\cos(x)}{\sqrt{4\,a\,c-b^2}}\right)}{\sqrt{4\,a\,c-b^2}} + \frac{b^3\,\ln\left(c\cos(x)^2+b\,\cos(x)+a\right)}{2\,\left(4\,a\,c^3-b^2\,c^2\right)} + \frac{b^2\,\mathrm{atan}\left(\frac{b}{\sqrt{4\,a\,c-b^2}}+\frac{2\,c\,\cos(x)}{\sqrt{4\,a\,c-b^2}}\right)}{c^2\,\sqrt{4\,a\,c-b^2}} - \frac{2\,a\,\mathrm{atan}\left(\frac{b}{\sqrt{4\,a\,c-b^2}}+\frac{2\,c\,\cos(x)}{\sqrt{4\,a\,c-b^2}}\right)}{c\,\sqrt{4\,a\,c-b^2}} - \frac{2\,a\,b\,c\,\ln\left(c\cos(x)^2+b\,\cos(x)+a\right)}{4\,a\,c^3-b^2\,c^2}$$

```
[In] int(sin(x)^3/(a + b*cos(x) + c*cos(x)^2),x)
```

```
[Out] cos(x)/c - (2*atan(b/(4*a*c - b^2)^(1/2) + (2*c*cos(x))/(4*a*c - b^2)^(1/2)
))/(4*a*c - b^2)^(1/2) + (b^3*log(a + b*cos(x) + c*cos(x)^2))/(2*(4*a*c^3 -
 b^2*c^2)) + (b^2*atan(b/(4*a*c - b^2)^(1/2) + (2*c*cos(x))/(4*a*c - b^2)^(
1/2)))/(c^2*(4*a*c - b^2)^(1/2)) - (2*a*atan(b/(4*a*c - b^2)^(1/2) + (2*c*c
os(x))/(4*a*c - b^2)^(1/2)))/(c*(4*a*c - b^2)^(1/2)) - (2*a*b*c*log(a + b*c
os(x) + c*cos(x)^2))/(4*a*c^3 - b^2*c^2)
```

3.3 $\int \frac{\sin(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 17, antiderivative size = 35

$$\int \frac{\sin(x)}{a+b\cos(x)+c\cos^2(x)}\,dx=\frac{2\text{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{\sqrt{b^2-4ac}}$$

```
[Out] 2*arctanh((b+2*c*cos(x))/(-4*a*c+b^2)^(1/2))/(-4*a*c+b^2)^(1/2)
```

Rubi [A] (verified)

Time = 0.05 (sec) , antiderivative size = 35, normalized size of antiderivative = 1.00, number of steps used = 3, number of rules used = 3, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.176, Rules used = {3340, 632, 212}

$$\int \frac{\sin(x)}{a+b\cos(x)+c\cos^2(x)}\,dx=\frac{2\text{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{\sqrt{b^2-4ac}}$$

```
[In] Int[Sin[x]/(a + b*Cos[x] + c*Cos[x]^2),x]
[Out] (2*ArcTanh[(b + 2*c*Cos[x])/Sqrt[b^2 - 4*a*c]])/Sqrt[b^2 - 4*a*c]
```

Rule 212

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(1/(Rt[a, 2]*Rt[-b, 2]))*
ArcTanh[Rt[-b, 2]*(x/Rt[a, 2])], x] /; FreeQ[{a, b}, x] && NegQ[a/b] && (Gt
Q[a, 0] || LtQ[b, 0])
```

Rule 632

```
Int[((a_.) + (b_.)*(x_) + (c_.)*(x_)^2)^(-1), x_Symbol] :> Dist[-2, Subst[I
nt[1/Simp[b^2 - 4*a*c - x^2, x], x], x, b + 2*c*x], x] /; FreeQ[{a, b, c},
```

```
x] && NeQ[b^2 - 4*a*c, 0]
```

Rule 3340

```
Int[((a_.) + (b_.)*(cos[(d_.) + (e_.)*(x_)]*(f_.))^(n_.) + (c_.)*(cos[(d_.)
 + (e_.)*(x_)]*(f_.))^(n2_.))^(p_.)*sin[(d_.) + (e_.)*(x_)]^(m_.), x_Symbol
] :> Module[{g = FreeFactors[Cos[d + e*x], x]}, Dist[-g/e, Subst[Int[(1 - g
^2*x^2)^((m - 1)/2)*(a + b*(f*g*x)^n + c*(f*g*x)^(2*n))^p, x], x, Cos[d + e
*x]/g], x]] /; FreeQ[{a, b, c, d, e, f, n, p}, x] && EqQ[n2, 2*n] && Intege
rQ[(m - 1)/2]
```

Rubi steps

$$\begin{aligned}\text{integral} &= -\text{Subst}\left(\int \frac{1}{a+bx+cx^2}\,dx, x, \cos(x)\right)\\ &= 2\text{Subst}\left(\int \frac{1}{b^2-4ac-x^2}\,dx, x, b+2c\cos(x)\right)\\ &= \frac{2\text{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{\sqrt{b^2-4ac}}\end{aligned}$$

Mathematica [A] (verified)

Time = 0.05 (sec) , antiderivative size = 39, normalized size of antiderivative = 1.11

$$\int \frac{\sin(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{2\arctan\left(\frac{b+2c\cos(x)}{\sqrt{-b^2+4ac}}\right)}{\sqrt{-b^2+4ac}}$$

```
[In] Integrate[Sin[x]/(a + b*Cos[x] + c*Cos[x]^2),x]
[Out] (-2*ArcTan[(b + 2*c*Cos[x])/Sqrt[-b^2 + 4*a*c]])/Sqrt[-b^2 + 4*a*c]
```

Maple [A] (verified)

Time = 0.28 (sec) , antiderivative size = 36, normalized size of antiderivative = 1.03

method	result	size
derivativedivides	$-\frac{2\arctan\left(\frac{b+2c\cos(x)}{\sqrt{4ac-b^2}}\right)}{\sqrt{4ac-b^2}}$	36
default	$-\frac{2\arctan\left(\frac{b+2c\cos(x)}{\sqrt{4ac-b^2}}\right)}{\sqrt{4ac-b^2}}$	36
risch	$-\frac{i\ln\left(e^{2ix}+\frac{\left(4iac-ib^2+b\sqrt{4ac-b^2}\right)e^{ix}}{c\sqrt{4ac-b^2}}+1\right)}{\sqrt{4ac-b^2}}+\frac{i\ln\left(e^{2ix}+\frac{\left(-4iac+ib^2+b\sqrt{4ac-b^2}\right)e^{ix}}{c\sqrt{4ac-b^2}}+1\right)}{\sqrt{4ac-b^2}}$	142

```
[In] int(sin(x)/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
[Out] -2/(4*a*c-b^2)^(1/2)*arctan((b+2*c*cos(x))/(4*a*c-b^2)^(1/2))
```

Fricas [A] (verification not implemented)

none

Time = 0.27 (sec) , antiderivative size = 126, normalized size of antiderivative = 3.60

$$\int \frac{\sin(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \left[\frac{\log\left(-\frac{2c^2\cos(x)^2+2bc\cos(x)+b^2-2ac+\sqrt{b^2-4ac}(2c\cos(x)+b)}{c\cos(x)^2+b\cos(x)+a}\right)}{\sqrt{b^2-4ac}}, \frac{2\sqrt{-b^2+4ac}\arctan\left(-\frac{\sqrt{-b^2+4ac}(2c\cos(x)+b)}{b^2-4ac}\right)}{b^2-4ac}\right]$$

```
[In] integrate(sin(x)/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
[Out] [log(-(2*c^2*cos(x)^2 + 2*b*c*cos(x) + b^2 - 2*a*c + sqrt(b^2 - 4*a*c)*(2*c
*cos(x) + b))/(c*cos(x)^2 + b*cos(x) + a))/sqrt(b^2 - 4*a*c), 2*sqrt(-b^2 +
 4*a*c)*arctan(-sqrt(-b^2 + 4*a*c)*(2*c*cos(x) + b)/(b^2 - 4*a*c))/(b^2 - 4
*a*c)]
```

Sympy [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 99 vs. 2(34) = 68.

Time = 1.28 (sec) , antiderivative size = 99, normalized size of antiderivative = 2.83

$$\int \frac{\sin(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \begin{cases} -\frac{\log\left(\frac{a}{b}+\cos(x)\right)}{b} & \text{for } c=0 \\ \frac{2}{b+2c\cos(x)} & \text{for } a=\frac{b^2}{4c} \\ -\frac{\log\left(\frac{b}{2c}+\cos(x)-\frac{\sqrt{-4ac+b^2}}{2c}\right)}{\sqrt{-4ac+b^2}}+\frac{\log\left(\frac{b}{2c}+\cos(x)+\frac{\sqrt{-4ac+b^2}}{2c}\right)}{\sqrt{-4ac+b^2}} & \text{otherwise} \end{cases}$$

```
[In] integrate(sin(x)/(a+b*cos(x)+c*cos(x)**2),x)
```

```
[Out] Piecewise((-log(a/b + cos(x))/b, Eq(c, 0)), (2/(b + 2*c*cos(x)), Eq(a, b**2
/(4*c))), (-log(b/(2*c) + cos(x) - sqrt(-4*a*c + b**2)/(2*c))/sqrt(-4*a*c +
 b**2) + log(b/(2*c) + cos(x) + sqrt(-4*a*c + b**2)/(2*c))/sqrt(-4*a*c + b*
*2), True))
```

Maxima [F(-2)]

Exception generated.

$$\int \frac{\sin(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Exception raised: ValueError}$$

```
[In] integrate(sin(x)/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
```

```
[Out] Exception raised: ValueError >> Computation failed since Maxima requested a
dditional constraints; using the 'assume' command before evaluation *may* h
elp (example of legal syntax is 'assume(4*a*c-b^2>0)', see `assume?` for mo
re deta
```

Giac [A] (verification not implemented)

none

Time = 0.30 (sec) , antiderivative size = 35, normalized size of antiderivative = 1.00

$$\int \frac{\sin(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = -\frac{2\arctan\left(\frac{2\,c\cos(x)+b}{\sqrt{-b^2+4\,ac}}\right)}{\sqrt{-b^2+4\,ac}}$$

```
[In] integrate(sin(x)/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] -2*arctan((2*c*cos(x) + b)/sqrt(-b^2 + 4*a*c))/sqrt(-b^2 + 4*a*c)
```

Mupad [B] (verification not implemented)

Time = 3.18 (sec) , antiderivative size = 47, normalized size of antiderivative = 1.34

$$\int \frac{\sin(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = -\frac{2\,\mathrm{atan}\left(\frac{b}{\sqrt{4\,a\,c-b^2}} + \frac{2\,c\,\cos(x)}{\sqrt{4\,a\,c-b^2}}\right)}{\sqrt{4\,a\,c-b^2}}$$

```
[In] int(sin(x)/(a + b*cos(x) + c*cos(x)^2),x)
```

```
[Out] -(2*atan(b/(4*a*c - b^2)^(1/2) + (2*c*cos(x))/(4*a*c - b^2)^(1/2)))/(4*a*c
- b^2)^(1/2)
```

3.4 $\int \frac{\csc(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 17, antiderivative size = 129

$$\int \frac{\csc(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{(b^2-2ac-2c^2)\operatorname{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{(a-b+c)(a+b+c)\sqrt{b^2-4ac}}+\frac{\log(1-\cos(x))}{2(a+b+c)} - \frac{\log(1+\cos(x))}{2(a-b+c)}+\frac{b\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2(a-b+c)(a+b+c)}$$

```
[Out] 1/2*ln(1-cos(x))/(a+b+c)-1/2*ln(1+cos(x))/(a-b+c)+1/2*b*ln(a+b*cos(x)+c*cos
(x)^2)/(a-b+c)/(a+b+c)-(-2*a*c+b^2-2*c^2)*arctanh((b+2*c*cos(x))/(-4*a*c+b^
2)^(1/2))/(a-b+c)/(a+b+c)/(-4*a*c+b^2)^(1/2)
```

Rubi [A] (verified)

Time = 0.20 (sec) , antiderivative size = 129, normalized size of antiderivative = 1.00, number of steps used = 9, number of rules used = 8, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.471, Rules used = {3340, 995, 648, 632, 212, 642, 647, 31}

$$\int \frac{\csc(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{(-2ac+b^2-2c^2)\operatorname{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{(a-b+c)(a+b+c)\sqrt{b^2-4ac}} + \frac{b\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2(a-b+c)(a+b+c)} + \frac{\log(1-\cos(x))}{2(a+b+c)}-\frac{\log(\cos(x)+1)}{2(a-b+c)}$$

```
[In] Int[Csc[x]/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] -(((b^2 - 2*a*c - 2*c^2)*ArcTanh[(b + 2*c*Cos[x])/Sqrt[b^2 - 4*a*c]])/((a -
 b + c)*(a + b + c)*Sqrt[b^2 - 4*a*c])) + Log[1 - Cos[x]]/(2*(a + b + c)) -
```

```
 Log[1 + Cos[x]]/(2*(a - b + c)) + (b*Log[a + b*Cos[x] + c*Cos[x]^2])/(2*(a
 - b + c)*(a + b + c))
```

Rule 31

```
Int[((a_) + (b_.)*(x_))^(-1), x_Symbol] :> Simp[Log[RemoveContent[a + b*x,
x]]/b, x] /; FreeQ[{a, b}, x]
```

Rule 212

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(1/(Rt[a, 2]*Rt[-b, 2]))*
ArcTanh[Rt[-b, 2]*(x/Rt[a, 2])], x] /; FreeQ[{a, b}, x] && NegQ[a/b] && (Gt
Q[a, 0] || LtQ[b, 0])
```

Rule 632

```
Int[((a_.) + (b_.)*(x_) + (c_.)*(x_)^2)^(-1), x_Symbol] :> Dist[-2, Subst[I
nt[1/Simp[b^2 - 4*a*c - x^2, x], x], x, b + 2*c*x], x] /; FreeQ[{a, b, c},
x] && NeQ[b^2 - 4*a*c, 0]
```

Rule 642

```
Int[((d_) + (e_.)*(x_))/((a_.) + (b_.)*(x_) + (c_.)*(x_)^2), x_Symbol] :> S
imp[d*(Log[RemoveContent[a + b*x + c*x^2, x]]/b), x] /; FreeQ[{a, b, c, d,
e}, x] && EqQ[2*c*d - b*e, 0]
```

Rule 647

```
Int[((d_) + (e_.)*(x_))/((a_) + (c_.)*(x_)^2), x_Symbol] :> With[{q = Rt[(-
a)*c, 2]}, Dist[e/2 + c*(d/(2*q)), Int[1/(-q + c*x), x], x] + Dist[e/2 - c*
(d/(2*q)), Int[1/(q + c*x), x], x]] /; FreeQ[{a, c, d, e}, x] && NiceSqrtQ[
(-a)*c]
```

Rule 648

```
Int[((d_.) + (e_.)*(x_))/((a_) + (b_.)*(x_) + (c_.)*(x_)^2), x_Symbol] :> D
ist[(2*c*d - b*e)/(2*c), Int[1/(a + b*x + c*x^2), x], x] + Dist[e/(2*c), In
t[(b + 2*c*x)/(a + b*x + c*x^2), x], x] /; FreeQ[{a, b, c, d, e}, x] && NeQ
[2*c*d - b*e, 0] && NeQ[b^2 - 4*a*c, 0] &&  !NiceSqrtQ[b^2 - 4*a*c]
```

Rule 995

```
Int[1/(((a_) + (b_.)*(x_) + (c_.)*(x_)^2)*((d_) + (f_.)*(x_)^2)), x_Symbol]
 :> With[{q = c^2*d^2 + b^2*d*f - 2*a*c*d*f + a^2*f^2}, Dist[1/q, Int[(c^2*
d + b^2*f - a*c*f + b*c*f*x)/(a + b*x + c*x^2), x], x] - Dist[1/q, Int[(c*d
*f - a*f^2 + b*f^2*x)/(d + f*x^2), x], x] /; NeQ[q, 0]] /; FreeQ[{a, b, c,
d, f}, x] && NeQ[b^2 - 4*a*c, 0]
```

Rule 3340

```
Int[((a_.) + (b_.)*(cos[(d_.) + (e_.)*(x_)]*(f_.))^(n_.) + (c_.)*(cos[(d_.)
 + (e_.)*(x_)]*(f_.))^(n2_.))^(p_.)*sin[(d_.) + (e_.)*(x_)]^(m_.), x_Symbol
] :> Module[{g = FreeFactors[Cos[d + e*x], x]}, Dist[-g/e, Subst[Int[(1 - g
^2*x^2)^((m - 1)/2)*(a + b*(f*g*x)^n + c*(f*g*x)^(2*n))^p, x], x, Cos[d + e
*x]/g], x]] /; FreeQ[{a, b, c, d, e, f, n, p}, x] && EqQ[n2, 2*n] && Intege
rQ[(m - 1)/2]
```

Rubi steps

$$\text{integral} = -\text{Subst}\left(\int \frac{1}{(1-x^2)(a+bx+cx^2)}\,dx, x, \cos(x)\right)$$

$$= \frac{\text{Subst}\left(\int \frac{-a-c+bx}{1-x^2}\,dx, x, \cos(x)\right)}{(a-b+c)(a+b+c)} - \frac{\text{Subst}\left(\int \frac{-b^2+ac+c^2-bcx}{a+bx+cx^2}\,dx, x, \cos(x)\right)}{(a-b+c)(a+b+c)}$$

$$= \frac{\text{Subst}\left(\int \frac{1}{-1-x}\,dx, x, \cos(x)\right)}{2(a-b+c)} - \frac{\text{Subst}\left(\int \frac{1}{1-x}\,dx, x, \cos(x)\right)}{2(a+b+c)} + \frac{b\text{Subst}\left(\int \frac{b+2cx}{a+bx+cx^2}\,dx, x, \cos(x)\right)}{2(a-b+c)(a+b+c)} + \frac{(b^2-2c(a+c))\,\text{Subst}\left(\int \frac{1}{a+bx+cx^2}\,dx, x, \cos(x)\right)}{2(a-b+c)(a+b+c)}$$

$$= \frac{\log(1-\cos(x))}{2(a+b+c)} - \frac{\log(1+\cos(x))}{2(a-b+c)} + \frac{b\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2(a-b+c)(a+b+c)} - \frac{(b^2-2c(a+c))\,\text{Subst}\left(\int \frac{1}{b^2-4ac-x^2}\,dx, x, b+2c\cos(x)\right)}{(a-b+c)(a+b+c)}$$

$$= -\frac{(b^2-2c(a+c))\,\text{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{(a-b+c)(a+b+c)\sqrt{b^2-4ac}} + \frac{\log(1-\cos(x))}{2(a+b+c)} - \frac{\log(1+\cos(x))}{2(a-b+c)} + \frac{b\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2(a-b+c)(a+b+c)}$$

Mathematica [A] (verified)

Time = 0.15 (sec) , antiderivative size = 126, normalized size of antiderivative = 0.98

$$\int \frac{\csc(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{(-2b^2+4c(a+c))\arctan\left(\frac{b+2c\cos(x)}{\sqrt{-b^2+4ac}}\right)+\sqrt{-b^2+4ac}(-((a-b+c)\log(1-\cos(x)))+(a+b+c)\log(}{2(a-b+c)(a+b+c)\sqrt{-b^2+4ac}}$$

```
[In] Integrate[Csc[x]/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] -1/2*((-2*b^2 + 4*c*(a + c))*ArcTan[(b + 2*c*Cos[x])/Sqrt[-b^2 + 4*a*c]] +
Sqrt[-b^2 + 4*a*c]*(-((a - b + c)*Log[1 - Cos[x]]) + (a + b + c)*Log[1 + Co
s[x]] - b*Log[a + b*Cos[x] + c*Cos[x]^2]))/((a - b + c)*(a + b + c)*Sqrt[-b
^2 + 4*a*c])
```

Maple [A] (verified)

Time = 0.87 (sec) , antiderivative size = 121, normalized size of antiderivative = 0.94

method	result	size
default	$\frac{\frac{b \ln\left(a+\cos(x) b+c\left(\cos^2(x)\right)\right)}{2}+\frac{2\left(-ac+\frac{1}{2}b^2-c^2\right)\arctan\left(\frac{b+2c\cos(x)}{\sqrt{4ac-b^2}}\right)}{\sqrt{4ac-b^2}}}{(a-b+c)(a+b+c)}-\frac{\ln(1+\cos(x))}{2a-2b+2c}+\frac{\ln(\cos(x)-1)}{2a+2b+2c}$	121
risch	Expression too large to display	1394

```
[In] int(csc(x)/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] 1/(a-b+c)/(a+b+c)*(1/2*b*ln(a+cos(x)*b+c*cos(x)^2)+2*(-a*c+1/2*b^2-c^2)/(4*
a*c-b^2)^(1/2)*arctan((b+2*c*cos(x))/(4*a*c-b^2)^(1/2)))-1/(2*a-2*b+2*c)*ln
(1+cos(x))+1/(2*a+2*b+2*c)*ln(cos(x)-1)
```

Fricas [A] (verification not implemented)

none

Time = 0.68 (sec) , antiderivative size = 470, normalized size of antiderivative = 3.64

$$\int \frac{\csc(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$$
$$= \Bigg[-\frac{(b^2-2\,ac-2\,c^2)\sqrt{b^2-4\,ac}\log\left(-\frac{2\,c^2\cos(x)^2+2\,bc\cos(x)+b^2-2\,ac+\sqrt{b^2-4\,ac}(2\,c\cos(x)+b)}{c\cos(x)^2+b\cos(x)+a}\right)-(b^3-4\,abc)\log\bigl(c\cos \cdots}{2\,(a^2b^2 \cdots}$$
$$-\frac{2\,(b^2-2\,ac-2\,c^2)\sqrt{-b^2+4\,ac}\arctan\left(-\frac{\sqrt{-b^2+4\,ac}(2\,c\cos(x)+b)}{b^2-4\,ac}\right)-(b^3-4\,abc)\log\bigl(c\cos(x)^2+b\cos(x)+ \cdots}{2\,(a^2b^2-b^4-4 \cdots}$$

```
[In] integrate(csc(x)/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
```

```
[Out] [-1/2*((b^2 - 2*a*c - 2*c^2)*sqrt(b^2 - 4*a*c)*log(-(2*c^2*cos(x)^2 + 2*b*c
*cos(x) + b^2 - 2*a*c + sqrt(b^2 - 4*a*c)*(2*c*cos(x) + b))/(c*cos(x)^2 + b
*cos(x) + a)) - (b^3 - 4*a*b*c)*log(c*cos(x)^2 + b*cos(x) + a) + (a*b^2 + b
^3 - 4*a*c^2 - (4*a^2 + 4*a*b - b^2)*c)*log(1/2*cos(x) + 1/2) - (a*b^2 - b^
3 - 4*a*c^2 - (4*a^2 - 4*a*b - b^2)*c)*log(-1/2*cos(x) + 1/2))/(a^2*b^2 - b
^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c), -1/2*(2*(b^2 - 2
```

```
*a*c - 2*c^2)*sqrt(-b^2 + 4*a*c)*arctan(-sqrt(-b^2 + 4*a*c)*(2*c*cos(x) + b
)/(b^2 - 4*a*c)) - (b^3 - 4*a*b*c)*log(c*cos(x)^2 + b*cos(x) + a) + (a*b^2
+ b^3 - 4*a*c^2 - (4*a^2 + 4*a*b - b^2)*c)*log(1/2*cos(x) + 1/2) - (a*b^2 -
 b^3 - 4*a*c^2 - (4*a^2 - 4*a*b - b^2)*c)*log(-1/2*cos(x) + 1/2))/(a^2*b^2
- b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)]
```

Sympy [F]

$$\int \frac{\csc(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \int \frac{\csc(x)}{a + b\cos(x) + c\cos^2(x)}\,dx$$

```
[In] integrate(csc(x)/(a+b*cos(x)+c*cos(x)**2),x)
[Out] Integral(csc(x)/(a + b*cos(x) + c*cos(x)**2), x)
```

Maxima [F(-2)]

Exception generated.

$$\int \frac{\csc(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Exception raised: ValueError}$$

```
[In] integrate(csc(x)/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
[Out] Exception raised: ValueError >> Computation failed since Maxima requested a
dditional constraints; using the 'assume' command before evaluation *may* h
elp (example of legal syntax is 'assume(4*a*c-b^2>0)', see `assume?` for mo
re deta
```

Giac [A] (verification not implemented)

none

Time = 0.28 (sec) , antiderivative size = 130, normalized size of antiderivative = 1.01

$$\int \frac{\csc(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \frac{b\log\left(c\cos(x)^2 + b\cos(x) + a\right)}{2\,(a^2 - b^2 + 2\,ac + c^2)} + \frac{(b^2 - 2\,ac - 2\,c^2)\arctan\left(\frac{2\,c\cos(x) + b}{\sqrt{-b^2 + 4\,ac}}\right)}{(a^2 - b^2 + 2\,ac + c^2)\sqrt{-b^2 + 4\,ac}} - \frac{\log(\cos(x) + 1)}{2\,(a - b + c)} + \frac{\log(-\cos(x) + 1)}{2\,(a + b + c)}$$

```
[In] integrate(csc(x)/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] 1/2*b*log(c*cos(x)^2 + b*cos(x) + a)/(a^2 - b^2 + 2*a*c + c^2) + (b^2 - 2*a
*c - 2*c^2)*arctan((2*c*cos(x) + b)/sqrt(-b^2 + 4*a*c))/((a^2 - b^2 + 2*a*c
 + c^2)*sqrt(-b^2 + 4*a*c)) - 1/2*log(cos(x) + 1)/(a - b + c) + 1/2*log(-co
s(x) + 1)/(a + b + c)
```

Mupad [B] (verification not implemented)

Time = 5.48 (sec) , antiderivative size = 1003, normalized size of antiderivative = 7.78

$$\int \frac{\csc(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{\ln\left(\cos\left(x\right)-1\right)}{2\,\left(a+b+c\right)} - \frac{\ln\left(\cos\left(x\right)+1\right)}{2\,\left(a-b+c\right)} - \frac{\ln\left(b\,c^2+4\,c^3\,\cos\left(x\right)+\frac{\left(a\,\left(4\,b\,c-2\,c\,\sqrt{b^2-4\,a\,c}\right)-b^3+b^2\,\sqrt{b^2-4\,a\,c}-2\,c^2\,\sqrt{b^2-4\,a\,c}\right)\,\left(8\,a\,c^3+\cos\left(x\right)\,\left(-3\,b^3\,c+12\,b\,c^3+12\,a\,b\,c^2\right)+4\,c^4 \cdots\right.}{\cdots}\right.}{\cdots} - \frac{\ln\left(b\,c^2+4\,c^3\,\cos\left(x\right)+\frac{\left(a\,\left(4\,b\,c+2\,c\,\sqrt{b^2-4\,a\,c}\right)-b^3-b^2\,\sqrt{b^2-4\,a\,c}+2\,c^2\,\sqrt{b^2-4\,a\,c}\right)\,\left(8\,a\,c^3+\cos\left(x\right)\,\left(-3\,b^3\,c+12\,b\,c^3+12\,a\,b\,c^2\right)+4\,c^4 \cdots\right.}{\cdots}\right.}{\cdots}$$

```
[In] int(1/(sin(x)*(a + b*cos(x) + c*cos(x)^2)),x)
```

```
[Out] log(cos(x) - 1)/(2*(a + b + c)) - log(cos(x) + 1)/(2*(a - b + c)) - (log(b*
c^2 + 4*c^3*cos(x) + ((a*(4*b*c - 2*c*(b^2 - 4*a*c)^(1/2)) - b^3 + b^2*(b^2
 - 4*a*c)^(1/2) - 2*c^2*(b^2 - 4*a*c)^(1/2))*(8*a*c^3 + cos(x)*(12*b*c^3 -
3*b^3*c + 12*a*b*c^2) + 4*c^4 + 4*a^2*c^2 + 3*b^2*c^2 - ((a*(4*b*c - 2*c*(b
^2 - 4*a*c)^(1/2)) - b^3 + b^2*(b^2 - 4*a*c)^(1/2) - 2*c^2*(b^2 - 4*a*c)^(1
/2))*(4*b*c^4 + 4*b^3*c^2 + cos(x)*(8*a*c^4 + 6*b^4*c + 8*c^5 - 8*a^2*c^3 -
 8*a^3*c^2 - 6*b^2*c^3 - 20*a*b^2*c^2 + 2*a^2*b^2*c) - 28*a^2*b*c^2 - 24*a*
b*c^3 + 8*a*b^3*c))/(b^2*(12*a*c + 2*a^2 - 2*b^2 + 2*c^2) - 4*a*c*(4*a*c +
2*a^2 + 2*c^2)) - a*b^2*c))/(b^2*(12*a*c + 2*a^2 - 2*b^2 + 2*c^2) - 4*a*c*(
4*a*c + 2*a^2 + 2*c^2)))*(a*(4*b*c - 2*c*(b^2 - 4*a*c)^(1/2)) - b^3 + b^2*(
b^2 - 4*a*c)^(1/2) - 2*c^2*(b^2 - 4*a*c)^(1/2)))/(b^2*(12*a*c + 2*a^2 - 2*b
^2 + 2*c^2) - 4*a*c*(4*a*c + 2*a^2 + 2*c^2)) - (log(b*c^2 + 4*c^3*cos(x) +
((a*(4*b*c + 2*c*(b^2 - 4*a*c)^(1/2)) - b^3 - b^2*(b^2 - 4*a*c)^(1/2) + 2*c
^2*(b^2 - 4*a*c)^(1/2))*(8*a*c^3 + cos(x)*(12*b*c^3 - 3*b^3*c + 12*a*b*c^2)
 + 4*c^4 + 4*a^2*c^2 + 3*b^2*c^2 - ((a*(4*b*c + 2*c*(b^2 - 4*a*c)^(1/2)) -
b^3 - b^2*(b^2 - 4*a*c)^(1/2) + 2*c^2*(b^2 - 4*a*c)^(1/2))*(4*b*c^4 + 4*b^3
*c^2 + cos(x)*(8*a*c^4 + 6*b^4*c + 8*c^5 - 8*a^2*c^3 - 8*a^3*c^2 - 6*b^2*c^
3 - 20*a*b^2*c^2 + 2*a^2*b^2*c) - 28*a^2*b*c^2 - 24*a*b*c^3 + 8*a*b^3*c))/(
b^2*(12*a*c + 2*a^2 - 2*b^2 + 2*c^2) - 4*a*c*(4*a*c + 2*a^2 + 2*c^2)) - a*b
^2*c))/(b^2*(12*a*c + 2*a^2 - 2*b^2 + 2*c^2) - 4*a*c*(4*a*c + 2*a^2 + 2*c^2
)))*(a*(4*b*c + 2*c*(b^2 - 4*a*c)^(1/2)) - b^3 - b^2*(b^2 - 4*a*c)^(1/2) +
```

```
2*c^2*(b^2 - 4*a*c)^(1/2)))/(b^2*(12*a*c + 2*a^2 - 2*b^2 + 2*c^2) - 4*a*c*(
4*a*c + 2*a^2 + 2*c^2))
```

3.5 $\int \frac{\csc^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 19, antiderivative size = 205

$$\int \frac{\csc^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{(b^4+2c^2(a+c)^2-2b^2c(2a+c))\operatorname{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{\sqrt{b^2-4ac}\,(a^2-b^2+2ac+c^2)^2} + \frac{(b-(a+c)\cos(x))\csc^2(x)}{2(a-b+c)(a+b+c)} + \frac{(a+2b+3c)\log(1-\cos(x))}{4(a+b+c)^2} - \frac{(a-2b+3c)\log(1+\cos(x))}{4(a-b+c)^2} - \frac{b(b^2-2c(a+c))\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2\,(a^2-b^2+2ac+c^2)^2}$$

```
[Out] 1/2*(b-(a+c)*cos(x))*csc(x)^2/(a-b+c)/(a+b+c)+1/4*(a+2*b+3*c)*ln(1-cos(x))/
(a+b+c)^2-1/4*(a-2*b+3*c)*ln(1+cos(x))/(a-b+c)^2-1/2*b*(b^2-2*c*(a+c))*ln(a
+b*cos(x)+c*cos(x)^2)/(a^2+2*a*c-b^2+c^2)^2+(b^4+2*c^2*(a+c)^2-2*b^2*c*(2*a
+c))*arctanh((b+2*c*cos(x))/(-4*a*c+b^2)^(1/2))/(a^2+2*a*c-b^2+c^2)^2/(-4*a
*c+b^2)^(1/2)
```

Rubi [A] (verified)

Time = 0.57 (sec) , antiderivative size = 205, normalized size of antiderivative = 1.00, number of steps used = 10, number of rules used = 9, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.474, Rules used = {3340, 990, 1088, 648, 632, 212, 642, 647, 31}

$$\int \frac{\csc^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{\left(-2b^2c(2a+c)+2c^2(a+c)^2+b^4\right)\text{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{\sqrt{b^2-4ac}\,\left(a^2+2ac-b^2+c^2\right)^2} - \frac{b(b^2-2c(a+c))\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2\left(a^2+2ac-b^2+c^2\right)^2} + \frac{(a+2b+3c)\log(1-\cos(x))}{4(a+b+c)^2} - \frac{(a-2b+3c)\log(\cos(x)+1)}{4(a-b+c)^2} + \frac{\csc^2(x)(b-(a+c)\cos(x))}{2(a-b+c)(a+b+c)}$$

```
[In] Int[Csc[x]^3/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] ((b^4 + 2*c^2*(a + c)^2 - 2*b^2*c*(2*a + c))*ArcTanh[(b + 2*c*Cos[x])/Sqrt[
b^2 - 4*a*c]])/(Sqrt[b^2 - 4*a*c]*(a^2 - b^2 + 2*a*c + c^2)^2) + ((b - (a +
 c)*Cos[x])*Csc[x]^2)/(2*(a - b + c)*(a + b + c)) + ((a + 2*b + 3*c)*Log[1
- Cos[x]])/(4*(a + b + c)^2) - ((a - 2*b + 3*c)*Log[1 + Cos[x]])/(4*(a - b
+ c)^2) - (b*(b^2 - 2*c*(a + c))*Log[a + b*Cos[x] + c*Cos[x]^2])/(2*(a^2 -
b^2 + 2*a*c + c^2)^2)
```

Rule 31

```
Int[((a_) + (b_.)*(x_))^(-1), x_Symbol] :> Simp[Log[RemoveContent[a + b*x,
x]]/b, x] /; FreeQ[{a, b}, x]
```

Rule 212

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(1/(Rt[a, 2]*Rt[-b, 2]))*
ArcTanh[Rt[-b, 2]*(x/Rt[a, 2])], x] /; FreeQ[{a, b}, x] && NegQ[a/b] && (Gt
Q[a, 0] || LtQ[b, 0])
```

Rule 632

```
Int[((a_.) + (b_.)*(x_) + (c_.)*(x_)^2)^(-1), x_Symbol] :> Dist[-2, Subst[I
nt[1/Simp[b^2 - 4*a*c - x^2, x], x], x, b + 2*c*x], x] /; FreeQ[{a, b, c},
x] && NeQ[b^2 - 4*a*c, 0]
```

Rule 642

```
Int[((d_) + (e_.)*(x_))/((a_.) + (b_.)*(x_) + (c_.)*(x_)^2), x_Symbol] :> S
imp[d*(Log[RemoveContent[a + b*x + c*x^2, x]]/b), x] /; FreeQ[{a, b, c, d,
e}, x] && EqQ[2*c*d - b*e, 0]
```

Rule 647

```
Int[((d_) + (e_.)*(x_))/((a_) + (c_.)*(x_)^2), x_Symbol] :> With[{q = Rt[(-
a)*c, 2]}, Dist[e/2 + c*(d/(2*q)), Int[1/(-q + c*x), x], x] + Dist[e/2 - c*
(d/(2*q)), Int[1/(q + c*x), x], x]] /; FreeQ[{a, c, d, e}, x] && NiceSqrtQ[
(-a)*c]
```

Rule 648

```
Int[((d_.) + (e_.)*(x_))/((a_) + (b_.)*(x_) + (c_.)*(x_)^2), x_Symbol] :> D
ist[(2*c*d - b*e)/(2*c), Int[1/(a + b*x + c*x^2), x], x] + Dist[e/(2*c), In
t[(b + 2*c*x)/(a + b*x + c*x^2), x], x] /; FreeQ[{a, b, c, d, e}, x] && NeQ
[2*c*d - b*e, 0] && NeQ[b^2 - 4*a*c, 0] &&  !NiceSqrtQ[b^2 - 4*a*c]
```

Rule 990

```
Int[((a_.) + (c_.)*(x_)^2)^(p_)*((d_.) + (e_.)*(x_) + (f_.)*(x_)^2)^(q_), x
_Symbol] :> Simp[(2*a*c^2*e + c*(2*c^2*d - c*(2*a*f))*x)*(a + c*x^2)^(p + 1
)*((d + e*x + f*x^2)^(q + 1)/((-4*a*c)*(a*c*e^2 + (c*d - a*f)^2)*(p + 1))),
 x] - Dist[1/((-4*a*c)*(a*c*e^2 + (c*d - a*f)^2)*(p + 1)), Int[(a + c*x^2)^
(p + 1)*(d + e*x + f*x^2)^q*Simp[2*c*((c*d - a*f)^2 - ((-a)*e)*(c*e))*(p +
1) - (2*c^2*d - c*(2*a*f))*(a*f*(p + 1) - c*d*(p + 2)) - e*(-2*a*c^2*e)*(p
+ q + 2) + (2*f*(2*a*c^2*e)*(p + q + 2) - (2*c^2*d - c*(2*a*f))*((-c)*e*(2*
p + q + 4)))*x + c*f*(2*c^2*d - c*(2*a*f))*(2*p + 2*q + 5)*x^2, x], x], x]
/; FreeQ[{a, c, d, e, f, q}, x] && NeQ[e^2 - 4*d*f, 0] && LtQ[p, -1] && NeQ
[a*c*e^2 + (c*d - a*f)^2, 0] &&  !( !IntegerQ[p] && ILtQ[q, -1]) &&  !IGtQ[
q, 0]
```

Rule 1088

```
Int[((A_.) + (B_.)*(x_) + (C_.)*(x_)^2)/(((a_) + (b_.)*(x_) + (c_.)*(x_)^2)
*((d_) + (f_.)*(x_)^2)), x_Symbol] :> With[{q = c^2*d^2 + b^2*d*f - 2*a*c*d
*f + a^2*f^2}, Dist[1/q, Int[(A*c^2*d - a*c*C*d + A*b^2*f - a*b*B*f - a*A*c
*f + a^2*C*f + c*(B*c*d - b*C*d + A*b*f - a*B*f)*x)/(a + b*x + c*x^2), x],
x] + Dist[1/q, Int[(c*C*d^2 + b*B*d*f - A*c*d*f - a*C*d*f + a*A*f^2 - f*(B*
c*d - b*C*d + A*b*f - a*B*f)*x)/(d + f*x^2), x], x] /; NeQ[q, 0]] /; FreeQ[
{a, b, c, d, f, A, B, C}, x] && NeQ[b^2 - 4*a*c, 0]
```

Rule 3340

```
Int[((a_.) + (b_.)*(cos[(d_.) + (e_.)*(x_)]*(f_.))^(n_.) + (c_.)*(cos[(d_.)
 + (e_.)*(x_)]*(f_.))^(n2_.))^(p_.)*sin[(d_.) + (e_.)*(x_)]^(m_.), x_Symbol
```

```
] :> Module[{g = FreeFactors[Cos[d + e*x], x]}, Dist[-g/e, Subst[Int[(1 - g
^2*x^2)^((m - 1)/2)*(a + b*(f*g*x)^n + c*(f*g*x)^(2*n))^p, x], x, Cos[d + e
*x]/g], x]] /; FreeQ[{a, b, c, d, e, f, n, p}, x] && EqQ[n2, 2*n] && Intege
rQ[(m - 1)/2]
```

Rubi steps

$$\text{integral} = -\text{Subst}\left(\int \frac{1}{(1-x^2)^2\,(a+bx+cx^2)}\,dx, x, \cos(x)\right)$$

$$= \frac{(b-(a+c)\cos(x))\csc^2(x)}{2(a-b+c)(a+b+c)} - \frac{\text{Subst}\left(\int \frac{2(a^2-2b^2+3ac+2c^2)+2b(a-c)x+2c(a+c)x^2}{(1-x^2)(a+bx+cx^2)}\,dx, x, \cos(x)\right)}{4(a-b+c)(a+b+c)}$$

$$= \frac{(b-(a+c)\cos(x))\csc^2(x)}{2(a-b+c)(a+b+c)} - \frac{\text{Subst}\left(\int \frac{-2b^2(a-c)+2ac(a+c)+2c^2(a+c)+2a(a^2-2b^2+3ac+2c^2)+2c(a^2-2b^2+3ac+2c^2)+(2ab(a-c)+2b(a-c)c-2bc(a+c)-}{1-x^2}\right.}{4(a-b+c)^2(a+b+c)^2} - \frac{\text{Subst}\left(\int \frac{2ab^2(a-c)-2a^2c(a+c)-2ac^2(a+c)-2b^2(a^2-2b^2+3ac+2c^2)+2ac(a^2-2b^2+3ac+2c^2)+2c^2(a^2-2b^2+3ac+2c^2)+c(2a}{a+bx+cx^2}\right.}{4(a-b+c)^2(a+b+c)^2}$$

$$= \frac{(b-(a+c)\cos(x))\csc^2(x)}{2(a-b+c)(a+b+c)} + \frac{(a-2b+3c)\text{Subst}\left(\int \frac{1}{-1-x}\,dx, x, \cos(x)\right)}{4(a-b+c)^2} - \frac{(a+2b+3c)\text{Subst}\left(\int \frac{1}{1-x}\,dx, x, \cos(x)\right)}{4(a+b+c)^2} - \frac{(b(b^2-2c(a+c)))\,\text{Subst}\left(\int \frac{b+2cx}{a+bx+cx^2}\,dx, x, \cos(x)\right)}{2(a-b+c)^2(a+b+c)^2} - \frac{(b^4+2c^2(a+c)^2-2b^2c(2a+c))\,\text{Subst}\left(\int \frac{1}{a+bx+cx^2}\,dx, x, \cos(x)\right)}{2(a-b+c)^2(a+b+c)^2}$$

$$= \frac{(b-(a+c)\cos(x))\csc^2(x)}{2(a-b+c)(a+b+c)} + \frac{(a+2b+3c)\log(1-\cos(x))}{4(a+b+c)^2} - \frac{(a-2b+3c)\log(1+\cos(x))}{4(a-b+c)^2} - \frac{b(b^2-2c(a+c))\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2(a-b+c)^2(a+b+c)^2} + \frac{(b^4+2c^2(a+c)^2-2b^2c(2a+c))\,\text{Subst}\left(\int \frac{1}{b^2-4ac-x^2}\,dx, x, b+2c\cos(x)\right)}{(a-b+c)^2(a+b+c)^2}$$

$$= \frac{(b^4+2c^2(a+c)^2-2b^2c(2a+c))\,\text{arctanh}\left(\frac{b+2c\cos(x)}{\sqrt{b^2-4ac}}\right)}{(a-b+c)^2(a+b+c)^2\sqrt{b^2-4ac}} + \frac{(b-(a+c)\cos(x))\csc^2(x)}{2(a-b+c)(a+b+c)} + \frac{(a+2b+3c)\log(1-\cos(x))}{4(a+b+c)^2} - \frac{(a-2b+3c)\log(1+\cos(x))}{4(a-b+c)^2} - \frac{b(b^2-2c(a+c))\log\left(a+b\cos(x)+c\cos^2(x)\right)}{2(a-b+c)^2(a+b+c)^2}$$

Mathematica [C] (verified)

Result contains complex when optimal does not.

Time = 1.76 (sec) , antiderivative size = 392, normalized size of antiderivative = 1.91

$$\int \frac{\csc^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$$
$$=\frac{1}{8}\left(\frac{16i(b^3-2bc(a+c))\,x}{(a-b+c)^2(a+b+c)^2}+\frac{4i(a-2b+3c)\arctan(\tan(x))}{(a-b+c)^2}\right.$$
$$-\frac{4i(a+2b+3c)\arctan(\tan(x))}{(a+b+c)^2}-\frac{\csc^2\left(\frac{x}{2}\right)}{a+b+c}-\frac{2(a-2b+3c)\log\left(\cos^2\left(\frac{x}{2}\right)\right)}{(a-b+c)^2}$$
$$-\frac{4\left(b^4+2c^2(a+c)^2-2b^2c(2a+c)+b^3\sqrt{b^2-4ac}-2bc(a+c)\sqrt{b^2-4ac}\right)\log\left(-b+\sqrt{b^2-4ac}-2c\cos(\right.}{\sqrt{b^2-4ac}\,(a^2-b^2+2ac+c^2)^2}$$
$$-\frac{4\left(-b^4-2c^2(a+c)^2+2b^2c(2a+c)+b^3\sqrt{b^2-4ac}-2bc(a+c)\sqrt{b^2-4ac}\right)\log\left(b+\sqrt{b^2-4ac}+2c\cos(\right.}{\sqrt{b^2-4ac}\,(a^2-b^2+2ac+c^2)^2}$$
$$\left.+\frac{2(a+2b+3c)\log\left(\sin^2\left(\frac{x}{2}\right)\right)}{(a+b+c)^2}+\frac{\sec^2\left(\frac{x}{2}\right)}{a-b+c}\right)$$

```
[In] Integrate[Csc[x]^3/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (((16*I)*(b^3 - 2*b*c*(a + c))*x)/((a - b + c)^2*(a + b + c)^2) + ((4*I)*(a
 - 2*b + 3*c)*ArcTan[Tan[x]])/(a - b + c)^2 - ((4*I)*(a + 2*b + 3*c)*ArcTan
[Tan[x]])/(a + b + c)^2 - Csc[x/2]^2/(a + b + c) - (2*(a - 2*b + 3*c)*Log[C
os[x/2]^2])/(a - b + c)^2 - (4*(b^4 + 2*c^2*(a + c)^2 - 2*b^2*c*(2*a + c) +
 b^3*Sqrt[b^2 - 4*a*c] - 2*b*c*(a + c)*Sqrt[b^2 - 4*a*c])*Log[-b + Sqrt[b^2
 - 4*a*c] - 2*c*Cos[x]])/(Sqrt[b^2 - 4*a*c]*(a^2 - b^2 + 2*a*c + c^2)^2) -
(4*(-b^4 - 2*c^2*(a + c)^2 + 2*b^2*c*(2*a + c) + b^3*Sqrt[b^2 - 4*a*c] - 2*
b*c*(a + c)*Sqrt[b^2 - 4*a*c])*Log[b + Sqrt[b^2 - 4*a*c] + 2*c*Cos[x]])/(Sq
rt[b^2 - 4*a*c]*(a^2 - b^2 + 2*a*c + c^2)^2) + (2*(a + 2*b + 3*c)*Log[Sin[x
/2]^2])/(a + b + c)^2 + Sec[x/2]^2/(a - b + c))/8
```

Maple [A] (verified)

Time = 3.21 (sec) , antiderivative size = 241, normalized size of antiderivative = 1.18

method	result
default	$\frac{\frac{\left(2ab\,c^2-b^3c+2b\,c^3\right)\ln\left(a+\cos(x)b+c\left(\cos^2(x)\right)\right)}{2c}+\frac{2\left(-a^2c^2+3a\,b^2c-2a\,c^3-b^4+2b^2c^2-c^4-\frac{\left(2ab\,c^2-b^3c+2b\,c^3\right)b}{2c}\right)\arctan\left(\frac{b+2c\cos(x)}{\sqrt{4ac-b^2}}\right)}{\sqrt{4ac-b^2}}}{(a-b+c)^2(a+b+c)^2}+\frac{}{(4a-}$
risch	Expression too large to display

```
[In] int(csc(x)^3/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] 1/(a-b+c)^2/(a+b+c)^2*(1/2*(2*a*b*c^2-b^3*c+2*b*c^3)/c*ln(a+cos(x)*b+c*cos(
x)^2)+2*(-a^2*c^2+3*a*b^2*c-2*a*c^3-b^4+2*b^2*c^2-c^4-1/2*(2*a*b*c^2-b^3*c+
2*b*c^3)*b/c)/(4*a*c-b^2)^(1/2)*arctan((b+2*c*cos(x))/(4*a*c-b^2)^(1/2)))+1
/(4*a-4*b+4*c)/(1+cos(x))+1/4/(a-b+c)^2*(-a+2*b-3*c)*ln(1+cos(x))+1/(4*a+4*
b+4*c)/(cos(x)-1)+1/4*(a+2*b+3*c)/(a+b+c)^2*ln(cos(x)-1)
```

Fricas [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 983 vs. 2(194) = 388.

Time = 5.43 (sec) , antiderivative size = 1991, normalized size of antiderivative = 9.71

$$\int \frac{\csc^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(csc(x)^3/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
```

```
[Out] [1/4*(2*a^2*b^3 - 2*b^5 - 8*a*b*c^3 - 2*(8*a^2*b - b^3)*c^2 + 2*(b^4 - 4*a*
b^2*c + 4*a*c^3 + 2*c^4 + 2*(a^2 - b^2)*c^2 - (b^4 - 4*a*b^2*c + 4*a*c^3 +
2*c^4 + 2*(a^2 - b^2)*c^2)*cos(x)^2)*sqrt(b^2 - 4*a*c)*log(-(2*c^2*cos(x)^2
 + 2*b*c*cos(x) + b^2 - 2*a*c + sqrt(b^2 - 4*a*c)*(2*c*cos(x) + b))/(c*cos(
x)^2 + b*cos(x) + a)) - 4*(2*a^3*b - 3*a*b^3)*c - 2*(a^3*b^2 - a*b^4 - 4*a*
c^4 - (12*a^2 - b^2)*c^3 - (12*a^3 - 7*a*b^2)*c^2 - (4*a^4 - 7*a^2*b^2 + b^
4)*c)*cos(x) - 2*(b^5 - 6*a*b^3*c + 8*a*b*c^3 + 2*(4*a^2*b - b^3)*c^2 - (b^
5 - 6*a*b^3*c + 8*a*b*c^3 + 2*(4*a^2*b - b^3)*c^2)*cos(x)^2)*log(c*cos(x)^2
 + b*cos(x) + a) - (a^3*b^2 - 3*a*b^4 - 2*b^5 - 12*a*c^4 - (28*a^2 + 16*a*b
 - 3*b^2)*c^3 - (20*a^3 + 16*a^2*b - 11*a*b^2 - 4*b^3)*c^2 - (a^3*b^2 - 3*a
*b^4 - 2*b^5 - 12*a*c^4 - (28*a^2 + 16*a*b - 3*b^2)*c^3 - (20*a^3 + 16*a^2*
b - 11*a*b^2 - 4*b^3)*c^2 - (4*a^4 - 17*a^2*b^2 - 12*a*b^3 + b^4)*c)*cos(x)
^2 - (4*a^4 - 17*a^2*b^2 - 12*a*b^3 + b^4)*c)*log(1/2*cos(x) + 1/2) + (a^3*
b^2 - 3*a*b^4 + 2*b^5 - 12*a*c^4 - (28*a^2 - 16*a*b - 3*b^2)*c^3 - (20*a^3
- 16*a^2*b - 11*a*b^2 + 4*b^3)*c^2 - (a^3*b^2 - 3*a*b^4 + 2*b^5 - 12*a*c^4
- (28*a^2 - 16*a*b - 3*b^2)*c^3 - (20*a^3 - 16*a^2*b - 11*a*b^2 + 4*b^3)*c^
2 - (4*a^4 - 17*a^2*b^2 + 12*a*b^3 + b^4)*c)*cos(x)^2 - (4*a^4 - 17*a^2*b^2
 + 12*a*b^3 + b^4)*c)*log(-1/2*cos(x) + 1/2))/(a^4*b^2 - 2*a^2*b^4 + b^6 -
4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b
^2 + b^4)*c^2 - (a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 -
 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3
*b^2 + 2*a*b^4)*c)*cos(x)^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c), 1/4*(2*a^2*
b^3 - 2*b^5 - 8*a*b*c^3 - 2*(8*a^2*b - b^3)*c^2 + 4*(b^4 - 4*a*b^2*c + 4*a*
c^3 + 2*c^4 + 2*(a^2 - b^2)*c^2 - (b^4 - 4*a*b^2*c + 4*a*c^3 + 2*c^4 + 2*(a
^2 - b^2)*c^2)*cos(x)^2)*sqrt(-b^2 + 4*a*c)*arctan(-sqrt(-b^2 + 4*a*c)*(2*c
*cos(x) + b)/(b^2 - 4*a*c)) - 4*(2*a^3*b - 3*a*b^3)*c - 2*(a^3*b^2 - a*b^4
- 4*a*c^4 - (12*a^2 - b^2)*c^3 - (12*a^3 - 7*a*b^2)*c^2 - (4*a^4 - 7*a^2*b^
```

```
2 + b^4)*c)*cos(x) - 2*(b^5 - 6*a*b^3*c + 8*a*b*c^3 + 2*(4*a^2*b - b^3)*c^2
 - (b^5 - 6*a*b^3*c + 8*a*b*c^3 + 2*(4*a^2*b - b^3)*c^2)*cos(x)^2)*log(c*co
s(x)^2 + b*cos(x) + a) - (a^3*b^2 - 3*a*b^4 - 2*b^5 - 12*a*c^4 - (28*a^2 +
16*a*b - 3*b^2)*c^3 - (20*a^3 + 16*a^2*b - 11*a*b^2 - 4*b^3)*c^2 - (a^3*b^2
 - 3*a*b^4 - 2*b^5 - 12*a*c^4 - (28*a^2 + 16*a*b - 3*b^2)*c^3 - (20*a^3 + 1
6*a^2*b - 11*a*b^2 - 4*b^3)*c^2 - (4*a^4 - 17*a^2*b^2 - 12*a*b^3 + b^4)*c)*
cos(x)^2 - (4*a^4 - 17*a^2*b^2 - 12*a*b^3 + b^4)*c)*log(1/2*cos(x) + 1/2) +
 (a^3*b^2 - 3*a*b^4 + 2*b^5 - 12*a*c^4 - (28*a^2 - 16*a*b - 3*b^2)*c^3 - (2
0*a^3 - 16*a^2*b - 11*a*b^2 + 4*b^3)*c^2 - (a^3*b^2 - 3*a*b^4 + 2*b^5 - 12*
a*c^4 - (28*a^2 - 16*a*b - 3*b^2)*c^3 - (20*a^3 - 16*a^2*b - 11*a*b^2 + 4*b
^3)*c^2 - (4*a^4 - 17*a^2*b^2 + 12*a*b^3 + b^4)*c)*cos(x)^2 - (4*a^4 - 17*a
^2*b^2 + 12*a*b^3 + b^4)*c)*log(-1/2*cos(x) + 1/2))/(a^4*b^2 - 2*a^2*b^4 +
b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11
*a^2*b^2 + b^4)*c^2 - (a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)
*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 -
 3*a^3*b^2 + 2*a*b^4)*c)*cos(x)^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c)]
```

Sympy [F]

$$\int \frac{\csc^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx=\int \frac{\csc^3{(x)}}{a+b\cos{(x)}+c\cos^2{(x)}}\,dx$$

```
[In] integrate(csc(x)**3/(a+b*cos(x)+c*cos(x)**2),x)
```

```
[Out] Integral(csc(x)**3/(a + b*cos(x) + c*cos(x)**2), x)
```

Maxima [F(-2)]

Exception generated.

$$\int \frac{\csc^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx=\text{Exception raised: ValueError}$$

```
[In] integrate(csc(x)^3/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
```

```
[Out] Exception raised: ValueError >> Computation failed since Maxima requested a
dditional constraints; using the 'assume' command before evaluation *may* h
elp (example of legal syntax is 'assume(4*a*c-b^2>0)', see `assume?` for mo
re deta
```

Giac [A] (verification not implemented)

none

Time = 0.30 (sec) , antiderivative size = 378, normalized size of antiderivative = 1.84

$$\int \frac{\csc^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$$
$$= -\frac{(b^3-2\,abc-2\,bc^2)\log\left(c\cos(x)^2+b\cos(x)+a\right)}{2\,(a^4-2\,a^2b^2+b^4+4\,a^3c-4\,ab^2c+6\,a^2c^2-2\,b^2c^2+4\,ac^3+c^4)} - \frac{(a-2\,b+3\,c)\log\left(\cos(x)+1\right)}{4\,(a^2-2\,ab+b^2+2\,ac-2\,bc+c^2)} + \frac{(a+2\,b+3\,c)\log\left(-\cos(x)+1\right)}{4\,(a^2+2\,ab+b^2+2\,ac+2\,bc+c^2)} - \frac{(b^4-4\,ab^2c+2\,a^2c^2-2\,b^2c^2+4\,ac^3+2\,c^4)\arctan\left(\frac{2\,c\cos(x)+b}{\sqrt{-b^2+4\,ac}}\right)}{(a^4-2\,a^2b^2+b^4+4\,a^3c-4\,ab^2c+6\,a^2c^2-2\,b^2c^2+4\,ac^3+c^4)\sqrt{-b^2+4\,ac}} - \frac{a^2b-b^3+2\,abc+bc^2-(a^3-ab^2+3\,a^2c-b^2c+3\,ac^2+c^3)\cos(x)}{2\,(a+b+c)^2(a-b+c)^2(\cos(x)+1)(\cos(x)-1)}$$

```
[In] integrate(csc(x)^3/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] -1/2*(b^3 - 2*a*b*c - 2*b*c^2)*log(c*cos(x)^2 + b*cos(x) + a)/(a^4 - 2*a^2*
b^2 + b^4 + 4*a^3*c - 4*a*b^2*c + 6*a^2*c^2 - 2*b^2*c^2 + 4*a*c^3 + c^4) -
1/4*(a - 2*b + 3*c)*log(cos(x) + 1)/(a^2 - 2*a*b + b^2 + 2*a*c - 2*b*c + c^
2) + 1/4*(a + 2*b + 3*c)*log(-cos(x) + 1)/(a^2 + 2*a*b + b^2 + 2*a*c + 2*b*
c + c^2) - (b^4 - 4*a*b^2*c + 2*a^2*c^2 - 2*b^2*c^2 + 4*a*c^3 + 2*c^4)*arct
an((2*c*cos(x) + b)/sqrt(-b^2 + 4*a*c))/((a^4 - 2*a^2*b^2 + b^4 + 4*a^3*c -
 4*a*b^2*c + 6*a^2*c^2 - 2*b^2*c^2 + 4*a*c^3 + c^4)*sqrt(-b^2 + 4*a*c)) - 1
/2*(a^2*b - b^3 + 2*a*b*c + b*c^2 - (a^3 - a*b^2 + 3*a^2*c - b^2*c + 3*a*c^
2 + c^3)*cos(x))/((a + b + c)^2*(a - b + c)^2*(cos(x) + 1)*(cos(x) - 1))
```

Mupad [B] (verification not implemented)

Time = 21.56 (sec) , antiderivative size = 2742, normalized size of antiderivative = 13.38

$$\int \frac{\csc^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] int(1/(sin(x)^3*(a + b*cos(x) + c*cos(x)^2)),x)
```

```
[Out] (b/(2*(2*a*c + a^2 - b^2 + c^2)) - (cos(x)*(a + c))/(2*(2*a*c + a^2 - b^2 +
 c^2)))/sin(x)^2 - log(cos(x) + 1)*(1/(4*(a - b + c)) - (b/4 - c/2)/(a - b
+ c)^2) + log(cos(x) - 1)*((b/4 + c/2)/(a + b + c)^2 + 1/(4*(a + b + c))) -
 (log((c^4*(4*a*c + a^2 - 4*b^2 + 3*c^2))/(4*(2*a*c + a^2 - b^2 + c^2)^2) -
 (b*c^5*cos(x))/(2*a*c + a^2 - b^2 + c^2)^2 - (((((c*(a*b^4 + 28*a*c^4 + 4*
a^4*c - 5*b^4*c + 8*c^5 - a^3*b^2 + 36*a^2*c^3 + 20*a^3*c^2 + 5*b^2*c^3 - 3
```

```
*a*b^2*c^2 - 9*a^2*b^2*c))/(2*(2*a*c + a^2 - b^2 + c^2)) - (2*c*((b^4*(b^2
- 4*a*c)^(1/2))/2 - b^5/2 + c^4*(b^2 - 4*a*c)^(1/2) + b^3*c^2 + 2*a*c^3*(b^
2 - 4*a*c)^(1/2) - 4*a^2*b*c^2 + a^2*c^2*(b^2 - 4*a*c)^(1/2) - b^2*c^2*(b^2
 - 4*a*c)^(1/2) - 4*a*b*c^3 + 3*a*b^3*c - 2*a*b^2*c*(b^2 - 4*a*c)^(1/2))*(4
*a*b^3 + 2*b*c^3 + 2*b^3*c + 3*b^4*cos(x) + 4*c^4*cos(x) + 4*a*c^3*cos(x) -
 4*a^3*c*cos(x) + a^2*b^2*cos(x) - 4*a^2*c^2*cos(x) - 3*b^2*c^2*cos(x) - 12
*a*b*c^2 - 14*a^2*b*c - 10*a*b^2*c*cos(x)))/((4*a*c - b^2)*(2*a*c + a^2 - b
^2 + c^2)^2) + (b*c*cos(x)*(36*a*c^3 + 4*a^3*c + 3*b^4 + 16*c^4 - a^2*b^2 +
 24*a^2*c^2 - 13*b^2*c^2 - 18*a*b^2*c))/(2*a*c + a^2 - b^2 + c^2))*((b^4*(b
^2 - 4*a*c)^(1/2))/2 - b^5/2 + c^4*(b^2 - 4*a*c)^(1/2) + b^3*c^2 + 2*a*c^3*
(b^2 - 4*a*c)^(1/2) - 4*a^2*b*c^2 + a^2*c^2*(b^2 - 4*a*c)^(1/2) - b^2*c^2*(
b^2 - 4*a*c)^(1/2) - 4*a*b*c^3 + 3*a*b^3*c - 2*a*b^2*c*(b^2 - 4*a*c)^(1/2))
)/((4*a*c - b^2)*(2*a*c + a^2 - b^2 + c^2)^2) - (b*c*(2*a*b^4 - 20*a*c^4 +
3*a^4*c - 6*b^4*c + 7*c^5 - a^3*b^2 - 26*a^2*c^3 + 4*a^3*c^2 + 23*a*b^2*c^2
 - 6*a^2*b^2*c))/(4*(2*a*c + a^2 - b^2 + c^2)^2) + (c*cos(x)*(64*a*c^5 + 26
*c^6 + a^2*b^4 + 52*a^2*c^4 + 16*a^3*c^3 + 2*a^4*c^2 - 18*b^2*c^4 + 9*b^4*c
^2 - 32*a*b^2*c^3 - 4*a^3*b^2*c - 2*a^2*b^2*c^2 - 2*a*b^4*c))/(4*(2*a*c + a
^2 - b^2 + c^2)^2))*((b^4*(b^2 - 4*a*c)^(1/2))/2 - b^5/2 + c^4*(b^2 - 4*a*c
)^(1/2) + b^3*c^2 + 2*a*c^3*(b^2 - 4*a*c)^(1/2) - 4*a^2*b*c^2 + a^2*c^2*(b^
2 - 4*a*c)^(1/2) - b^2*c^2*(b^2 - 4*a*c)^(1/2) - 4*a*b*c^3 + 3*a*b^3*c - 2*
a*b^2*c*(b^2 - 4*a*c)^(1/2)))/((4*a*c - b^2)*(2*a*c + a^2 - b^2 + c^2)^2))*
(b^3*(3*a*c + c^2) - b^2*(c^2*(b^2 - 4*a*c)^(1/2) + 2*a*c*(b^2 - 4*a*c)^(1/
2)) - b*(4*a*c^3 + 4*a^2*c^2) - b^5/2 + (b^4*(b^2 - 4*a*c)^(1/2))/2 + c^4*(
b^2 - 4*a*c)^(1/2) + 2*a*c^3*(b^2 - 4*a*c)^(1/2) + a^2*c^2*(b^2 - 4*a*c)^(1
/2)))/(4*a*c^5 + 4*a^5*c - b^6 + 2*a^2*b^4 - a^4*b^2 + 16*a^2*c^4 + 24*a^3*
c^3 + 16*a^4*c^2 - b^2*c^4 + 2*b^4*c^2 - 12*a*b^2*c^3 - 12*a^3*b^2*c - 22*a
^2*b^2*c^2 + 8*a*b^4*c) + (log((c^4*(4*a*c + a^2 - 4*b^2 + 3*c^2))/(4*(2*a*
c + a^2 - b^2 + c^2)^2) - (b*c^5*cos(x))/(2*a*c + a^2 - b^2 + c^2)^2 - ((((
(c*(a*b^4 + 28*a*c^4 + 4*a^4*c - 5*b^4*c + 8*c^5 - a^3*b^2 + 36*a^2*c^3 + 2
0*a^3*c^2 + 5*b^2*c^3 - 3*a*b^2*c^2 - 9*a^2*b^2*c))/(2*(2*a*c + a^2 - b^2 +
 c^2)) + (2*c*(b^5/2 + (b^4*(b^2 - 4*a*c)^(1/2))/2 + c^4*(b^2 - 4*a*c)^(1/2
) - b^3*c^2 + 2*a*c^3*(b^2 - 4*a*c)^(1/2) + 4*a^2*b*c^2 + a^2*c^2*(b^2 - 4*
a*c)^(1/2) - b^2*c^2*(b^2 - 4*a*c)^(1/2) + 4*a*b*c^3 - 3*a*b^3*c - 2*a*b^2*
c*(b^2 - 4*a*c)^(1/2))*(4*a*b^3 + 2*b*c^3 + 2*b^3*c + 3*b^4*cos(x) + 4*c^4*
cos(x) + 4*a*c^3*cos(x) - 4*a^3*c*cos(x) + a^2*b^2*cos(x) - 4*a^2*c^2*cos(x
) - 3*b^2*c^2*cos(x) - 12*a*b*c^2 - 14*a^2*b*c - 10*a*b^2*c*cos(x)))/((4*a*
c - b^2)*(2*a*c + a^2 - b^2 + c^2)^2) + (b*c*cos(x)*(36*a*c^3 + 4*a^3*c + 3
*b^4 + 16*c^4 - a^2*b^2 + 24*a^2*c^2 - 13*b^2*c^2 - 18*a*b^2*c))/(2*a*c + a
^2 - b^2 + c^2))*(b^5/2 + (b^4*(b^2 - 4*a*c)^(1/2))/2 + c^4*(b^2 - 4*a*c)^(
1/2) - b^3*c^2 + 2*a*c^3*(b^2 - 4*a*c)^(1/2) + 4*a^2*b*c^2 + a^2*c^2*(b^2 -
 4*a*c)^(1/2) - b^2*c^2*(b^2 - 4*a*c)^(1/2) + 4*a*b*c^3 - 3*a*b^3*c - 2*a*b
^2*c*(b^2 - 4*a*c)^(1/2)))/((4*a*c - b^2)*(2*a*c + a^2 - b^2 + c^2)^2) + (b
*c*(2*a*b^4 - 20*a*c^4 + 3*a^4*c - 6*b^4*c + 7*c^5 - a^3*b^2 - 26*a^2*c^3 +
 4*a^3*c^2 + 23*a*b^2*c^2 - 6*a^2*b^2*c))/(4*(2*a*c + a^2 - b^2 + c^2)^2) -
 (c*cos(x)*(64*a*c^5 + 26*c^6 + a^2*b^4 + 52*a^2*c^4 + 16*a^3*c^3 + 2*a^4*c
```

```
^2 - 18*b^2*c^4 + 9*b^4*c^2 - 32*a*b^2*c^3 - 4*a^3*b^2*c - 2*a^2*b^2*c^2 -
2*a*b^4*c))/(4*(2*a*c + a^2 - b^2 + c^2)^2))*(b^5/2 + (b^4*(b^2 - 4*a*c)^(1
/2))/2 + c^4*(b^2 - 4*a*c)^(1/2) - b^3*c^2 + 2*a*c^3*(b^2 - 4*a*c)^(1/2) +
4*a^2*b*c^2 + a^2*c^2*(b^2 - 4*a*c)^(1/2) - b^2*c^2*(b^2 - 4*a*c)^(1/2) + 4
*a*b*c^3 - 3*a*b^3*c - 2*a*b^2*c*(b^2 - 4*a*c)^(1/2)))/((4*a*c - b^2)*(2*a*
c + a^2 - b^2 + c^2)^2))*(b*(4*a*c^3 + 4*a^2*c^2) - b^3*(3*a*c + c^2) - b^2
*(c^2*(b^2 - 4*a*c)^(1/2) + 2*a*c*(b^2 - 4*a*c)^(1/2)) + b^5/2 + (b^4*(b^2
- 4*a*c)^(1/2))/2 + c^4*(b^2 - 4*a*c)^(1/2) + 2*a*c^3*(b^2 - 4*a*c)^(1/2) +
 a^2*c^2*(b^2 - 4*a*c)^(1/2)))/(4*a*c^5 + 4*a^5*c - b^6 + 2*a^2*b^4 - a^4*b
^2 + 16*a^2*c^4 + 24*a^3*c^3 + 16*a^4*c^2 - b^2*c^4 + 2*b^4*c^2 - 12*a*b^2*
c^3 - 12*a^3*b^2*c - 22*a^2*b^2*c^2 + 8*a*b^4*c)
```

3.6 $\int \frac{\sin^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 19, antiderivative size = 388

$$\int \frac{\sin^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{x}{2c}+\frac{(b^2-c(a+2c))\,x}{c^3} + \frac{2\left(b^2(b^2-2c(a+c))-b\sqrt{b^2-4ac}(b^2-2c(a+c))-2c(ab^2-c(a+c)^2)\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{c^3\sqrt{b^2-4ac}\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} - \frac{2\left(b^4+2c^2(a+c)^2-2b^2c(2a+c)+b^3\sqrt{b^2-4ac}-2bc(a+c)\sqrt{b^2-4ac}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{c^3\sqrt{b^2-4ac}\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}} - \frac{b\sin(x)}{c^2}+\frac{\cos(x)\sin(x)}{2c}$$

```
[Out] 1/2*x/c+(b^2-c*(a+2*c))*x/c^3-b*sin(x)/c^2+1/2*cos(x)*sin(x)/c-2*arctan((b-
2*c-(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2))*
(b*(b^2-2*c*(a+c))+(-b^4-2*c^2*(a+c)^2+2*b^2*c*(2*a+c))/(-4*a*c+b^2)^(1/2))
/c^3/(b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2)-2*ar
ctan((b-2*c+(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c+(-4*a*c+b^2)^(1/2))
^(1/2))*(b^4+2*c^2*(a+c)^2-2*b^2*c*(2*a+c)+b^3*(-4*a*c+b^2)^(1/2)-2*b*c*(a+
c)*(-4*a*c+b^2)^(1/2))/c^3/(-4*a*c+b^2)^(1/2)/(b-2*c+(-4*a*c+b^2)^(1/2))^(1
/2)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2)
```

Rubi [A] (verified)

Time = 11.37 (sec) , antiderivative size = 386, normalized size of antiderivative = 0.99, number of steps used = 10, number of rules used = 7, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.368, Rules used = {3348, 2717, 2715, 8, 3374, 2738, 211}

$$\int \frac{\sin^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$$
$$=\frac{2\left(-2b^2c(a+c)-b\sqrt{b^2-4ac}(b^2-2c(a+c))-2c(ab^2-c(a+c)^2)+b^4\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{-\sqrt{b^2-4ac}+b-2c}}{\sqrt{-\sqrt{b^2-4ac}+b+2c}}\right)}{c^3\sqrt{b^2-4ac}\sqrt{-\sqrt{b^2-4ac}+b-2c}\sqrt{-\sqrt{b^2-4ac}+b+2c}}$$
$$-\frac{2\left(-2b^2c(2a+c)-2bc(a+c)\sqrt{b^2-4ac}+b^3\sqrt{b^2-4ac}+2c^2(a+c)^2+b^4\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{\sqrt{b^2-4ac}+b-2c}}{\sqrt{\sqrt{b^2-4ac}+b+2c}}\right)}{c^3\sqrt{b^2-4ac}\sqrt{\sqrt{b^2-4ac}+b-2c}\sqrt{\sqrt{b^2-4ac}+b+2c}}$$
$$+\frac{x(b^2-c(a+2c))}{c^3}-\frac{b\sin(x)}{c^2}+\frac{x}{2c}+\frac{\sin(x)\cos(x)}{2c}$$

```
[In] Int[Sin[x]^4/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] x/(2*c) + ((b^2 - c*(a + 2*c))*x)/c^3 + (2*(b^4 - 2*b^2*c*(a + c) - b*Sqrt[
b^2 - 4*a*c]*(b^2 - 2*c*(a + c)) - 2*c*(a*b^2 - c*(a + c)^2))*ArcTan[(Sqrt[
b - 2*c - Sqrt[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]])/
(c^3*Sqrt[b^2 - 4*a*c]*Sqrt[b - 2*c - Sqrt[b^2 - 4*a*c]]*Sqrt[b + 2*c - Sqr
t[b^2 - 4*a*c]]) - (2*(b^4 + 2*c^2*(a + c)^2 - 2*b^2*c*(2*a + c) + b^3*Sqrt
[b^2 - 4*a*c] - 2*b*c*(a + c)*Sqrt[b^2 - 4*a*c])*ArcTan[(Sqrt[b - 2*c + Sqr
t[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]]])/(c^3*Sqrt[b^2
 - 4*a*c]*Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]*Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c
]]) - (b*Sin[x])/c^2 + (Cos[x]*Sin[x])/(2*c)
```

Rule 8

```
Int[a_, x_Symbol] :> Simp[a*x, x] /; FreeQ[a, x]
```

Rule 211

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(Rt[a/b, 2]/a)*ArcTan[x/R
t[a/b, 2]], x] /; FreeQ[{a, b}, x] && PosQ[a/b]
```

Rule 2715

```
Int[((b_.)*sin[(c_.) + (d_.)*(x_)])^(n_), x_Symbol] :> Simp[(-b)*Cos[c + d*
x]*((b*Sin[c + d*x])^(n - 1)/(d*n)), x] + Dist[b^2*((n - 1)/n), Int[(b*Sin[
c + d*x])^(n - 2), x], x] /; FreeQ[{b, c, d}, x] && GtQ[n, 1] && IntegerQ[2
*n]
```

Rule 2717

```
Int[sin[Pi/2 + (c_.) + (d_.)*(x_)], x_Symbol] :> Simp[Sin[c + d*x]/d, x] /;
 FreeQ[{c, d}, x]
```

Rule 2738

```
Int[((a_) + (b_.)*sin[Pi/2 + (c_.) + (d_.)*(x_)])^(-1), x_Symbol] :> With[{
e = FreeFactors[Tan[(c + d*x)/2], x]}, Dist[2*(e/d), Subst[Int[1/(a + b + (
a - b)*e^2*x^2), x], x, Tan[(c + d*x)/2]/e], x]] /; FreeQ[{a, b, c, d}, x]
&& NeQ[a^2 - b^2, 0]
```

Rule 3348

```
Int[((a_.) + cos[(d_.) + (e_.)*(x_)]^(n_.)*(b_.) + cos[(d_.) + (e_.)*(x_)]^
(n2_.)*(c_.))^(p_.)*sin[(d_.) + (e_.)*(x_)]^(m_.), x_Symbol] :> Int[ExpandT
rig[(1 - cos[d + e*x]^2)^(m/2)*(a + b*cos[d + e*x]^n + c*cos[d + e*x]^(2*n)
)^p, x], x] /; FreeQ[{a, b, c, d, e}, x] && EqQ[n2, 2*n] && IntegerQ[m/2] &
& NeQ[b^2 - 4*a*c, 0] && IntegersQ[n, p]
```

Rule 3374

```
Int[(cos[(d_.) + (e_.)*(x_)]*(B_.) + (A_))/((a_.) + cos[(d_.) + (e_.)*(x_)]
*(b_.) + cos[(d_.) + (e_.)*(x_)]^2*(c_.)), x_Symbol] :> Module[{q = Rt[b^2
- 4*a*c, 2]}, Dist[B + (b*B - 2*A*c)/q, Int[1/(b + q + 2*c*Cos[d + e*x]), x
], x] + Dist[B - (b*B - 2*A*c)/q, Int[1/(b - q + 2*c*Cos[d + e*x]), x], x]]
 /; FreeQ[{a, b, c, d, e, A, B}, x] && NeQ[b^2 - 4*a*c, 0]
```

Rubi steps

$$\text{integral} = \int \left(\frac{b^2-c(a+2c)}{c^3} - \frac{b\cos(x)}{c^2} + \frac{\cos^2(x)}{c} + \frac{-ab^2\left(1-\frac{c(a+c)^2}{ab^2}\right) - b^3\left(1-\frac{2c(a+c)}{b^2}\right)\cos(x)}{c^3\left(a+b\cos(x)+c\cos^2(x)\right)} \right) dx$$

$$= \frac{(b^2-c(a+2c))\,x}{c^3} + \frac{\int \frac{-ab^2\left(1-\frac{c(a+c)^2}{ab^2}\right)-b^3\left(1-\frac{2c(a+c)}{b^2}\right)\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx}{c^3} - \frac{b\int\cos(x)\,dx}{c^2} + \frac{\int\cos^2(x)\,dx}{c}$$

$$= \frac{(b^2-c(a+2c))\,x}{c^3} - \frac{b\sin(x)}{c^2} + \frac{\cos(x)\sin(x)}{2c} + \frac{\int 1\,dx}{2c} - \frac{\left(b^4+2c^2(a+c)^2-2b^2c(2a+c)+b\sqrt{b^2-4ac}\,(b^2-2c(a+c))\right)\int\frac{1}{b+\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{c^3\sqrt{b^2-4ac}} + \frac{\left(b^4-2b^2c(a+c)-b\sqrt{b^2-4ac}\,(b^2-2c(a+c))-2c(ab^2-c(a+c)^2)\right)\int\frac{1}{b-\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{c^3\sqrt{b^2-4ac}}$$

$$= \frac{x}{2c} + \frac{(b^2 - c(a+2c))\,x}{c^3} - \frac{b\sin(x)}{c^2} + \frac{\cos(x)\sin(x)}{2c}$$
$$- \frac{\left(2\left(b^4 + 2c^2(a+c)^2 - 2b^2c(2a+c) + b\sqrt{b^2-4ac}(b^2 - 2c(a+c))\right)\right)\text{Subst}\left(\int \frac{1}{b+2c+\sqrt{b^2-4ac}+\left(b-2c\right.}\right.}{c^3\sqrt{b^2-4ac}}$$
$$+ \frac{\left(2\left(b^4 - 2b^2c(a+c) - b\sqrt{b^2-4ac}(b^2 - 2c(a+c)) - 2c(ab^2 - c(a+c)^2)\right)\right)\text{Subst}\left(\int \frac{}{b+2c-\sqrt{b^2-4a}}\right.}{c^3\sqrt{b^2-4ac}}$$
$$= \frac{x}{2c} + \frac{(b^2 - c(a+2c))\,x}{c^3}$$
$$+ \frac{2\left(b^4 - 2b^2c(a+c) - b\sqrt{b^2-4ac}(b^2 - 2c(a+c)) - 2c(ab^2 - c(a+c)^2)\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}}{\sqrt{b+2c-\sqrt{b^2-}}}\right.}{c^3\sqrt{b^2-4ac}\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}}$$
$$- \frac{2\left(b^4 + 2c^2(a+c)^2 - 2b^2c(2a+c) + b\sqrt{b^2-4ac}(b^2 - 2c(a+c))\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{c^3\sqrt{b^2-4ac}\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}$$
$$- \frac{b\sin(x)}{c^2} + \frac{\cos(x)\sin(x)}{2c}$$

Mathematica [A] (verified)

Time = 0.65 (sec) , antiderivative size = 374, normalized size of antiderivative = 0.96

$$\int \frac{\sin^4(x)}{a + b\cos(x) + c\cos^2(x)}\,dx$$
$$= \frac{4b^2x - 2c(2a+3c)x + \frac{4\sqrt{2}\left(b^4+2c^2(a+c)^2-2b^2c(2a+c)+b^3\sqrt{b^2-4ac}-2bc(a+c)\sqrt{b^2-4ac}\right)\operatorname{arctanh}\left(\frac{\left(b-2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)-2b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{-b^2+2c(a+c)-b\sqrt{b^2-4ac}}}}{}$$

```
[In] Integrate[Sin[x]^4/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (4*b^2*x - 2*c*(2*a + 3*c)*x + (4*Sqrt[2]*(b^4 + 2*c^2*(a + c)^2 - 2*b^2*c*
(2*a + c) + b^3*Sqrt[b^2 - 4*a*c] - 2*b*c*(a + c)*Sqrt[b^2 - 4*a*c])*ArcTan
h[((b - 2*c + Sqrt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) - 2*b*
Sqrt[b^2 - 4*a*c]]])/(Sqrt[b^2 - 4*a*c]*Sqrt[-b^2 + 2*c*(a + c) - b*Sqrt[b^
2 - 4*a*c]]) - (4*Sqrt[2]*(-b^4 - 2*c^2*(a + c)^2 + 2*b^2*c*(2*a + c) + b^3
*Sqrt[b^2 - 4*a*c] - 2*b*c*(a + c)*Sqrt[b^2 - 4*a*c])*ArcTanh[((-b + 2*c +
Sqrt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) + 2*b*Sqrt[b^2 - 4*a
*c]]])/(Sqrt[b^2 - 4*a*c]*Sqrt[-b^2 + 2*c*(a + c) + b*Sqrt[b^2 - 4*a*c]]) -
 4*b*c*Sin[x] + c^2*Sin[2*x])/(4*c^3)
```

Maple [A] (verified)

Time = 5.47 (sec) , antiderivative size = 406, normalized size of antiderivative = 1.05

method	result
default	$\frac{2(a-b+c)\left(\frac{\left(\sqrt{-4ac+b^2}\,ac-\sqrt{-4ac+b^2}\,b^2-\sqrt{-4ac+b^2}\,bc+\sqrt{-4ac+b^2}\,c^2+3cab+2a\,c^2-b^3-b^2c+b\,c^2+2c^3\right)\operatorname{arctanh}\left(\frac{(-a+b-c)\tan\left(\frac{x}{2}\right)}{\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+\cdots}}\right.}{2\sqrt{-4ac+b^2}\,\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}\right.}{c^3\cdots}$
risch	Expression too large to display

```
[In] int(sin(x)^4/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] 2/c^3*(a-b+c)*(1/2*((-4*a*c+b^2)^(1/2)*a*c-(-4*a*c+b^2)^(1/2)*b^2-(-4*a*c+b
^2)^(1/2)*b*c+(-4*a*c+b^2)^(1/2)*c^2+3*c*a*b+2*a*c^2-b^3-b^2*c+b*c^2+2*c^3)
/(-4*a*c+b^2)^(1/2)/(((-4*a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2)*arctanh((-a+b-
c)*tan(1/2*x)/(((-4*a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2))+1/2*((-4*a*c+b^2)^(
1/2)*a*c-(-4*a*c+b^2)^(1/2)*b^2-(-4*a*c+b^2)^(1/2)*b*c+(-4*a*c+b^2)^(1/2)*c
^2-3*c*a*b-2*a*c^2+b^3+b^2*c-b*c^2-2*c^3)/(-4*a*c+b^2)^(1/2)/(((-4*a*c+b^2)
^(1/2)+a-c)*(a-b+c))^(1/2)*arctan((a-b+c)*tan(1/2*x)/(((-4*a*c+b^2)^(1/2)+a
-c)*(a-b+c))^(1/2)))-2/c^3*(((c*b+1/2*c^2)*tan(1/2*x)^3+(c*b-1/2*c^2)*tan(1
/2*x))/(1+tan(1/2*x)^2)^2+1/2*(2*a*c-2*b^2+3*c^2)*arctan(tan(1/2*x)))
```

Fricas [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 5045 vs. $2(323) = 646$.

Time = 2.40 (sec) , antiderivative size = 5045, normalized size of antiderivative = 13.00

$$\int \frac{\sin^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(sin(x)^4/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
```

```
[Out] Too large to include
```

Sympy [F(-1)]

Timed out.

$$\int \frac{\sin^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Timed out}$$

```
[In] integrate(sin(x)**4/(a+b*cos(x)+c*cos(x)**2),x)
```

```
[Out] Timed out
```

Maxima [F]

$$\int \frac{\sin^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \int \frac{\sin(x)^4}{c\cos(x)^2+b\cos(x)+a}\,dx$$

```
[In] integrate(sin(x)^4/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
```

```
[Out] 1/4*(4*c^3*integrate(-2*(2*(b^4 - 2*a*b^2*c - 2*b^2*c^2)*cos(3*x)^2 + 4*(2*
a^2*b^2 - 5*a^2*c^2 - 4*a*c^3 - c^4 - (2*a^3 - a*b^2)*c)*cos(2*x)^2 + 2*(b^
4 - 2*a*b^2*c - 2*b^2*c^2)*cos(x)^2 + 2*(b^4 - 2*a*b^2*c - 2*b^2*c^2)*sin(3
*x)^2 + 4*(2*a^2*b^2 - 5*a^2*c^2 - 4*a*c^3 - c^4 - (2*a^3 - a*b^2)*c)*sin(2
*x)^2 + 2*(4*a*b^3 - 10*a*b*c^2 - 4*b*c^3 - (6*a^2*b - b^3)*c)*sin(2*x)*sin
(x) + 2*(b^4 - 2*a*b^2*c - 2*b^2*c^2)*sin(x)^2 + ((b^3*c - 2*a*b*c^2 - 2*b*
c^3)*cos(3*x) + 2*(a*b^2*c - a^2*c^2 - 2*a*c^3 - c^4)*cos(2*x) + (b^3*c - 2
*a*b*c^2 - 2*b*c^3)*cos(x))*cos(4*x) + (b^3*c - 2*a*b*c^2 - 2*b*c^3 + 2*(4*
a*b^3 - 10*a*b*c^2 - 4*b*c^3 - (6*a^2*b - b^3)*c)*cos(2*x) + 4*(b^4 - 2*a*b
^2*c - 2*b^2*c^2)*cos(x))*cos(3*x) + 2*(a*b^2*c - a^2*c^2 - 2*a*c^3 - c^4 +
 (4*a*b^3 - 10*a*b*c^2 - 4*b*c^3 - (6*a^2*b - b^3)*c)*cos(x))*cos(2*x) + (b
^3*c - 2*a*b*c^2 - 2*b*c^3)*cos(x) + ((b^3*c - 2*a*b*c^2 - 2*b*c^3)*sin(3*x
) + 2*(a*b^2*c - a^2*c^2 - 2*a*c^3 - c^4)*sin(2*x) + (b^3*c - 2*a*b*c^2 - 2
*b*c^3)*sin(x))*sin(4*x) + 2*((4*a*b^3 - 10*a*b*c^2 - 4*b*c^3 - (6*a^2*b -
b^3)*c)*sin(2*x) + 2*(b^4 - 2*a*b^2*c - 2*b^2*c^2)*sin(x))*sin(3*x))/(c^5*c
os(4*x)^2 + 4*b^2*c^3*cos(3*x)^2 + 4*b^2*c^3*cos(x)^2 + c^5*sin(4*x)^2 + 4*
b^2*c^3*sin(3*x)^2 + 4*b^2*c^3*sin(x)^2 + 4*b*c^4*cos(x) + c^5 + 4*(4*a^2*c
^3 + 4*a*c^4 + c^5)*cos(2*x)^2 + 4*(4*a^2*c^3 + 4*a*c^4 + c^5)*sin(2*x)^2 +
 8*(2*a*b*c^3 + b*c^4)*sin(2*x)*sin(x) + 2*(2*b*c^4*cos(3*x) + 2*b*c^4*cos(
x) + c^5 + 2*(2*a*c^4 + c^5)*cos(2*x))*cos(4*x) + 4*(2*b^2*c^3*cos(x) + b*c
^4 + 2*(2*a*b*c^3 + b*c^4)*cos(2*x))*cos(3*x) + 4*(2*a*c^4 + c^5 + 2*(2*a*b
*c^3 + b*c^4)*cos(x))*cos(2*x) + 4*(b*c^4*sin(3*x) + b*c^4*sin(x) + (2*a*c^
4 + c^5)*sin(2*x))*sin(4*x) + 8*(b^2*c^3*sin(x) + (2*a*b*c^3 + b*c^4)*sin(2
*x))*sin(3*x)), x) + c^2*sin(2*x) - 4*b*c*sin(x) + 2*(2*b^2 - 2*a*c - 3*c^2
)*x)/c^3
```

Giac [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 11375 vs. $2(323) = 646$.

Time = 2.80 (sec) , antiderivative size = 11375, normalized size of antiderivative = 29.32

$$\int \frac{\sin^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(sin(x)^4/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] ((2*a^2*b^6 - 4*a*b^7 + 2*b^8 - 18*a^3*b^4*c + 38*a^2*b^5*c - 18*a*b^6*c -
2*b^7*c + 48*a^4*b^2*c^2 - 112*a^3*b^3*c^2 + 42*a^2*b^4*c^2 + 28*a*b^5*c^2
- 4*b^6*c^2 - 32*a^5*c^3 + 96*a^4*b*c^3 + 16*a^3*b^2*c^3 - 128*a^2*b^3*c^3
+ 26*a*b^4*c^3 + 6*b^5*c^3 - 96*a^4*c^4 + 192*a^3*b*c^4 - 16*a^2*b^2*c^4 -
48*a*b^3*c^4 - 2*b^4*c^4 - 96*a^3*c^5 + 96*a^2*b*c^5 + 16*a*b^2*c^5 - 32*a^
2*c^6 + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(
b^2 - 4*a*c)*a^2*b^4 - 2*(b^2 - 4*a*c)*a^2*b^4 - 2*sqrt(a^2 - a*b + b*c - c
^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^5 + 4*(b^2 - 4*a*
c)*a*b^5 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sq
rt(b^2 - 4*a*c)*b^6 - 2*(b^2 - 4*a*c)*b^6 - 15*sqrt(a^2 - a*b + b*c - c^2 +
 sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c + 10*(b^2 - 4*a
*c)*a^3*b^2*c + 13*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*sqrt(b^2 - 4*a*c)*a^2*b^3*c - 22*(b^2 - 4*a*c)*a^2*b^3*c + 37*sqrt(a^2
- a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4*
c + 10*(b^2 - 4*a*c)*a*b^4*c + sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*
c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5*c + 2*(b^2 - 4*a*c)*b^5*c + 12*sqrt(a
^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4
*c^2 - 8*(b^2 - 4*a*c)*a^4*c^2 - 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^2 + 24*(b^2 - 4*a*c)*a^3*b*c
^2 - 85*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^
2 - 4*a*c)*a^2*b^2*c^2 - 2*(b^2 - 4*a*c)*a^2*b^2*c^2 + 6*sqrt(a^2 - a*b + b
*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c^2 - 20*
(b^2 - 4*a*c)*a*b^3*c^2 + 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*
(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c^2 + 4*(b^2 - 4*a*c)*b^4*c^2 + 68*sqrt(
a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^
3*c^3 - 24*(b^2 - 4*a*c)*a^3*c^3 - 40*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^3 + 48*(b^2 - 4*a*c)*a^2*b
*c^3 - 33*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(
b^2 - 4*a*c)*a*b^2*c^3 - 10*(b^2 - 4*a*c)*a*b^2*c^3 - 11*sqrt(a^2 - a*b + b
*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^3*c^3 - 6*(b^
2 - 4*a*c)*b^3*c^3 + 36*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
 b + c))*sqrt(b^2 - 4*a*c)*a^2*c^4 - 24*(b^2 - 4*a*c)*a^2*c^4 + 44*sqrt(a^2
 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*c
^4 + 24*(b^2 - 4*a*c)*a*b*c^4 + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4
*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c^4 + 2*(b^2 - 4*a*c)*b^2*c^4 - 20
*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a
*c)*a*c^5 - 8*(b^2 - 4*a*c)*a*c^5)*c^2*abs(a - b + c) - (4*a^2*b^6*c - 4*b^
8*c - 36*a^3*b^4*c^2 - 4*a^2*b^5*c^2 + 44*a*b^6*c^2 + 4*b^7*c^2 + 96*a^4*b^
2*c^3 + 32*a^3*b^3*c^3 - 172*a^2*b^4*c^3 - 40*a*b^5*c^3 + 8*b^6*c^3 - 64*a^
5*c^4 - 64*a^4*b*c^4 + 288*a^3*b^2*c^4 + 128*a^2*b^3*c^4 - 76*a*b^4*c^4 - 4
*b^5*c^4 - 192*a^4*c^5 - 128*a^3*b*c^5 + 224*a^2*b^2*c^5 + 32*a*b^3*c^5 - 4
*b^4*c^5 - 192*a^3*c^6 - 64*a^2*b*c^6 + 32*a*b^2*c^6 - 64*a^2*c^7 + 3*sqrt(
a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^4*c + sqrt(a^2
 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^5*c - 7*sqrt(a^2
- a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^6*c - 5*sqrt(a^2 - a
```

```
*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^7*c - 15*sqrt(a^2 - a*b +
 b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*b^2*c^2 - 8*sqrt(a^2 - a*b
+ b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^3*c^2 + 51*sqrt(a^2 - a*
b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^4*c^2 + 50*sqrt(a^2 -
a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^5*c^2 + 6*sqrt(a^2 - a
*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^6*c^2 + 12*sqrt(a^2 - a*b
 + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^5*c^3 + 16*sqrt(a^2 - a*b +
 b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*b*c^3 - 112*sqrt(a^2 - a*b
+ b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^2*c^3 - 156*sqrt(a^2 - a
*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^3*c^3 - 27*sqrt(a^2 -
 a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^4*c^3 + 5*sqrt(a^2 -
a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^5*c^3 + 80*sqrt(a^2 - a*
b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*c^4 + 144*sqrt(a^2 - a*b
 + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b*c^4 - 14*sqrt(a^2 - a*b
 + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^2*c^4 - 48*sqrt(a^2 - a
*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^3*c^4 - 7*sqrt(a^2 - a*
b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^4*c^4 + 104*sqrt(a^2 - a*b
 + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*c^5 + 112*sqrt(a^2 - a*b
+ b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b*c^5 + 24*sqrt(a^2 - a*b
+ b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^2*c^5 + 4*sqrt(a^2 - a*b +
 b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^3*c^5 + 16*sqrt(a^2 - a*b + b
*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*c^6 - 16*sqrt(a^2 - a*b + b*c
 - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b*c^6 + 5*sqrt(a^2 - a*b + b*c -
c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^2*c^6 - 20*sqrt(a^2 - a*b + b*c - c^
2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*c^7 - 4*(b^2 - 4*a*c)*a^2*b^4*c + 4*(b
^2 - 4*a*c)*b^6*c + 20*(b^2 - 4*a*c)*a^3*b^2*c^2 + 4*(b^2 - 4*a*c)*a^2*b^3*
c^2 - 28*(b^2 - 4*a*c)*a*b^4*c^2 - 4*(b^2 - 4*a*c)*b^5*c^2 - 16*(b^2 - 4*a*
c)*a^4*c^3 - 16*(b^2 - 4*a*c)*a^3*b*c^3 + 60*(b^2 - 4*a*c)*a^2*b^2*c^3 + 24
*(b^2 - 4*a*c)*a*b^3*c^3 - 8*(b^2 - 4*a*c)*b^4*c^3 - 48*(b^2 - 4*a*c)*a^3*c
^4 - 32*(b^2 - 4*a*c)*a^2*b*c^4 + 44*(b^2 - 4*a*c)*a*b^2*c^4 + 4*(b^2 - 4*a
*c)*b^3*c^4 - 48*(b^2 - 4*a*c)*a^2*c^5 - 16*(b^2 - 4*a*c)*a*b*c^5 + 4*(b^2
- 4*a*c)*b^2*c^5 - 16*(b^2 - 4*a*c)*a*c^6)*abs(a - b + c)*abs(c) + (2*a^3*b
^5*c^2 - 4*a^2*b^6*c^2 + 2*a*b^7*c^2 - 14*a^4*b^3*c^3 + 30*a^3*b^4*c^3 - 16
*a^2*b^5*c^3 + 2*a*b^6*c^3 - 2*b^7*c^3 + 24*a^5*b*c^4 - 60*a^4*b^2*c^4 + 32
*a^3*b^3*c^4 - 6*a^2*b^4*c^4 + 10*a*b^5*c^4 + 2*b^6*c^4 + 16*a^5*c^5 - 16*a
^3*b^2*c^5 + 4*a^2*b^3*c^5 - 22*a*b^4*c^5 + 4*b^5*c^5 + 32*a^4*c^6 - 48*a^3
*b*c^6 + 56*a^2*b^2*c^6 - 16*a*b^3*c^6 - 2*b^4*c^6 + 16*a*b^2*c^7 - 6*b^3*c
^7 - 32*a^2*c^8 + 24*a*b*c^8 + 4*b^2*c^8 - 16*a*c^9 + 3*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^3*c^2 - 2*
(b^2 - 4*a*c)*a^3*b^3*c^2 - 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^4*c^2 + 4*(b^2 - 4*a*c)*a^2*b^4*c^2
- 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
4*a*c)*a*b^5*c^2 - 2*(b^2 - 4*a*c)*a*b^5*c^2 - 9*sqrt(a^2 - a*b + b*c - c^2
 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b*c^3 + 6*(b^2 - 4*
a*c)*a^4*b*c^3 + 9*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
```

c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c^3 - 14*(b^2 - 4*a*c)*a^3*b^2*c^3 + 24*sqrt(
a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^
2*b^3*c^3 + 8*(b^2 - 4*a*c)*a^2*b^3*c^3 + 3*sqrt(a^2 - a*b + b*c - c^2 + sq
rt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4*c^3 - 2*(b^2 - 4*a*c)*
a*b^4*c^3 + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*s
qrt(b^2 - 4*a*c)*b^5*c^3 + 2*(b^2 - 4*a*c)*b^5*c^3 - 6*sqrt(a^2 - a*b + b*c
 - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*c^4 + 4*(b^2
- 4*a*c)*a^4*c^4 - 32*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
 + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^4 - sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
- 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c^4 - 2*(b^2 - 4*a*c)*a^2*b
^2*c^4 - 21*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*a*b^3*c^4 - 2*(b^2 - 4*a*c)*a*b^3*c^4 - sqrt(a^2 - a*b + b*c
 - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c^4 - 2*(b^2
- 4*a*c)*b^4*c^4 - 28*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
 + c))*sqrt(b^2 - 4*a*c)*a^3*c^5 + 8*(b^2 - 4*a*c)*a^3*c^5 + 34*sqrt(a^2 -
a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^
5 - 12*(b^2 - 4*a*c)*a^2*b*c^5 - 9*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^5 + 14*(b^2 - 4*a*c)*a*b^2*c^
5 - 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*b^3*c^5 - 4*(b^2 - 4*a*c)*b^3*c^5 + 16*sqrt(a^2 - a*b + b*c - c^2
+ sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^6 + sqrt(a^2 - a*b
 + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c^6 + 2
*(b^2 - 4*a*c)*b^2*c^6 + 28*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*
(a - b + c))*sqrt(b^2 - 4*a*c)*a*c^7 - 8*(b^2 - 4*a*c)*a*c^7 + 7*sqrt(a^2 -
 a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b*c^7 +
 6*(b^2 - 4*a*c)*b*c^7 - 10*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*
(a - b + c))*sqrt(b^2 - 4*a*c)*c^8 - 4*(b^2 - 4*a*c)*c^8)*abs(a - b + c))*(
pi*floor(1/2*x/pi + 1/2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a*c^3 - 2*
c^4 + sqrt(-4*(a*c^3 + b*c^3 + c^4)*(a*c^3 - b*c^3 + c^4) + 4*(a*c^3 - c^4)
^2))/(a*c^3 - b*c^3 + c^4))))/((3*a^5*b^2*c^4 - 5*a^4*b^3*c^4 - 6*a^3*b^4*c
^4 + 10*a^2*b^5*c^4 + 3*a*b^6*c^4 - 5*b^7*c^4 - 12*a^6*c^5 + 20*a^5*b*c^5 +
 47*a^4*b^2*c^5 - 60*a^3*b^3*c^5 - 46*a^2*b^4*c^5 + 40*a*b^5*c^5 + 11*b^6*c
^5 - 92*a^5*c^6 + 80*a^4*b*c^6 + 182*a^3*b^2*c^6 - 94*a^2*b^3*c^6 - 78*a*b^
4*c^6 - 6*b^5*c^6 - 184*a^4*c^7 + 56*a^3*b*c^7 + 166*a^2*b^2*c^7 + 36*a*b^3
*c^7 - 6*b^4*c^7 - 120*a^3*c^8 - 48*a^2*b*c^8 + 23*a*b^2*c^8 + 11*b^3*c^8 +
 4*a^2*c^9 - 44*a*b*c^9 - 5*b^2*c^9 + 20*a*c^10)*abs(c)) - ((2*a^2*b^6 - 4*
a*b^7 + 2*b^8 - 18*a^3*b^4*c + 38*a^2*b^5*c - 18*a*b^6*c - 2*b^7*c + 48*a^4
*b^2*c^2 - 112*a^3*b^3*c^2 + 42*a^2*b^4*c^2 + 28*a*b^5*c^2 - 4*b^6*c^2 - 32
*a^5*c^3 + 96*a^4*b*c^3 + 16*a^3*b^2*c^3 - 128*a^2*b^3*c^3 + 26*a*b^4*c^3 +
 6*b^5*c^3 - 96*a^4*c^4 + 192*a^3*b*c^4 - 16*a^2*b^2*c^4 - 48*a*b^3*c^4 - 2
*b^4*c^4 - 96*a^3*c^5 + 96*a^2*b*c^5 + 16*a*b^2*c^5 - 32*a^2*c^6 + 3*sqrt(a
^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2
b^4 - 2(b^2 - 4*a*c)*a^2*b^4 - 2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^5 + 4*(b^2 - 4*a*c)*a*b^5 - 5*sqr
t(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*

```
b^6 - 2*(b^2 - 4*a*c)*b^6 - 15*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c + 10*(b^2 - 4*a*c)*a^3*b^2*c +
13*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4
*a*c)*a^2*b^3*c - 22*(b^2 - 4*a*c)*a^2*b^3*c + 37*sqrt(a^2 - a*b + b*c - c^
2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4*c + 10*(b^2 - 4*
a*c)*a*b^4*c + sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*
sqrt(b^2 - 4*a*c)*b^5*c + 2*(b^2 - 4*a*c)*b^5*c + 12*sqrt(a^2 - a*b + b*c -
 c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*c^2 - 8*(b^2 -
4*a*c)*a^4*c^2 - 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
 c))*sqrt(b^2 - 4*a*c)*a^3*b*c^2 + 24*(b^2 - 4*a*c)*a^3*b*c^2 - 85*sqrt(a^2
 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b
^2*c^2 - 2*(b^2 - 4*a*c)*a^2*b^2*c^2 + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(
b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c^2 - 20*(b^2 - 4*a*c)*a*
b^3*c^2 + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*b^4*c^2 + 4*(b^2 - 4*a*c)*b^4*c^2 + 68*sqrt(a^2 - a*b + b*c
- c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^3 - 24*(b^2
- 4*a*c)*a^3*c^3 - 40*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
 + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^3 + 48*(b^2 - 4*a*c)*a^2*b*c^3 - 33*sqrt(a
^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b
^2*c^3 - 10*(b^2 - 4*a*c)*a*b^2*c^3 - 11*sqrt(a^2 - a*b + b*c - c^2 - sqrt(
b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^3*c^3 - 6*(b^2 - 4*a*c)*b^3*c
^3 + 36*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^
2 - 4*a*c)*a^2*c^4 - 24*(b^2 - 4*a*c)*a^2*c^4 + 44*sqrt(a^2 - a*b + b*c - c
^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*c^4 + 24*(b^2 - 4
*a*c)*a*b*c^4 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c
))*sqrt(b^2 - 4*a*c)*b^2*c^4 + 2*(b^2 - 4*a*c)*b^2*c^4 - 20*sqrt(a^2 - a*b
+ b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*c^5 - 8*(b
^2 - 4*a*c)*a*c^5)*c^2*abs(a - b + c) - (4*a^2*b^6*c - 4*b^8*c - 36*a^3*b^4
*c^2 - 4*a^2*b^5*c^2 + 44*a*b^6*c^2 + 4*b^7*c^2 + 96*a^4*b^2*c^3 + 32*a^3*b
^3*c^3 - 172*a^2*b^4*c^3 - 40*a*b^5*c^3 + 8*b^6*c^3 - 64*a^5*c^4 - 64*a^4*b
*c^4 + 288*a^3*b^2*c^4 + 128*a^2*b^3*c^4 - 76*a*b^4*c^4 - 4*b^5*c^4 - 192*a
^4*c^5 - 128*a^3*b*c^5 + 224*a^2*b^2*c^5 + 32*a*b^3*c^5 - 4*b^4*c^5 - 192*a
^3*c^6 - 64*a^2*b*c^6 + 32*a*b^2*c^6 - 64*a^2*c^7 - 3*sqrt(a^2 - a*b + b*c
- c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^4*c - sqrt(a^2 - a*b + b*c - c
^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^5*c + 7*sqrt(a^2 - a*b + b*c - c^
2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^6*c + 5*sqrt(a^2 - a*b + b*c - c^2 -
 sqrt(b^2 - 4*a*c)*(a - b + c))*b^7*c + 15*sqrt(a^2 - a*b + b*c - c^2 - sqr
t(b^2 - 4*a*c)*(a - b + c))*a^4*b^2*c^2 + 8*sqrt(a^2 - a*b + b*c - c^2 - sq
rt(b^2 - 4*a*c)*(a - b + c))*a^3*b^3*c^2 - 51*sqrt(a^2 - a*b + b*c - c^2 -
sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^4*c^2 - 50*sqrt(a^2 - a*b + b*c - c^2
- sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^5*c^2 - 6*sqrt(a^2 - a*b + b*c - c^2 -
 sqrt(b^2 - 4*a*c)*(a - b + c))*b^6*c^2 - 12*sqrt(a^2 - a*b + b*c - c^2 - s
qrt(b^2 - 4*a*c)*(a - b + c))*a^5*c^3 - 16*sqrt(a^2 - a*b + b*c - c^2 - sqr
t(b^2 - 4*a*c)*(a - b + c))*a^4*b*c^3 + 112*sqrt(a^2 - a*b + b*c - c^2 - sq
rt(b^2 - 4*a*c)*(a - b + c))*a^3*b^2*c^3 + 156*sqrt(a^2 - a*b + b*c - c^2 -
```

```
 sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^3*c^3 + 27*sqrt(a^2 - a*b + b*c - c^2
 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^4*c^3 - 5*sqrt(a^2 - a*b + b*c - c^2
- sqrt(b^2 - 4*a*c)*(a - b + c))*b^5*c^3 - 80*sqrt(a^2 - a*b + b*c - c^2 -
sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*c^4 - 144*sqrt(a^2 - a*b + b*c - c^2 - s
qrt(b^2 - 4*a*c)*(a - b + c))*a^3*b*c^4 + 14*sqrt(a^2 - a*b + b*c - c^2 - s
qrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^2*c^4 + 48*sqrt(a^2 - a*b + b*c - c^2 -
 sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^3*c^4 + 7*sqrt(a^2 - a*b + b*c - c^2 -
sqrt(b^2 - 4*a*c)*(a - b + c))*b^4*c^4 - 104*sqrt(a^2 - a*b + b*c - c^2 - s
qrt(b^2 - 4*a*c)*(a - b + c))*a^3*c^5 - 112*sqrt(a^2 - a*b + b*c - c^2 - sq
rt(b^2 - 4*a*c)*(a - b + c))*a^2*b*c^5 - 24*sqrt(a^2 - a*b + b*c - c^2 - sq
rt(b^2 - 4*a*c)*(a - b + c))*a*b^2*c^5 - 4*sqrt(a^2 - a*b + b*c - c^2 - sqr
t(b^2 - 4*a*c)*(a - b + c))*b^3*c^5 - 16*sqrt(a^2 - a*b + b*c - c^2 - sqrt(
b^2 - 4*a*c)*(a - b + c))*a^2*c^6 + 16*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^
2 - 4*a*c)*(a - b + c))*a*b*c^6 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
 4*a*c)*(a - b + c))*b^2*c^6 + 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4
*a*c)*(a - b + c))*a*c^7 - 4*(b^2 - 4*a*c)*a^2*b^4*c + 4*(b^2 - 4*a*c)*b^6*
c + 20*(b^2 - 4*a*c)*a^3*b^2*c^2 + 4*(b^2 - 4*a*c)*a^2*b^3*c^2 - 28*(b^2 -
4*a*c)*a*b^4*c^2 - 4*(b^2 - 4*a*c)*b^5*c^2 - 16*(b^2 - 4*a*c)*a^4*c^3 - 16*
(b^2 - 4*a*c)*a^3*b*c^3 + 60*(b^2 - 4*a*c)*a^2*b^2*c^3 + 24*(b^2 - 4*a*c)*a
*b^3*c^3 - 8*(b^2 - 4*a*c)*b^4*c^3 - 48*(b^2 - 4*a*c)*a^3*c^4 - 32*(b^2 - 4
*a*c)*a^2*b*c^4 + 44*(b^2 - 4*a*c)*a*b^2*c^4 + 4*(b^2 - 4*a*c)*b^3*c^4 - 48
*(b^2 - 4*a*c)*a^2*c^5 - 16*(b^2 - 4*a*c)*a*b*c^5 + 4*(b^2 - 4*a*c)*b^2*c^5
 - 16*(b^2 - 4*a*c)*a*c^6)*abs(a - b + c)*abs(c) + (2*a^3*b^5*c^2 - 4*a^2*b
^6*c^2 + 2*a*b^7*c^2 - 14*a^4*b^3*c^3 + 30*a^3*b^4*c^3 - 16*a^2*b^5*c^3 + 2
*a*b^6*c^3 - 2*b^7*c^3 + 24*a^5*b*c^4 - 60*a^4*b^2*c^4 + 32*a^3*b^3*c^4 - 6
*a^2*b^4*c^4 + 10*a*b^5*c^4 + 2*b^6*c^4 + 16*a^5*c^5 - 16*a^3*b^2*c^5 + 4*a
^2*b^3*c^5 - 22*a*b^4*c^5 + 4*b^5*c^5 + 32*a^4*c^6 - 48*a^3*b*c^6 + 56*a^2*
b^2*c^6 - 16*a*b^3*c^6 - 2*b^4*c^6 + 16*a*b^2*c^7 - 6*b^3*c^7 - 32*a^2*c^8
+ 24*a*b*c^8 + 4*b^2*c^8 - 16*a*c^9 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b
^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^3*c^2 - 2*(b^2 - 4*a*c)*a^
3*b^3*c^2 - 2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*s
qrt(b^2 - 4*a*c)*a^2*b^4*c^2 + 4*(b^2 - 4*a*c)*a^2*b^4*c^2 - 5*sqrt(a^2 - a
*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^5*c^2
 - 2*(b^2 - 4*a*c)*a*b^5*c^2 - 9*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b*c^3 + 6*(b^2 - 4*a*c)*a^4*b*c^3 +
 9*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4
*a*c)*a^3*b^2*c^3 - 14*(b^2 - 4*a*c)*a^3*b^2*c^3 + 24*sqrt(a^2 - a*b + b*c
- c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3*c^3 + 8*(b
^2 - 4*a*c)*a^2*b^3*c^3 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*
(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4*c^3 - 2*(b^2 - 4*a*c)*a*b^4*c^3 + 5*sq
rt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)
*b^5*c^3 + 2*(b^2 - 4*a*c)*b^5*c^3 - 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^
2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*c^4 + 4*(b^2 - 4*a*c)*a^4*c^4
 - 32*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*a^3*b*c^4 - sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
```

```
+ c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c^4 - 2*(b^2 - 4*a*c)*a^2*b^2*c^4 - 21*sqrt
(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a
*b^3*c^4 - 2*(b^2 - 4*a*c)*a*b^3*c^4 - sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^
2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c^4 - 2*(b^2 - 4*a*c)*b^4*c^4
 - 28*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*a^3*c^5 + 8*(b^2 - 4*a*c)*a^3*c^5 + 34*sqrt(a^2 - a*b + b*c - c^2
- sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^5 - 12*(b^2 - 4*
a*c)*a^2*b*c^5 - 9*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
c))*sqrt(b^2 - 4*a*c)*a*b^2*c^5 + 14*(b^2 - 4*a*c)*a*b^2*c^5 - 6*sqrt(a^2 -
 a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^3*c^5
 - 4*(b^2 - 4*a*c)*b^3*c^5 + 16*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a
*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^6 + sqrt(a^2 - a*b + b*c - c^2 - s
qrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c^6 + 2*(b^2 - 4*a*c)*b
^2*c^6 + 28*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*a*c^7 - 8*(b^2 - 4*a*c)*a*c^7 + 7*sqrt(a^2 - a*b + b*c - c^2
 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b*c^7 + 6*(b^2 - 4*a*c)
*b*c^7 - 10*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*c^8 - 4*(b^2 - 4*a*c)*c^8)*abs(a - b + c))*(pi*floor(1/2*x/p
i + 1/2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a*c^3 - 2*c^4 - sqrt(-4*(a
*c^3 + b*c^3 + c^4)*(a*c^3 - b*c^3 + c^4) + 4*(a*c^3 - c^4)^2))/(a*c^3 - b*
c^3 + c^4))))/((3*a^5*b^2*c^4 - 5*a^4*b^3*c^4 - 6*a^3*b^4*c^4 + 10*a^2*b^5*
c^4 + 3*a*b^6*c^4 - 5*b^7*c^4 - 12*a^6*c^5 + 20*a^5*b*c^5 + 47*a^4*b^2*c^5
- 60*a^3*b^3*c^5 - 46*a^2*b^4*c^5 + 40*a*b^5*c^5 + 11*b^6*c^5 - 92*a^5*c^6
+ 80*a^4*b*c^6 + 182*a^3*b^2*c^6 - 94*a^2*b^3*c^6 - 78*a*b^4*c^6 - 6*b^5*c^
6 - 184*a^4*c^7 + 56*a^3*b*c^7 + 166*a^2*b^2*c^7 + 36*a*b^3*c^7 - 6*b^4*c^7
 - 120*a^3*c^8 - 48*a^2*b*c^8 + 23*a*b^2*c^8 + 11*b^3*c^8 + 4*a^2*c^9 - 44*
a*b*c^9 - 5*b^2*c^9 + 20*a*c^10)*abs(c)) + 1/2*(2*b^2 - 2*a*c - 3*c^2)*x/c^
3 - (2*b*tan(1/2*x)^3 + c*tan(1/2*x)^3 + 2*b*tan(1/2*x) - c*tan(1/2*x))/((t
an(1/2*x)^2 + 1)^2*c^2)
```

Mupad [B] (verification not implemented)

Time = 15.41 (sec) , antiderivative size = 46613, normalized size of antiderivative = 120.14

$$\int \frac{\sin^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] int(sin(x)^4/(a + b*cos(x) + c*cos(x)^2),x)
```

```
[Out] atan((((((((2048*(48*a*c^15 + 272*a^2*c^14 + 576*a^3*c^13 + 576*a^4*c^12 + 2
72*a^5*c^11 + 48*a^6*c^10 - 12*b^2*c^14 + 20*b^3*c^13 + 18*b^4*c^12 - 46*b^
5*c^11 + 6*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 - 140*a*b^2*c^13 + 288*a*b^3*
c^12 + 30*a*b^4*c^11 - 240*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 - 416*a
^2*b*c^13 - 736*a^3*b*c^12 - 544*a^4*b*c^11 - 144*a^5*b*c^10 - 360*a^2*b^2*
c^12 + 728*a^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8
```

- 360*a^3*b^2*c^11 + 544*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 -
172*a^4*b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9 - 80*a*
b*c^14))/c^8 - (2048*tan(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 +
8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^
2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2
*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(
1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a
*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b
^4*c^6 - 8*a*b^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*
a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^
12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4
*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32
*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b
^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*
c^11 - 96*a*b*c^15))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^
4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 -
18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2
*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2)
 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c -
 b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c
^6 - 8*a*b^2*c^7)))^(1/2) + (2048*tan(x/2)*(24*b*c^14 - 96*a*c^14 - 8*c^15
+ 152*a^2*c^13 + 952*a^3*c^12 + 1096*a^4*c^11 + 304*a^5*c^10 - 152*a^6*c^9
- 72*a^7*c^8 + 2*b^2*c^13 - 38*b^3*c^12 - 7*b^4*c^11 + 39*b^5*c^10 - 15*b^6
*c^9 + 35*b^7*c^8 - 44*b^8*c^7 - 4*b^9*c^6 + 24*b^10*c^5 - 8*b^11*c^4 + 68*
a*b^2*c^12 + 42*a*b^3*c^11 - 159*a*b^4*c^10 - 400*a*b^5*c^9 + 537*a*b^6*c^8
 + 68*a*b^7*c^7 - 276*a*b^8*c^6 + 72*a*b^9*c^5 + 8*a*b^10*c^4 - 944*a^2*b*c
^12 - 2520*a^3*b*c^11 - 1824*a^4*b*c^10 - 272*a^5*b*c^9 + 88*a^6*b*c^8 + 58
4*a^2*b^2*c^11 + 1742*a^2*b^3*c^10 - 1645*a^2*b^4*c^9 - 795*a^2*b^5*c^8 + 1
132*a^2*b^6*c^7 - 112*a^2*b^7*c^6 - 112*a^2*b^8*c^5 + 8*a^2*b^9*c^4 + 476*a
^3*b^2*c^10 + 2766*a^3*b^3*c^9 - 1705*a^3*b^4*c^8 - 396*a^3*b^5*c^7 + 456*a
^3*b^6*c^6 - 56*a^3*b^7*c^5 - 8*a^3*b^8*c^4 + 230*a^4*b^2*c^9 + 880*a^4*b^3
*c^8 - 656*a^4*b^4*c^7 + 140*a^4*b^5*c^6 + 72*a^4*b^6*c^5 + 464*a^5*b^2*c^8
 - 192*a^5*b^3*c^7 - 220*a^5*b^4*c^6 + 256*a^6*b^2*c^7 + 136*a*b*c^13))/c^8
)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b
^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*
c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 -
38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*
c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3
c(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2
) + (2048*(236*a*c^13 - 32*b*c^13 + 12*c^14 + 1084*a^2*c^12 + 2328*a^3*c^11
 + 2784*a^4*c^10 + 1948*a^5*c^9 + 780*a^6*c^8 + 160*a^7*c^7 + 12*a^8*c^6 -
39*b^2*c^12 + 121*b^3*c^11 + 61*b^4*c^10 - 220*b^5*c^9 - 36*b^6*c^8 + 232*b
^7*c^7 - 28*b^8*c^6 - 127*b^9*c^5 + 42*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2
 - 635*a*b^2*c^11 + 1300*a*b^3*c^10 + 608*a*b^4*c^9 - 1792*a*b^5*c^8 - 60*a
*b^6*c^7 + 1218*a*b^7*c^6 - 249*a*b^8*c^5 - 340*a*b^9*c^4 + 98*a*b^10*c^3 +

20*a*b^11*c^2 - 1616*a^2*b*c^11 - 3160*a^3*b*c^10 - 3440*a^4*b*c^9 - 2132*
a^5*b*c^8 - 704*a^6*b*c^7 - 96*a^7*b*c^6 - 2242*a^2*b^2*c^10 + 4146*a^2*b^3
*c^9 + 1420*a^2*b^4*c^8 - 4158*a^2*b^5*c^7 + 77*a^2*b^6*c^6 + 1735*a^2*b^7*
c^5 - 234*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 3714*a^3*b^2*c^9
+ 6252*a^3*b^3*c^8 + 1730*a^3*b^4*c^7 - 4300*a^3*b^5*c^6 - 79*a^3*b^6*c^5
+ 968*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 3523*a^4*b^2*c^8 + 502
5*a^4*b^3*c^7 + 1339*a^4*b^4*c^6 - 2082*a^4*b^5*c^5 - 192*a^4*b^6*c^4 + 156
*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 2031*a^5*b^2*c^7 + 2104*a^5*b^3*c^6 + 634*a^
5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 676*a^6*b^2*c^6 + 364*a^6*b^
3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^2*c^5 - 404*a*b*c^12))/c^8)*(-(8*a*c^7
+ b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2)
- 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4
*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^
3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c
- b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c -
b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) - (2048*tan
(x/2)*(20*a*b^12 + 42*a*c^12 - 58*b*c^12 + 4*b^12*c - 4*b^13 + 22*c^13 - 40
*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 - 214*a^2*c^11 - 938*a^3*c
^10 - 1538*a^4*c^9 - 1278*a^5*c^8 - 498*a^6*c^7 - 14*a^7*c^6 + 52*a^8*c^5 +
12*a^9*c^4 + 14*b^2*c^11 + 34*b^3*c^10 + 59*b^4*c^9 - 39*b^5*c^8 - 160*b^6
*c^7 + 112*b^7*c^6 + 105*b^8*c^5 - 89*b^9*c^4 - 28*b^10*c^3 + 28*b^11*c^2 -
518*a*b^2*c^10 - 264*a*b^3*c^9 + 1339*a*b^4*c^8 - 92*a*b^5*c^7 - 1312*a*b^
6*c^6 + 268*a*b^7*c^5 + 649*a*b^8*c^4 - 124*a*b^9*c^3 - 180*a*b^10*c^2 + 15
50*a^2*b*c^10 - 160*a^2*b^10*c + 3488*a^3*b*c^9 + 320*a^3*b^9*c + 3350*a^4*
b*c^8 - 300*a^4*b^8*c + 1092*a^5*b*c^7 + 136*a^5*b^7*c - 462*a^6*b*c^6 - 24
*a^6*b^6*c - 440*a^7*b*c^5 - 92*a^8*b*c^4 - 1568*a^2*b^2*c^9 - 2708*a^2*b^3
*c^8 + 3564*a^2*b^4*c^7 + 1964*a^2*b^5*c^6 - 2790*a^2*b^6*c^5 - 922*a^2*b^7
*c^4 + 1048*a^2*b^8*c^3 + 276*a^2*b^9*c^2 - 652*a^3*b^2*c^8 - 6280*a^3*b^3*
c^7 + 2020*a^3*b^4*c^6 + 4988*a^3*b^5*c^5 - 1118*a^3*b^6*c^4 - 2008*a^3*b^7
*c^3 + 140*a^3*b^8*c^2 + 2350*a^4*b^2*c^7 - 5630*a^4*b^3*c^6 - 2295*a^4*b^4
*c^5 + 3563*a^4*b^5*c^4 + 1260*a^4*b^6*c^3 - 740*a^4*b^7*c^2 + 3314*a^5*b^2
*c^6 - 1456*a^5*b^3*c^5 - 2771*a^5*b^4*c^4 + 308*a^5*b^5*c^3 + 732*a^5*b^6*
c^2 + 1572*a^6*b^2*c^5 + 576*a^6*b^3*c^4 - 696*a^6*b^4*c^3 - 300*a^6*b^5*c^
2 + 192*a^7*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 1
48*a*b*c^11 + 24*a*b^11*c))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5
+ 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6
*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*
a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3
)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(
4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8
+ b^4*c^6 - 8*a*b^2*c^7)))^(1/2)*1i - (((((2048*(48*a*c^15 + 272*a^2*c^14 +
576*a^3*c^13 + 576*a^4*c^12 + 272*a^5*c^11 + 48*a^6*c^10 - 12*b^2*c^14 + 2
0*b^3*c^13 + 18*b^4*c^12 - 46*b^5*c^11 + 6*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c
^8 - 140*a*b^2*c^13 + 288*a*b^3*c^12 + 30*a*b^4*c^11 - 240*a*b^5*c^10 + 74*
a*b^6*c^9 + 20*a*b^7*c^8 - 416*a^2*b*c^13 - 736*a^3*b*c^12 - 544*a^4*b*c^11

- 144*a^5*b*c^10 - 360*a^2*b^2*c^12 + 728*a^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 360*a^3*b^2*c^11 + 544*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 172*a^4*b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9 - 80*a*b*c^14))/c^8 + (2048*tan(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(24*b*c^14 - 96*a*c^14 - 8*c^15 + 152*a^2*c^13 + 952*a^3*c^12 + 1096*a^4*c^11 + 304*a^5*c^10 - 152*a^6*c^9 - 72*a^7*c^8 + 2*b^2*c^13 - 38*b^3*c^12 - 7*b^4*c^11 + 39*b^5*c^10 - 15*b^6*c^9 + 35*b^7*c^8 - 44*b^8*c^7 - 4*b^9*c^6 + 24*b^10*c^5 - 8*b^11*c^4 + 68*a*b^2*c^12 + 42*a*b^3*c^11 - 159*a*b^4*c^10 - 400*a*b^5*c^9 + 537*a*b^6*c^8 + 68*a*b^7*c^7 - 276*a*b^8*c^6 + 72*a*b^9*c^5 + 8*a*b^10*c^4 - 944*a^2*b*c^12 - 2520*a^3*b*c^11 - 1824*a^4*b*c^10 - 272*a^5*b*c^9 + 88*a^6*b*c^8 + 584*a^2*b^2*c^11 + 1742*a^2*b^3*c^10 - 1645*a^2*b^4*c^9 - 795*a^2*b^5*c^8 + 1132*a^2*b^6*c^7 - 112*a^2*b^7*c^6 - 112*a^2*b^8*c^5 + 8*a^2*b^9*c^4 + 476*a^3*b^2*c^10 + 2766*a^3*b^3*c^9 - 1705*a^3*b^4*c^8 - 396*a^3*b^5*c^7 + 456*a^3*b^6*c^6 - 56*a^3*b^7*c^5 - 8*a^3*b^8*c^4 + 230*a^4*b^2*c^9 + 880*a^4*b^3*c^8 - 656*a^4*b^4*c^7 + 140*a^4*b^5*c^6 + 72*a^4*b^6*c^5 + 464*a^5*b^2*c^8 - 192*a^5*b^3*c^7 - 220*a^5*b^4*c^6 + 256*a^6*b^2*c^7 + 136*a*b*c^13))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) + (2048*(236*a*c^13 - 32*b*c^13 + 12*c^14 + 1084*a^2*c^12 + 2328*a^3*c^11 + 2784*a^4*c^10 + 1948*a^5*c^9 + 780*a^6*c^8 + 160*a^7*c^7 + 12*a^8*c^6 - 39*b^2*c^12 + 121*b^3*c^11 + 61*b^4*c^10 - 220*b^5*c^9 - 36*b^6*c^8 + 232*b^7*c^7 - 28*b^8*c^6 - 127*b^9*c^5 + 42*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 635*a*b^2*c^11 + 1300*a*b^3*c^10 + 608*a

```
*b^4*c^9 - 1792*a*b^5*c^8 - 60*a*b^6*c^7 + 1218*a*b^7*c^6 - 249*a*b^8*c^5 -
 340*a*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^11*c^2 - 1616*a^2*b*c^11 - 3160*a^3
*b*c^10 - 3440*a^4*b*c^9 - 2132*a^5*b*c^8 - 704*a^6*b*c^7 - 96*a^7*b*c^6 -
2242*a^2*b^2*c^10 + 4146*a^2*b^3*c^9 + 1420*a^2*b^4*c^8 - 4158*a^2*b^5*c^7
+ 77*a^2*b^6*c^6 + 1735*a^2*b^7*c^5 - 234*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4
*a^2*b^10*c^2 - 3714*a^3*b^2*c^9 + 6252*a^3*b^3*c^8 + 1730*a^3*b^4*c^7 - 43
00*a^3*b^5*c^6 - 79*a^3*b^6*c^5 + 968*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*
b^9*c^2 - 3523*a^4*b^2*c^8 + 5025*a^4*b^3*c^7 + 1339*a^4*b^4*c^6 - 2082*a^4
*b^5*c^5 - 192*a^4*b^6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 2031*a^5*b^2
*c^7 + 2104*a^5*b^3*c^6 + 634*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^
3 - 676*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^2*c^5 -
 404*a*b*c^12))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4
 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*
b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4
+ 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10
*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)
^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 -
8*a*b^2*c^7)))^(1/2) + (2048*tan(x/2)*(20*a*b^12 + 42*a*c^12 - 58*b*c^12 +
4*b^12*c - 4*b^13 + 22*c^13 - 40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^
5*b^8 - 214*a^2*c^11 - 938*a^3*c^10 - 1538*a^4*c^9 - 1278*a^5*c^8 - 498*a^6
*c^7 - 14*a^7*c^6 + 52*a^8*c^5 + 12*a^9*c^4 + 14*b^2*c^11 + 34*b^3*c^10 + 5
9*b^4*c^9 - 39*b^5*c^8 - 160*b^6*c^7 + 112*b^7*c^6 + 105*b^8*c^5 - 89*b^9*c
^4 - 28*b^10*c^3 + 28*b^11*c^2 - 518*a*b^2*c^10 - 264*a*b^3*c^9 + 1339*a*b^
4*c^8 - 92*a*b^5*c^7 - 1312*a*b^6*c^6 + 268*a*b^7*c^5 + 649*a*b^8*c^4 - 124
*a*b^9*c^3 - 180*a*b^10*c^2 + 1550*a^2*b*c^10 - 160*a^2*b^10*c + 3488*a^3*b
*c^9 + 320*a^3*b^9*c + 3350*a^4*b*c^8 - 300*a^4*b^8*c + 1092*a^5*b*c^7 + 13
6*a^5*b^7*c - 462*a^6*b*c^6 - 24*a^6*b^6*c - 440*a^7*b*c^5 - 92*a^8*b*c^4 -
 1568*a^2*b^2*c^9 - 2708*a^2*b^3*c^8 + 3564*a^2*b^4*c^7 + 1964*a^2*b^5*c^6
- 2790*a^2*b^6*c^5 - 922*a^2*b^7*c^4 + 1048*a^2*b^8*c^3 + 276*a^2*b^9*c^2 -
 652*a^3*b^2*c^8 - 6280*a^3*b^3*c^7 + 2020*a^3*b^4*c^6 + 4988*a^3*b^5*c^5 -
 1118*a^3*b^6*c^4 - 2008*a^3*b^7*c^3 + 140*a^3*b^8*c^2 + 2350*a^4*b^2*c^7 -
 5630*a^4*b^3*c^6 - 2295*a^4*b^4*c^5 + 3563*a^4*b^5*c^4 + 1260*a^4*b^6*c^3
- 740*a^4*b^7*c^2 + 3314*a^5*b^2*c^6 - 1456*a^5*b^3*c^5 - 2771*a^5*b^4*c^4
+ 308*a^5*b^5*c^3 + 732*a^5*b^6*c^2 + 1572*a^6*b^2*c^5 + 576*a^6*b^3*c^4 -
696*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 192*a^7*b^2*c^4 + 272*a^7*b^3*c^3 + 44*
a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 148*a*b*c^11 + 24*a*b^11*c))/c^8)*(-(8*a*c^7
 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2)
 - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^
4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c
^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c
 - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c
- b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2)*1i)/((4096
*(16*a*b^11 + 274*a*c^11 - 78*b*c^11 + 4*b^11*c - 4*b^12 + 33*c^12 - 16*a^2
*b^10 - 16*a^3*b^9 + 40*a^4*b^8 - 16*a^5*b^7 - 16*a^6*b^6 + 16*a^7*b^5 - 4*
a^8*b^4 + 1008*a^2*c^10 + 2156*a^3*c^9 + 2954*a^4*c^8 + 2688*a^5*c^7 + 1624
```

*a^6*c^6 + 628*a^7*c^5 + 141*a^8*c^4 + 14*a^9*c^3 - 64*b^2*c^10 + 268*b^3*c
^9 - 26*b^4*c^8 - 348*b^5*c^7 + 144*b^6*c^6 + 208*b^7*c^5 - 123*b^8*c^4 - 5
4*b^9*c^3 + 40*b^10*c^2 - 520*a*b^2*c^9 + 1516*a*b^3*c^8 + 144*a*b^4*c^7 -
1564*a*b^5*c^6 + 228*a*b^6*c^5 + 740*a*b^7*c^4 - 146*a*b^8*c^3 - 164*a*b^9*
c^2 - 1624*a^2*b*c^9 - 112*a^2*b^9*c - 2676*a^3*b*c^8 + 128*a^3*b^8*c - 258
8*a^4*b*c^7 + 56*a^4*b^7*c - 1388*a^5*b*c^6 - 184*a^5*b^6*c - 264*a^6*b*c^5
 + 80*a^6*b^5*c + 116*a^7*b*c^4 + 32*a^7*b^4*c + 74*a^8*b*c^3 - 28*a^8*b^3*
c + 12*a^9*b*c^2 + 4*a^9*b^2*c - 1820*a^2*b^2*c^8 + 3576*a^2*b^3*c^7 + 1032
*a^2*b^4*c^6 - 2792*a^2*b^5*c^5 - 236*a^2*b^6*c^4 + 920*a^2*b^7*c^3 + 64*a^
2*b^8*c^2 - 3584*a^3*b^2*c^7 + 4472*a^3*b^3*c^6 + 2236*a^3*b^4*c^5 - 2436*a
^3*b^5*c^4 - 744*a^3*b^6*c^3 + 464*a^3*b^7*c^2 - 4336*a^4*b^2*c^6 + 3040*a^
4*b^3*c^5 + 2390*a^4*b^4*c^4 - 964*a^4*b^5*c^3 - 592*a^4*b^6*c^2 - 3284*a^5
*b^2*c^5 + 908*a^5*b^3*c^4 + 1364*a^5*b^4*c^3 - 40*a^5*b^5*c^2 - 1500*a^6*b
^2*c^4 - 104*a^6*b^3*c^3 + 384*a^6*b^4*c^2 - 360*a^7*b^2*c^3 - 144*a^7*b^3*
c^2 - 24*a^8*b^2*c^2 - 544*a*b*c^10 + 20*a*b^10*c))/c^8 + (((((2048*(48*a*c
^15 + 272*a^2*c^14 + 576*a^3*c^13 + 576*a^4*c^12 + 272*a^5*c^11 + 48*a^6*c^
10 - 12*b^2*c^14 + 20*b^3*c^13 + 18*b^4*c^12 - 46*b^5*c^11 + 6*b^6*c^10 + 2
6*b^7*c^9 - 12*b^8*c^8 - 140*a*b^2*c^13 + 288*a*b^3*c^12 + 30*a*b^4*c^11 -
240*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 - 416*a^2*b*c^13 - 736*a^3*b*c
^12 - 544*a^4*b*c^11 - 144*a^5*b*c^10 - 360*a^2*b^2*c^12 + 728*a^2*b^3*c^11
 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 360*a^3*b^2*c^11 + 5
44*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 172*a^4*b^2*c^10 + 116*
a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9 - 80*a*b*c^14))/c^8 - (2048*ta
n(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*
c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a
*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c
^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a
^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*
a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))
^(1/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12
 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^
7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 -
 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b
^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c
^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^
8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c -
b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4
*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 -
 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b
c^2(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^
3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/
2) + (2048*tan(x/2)*(24*b*c^14 - 96*a*c^14 - 8*c^15 + 152*a^2*c^13 + 952*a^
3*c^12 + 1096*a^4*c^11 + 304*a^5*c^10 - 152*a^6*c^9 - 72*a^7*c^8 + 2*b^2*c^
13 - 38*b^3*c^12 - 7*b^4*c^11 + 39*b^5*c^10 - 15*b^6*c^9 + 35*b^7*c^8 - 44*
b^8*c^7 - 4*b^9*c^6 + 24*b^10*c^5 - 8*b^11*c^4 + 68*a*b^2*c^12 + 42*a*b^3*c

```
^11 - 159*a*b^4*c^10 - 400*a*b^5*c^9 + 537*a*b^6*c^8 + 68*a*b^7*c^7 - 276*a
*b^8*c^6 + 72*a*b^9*c^5 + 8*a*b^10*c^4 - 944*a^2*b*c^12 - 2520*a^3*b*c^11 -
 1824*a^4*b*c^10 - 272*a^5*b*c^9 + 88*a^6*b*c^8 + 584*a^2*b^2*c^11 + 1742*a
^2*b^3*c^10 - 1645*a^2*b^4*c^9 - 795*a^2*b^5*c^8 + 1132*a^2*b^6*c^7 - 112*a
^2*b^7*c^6 - 112*a^2*b^8*c^5 + 8*a^2*b^9*c^4 + 476*a^3*b^2*c^10 + 2766*a^3*
b^3*c^9 - 1705*a^3*b^4*c^8 - 396*a^3*b^5*c^7 + 456*a^3*b^6*c^6 - 56*a^3*b^7
*c^5 - 8*a^3*b^8*c^4 + 230*a^4*b^2*c^9 + 880*a^4*b^3*c^8 - 656*a^4*b^4*c^7
+ 140*a^4*b^5*c^6 + 72*a^4*b^6*c^5 + 464*a^5*b^2*c^8 - 192*a^5*b^3*c^7 - 22
0*a^5*b^4*c^6 + 256*a^6*b^2*c^7 + 136*a*b*c^13))/c^8)*(-(8*a*c^7 + b^8 + 24
*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^
6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c
- b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*
c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^
(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(
1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) + (2048*(236*a*c^13 -
 32*b*c^13 + 12*c^14 + 1084*a^2*c^12 + 2328*a^3*c^11 + 2784*a^4*c^10 + 1948
*a^5*c^9 + 780*a^6*c^8 + 160*a^7*c^7 + 12*a^8*c^6 - 39*b^2*c^12 + 121*b^3*c
^11 + 61*b^4*c^10 - 220*b^5*c^9 - 36*b^6*c^8 + 232*b^7*c^7 - 28*b^8*c^6 - 1
27*b^9*c^5 + 42*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 635*a*b^2*c^11 + 130
0*a*b^3*c^10 + 608*a*b^4*c^9 - 1792*a*b^5*c^8 - 60*a*b^6*c^7 + 1218*a*b^7*c
^6 - 249*a*b^8*c^5 - 340*a*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^11*c^2 - 1616*a
^2*b*c^11 - 3160*a^3*b*c^10 - 3440*a^4*b*c^9 - 2132*a^5*b*c^8 - 704*a^6*b*c
^7 - 96*a^7*b*c^6 - 2242*a^2*b^2*c^10 + 4146*a^2*b^3*c^9 + 1420*a^2*b^4*c^8
 - 4158*a^2*b^5*c^7 + 77*a^2*b^6*c^6 + 1735*a^2*b^7*c^5 - 234*a^2*b^8*c^4 -
 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 3714*a^3*b^2*c^9 + 6252*a^3*b^3*c^8 + 1
730*a^3*b^4*c^7 - 4300*a^3*b^5*c^6 - 79*a^3*b^6*c^5 + 968*a^3*b^7*c^4 + 2*a
^3*b^8*c^3 - 20*a^3*b^9*c^2 - 3523*a^4*b^2*c^8 + 5025*a^4*b^3*c^7 + 1339*a^
4*b^4*c^6 - 2082*a^4*b^5*c^5 - 192*a^4*b^6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^
8*c^2 - 2031*a^5*b^2*c^7 + 2104*a^5*b^3*c^6 + 634*a^5*b^4*c^5 - 388*a^5*b^5
*c^4 - 60*a^5*b^6*c^3 - 676*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a^6*b^4*c^4
 - 100*a^7*b^2*c^5 - 404*a*b*c^12))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24
*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4
 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/
2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c
- b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b
*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*
a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(20*a*b^12 + 42*a
*c^12 - 58*b*c^12 + 4*b^12*c - 4*b^13 + 22*c^13 - 40*a^2*b^11 + 40*a^3*b^10
 - 20*a^4*b^9 + 4*a^5*b^8 - 214*a^2*c^11 - 938*a^3*c^10 - 1538*a^4*c^9 - 12
78*a^5*c^8 - 498*a^6*c^7 - 14*a^7*c^6 + 52*a^8*c^5 + 12*a^9*c^4 + 14*b^2*c^
11 + 34*b^3*c^10 + 59*b^4*c^9 - 39*b^5*c^8 - 160*b^6*c^7 + 112*b^7*c^6 + 10
5*b^8*c^5 - 89*b^9*c^4 - 28*b^10*c^3 + 28*b^11*c^2 - 518*a*b^2*c^10 - 264*a
*b^3*c^9 + 1339*a*b^4*c^8 - 92*a*b^5*c^7 - 1312*a*b^6*c^6 + 268*a*b^7*c^5 +
 649*a*b^8*c^4 - 124*a*b^9*c^3 - 180*a*b^10*c^2 + 1550*a^2*b*c^10 - 160*a^2
*b^10*c + 3488*a^3*b*c^9 + 320*a^3*b^9*c + 3350*a^4*b*c^8 - 300*a^4*b^8*c +
```

1092*a^5*b*c^7 + 136*a^5*b^7*c - 462*a^6*b*c^6 - 24*a^6*b^6*c - 440*a^7*b*
c^5 - 92*a^8*b*c^4 - 1568*a^2*b^2*c^9 - 2708*a^2*b^3*c^8 + 3564*a^2*b^4*c^7
 + 1964*a^2*b^5*c^6 - 2790*a^2*b^6*c^5 - 922*a^2*b^7*c^4 + 1048*a^2*b^8*c^3
 + 276*a^2*b^9*c^2 - 652*a^3*b^2*c^8 - 6280*a^3*b^3*c^7 + 2020*a^3*b^4*c^6
+ 4988*a^3*b^5*c^5 - 1118*a^3*b^6*c^4 - 2008*a^3*b^7*c^3 + 140*a^3*b^8*c^2
+ 2350*a^4*b^2*c^7 - 5630*a^4*b^3*c^6 - 2295*a^4*b^4*c^5 + 3563*a^4*b^5*c^4
 + 1260*a^4*b^6*c^3 - 740*a^4*b^7*c^2 + 3314*a^5*b^2*c^6 - 1456*a^5*b^3*c^5
 - 2771*a^5*b^4*c^4 + 308*a^5*b^5*c^3 + 732*a^5*b^6*c^2 + 1572*a^6*b^2*c^5
+ 576*a^6*b^3*c^4 - 696*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 192*a^7*b^2*c^4 + 2
72*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 148*a*b*c^11 + 24*a*b^11
c))/c^8)(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4
*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 2
4*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^
4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c +
3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) -
 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7
)))^(1/2) + (((((2048*(48*a*c^15 + 272*a^2*c^14 + 576*a^3*c^13 + 576*a^4*c^
12 + 272*a^5*c^11 + 48*a^6*c^10 - 12*b^2*c^14 + 20*b^3*c^13 + 18*b^4*c^12 -
 46*b^5*c^11 + 6*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 - 140*a*b^2*c^13 + 288*
a*b^3*c^12 + 30*a*b^4*c^11 - 240*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 -
 416*a^2*b*c^13 - 736*a^3*b*c^12 - 544*a^4*b*c^11 - 144*a^5*b*c^10 - 360*a^
2*b^2*c^12 + 728*a^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b
^6*c^8 - 360*a^3*b^2*c^11 + 544*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*
c^8 - 172*a^4*b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9 -
 80*a*b*c^14))/c^8 + (2048*tan(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*
c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*
b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) -
54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2
)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*
(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c
^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14
 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*
b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184
*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^1
3 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8
*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^
4*b^2*c^11 - 96*a*b*c^15))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5
+ 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*
c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a
^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)
^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4
*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 +
 b^4*c^6 - 8*a*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(24*b*c^14 - 96*a*c^14 - 8
*c^15 + 152*a^2*c^13 + 952*a^3*c^12 + 1096*a^4*c^11 + 304*a^5*c^10 - 152*a^
6*c^9 - 72*a^7*c^8 + 2*b^2*c^13 - 38*b^3*c^12 - 7*b^4*c^11 + 39*b^5*c^10 -

15*b^6*c^9 + 35*b^7*c^8 - 44*b^8*c^7 - 4*b^9*c^6 + 24*b^10*c^5 - 8*b^11*c^4
 + 68*a*b^2*c^12 + 42*a*b^3*c^11 - 159*a*b^4*c^10 - 400*a*b^5*c^9 + 537*a*b
^6*c^8 + 68*a*b^7*c^7 - 276*a*b^8*c^6 + 72*a*b^9*c^5 + 8*a*b^10*c^4 - 944*a
^2*b*c^12 - 2520*a^3*b*c^11 - 1824*a^4*b*c^10 - 272*a^5*b*c^9 + 88*a^6*b*c^
8 + 584*a^2*b^2*c^11 + 1742*a^2*b^3*c^10 - 1645*a^2*b^4*c^9 - 795*a^2*b^5*c
^8 + 1132*a^2*b^6*c^7 - 112*a^2*b^7*c^6 - 112*a^2*b^8*c^5 + 8*a^2*b^9*c^4 +
 476*a^3*b^2*c^10 + 2766*a^3*b^3*c^9 - 1705*a^3*b^4*c^8 - 396*a^3*b^5*c^7 +
 456*a^3*b^6*c^6 - 56*a^3*b^7*c^5 - 8*a^3*b^8*c^4 + 230*a^4*b^2*c^9 + 880*a
^4*b^3*c^8 - 656*a^4*b^4*c^7 + 140*a^4*b^5*c^6 + 72*a^4*b^6*c^5 + 464*a^5*b
^2*c^8 - 192*a^5*b^3*c^7 - 220*a^5*b^4*c^6 + 256*a^6*b^2*c^7 + 136*a*b*c^13
))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a
*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*
a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*
c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*
a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4
*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7))
)^(1/2) + (2048*(236*a*c^13 - 32*b*c^13 + 12*c^14 + 1084*a^2*c^12 + 2328*a^
3*c^11 + 2784*a^4*c^10 + 1948*a^5*c^9 + 780*a^6*c^8 + 160*a^7*c^7 + 12*a^8*
c^6 - 39*b^2*c^12 + 121*b^3*c^11 + 61*b^4*c^10 - 220*b^5*c^9 - 36*b^6*c^8 +
 232*b^7*c^7 - 28*b^8*c^6 - 127*b^9*c^5 + 42*b^10*c^4 + 26*b^11*c^3 - 12*b^
12*c^2 - 635*a*b^2*c^11 + 1300*a*b^3*c^10 + 608*a*b^4*c^9 - 1792*a*b^5*c^8
- 60*a*b^6*c^7 + 1218*a*b^7*c^6 - 249*a*b^8*c^5 - 340*a*b^9*c^4 + 98*a*b^10
*c^3 + 20*a*b^11*c^2 - 1616*a^2*b*c^11 - 3160*a^3*b*c^10 - 3440*a^4*b*c^9 -
 2132*a^5*b*c^8 - 704*a^6*b*c^7 - 96*a^7*b*c^6 - 2242*a^2*b^2*c^10 + 4146*a
^2*b^3*c^9 + 1420*a^2*b^4*c^8 - 4158*a^2*b^5*c^7 + 77*a^2*b^6*c^6 + 1735*a^
2*b^7*c^5 - 234*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 3714*a^3*b
^2*c^9 + 6252*a^3*b^3*c^8 + 1730*a^3*b^4*c^7 - 4300*a^3*b^5*c^6 - 79*a^3*b^
6*c^5 + 968*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 3523*a^4*b^2*c^8
 + 5025*a^4*b^3*c^7 + 1339*a^4*b^4*c^6 - 2082*a^4*b^5*c^5 - 192*a^4*b^6*c^4
 + 156*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 2031*a^5*b^2*c^7 + 2104*a^5*b^3*c^6 +
634*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 676*a^6*b^2*c^6 + 364*
a^6*b^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^2*c^5 - 404*a*b*c^12))/c^8)*(-(8*
a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^
(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3
*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*
b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(
4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4
*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) + (20
48*tan(x/2)*(20*a*b^12 + 42*a*c^12 - 58*b*c^12 + 4*b^12*c - 4*b^13 + 22*c^1
3 - 40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 - 214*a^2*c^11 - 938
*a^3*c^10 - 1538*a^4*c^9 - 1278*a^5*c^8 - 498*a^6*c^7 - 14*a^7*c^6 + 52*a^8
*c^5 + 12*a^9*c^4 + 14*b^2*c^11 + 34*b^3*c^10 + 59*b^4*c^9 - 39*b^5*c^8 - 1
60*b^6*c^7 + 112*b^7*c^6 + 105*b^8*c^5 - 89*b^9*c^4 - 28*b^10*c^3 + 28*b^11
*c^2 - 518*a*b^2*c^10 - 264*a*b^3*c^9 + 1339*a*b^4*c^8 - 92*a*b^5*c^7 - 131
2*a*b^6*c^6 + 268*a*b^7*c^5 + 649*a*b^8*c^4 - 124*a*b^9*c^3 - 180*a*b^10*c^

2 + 1550*a^2*b*c^10 - 160*a^2*b^10*c + 3488*a^3*b*c^9 + 320*a^3*b^9*c + 335
0*a^4*b*c^8 - 300*a^4*b^8*c + 1092*a^5*b*c^7 + 136*a^5*b^7*c - 462*a^6*b*c^
6 - 24*a^6*b^6*c - 440*a^7*b*c^5 - 92*a^8*b*c^4 - 1568*a^2*b^2*c^9 - 2708*a
^2*b^3*c^8 + 3564*a^2*b^4*c^7 + 1964*a^2*b^5*c^6 - 2790*a^2*b^6*c^5 - 922*a
^2*b^7*c^4 + 1048*a^2*b^8*c^3 + 276*a^2*b^9*c^2 - 652*a^3*b^2*c^8 - 6280*a^
3*b^3*c^7 + 2020*a^3*b^4*c^6 + 4988*a^3*b^5*c^5 - 1118*a^3*b^6*c^4 - 2008*a
^3*b^7*c^3 + 140*a^3*b^8*c^2 + 2350*a^4*b^2*c^7 - 5630*a^4*b^3*c^6 - 2295*a
^4*b^4*c^5 + 3563*a^4*b^5*c^4 + 1260*a^4*b^6*c^3 - 740*a^4*b^7*c^2 + 3314*a
^5*b^2*c^6 - 1456*a^5*b^3*c^5 - 2771*a^5*b^4*c^4 + 308*a^5*b^5*c^3 + 732*a^
5*b^6*c^2 + 1572*a^6*b^2*c^5 + 576*a^6*b^3*c^4 - 696*a^6*b^4*c^3 - 300*a^6*
b^5*c^2 + 192*a^7*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c
^3 + 148*a*b*c^11 + 24*a*b^11*c))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a
^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 -
3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2)
- 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c -
b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c
^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^
2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2)))*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24
*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4
- 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/
2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c
- b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b
c^3(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*
a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2)*2i - ((tan(x/2)*(2*b - c))/c^2 + (
tan(x/2)^3*(2*b + c))/c^2)/(2*tan(x/2)^2 + tan(x/2)^4 + 1) + atan((((((204
8*(48*a*c^15 + 272*a^2*c^14 + 576*a^3*c^13 + 576*a^4*c^12 + 272*a^5*c^11 +
48*a^6*c^10 - 12*b^2*c^14 + 20*b^3*c^13 + 18*b^4*c^12 - 46*b^5*c^11 + 6*b^6
*c^10 + 26*b^7*c^9 - 12*b^8*c^8 - 140*a*b^2*c^13 + 288*a*b^3*c^12 + 30*a*b^
4*c^11 - 240*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 - 416*a^2*b*c^13 - 73
6*a^3*b*c^12 - 544*a^4*b*c^11 - 144*a^5*b*c^10 - 360*a^2*b^2*c^12 + 728*a^2
*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 360*a^3*b^2
*c^11 + 544*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 172*a^4*b^2*c^
10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9 - 80*a*b*c^14))/c^8 -
(2048*tan(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^
5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c
^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*
a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^
6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(
1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b
^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13 + 96*
a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b^6*c^
11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 56*a*b
^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^12 -
320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 - 272*
a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 96*a*b*

c^15))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) + (2048*tan(x/2)*(24*b*c^14 - 96*a*c^14 - 8*c^15 + 152*a^2*c^13 + 952*a^3*c^12 + 1096*a^4*c^11 + 304*a^5*c^10 - 152*a^6*c^9 - 72*a^7*c^8 + 2*b^2*c^13 - 38*b^3*c^12 - 7*b^4*c^11 + 39*b^5*c^10 - 15*b^6*c^9 + 35*b^7*c^8 - 44*b^8*c^7 - 4*b^9*c^6 + 24*b^10*c^5 - 8*b^11*c^4 + 68*a*b^2*c^12 + 42*a*b^3*c^11 - 159*a*b^4*c^10 - 400*a*b^5*c^9 + 537*a*b^6*c^8 + 68*a*b^7*c^7 - 276*a*b^8*c^6 + 72*a*b^9*c^5 + 8*a*b^10*c^4 - 944*a^2*b*c^12 - 2520*a^3*b*c^11 - 1824*a^4*b*c^10 - 272*a^5*b*c^9 + 88*a^6*b*c^8 + 584*a^2*b^2*c^11 + 1742*a^2*b^3*c^10 - 1645*a^2*b^4*c^9 - 795*a^2*b^5*c^8 + 1132*a^2*b^6*c^7 - 112*a^2*b^7*c^6 - 112*a^2*b^8*c^5 + 8*a^2*b^9*c^4 + 476*a^3*b^2*c^10 + 2766*a^3*b^3*c^9 - 1705*a^3*b^4*c^8 - 396*a^3*b^5*c^7 + 456*a^3*b^6*c^6 - 56*a^3*b^7*c^5 - 8*a^3*b^8*c^4 + 230*a^4*b^2*c^9 + 880*a^4*b^3*c^8 - 656*a^4*b^4*c^7 + 140*a^4*b^5*c^6 + 72*a^4*b^6*c^5 + 464*a^5*b^2*c^8 - 192*a^5*b^3*c^7 - 220*a^5*b^4*c^6 + 256*a^6*b^2*c^7 + 136*a*b*c^13))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) + (2048*(236*a*c^13 - 32*b*c^13 + 12*c^14 + 1084*a^2*c^12 + 2328*a^3*c^11 + 2784*a^4*c^10 + 1948*a^5*c^9 + 780*a^6*c^8 + 160*a^7*c^7 + 12*a^8*c^6 - 39*b^2*c^12 + 121*b^3*c^11 + 61*b^4*c^10 - 220*b^5*c^9 - 36*b^6*c^8 + 232*b^7*c^7 - 28*b^8*c^6 - 127*b^9*c^5 + 42*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 635*a*b^2*c^11 + 1300*a*b^3*c^10 + 608*a*b^4*c^9 - 1792*a*b^5*c^8 - 60*a*b^6*c^7 + 1218*a*b^7*c^6 - 249*a*b^8*c^5 - 340*a*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^11*c^2 - 1616*a^2*b*c^11 - 3160*a^3*b*c^10 - 3440*a^4*b*c^9 - 2132*a^5*b*c^8 - 704*a^6*b*c^7 - 96*a^7*b*c^6 - 2242*a^2*b^2*c^10 + 4146*a^2*b^3*c^9 + 1420*a^2*b^4*c^8 - 4158*a^2*b^5*c^7 + 77*a^2*b^6*c^6 + 1735*a^2*b^7*c^5 - 234*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 3714*a^3*b^2*c^9 + 6252*a^3*b^3*c^8 + 1730*a^3*b^4*c^7 - 4300*a^3*b^5*c^6 - 79*a^3*b^6*c^5 + 968*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 3523*a^4*b^2*c^8 + 5025*a^4*b^3*c^7 + 1339*a^4*b^4*c^6 - 2082*a^4*b^5*c^5 - 192*a^4*b^6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 2031*a^5*b^2*c^7 + 2104*a^5*b^3*c^6 + 634*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 676*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^2*c^5 - 404*a*b*c^12))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2

```
) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2)
)/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(20*a*b^
12 + 42*a*c^12 - 58*b*c^12 + 4*b^12*c - 4*b^13 + 22*c^13 - 40*a^2*b^11 + 40
*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 - 214*a^2*c^11 - 938*a^3*c^10 - 1538*a^4
*c^9 - 1278*a^5*c^8 - 498*a^6*c^7 - 14*a^7*c^6 + 52*a^8*c^5 + 12*a^9*c^4 +
14*b^2*c^11 + 34*b^3*c^10 + 59*b^4*c^9 - 39*b^5*c^8 - 160*b^6*c^7 + 112*b^7
*c^6 + 105*b^8*c^5 - 89*b^9*c^4 - 28*b^10*c^3 + 28*b^11*c^2 - 518*a*b^2*c^1
0 - 264*a*b^3*c^9 + 1339*a*b^4*c^8 - 92*a*b^5*c^7 - 1312*a*b^6*c^6 + 268*a*
b^7*c^5 + 649*a*b^8*c^4 - 124*a*b^9*c^3 - 180*a*b^10*c^2 + 1550*a^2*b*c^10
- 160*a^2*b^10*c + 3488*a^3*b*c^9 + 320*a^3*b^9*c + 3350*a^4*b*c^8 - 300*a^
4*b^8*c + 1092*a^5*b*c^7 + 136*a^5*b^7*c - 462*a^6*b*c^6 - 24*a^6*b^6*c - 4
40*a^7*b*c^5 - 92*a^8*b*c^4 - 1568*a^2*b^2*c^9 - 2708*a^2*b^3*c^8 + 3564*a^
2*b^4*c^7 + 1964*a^2*b^5*c^6 - 2790*a^2*b^6*c^5 - 922*a^2*b^7*c^4 + 1048*a^
2*b^8*c^3 + 276*a^2*b^9*c^2 - 652*a^3*b^2*c^8 - 6280*a^3*b^3*c^7 + 2020*a^3
*b^4*c^6 + 4988*a^3*b^5*c^5 - 1118*a^3*b^6*c^4 - 2008*a^3*b^7*c^3 + 140*a^3
*b^8*c^2 + 2350*a^4*b^2*c^7 - 5630*a^4*b^3*c^6 - 2295*a^4*b^4*c^5 + 3563*a^
4*b^5*c^4 + 1260*a^4*b^6*c^3 - 740*a^4*b^7*c^2 + 3314*a^5*b^2*c^6 - 1456*a^
5*b^3*c^5 - 2771*a^5*b^4*c^4 + 308*a^5*b^5*c^3 + 732*a^5*b^6*c^2 + 1572*a^6
*b^2*c^5 + 576*a^6*b^3*c^4 - 696*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 192*a^7*b^
2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 148*a*b*c^11 +
24*a*b^11*c))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 -
 b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^
2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 +
33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a
*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3
)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*
a*b^2*c^7)))^(1/2)*1i - (((((2048*(48*a*c^15 + 272*a^2*c^14 + 576*a^3*c^13
+ 576*a^4*c^12 + 272*a^5*c^11 + 48*a^6*c^10 - 12*b^2*c^14 + 20*b^3*c^13 + 1
8*b^4*c^12 - 46*b^5*c^11 + 6*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 - 140*a*b^2
*c^13 + 288*a*b^3*c^12 + 30*a*b^4*c^11 - 240*a*b^5*c^10 + 74*a*b^6*c^9 + 20
*a*b^7*c^8 - 416*a^2*b*c^13 - 736*a^3*b*c^12 - 544*a^4*b*c^11 - 144*a^5*b*c
^10 - 360*a^2*b^2*c^12 + 728*a^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c
^9 + 4*a^2*b^6*c^8 - 360*a^3*b^2*c^11 + 544*a^3*b^3*c^10 + 10*a^3*b^4*c^9 -
 20*a^3*b^5*c^8 - 172*a^4*b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a
^5*b^2*c^9 - 80*a*b*c^14))/c^8 + (2048*tan(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c
^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*
b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)
^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-
(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2)
- 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/
(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^15 -
128*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^
4*c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^
3*c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 3
52*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*
```

b^4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*
c^10 - 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 +
 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*
c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^
(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a
*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*
a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(
16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(24*b*c^14 - 9
6*a*c^14 - 8*c^15 + 152*a^2*c^13 + 952*a^3*c^12 + 1096*a^4*c^11 + 304*a^5*c
^10 - 152*a^6*c^9 - 72*a^7*c^8 + 2*b^2*c^13 - 38*b^3*c^12 - 7*b^4*c^11 + 39
*b^5*c^10 - 15*b^6*c^9 + 35*b^7*c^8 - 44*b^8*c^7 - 4*b^9*c^6 + 24*b^10*c^5
- 8*b^11*c^4 + 68*a*b^2*c^12 + 42*a*b^3*c^11 - 159*a*b^4*c^10 - 400*a*b^5*c
^9 + 537*a*b^6*c^8 + 68*a*b^7*c^7 - 276*a*b^8*c^6 + 72*a*b^9*c^5 + 8*a*b^10
*c^4 - 944*a^2*b*c^12 - 2520*a^3*b*c^11 - 1824*a^4*b*c^10 - 272*a^5*b*c^9 +
 88*a^6*b*c^8 + 584*a^2*b^2*c^11 + 1742*a^2*b^3*c^10 - 1645*a^2*b^4*c^9 - 7
95*a^2*b^5*c^8 + 1132*a^2*b^6*c^7 - 112*a^2*b^7*c^6 - 112*a^2*b^8*c^5 + 8*a
^2*b^9*c^4 + 476*a^3*b^2*c^10 + 2766*a^3*b^3*c^9 - 1705*a^3*b^4*c^8 - 396*a
^3*b^5*c^7 + 456*a^3*b^6*c^6 - 56*a^3*b^7*c^5 - 8*a^3*b^8*c^4 + 230*a^4*b^2
*c^9 + 880*a^4*b^3*c^8 - 656*a^4*b^4*c^7 + 140*a^4*b^5*c^6 + 72*a^4*b^6*c^5
 + 464*a^5*b^2*c^8 - 192*a^5*b^3*c^7 - 220*a^5*b^4*c^6 + 256*a^6*b^2*c^7 +
136*a*b*c^13))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4
- b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b
^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 +
 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*
a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^
3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8
*a*b^2*c^7)))^(1/2) + (2048*(236*a*c^13 - 32*b*c^13 + 12*c^14 + 1084*a^2*c^
12 + 2328*a^3*c^11 + 2784*a^4*c^10 + 1948*a^5*c^9 + 780*a^6*c^8 + 160*a^7*c
^7 + 12*a^8*c^6 - 39*b^2*c^12 + 121*b^3*c^11 + 61*b^4*c^10 - 220*b^5*c^9 -
36*b^6*c^8 + 232*b^7*c^7 - 28*b^8*c^6 - 127*b^9*c^5 + 42*b^10*c^4 + 26*b^11
*c^3 - 12*b^12*c^2 - 635*a*b^2*c^11 + 1300*a*b^3*c^10 + 608*a*b^4*c^9 - 179
2*a*b^5*c^8 - 60*a*b^6*c^7 + 1218*a*b^7*c^6 - 249*a*b^8*c^5 - 340*a*b^9*c^4
 + 98*a*b^10*c^3 + 20*a*b^11*c^2 - 1616*a^2*b*c^11 - 3160*a^3*b*c^10 - 3440
*a^4*b*c^9 - 2132*a^5*b*c^8 - 704*a^6*b*c^7 - 96*a^7*b*c^6 - 2242*a^2*b^2*c
^10 + 4146*a^2*b^3*c^9 + 1420*a^2*b^4*c^8 - 4158*a^2*b^5*c^7 + 77*a^2*b^6*c
^6 + 1735*a^2*b^7*c^5 - 234*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2
- 3714*a^3*b^2*c^9 + 6252*a^3*b^3*c^8 + 1730*a^3*b^4*c^7 - 4300*a^3*b^5*c^6
 - 79*a^3*b^6*c^5 + 968*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 3523
*a^4*b^2*c^8 + 5025*a^4*b^3*c^7 + 1339*a^4*b^4*c^6 - 2082*a^4*b^5*c^5 - 192
*a^4*b^6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 2031*a^5*b^2*c^7 + 2104*a^
5*b^3*c^6 + 634*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 676*a^6*b^
2*c^6 + 364*a^6*b^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^2*c^5 - 404*a*b*c^12)
)/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*
c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a
*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c

^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a
^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*
a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))
^(1/2) + (2048*tan(x/2)*(20*a*b^12 + 42*a*c^12 - 58*b*c^12 + 4*b^12*c - 4*b
^13 + 22*c^13 - 40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 - 214*a^
2*c^11 - 938*a^3*c^10 - 1538*a^4*c^9 - 1278*a^5*c^8 - 498*a^6*c^7 - 14*a^7*
c^6 + 52*a^8*c^5 + 12*a^9*c^4 + 14*b^2*c^11 + 34*b^3*c^10 + 59*b^4*c^9 - 39
*b^5*c^8 - 160*b^6*c^7 + 112*b^7*c^6 + 105*b^8*c^5 - 89*b^9*c^4 - 28*b^10*c
^3 + 28*b^11*c^2 - 518*a*b^2*c^10 - 264*a*b^3*c^9 + 1339*a*b^4*c^8 - 92*a*b
^5*c^7 - 1312*a*b^6*c^6 + 268*a*b^7*c^5 + 649*a*b^8*c^4 - 124*a*b^9*c^3 - 1
80*a*b^10*c^2 + 1550*a^2*b*c^10 - 160*a^2*b^10*c + 3488*a^3*b*c^9 + 320*a^3
*b^9*c + 3350*a^4*b*c^8 - 300*a^4*b^8*c + 1092*a^5*b*c^7 + 136*a^5*b^7*c -
462*a^6*b*c^6 - 24*a^6*b^6*c - 440*a^7*b*c^5 - 92*a^8*b*c^4 - 1568*a^2*b^2*
c^9 - 2708*a^2*b^3*c^8 + 3564*a^2*b^4*c^7 + 1964*a^2*b^5*c^6 - 2790*a^2*b^6
*c^5 - 922*a^2*b^7*c^4 + 1048*a^2*b^8*c^3 + 276*a^2*b^9*c^2 - 652*a^3*b^2*c
^8 - 6280*a^3*b^3*c^7 + 2020*a^3*b^4*c^6 + 4988*a^3*b^5*c^5 - 1118*a^3*b^6*
c^4 - 2008*a^3*b^7*c^3 + 140*a^3*b^8*c^2 + 2350*a^4*b^2*c^7 - 5630*a^4*b^3*
c^6 - 2295*a^4*b^4*c^5 + 3563*a^4*b^5*c^4 + 1260*a^4*b^6*c^3 - 740*a^4*b^7*
c^2 + 3314*a^5*b^2*c^6 - 1456*a^5*b^3*c^5 - 2771*a^5*b^4*c^4 + 308*a^5*b^5*
c^3 + 732*a^5*b^6*c^2 + 1572*a^6*b^2*c^5 + 576*a^6*b^3*c^4 - 696*a^6*b^4*c^
3 - 300*a^6*b^5*c^2 + 192*a^7*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 -
32*a^8*b^2*c^3 + 148*a*b*c^11 + 24*a*b^11*c))/c^8)*(-(8*a*c^7 + b^8 + 24*a^
2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 +
3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b
^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2
*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/
2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2
))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2)*1i)/((4096*(16*a*b^11 +
274*a*c^11 - 78*b*c^11 + 4*b^11*c - 4*b^12 + 33*c^12 - 16*a^2*b^10 - 16*a^3
*b^9 + 40*a^4*b^8 - 16*a^5*b^7 - 16*a^6*b^6 + 16*a^7*b^5 - 4*a^8*b^4 + 1008
*a^2*c^10 + 2156*a^3*c^9 + 2954*a^4*c^8 + 2688*a^5*c^7 + 1624*a^6*c^6 + 628
*a^7*c^5 + 141*a^8*c^4 + 14*a^9*c^3 - 64*b^2*c^10 + 268*b^3*c^9 - 26*b^4*c^
8 - 348*b^5*c^7 + 144*b^6*c^6 + 208*b^7*c^5 - 123*b^8*c^4 - 54*b^9*c^3 + 40
*b^10*c^2 - 520*a*b^2*c^9 + 1516*a*b^3*c^8 + 144*a*b^4*c^7 - 1564*a*b^5*c^6
+ 228*a*b^6*c^5 + 740*a*b^7*c^4 - 146*a*b^8*c^3 - 164*a*b^9*c^2 - 1624*a^2
*b*c^9 - 112*a^2*b^9*c - 2676*a^3*b*c^8 + 128*a^3*b^8*c - 2588*a^4*b*c^7 +
56*a^4*b^7*c - 1388*a^5*b*c^6 - 184*a^5*b^6*c - 264*a^6*b*c^5 + 80*a^6*b^5*
c + 116*a^7*b*c^4 + 32*a^7*b^4*c + 74*a^8*b*c^3 - 28*a^8*b^3*c + 12*a^9*b*c
^2 + 4*a^9*b^2*c - 1820*a^2*b^2*c^8 + 3576*a^2*b^3*c^7 + 1032*a^2*b^4*c^6 -
2792*a^2*b^5*c^5 - 236*a^2*b^6*c^4 + 920*a^2*b^7*c^3 + 64*a^2*b^8*c^2 - 35
84*a^3*b^2*c^7 + 4472*a^3*b^3*c^6 + 2236*a^3*b^4*c^5 - 2436*a^3*b^5*c^4 - 7
44*a^3*b^6*c^3 + 464*a^3*b^7*c^2 - 4336*a^4*b^2*c^6 + 3040*a^4*b^3*c^5 + 23
90*a^4*b^4*c^4 - 964*a^4*b^5*c^3 - 592*a^4*b^6*c^2 - 3284*a^5*b^2*c^5 + 908
*a^5*b^3*c^4 + 1364*a^5*b^4*c^3 - 40*a^5*b^5*c^2 - 1500*a^6*b^2*c^4 - 104*a
^6*b^3*c^3 + 384*a^6*b^4*c^2 - 360*a^7*b^2*c^3 - 144*a^7*b^3*c^2 - 24*a^8*b

^2*c^2 - 544*a*b*c^10 + 20*a*b^10*c))/c^8 + (((((2048*(48*a*c^15 + 272*a^2*
c^14 + 576*a^3*c^13 + 576*a^4*c^12 + 272*a^5*c^11 + 48*a^6*c^10 - 12*b^2*c^
14 + 20*b^3*c^13 + 18*b^4*c^12 - 46*b^5*c^11 + 6*b^6*c^10 + 26*b^7*c^9 - 12
*b^8*c^8 - 140*a*b^2*c^13 + 288*a*b^3*c^12 + 30*a*b^4*c^11 - 240*a*b^5*c^10
 + 74*a*b^6*c^9 + 20*a*b^7*c^8 - 416*a^2*b*c^13 - 736*a^3*b*c^12 - 544*a^4*
b*c^11 - 144*a^5*b*c^10 - 360*a^2*b^2*c^12 + 728*a^2*b^3*c^11 - 50*a^2*b^4*
c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 360*a^3*b^2*c^11 + 544*a^3*b^3*c^1
0 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 172*a^4*b^2*c^10 + 116*a^4*b^3*c^9 +
8*a^4*b^4*c^8 - 44*a^5*b^2*c^9 - 80*a*b*c^14))/c^8 - (2048*tan(x/2)*(-(8*a*
c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1
/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b
c^4(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^
2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*
a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a
c - b^2)^3)^(1/2))/(2(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2)*(32*a*c
^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 +
 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a
*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10
+ 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^
2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b
^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^8)*(-(8*a*c^7
+ b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2)
- 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4
*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^
3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c
- b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c -
 b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) + (2048*tan
(x/2)*(24*b*c^14 - 96*a*c^14 - 8*c^15 + 152*a^2*c^13 + 952*a^3*c^12 + 1096*
a^4*c^11 + 304*a^5*c^10 - 152*a^6*c^9 - 72*a^7*c^8 + 2*b^2*c^13 - 38*b^3*c^
12 - 7*b^4*c^11 + 39*b^5*c^10 - 15*b^6*c^9 + 35*b^7*c^8 - 44*b^8*c^7 - 4*b^
9*c^6 + 24*b^10*c^5 - 8*b^11*c^4 + 68*a*b^2*c^12 + 42*a*b^3*c^11 - 159*a*b^
4*c^10 - 400*a*b^5*c^9 + 537*a*b^6*c^8 + 68*a*b^7*c^7 - 276*a*b^8*c^6 + 72*
a*b^9*c^5 + 8*a*b^10*c^4 - 944*a^2*b*c^12 - 2520*a^3*b*c^11 - 1824*a^4*b*c^
10 - 272*a^5*b*c^9 + 88*a^6*b*c^8 + 584*a^2*b^2*c^11 + 1742*a^2*b^3*c^10 -
1645*a^2*b^4*c^9 - 795*a^2*b^5*c^8 + 1132*a^2*b^6*c^7 - 112*a^2*b^7*c^6 - 1
12*a^2*b^8*c^5 + 8*a^2*b^9*c^4 + 476*a^3*b^2*c^10 + 2766*a^3*b^3*c^9 - 1705
*a^3*b^4*c^8 - 396*a^3*b^5*c^7 + 456*a^3*b^6*c^6 - 56*a^3*b^7*c^5 - 8*a^3*b
^8*c^4 + 230*a^4*b^2*c^9 + 880*a^4*b^3*c^8 - 656*a^4*b^4*c^7 + 140*a^4*b^5*
c^6 + 72*a^4*b^6*c^5 + 464*a^5*b^2*c^8 - 192*a^5*b^3*c^7 - 220*a^5*b^4*c^6
+ 256*a^6*b^2*c^7 + 136*a*b*c^13))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*
a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4
- 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2
) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c -
 b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*
c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a

^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) + (2048*(236*a*c^13 - 32*b*c^13 + 1
2*c^14 + 1084*a^2*c^12 + 2328*a^3*c^11 + 2784*a^4*c^10 + 1948*a^5*c^9 + 780
*a^6*c^8 + 160*a^7*c^7 + 12*a^8*c^6 - 39*b^2*c^12 + 121*b^3*c^11 + 61*b^4*c
^10 - 220*b^5*c^9 - 36*b^6*c^8 + 232*b^7*c^7 - 28*b^8*c^6 - 127*b^9*c^5 + 4
2*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 635*a*b^2*c^11 + 1300*a*b^3*c^10 +
608*a*b^4*c^9 - 1792*a*b^5*c^8 - 60*a*b^6*c^7 + 1218*a*b^7*c^6 - 249*a*b^8
*c^5 - 340*a*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^11*c^2 - 1616*a^2*b*c^11 - 31
60*a^3*b*c^10 - 3440*a^4*b*c^9 - 2132*a^5*b*c^8 - 704*a^6*b*c^7 - 96*a^7*b*
c^6 - 2242*a^2*b^2*c^10 + 4146*a^2*b^3*c^9 + 1420*a^2*b^4*c^8 - 4158*a^2*b^
5*c^7 + 77*a^2*b^6*c^6 + 1735*a^2*b^7*c^5 - 234*a^2*b^8*c^4 - 222*a^2*b^9*c
^3 + 4*a^2*b^10*c^2 - 3714*a^3*b^2*c^9 + 6252*a^3*b^3*c^8 + 1730*a^3*b^4*c^
7 - 4300*a^3*b^5*c^6 - 79*a^3*b^6*c^5 + 968*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 2
0*a^3*b^9*c^2 - 3523*a^4*b^2*c^8 + 5025*a^4*b^3*c^7 + 1339*a^4*b^4*c^6 - 20
82*a^4*b^5*c^5 - 192*a^4*b^6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 2031*a
^5*b^2*c^7 + 2104*a^5*b^3*c^6 + 634*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*
b^6*c^3 - 676*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^2
*c^5 - 404*a*b*c^12))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a
^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 -
18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^
2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2
) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c
- b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*
c^6 - 8*a*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(20*a*b^12 + 42*a*c^12 - 58*b*c
^12 + 4*b^12*c - 4*b^13 + 22*c^13 - 40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9
+ 4*a^5*b^8 - 214*a^2*c^11 - 938*a^3*c^10 - 1538*a^4*c^9 - 1278*a^5*c^8 - 4
98*a^6*c^7 - 14*a^7*c^6 + 52*a^8*c^5 + 12*a^9*c^4 + 14*b^2*c^11 + 34*b^3*c^
10 + 59*b^4*c^9 - 39*b^5*c^8 - 160*b^6*c^7 + 112*b^7*c^6 + 105*b^8*c^5 - 89
*b^9*c^4 - 28*b^10*c^3 + 28*b^11*c^2 - 518*a*b^2*c^10 - 264*a*b^3*c^9 + 133
9*a*b^4*c^8 - 92*a*b^5*c^7 - 1312*a*b^6*c^6 + 268*a*b^7*c^5 + 649*a*b^8*c^4
- 124*a*b^9*c^3 - 180*a*b^10*c^2 + 1550*a^2*b*c^10 - 160*a^2*b^10*c + 3488
*a^3*b*c^9 + 320*a^3*b^9*c + 3350*a^4*b*c^8 - 300*a^4*b^8*c + 1092*a^5*b*c^
7 + 136*a^5*b^7*c - 462*a^6*b*c^6 - 24*a^6*b^6*c - 440*a^7*b*c^5 - 92*a^8*b
*c^4 - 1568*a^2*b^2*c^9 - 2708*a^2*b^3*c^8 + 3564*a^2*b^4*c^7 + 1964*a^2*b^
5*c^6 - 2790*a^2*b^6*c^5 - 922*a^2*b^7*c^4 + 1048*a^2*b^8*c^3 + 276*a^2*b^9
*c^2 - 652*a^3*b^2*c^8 - 6280*a^3*b^3*c^7 + 2020*a^3*b^4*c^6 + 4988*a^3*b^5
*c^5 - 1118*a^3*b^6*c^4 - 2008*a^3*b^7*c^3 + 140*a^3*b^8*c^2 + 2350*a^4*b^2
*c^7 - 5630*a^4*b^3*c^6 - 2295*a^4*b^4*c^5 + 3563*a^4*b^5*c^4 + 1260*a^4*b^
6*c^3 - 740*a^4*b^7*c^2 + 3314*a^5*b^2*c^6 - 1456*a^5*b^3*c^5 - 2771*a^5*b^
4*c^4 + 308*a^5*b^5*c^3 + 732*a^5*b^6*c^2 + 1572*a^6*b^2*c^5 + 576*a^6*b^3*
c^4 - 696*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 192*a^7*b^2*c^4 + 272*a^7*b^3*c^3
+ 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 148*a*b*c^11 + 24*a*b^11*c))/c^8)*(-(8
*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)
^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 -
3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3
*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-

```
(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(
4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) + ((
(((2048*(48*a*c^15 + 272*a^2*c^14 + 576*a^3*c^13 + 576*a^4*c^12 + 272*a^5*c
^11 + 48*a^6*c^10 - 12*b^2*c^14 + 20*b^3*c^13 + 18*b^4*c^12 - 46*b^5*c^11 +
 6*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 - 140*a*b^2*c^13 + 288*a*b^3*c^12 + 3
0*a*b^4*c^11 - 240*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 - 416*a^2*b*c^1
3 - 736*a^3*b*c^12 - 544*a^4*b*c^11 - 144*a^5*b*c^10 - 360*a^2*b^2*c^12 + 7
28*a^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 360*a
^3*b^2*c^11 + 544*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 172*a^4*
b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9 - 80*a*b*c^14))
/c^8 + (2048*tan(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^
4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a
*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4
 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 1
0*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2
)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 -
 8*a*b^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13
 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*
b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 -
56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c
^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10
- 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 9
6*a*b*c^15))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 -
b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2
*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 3
3*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*
b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)
^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a
*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(24*b*c^14 - 96*a*c^14 - 8*c^15 + 152*a^
2*c^13 + 952*a^3*c^12 + 1096*a^4*c^11 + 304*a^5*c^10 - 152*a^6*c^9 - 72*a^7
*c^8 + 2*b^2*c^13 - 38*b^3*c^12 - 7*b^4*c^11 + 39*b^5*c^10 - 15*b^6*c^9 + 3
5*b^7*c^8 - 44*b^8*c^7 - 4*b^9*c^6 + 24*b^10*c^5 - 8*b^11*c^4 + 68*a*b^2*c^
12 + 42*a*b^3*c^11 - 159*a*b^4*c^10 - 400*a*b^5*c^9 + 537*a*b^6*c^8 + 68*a*
b^7*c^7 - 276*a*b^8*c^6 + 72*a*b^9*c^5 + 8*a*b^10*c^4 - 944*a^2*b*c^12 - 25
20*a^3*b*c^11 - 1824*a^4*b*c^10 - 272*a^5*b*c^9 + 88*a^6*b*c^8 + 584*a^2*b^
2*c^11 + 1742*a^2*b^3*c^10 - 1645*a^2*b^4*c^9 - 795*a^2*b^5*c^8 + 1132*a^2*
b^6*c^7 - 112*a^2*b^7*c^6 - 112*a^2*b^8*c^5 + 8*a^2*b^9*c^4 + 476*a^3*b^2*c
^10 + 2766*a^3*b^3*c^9 - 1705*a^3*b^4*c^8 - 396*a^3*b^5*c^7 + 456*a^3*b^6*c
^6 - 56*a^3*b^7*c^5 - 8*a^3*b^8*c^4 + 230*a^4*b^2*c^9 + 880*a^4*b^3*c^8 - 6
56*a^4*b^4*c^7 + 140*a^4*b^5*c^6 + 72*a^4*b^6*c^5 + 464*a^5*b^2*c^8 - 192*a
^5*b^3*c^7 - 220*a^5*b^4*c^6 + 256*a^6*b^2*c^7 + 136*a*b*c^13))/c^8)*(-(8*a
*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(
1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*
b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b
^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4
```

```
*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*
a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) + (204
8*(236*a*c^13 - 32*b*c^13 + 12*c^14 + 1084*a^2*c^12 + 2328*a^3*c^11 + 2784*
a^4*c^10 + 1948*a^5*c^9 + 780*a^6*c^8 + 160*a^7*c^7 + 12*a^8*c^6 - 39*b^2*c
^12 + 121*b^3*c^11 + 61*b^4*c^10 - 220*b^5*c^9 - 36*b^6*c^8 + 232*b^7*c^7 -
 28*b^8*c^6 - 127*b^9*c^5 + 42*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 635*a
*b^2*c^11 + 1300*a*b^3*c^10 + 608*a*b^4*c^9 - 1792*a*b^5*c^8 - 60*a*b^6*c^7
 + 1218*a*b^7*c^6 - 249*a*b^8*c^5 - 340*a*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^
11*c^2 - 1616*a^2*b*c^11 - 3160*a^3*b*c^10 - 3440*a^4*b*c^9 - 2132*a^5*b*c^
8 - 704*a^6*b*c^7 - 96*a^7*b*c^6 - 2242*a^2*b^2*c^10 + 4146*a^2*b^3*c^9 + 1
420*a^2*b^4*c^8 - 4158*a^2*b^5*c^7 + 77*a^2*b^6*c^6 + 1735*a^2*b^7*c^5 - 23
4*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 3714*a^3*b^2*c^9 + 6252*
a^3*b^3*c^8 + 1730*a^3*b^4*c^7 - 4300*a^3*b^5*c^6 - 79*a^3*b^6*c^5 + 968*a^
3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 3523*a^4*b^2*c^8 + 5025*a^4*b^
3*c^7 + 1339*a^4*b^4*c^6 - 2082*a^4*b^5*c^5 - 192*a^4*b^6*c^4 + 156*a^4*b^7
*c^3 + 8*a^4*b^8*c^2 - 2031*a^5*b^2*c^7 + 2104*a^5*b^3*c^6 + 634*a^5*b^4*c^
5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 676*a^6*b^2*c^6 + 364*a^6*b^3*c^5 +
136*a^6*b^4*c^4 - 100*a^7*b^2*c^5 - 404*a*b*c^12))/c^8)*(-(8*a*c^7 + b^8 +
24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*
c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*
c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^
3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3
)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)
^(1/2))/(2*(16*a^2*c^8 + b^4*c^6 - 8*a*b^2*c^7)))^(1/2) + (2048*tan(x/2)*(2
0*a*b^12 + 42*a*c^12 - 58*b*c^12 + 4*b^12*c - 4*b^13 + 22*c^13 - 40*a^2*b^1
1 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 - 214*a^2*c^11 - 938*a^3*c^10 - 15
38*a^4*c^9 - 1278*a^5*c^8 - 498*a^6*c^7 - 14*a^7*c^6 + 52*a^8*c^5 + 12*a^9*
c^4 + 14*b^2*c^11 + 34*b^3*c^10 + 59*b^4*c^9 - 39*b^5*c^8 - 160*b^6*c^7 + 1
12*b^7*c^6 + 105*b^8*c^5 - 89*b^9*c^4 - 28*b^10*c^3 + 28*b^11*c^2 - 518*a*b
^2*c^10 - 264*a*b^3*c^9 + 1339*a*b^4*c^8 - 92*a*b^5*c^7 - 1312*a*b^6*c^6 +
268*a*b^7*c^5 + 649*a*b^8*c^4 - 124*a*b^9*c^3 - 180*a*b^10*c^2 + 1550*a^2*b
*c^10 - 160*a^2*b^10*c + 3488*a^3*b*c^9 + 320*a^3*b^9*c + 3350*a^4*b*c^8 -
300*a^4*b^8*c + 1092*a^5*b*c^7 + 136*a^5*b^7*c - 462*a^6*b*c^6 - 24*a^6*b^6
*c - 440*a^7*b*c^5 - 92*a^8*b*c^4 - 1568*a^2*b^2*c^9 - 2708*a^2*b^3*c^8 + 3
564*a^2*b^4*c^7 + 1964*a^2*b^5*c^6 - 2790*a^2*b^6*c^5 - 922*a^2*b^7*c^4 + 1
048*a^2*b^8*c^3 + 276*a^2*b^9*c^2 - 652*a^3*b^2*c^8 - 6280*a^3*b^3*c^7 + 20
20*a^3*b^4*c^6 + 4988*a^3*b^5*c^5 - 1118*a^3*b^6*c^4 - 2008*a^3*b^7*c^3 + 1
40*a^3*b^8*c^2 + 2350*a^4*b^2*c^7 - 5630*a^4*b^3*c^6 - 2295*a^4*b^4*c^5 + 3
563*a^4*b^5*c^4 + 1260*a^4*b^6*c^3 - 740*a^4*b^7*c^2 + 3314*a^5*b^2*c^6 - 1
456*a^5*b^3*c^5 - 2771*a^5*b^4*c^4 + 308*a^5*b^5*c^3 + 732*a^5*b^6*c^2 + 15
72*a^6*b^2*c^5 + 576*a^6*b^3*c^4 - 696*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 192*
a^7*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 148*a*b*c
^11 + 24*a*b^11*c))/c^8)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4
*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 1
8*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*
```

c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2)
- 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c -
b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*c^
6 - 8*a*b^2*c^7)))^(1/2)))*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a
^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 -
18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^
2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2
) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c
- b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + b^4*
c^6 - 8*a*b^2*c^7)))^(1/2)*2i + (atan((((((2048*(236*a*c^13 - 32*b*c^13 +
12*c^14 + 1084*a^2*c^12 + 2328*a^3*c^11 + 2784*a^4*c^10 + 1948*a^5*c^9 + 78
0*a^6*c^8 + 160*a^7*c^7 + 12*a^8*c^6 - 39*b^2*c^12 + 121*b^3*c^11 + 61*b^4*
c^10 - 220*b^5*c^9 - 36*b^6*c^8 + 232*b^7*c^7 - 28*b^8*c^6 - 127*b^9*c^5 +
42*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 635*a*b^2*c^11 + 1300*a*b^3*c^10
+ 608*a*b^4*c^9 - 1792*a*b^5*c^8 - 60*a*b^6*c^7 + 1218*a*b^7*c^6 - 249*a*b^
8*c^5 - 340*a*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^11*c^2 - 1616*a^2*b*c^11 - 3
160*a^3*b*c^10 - 3440*a^4*b*c^9 - 2132*a^5*b*c^8 - 704*a^6*b*c^7 - 96*a^7*b
*c^6 - 2242*a^2*b^2*c^10 + 4146*a^2*b^3*c^9 + 1420*a^2*b^4*c^8 - 4158*a^2*b
^5*c^7 + 77*a^2*b^6*c^6 + 1735*a^2*b^7*c^5 - 234*a^2*b^8*c^4 - 222*a^2*b^9*
c^3 + 4*a^2*b^10*c^2 - 3714*a^3*b^2*c^9 + 6252*a^3*b^3*c^8 + 1730*a^3*b^4*c
^7 - 4300*a^3*b^5*c^6 - 79*a^3*b^6*c^5 + 968*a^3*b^7*c^4 + 2*a^3*b^8*c^3 -
20*a^3*b^9*c^2 - 3523*a^4*b^2*c^8 + 5025*a^4*b^3*c^7 + 1339*a^4*b^4*c^6 - 2
082*a^4*b^5*c^5 - 192*a^4*b^6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 2031*
a^5*b^2*c^7 + 2104*a^5*b^3*c^6 + 634*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5
*b^6*c^3 - 676*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^
2*c^5 - 404*a*b*c^12))/c^8 + (((2048*tan(x/2)*(24*b*c^14 - 96*a*c^14 - 8*c^
15 + 152*a^2*c^13 + 952*a^3*c^12 + 1096*a^4*c^11 + 304*a^5*c^10 - 152*a^6*c
^9 - 72*a^7*c^8 + 2*b^2*c^13 - 38*b^3*c^12 - 7*b^4*c^11 + 39*b^5*c^10 - 15*
b^6*c^9 + 35*b^7*c^8 - 44*b^8*c^7 - 4*b^9*c^6 + 24*b^10*c^5 - 8*b^11*c^4 +
68*a*b^2*c^12 + 42*a*b^3*c^11 - 159*a*b^4*c^10 - 400*a*b^5*c^9 + 537*a*b^6*
c^8 + 68*a*b^7*c^7 - 276*a*b^8*c^6 + 72*a*b^9*c^5 + 8*a*b^10*c^4 - 944*a^2*
b*c^12 - 2520*a^3*b*c^11 - 1824*a^4*b*c^10 - 272*a^5*b*c^9 + 88*a^6*b*c^8 +
584*a^2*b^2*c^11 + 1742*a^2*b^3*c^10 - 1645*a^2*b^4*c^9 - 795*a^2*b^5*c^8
+ 1132*a^2*b^6*c^7 - 112*a^2*b^7*c^6 - 112*a^2*b^8*c^5 + 8*a^2*b^9*c^4 + 47
6*a^3*b^2*c^10 + 2766*a^3*b^3*c^9 - 1705*a^3*b^4*c^8 - 396*a^3*b^5*c^7 + 45
6*a^3*b^6*c^6 - 56*a^3*b^7*c^5 - 8*a^3*b^8*c^4 + 230*a^4*b^2*c^9 + 880*a^4*
b^3*c^8 - 656*a^4*b^4*c^7 + 140*a^4*b^5*c^6 + 72*a^4*b^6*c^5 + 464*a^5*b^2*
c^8 - 192*a^5*b^3*c^7 - 220*a^5*b^4*c^6 + 256*a^6*b^2*c^7 + 136*a*b*c^13))/
c^8 + (((2048*(48*a*c^15 + 272*a^2*c^14 + 576*a^3*c^13 + 576*a^4*c^12 + 272
*a^5*c^11 + 48*a^6*c^10 - 12*b^2*c^14 + 20*b^3*c^13 + 18*b^4*c^12 - 46*b^5*
c^11 + 6*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 - 140*a*b^2*c^13 + 288*a*b^3*c^
12 + 30*a*b^4*c^11 - 240*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 - 416*a^2
*b*c^13 - 736*a^3*b*c^12 - 544*a^4*b*c^11 - 144*a^5*b*c^10 - 360*a^2*b^2*c^
12 + 728*a^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 -
360*a^3*b^2*c^11 + 544*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 17

2*a^4*b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9 - 80*a*b*
c^14))/c^8 - (2048*tan(x/2)*(a*c*1i - b^2*1i + (c^2*3i)/2)*(32*a*c^16 - 64*
a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c
^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14
 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2
*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^1
2 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 +
 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^11)*(a*c*1i - b^2*1i +
(c^2*3i)/2))/c^3)*(a*c*1i - b^2*1i + (c^2*3i)/2))/c^3)*(a*c*1i - b^2*1i + (
c^2*3i)/2))/c^3 - (2048*tan(x/2)*(20*a*b^12 + 42*a*c^12 - 58*b*c^12 + 4*b^1
2*c - 4*b^13 + 22*c^13 - 40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8
 - 214*a^2*c^11 - 938*a^3*c^10 - 1538*a^4*c^9 - 1278*a^5*c^8 - 498*a^6*c^7
- 14*a^7*c^6 + 52*a^8*c^5 + 12*a^9*c^4 + 14*b^2*c^11 + 34*b^3*c^10 + 59*b^4
*c^9 - 39*b^5*c^8 - 160*b^6*c^7 + 112*b^7*c^6 + 105*b^8*c^5 - 89*b^9*c^4 -
28*b^10*c^3 + 28*b^11*c^2 - 518*a*b^2*c^10 - 264*a*b^3*c^9 + 1339*a*b^4*c^8
 - 92*a*b^5*c^7 - 1312*a*b^6*c^6 + 268*a*b^7*c^5 + 649*a*b^8*c^4 - 124*a*b^
9*c^3 - 180*a*b^10*c^2 + 1550*a^2*b*c^10 - 160*a^2*b^10*c + 3488*a^3*b*c^9
+ 320*a^3*b^9*c + 3350*a^4*b*c^8 - 300*a^4*b^8*c + 1092*a^5*b*c^7 + 136*a^5
*b^7*c - 462*a^6*b*c^6 - 24*a^6*b^6*c - 440*a^7*b*c^5 - 92*a^8*b*c^4 - 1568
*a^2*b^2*c^9 - 2708*a^2*b^3*c^8 + 3564*a^2*b^4*c^7 + 1964*a^2*b^5*c^6 - 279
0*a^2*b^6*c^5 - 922*a^2*b^7*c^4 + 1048*a^2*b^8*c^3 + 276*a^2*b^9*c^2 - 652*
a^3*b^2*c^8 - 6280*a^3*b^3*c^7 + 2020*a^3*b^4*c^6 + 4988*a^3*b^5*c^5 - 1118
*a^3*b^6*c^4 - 2008*a^3*b^7*c^3 + 140*a^3*b^8*c^2 + 2350*a^4*b^2*c^7 - 5630
*a^4*b^3*c^6 - 2295*a^4*b^4*c^5 + 3563*a^4*b^5*c^4 + 1260*a^4*b^6*c^3 - 740
*a^4*b^7*c^2 + 3314*a^5*b^2*c^6 - 1456*a^5*b^3*c^5 - 2771*a^5*b^4*c^4 + 308
*a^5*b^5*c^3 + 732*a^5*b^6*c^2 + 1572*a^6*b^2*c^5 + 576*a^6*b^3*c^4 - 696*a
^6*b^4*c^3 - 300*a^6*b^5*c^2 + 192*a^7*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b
^4*c^2 - 32*a^8*b^2*c^3 + 148*a*b*c^11 + 24*a*b^11*c))/c^8)*(a*c*1i - b^2*1
i + (c^2*3i)/2)*1i)/c^3 - (((((2048*(236*a*c^13 - 32*b*c^13 + 12*c^14 + 108
4*a^2*c^12 + 2328*a^3*c^11 + 2784*a^4*c^10 + 1948*a^5*c^9 + 780*a^6*c^8 + 1
60*a^7*c^7 + 12*a^8*c^6 - 39*b^2*c^12 + 121*b^3*c^11 + 61*b^4*c^10 - 220*b^
5*c^9 - 36*b^6*c^8 + 232*b^7*c^7 - 28*b^8*c^6 - 127*b^9*c^5 + 42*b^10*c^4 +
 26*b^11*c^3 - 12*b^12*c^2 - 635*a*b^2*c^11 + 1300*a*b^3*c^10 + 608*a*b^4*c
^9 - 1792*a*b^5*c^8 - 60*a*b^6*c^7 + 1218*a*b^7*c^6 - 249*a*b^8*c^5 - 340*a
*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^11*c^2 - 1616*a^2*b*c^11 - 3160*a^3*b*c^1
0 - 3440*a^4*b*c^9 - 2132*a^5*b*c^8 - 704*a^6*b*c^7 - 96*a^7*b*c^6 - 2242*a
^2*b^2*c^10 + 4146*a^2*b^3*c^9 + 1420*a^2*b^4*c^8 - 4158*a^2*b^5*c^7 + 77*a
^2*b^6*c^6 + 1735*a^2*b^7*c^5 - 234*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b
^10*c^2 - 3714*a^3*b^2*c^9 + 6252*a^3*b^3*c^8 + 1730*a^3*b^4*c^7 - 4300*a^3
*b^5*c^6 - 79*a^3*b^6*c^5 + 968*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^
2 - 3523*a^4*b^2*c^8 + 5025*a^4*b^3*c^7 + 1339*a^4*b^4*c^6 - 2082*a^4*b^5*c
^5 - 192*a^4*b^6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 2031*a^5*b^2*c^7 +
 2104*a^5*b^3*c^6 + 634*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 67
6*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^2*c^5 - 404*a
*b*c^12))/c^8 - (((2048*tan(x/2)*(24*b*c^14 - 96*a*c^14 - 8*c^15 + 152*a^2*

c^13 + 952*a^3*c^12 + 1096*a^4*c^11 + 304*a^5*c^10 - 152*a^6*c^9 - 72*a^7*c
^8 + 2*b^2*c^13 - 38*b^3*c^12 - 7*b^4*c^11 + 39*b^5*c^10 - 15*b^6*c^9 + 35*
b^7*c^8 - 44*b^8*c^7 - 4*b^9*c^6 + 24*b^10*c^5 - 8*b^11*c^4 + 68*a*b^2*c^12
 + 42*a*b^3*c^11 - 159*a*b^4*c^10 - 400*a*b^5*c^9 + 537*a*b^6*c^8 + 68*a*b^
7*c^7 - 276*a*b^8*c^6 + 72*a*b^9*c^5 + 8*a*b^10*c^4 - 944*a^2*b*c^12 - 2520
*a^3*b*c^11 - 1824*a^4*b*c^10 - 272*a^5*b*c^9 + 88*a^6*b*c^8 + 584*a^2*b^2*
c^11 + 1742*a^2*b^3*c^10 - 1645*a^2*b^4*c^9 - 795*a^2*b^5*c^8 + 1132*a^2*b^
6*c^7 - 112*a^2*b^7*c^6 - 112*a^2*b^8*c^5 + 8*a^2*b^9*c^4 + 476*a^3*b^2*c^1
0 + 2766*a^3*b^3*c^9 - 1705*a^3*b^4*c^8 - 396*a^3*b^5*c^7 + 456*a^3*b^6*c^6
 - 56*a^3*b^7*c^5 - 8*a^3*b^8*c^4 + 230*a^4*b^2*c^9 + 880*a^4*b^3*c^8 - 656
*a^4*b^4*c^7 + 140*a^4*b^5*c^6 + 72*a^4*b^6*c^5 + 464*a^5*b^2*c^8 - 192*a^5
*b^3*c^7 - 220*a^5*b^4*c^6 + 256*a^6*b^2*c^7 + 136*a*b*c^13))/c^8 - (((2048
*(48*a*c^15 + 272*a^2*c^14 + 576*a^3*c^13 + 576*a^4*c^12 + 272*a^5*c^11 + 4
8*a^6*c^10 - 12*b^2*c^14 + 20*b^3*c^13 + 18*b^4*c^12 - 46*b^5*c^11 + 6*b^6*
c^10 + 26*b^7*c^9 - 12*b^8*c^8 - 140*a*b^2*c^13 + 288*a*b^3*c^12 + 30*a*b^4
*c^11 - 240*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 - 416*a^2*b*c^13 - 736
*a^3*b*c^12 - 544*a^4*b*c^11 - 144*a^5*b*c^10 - 360*a^2*b^2*c^12 + 728*a^2*
b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 360*a^3*b^2*
c^11 + 544*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 172*a^4*b^2*c^1
0 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9 - 80*a*b*c^14))/c^8 +
(2048*tan(x/2)*(a*c*1i - b^2*1i + (c^2*3i)/2)*(32*a*c^16 - 64*a^2*c^15 - 12
8*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*
c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*
c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352
*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^
4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^
10 - 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^11)*(a*c*1i - b^2*1i + (c^2*3i)/2))/
c^3)*(a*c*1i - b^2*1i + (c^2*3i)/2))/c^3)*(a*c*1i - b^2*1i + (c^2*3i)/2))/c
^3 + (2048*tan(x/2)*(20*a*b^12 + 42*a*c^12 - 58*b*c^12 + 4*b^12*c - 4*b^13
+ 22*c^13 - 40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 - 214*a^2*c^
11 - 938*a^3*c^10 - 1538*a^4*c^9 - 1278*a^5*c^8 - 498*a^6*c^7 - 14*a^7*c^6
+ 52*a^8*c^5 + 12*a^9*c^4 + 14*b^2*c^11 + 34*b^3*c^10 + 59*b^4*c^9 - 39*b^5
*c^8 - 160*b^6*c^7 + 112*b^7*c^6 + 105*b^8*c^5 - 89*b^9*c^4 - 28*b^10*c^3 +
 28*b^11*c^2 - 518*a*b^2*c^10 - 264*a*b^3*c^9 + 1339*a*b^4*c^8 - 92*a*b^5*c
^7 - 1312*a*b^6*c^6 + 268*a*b^7*c^5 + 649*a*b^8*c^4 - 124*a*b^9*c^3 - 180*a
*b^10*c^2 + 1550*a^2*b*c^10 - 160*a^2*b^10*c + 3488*a^3*b*c^9 + 320*a^3*b^9
*c + 3350*a^4*b*c^8 - 300*a^4*b^8*c + 1092*a^5*b*c^7 + 136*a^5*b^7*c - 462*
a^6*b*c^6 - 24*a^6*b^6*c - 440*a^7*b*c^5 - 92*a^8*b*c^4 - 1568*a^2*b^2*c^9
- 2708*a^2*b^3*c^8 + 3564*a^2*b^4*c^7 + 1964*a^2*b^5*c^6 - 2790*a^2*b^6*c^5
 - 922*a^2*b^7*c^4 + 1048*a^2*b^8*c^3 + 276*a^2*b^9*c^2 - 652*a^3*b^2*c^8 -
 6280*a^3*b^3*c^7 + 2020*a^3*b^4*c^6 + 4988*a^3*b^5*c^5 - 1118*a^3*b^6*c^4
- 2008*a^3*b^7*c^3 + 140*a^3*b^8*c^2 + 2350*a^4*b^2*c^7 - 5630*a^4*b^3*c^6
- 2295*a^4*b^4*c^5 + 3563*a^4*b^5*c^4 + 1260*a^4*b^6*c^3 - 740*a^4*b^7*c^2
+ 3314*a^5*b^2*c^6 - 1456*a^5*b^3*c^5 - 2771*a^5*b^4*c^4 + 308*a^5*b^5*c^3
+ 732*a^5*b^6*c^2 + 1572*a^6*b^2*c^5 + 576*a^6*b^3*c^4 - 696*a^6*b^4*c^3 -

300*a^6*b^5*c^2 + 192*a^7*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a
^8*b^2*c^3 + 148*a*b*c^11 + 24*a*b^11*c))/c^8)*(a*c*1i - b^2*1i + (c^2*3i)/
2)*1i)/c^3)/((4096*(16*a*b^11 + 274*a*c^11 - 78*b*c^11 + 4*b^11*c - 4*b^12
+ 33*c^12 - 16*a^2*b^10 - 16*a^3*b^9 + 40*a^4*b^8 - 16*a^5*b^7 - 16*a^6*b^6
+ 16*a^7*b^5 - 4*a^8*b^4 + 1008*a^2*c^10 + 2156*a^3*c^9 + 2954*a^4*c^8 + 2
688*a^5*c^7 + 1624*a^6*c^6 + 628*a^7*c^5 + 141*a^8*c^4 + 14*a^9*c^3 - 64*b^
2*c^10 + 268*b^3*c^9 - 26*b^4*c^8 - 348*b^5*c^7 + 144*b^6*c^6 + 208*b^7*c^5
- 123*b^8*c^4 - 54*b^9*c^3 + 40*b^10*c^2 - 520*a*b^2*c^9 + 1516*a*b^3*c^8
+ 144*a*b^4*c^7 - 1564*a*b^5*c^6 + 228*a*b^6*c^5 + 740*a*b^7*c^4 - 146*a*b^
8*c^3 - 164*a*b^9*c^2 - 1624*a^2*b*c^9 - 112*a^2*b^9*c - 2676*a^3*b*c^8 + 1
28*a^3*b^8*c - 2588*a^4*b*c^7 + 56*a^4*b^7*c - 1388*a^5*b*c^6 - 184*a^5*b^6
*c - 264*a^6*b*c^5 + 80*a^6*b^5*c + 116*a^7*b*c^4 + 32*a^7*b^4*c + 74*a^8*b
*c^3 - 28*a^8*b^3*c + 12*a^9*b*c^2 + 4*a^9*b^2*c - 1820*a^2*b^2*c^8 + 3576*
a^2*b^3*c^7 + 1032*a^2*b^4*c^6 - 2792*a^2*b^5*c^5 - 236*a^2*b^6*c^4 + 920*a
^2*b^7*c^3 + 64*a^2*b^8*c^2 - 3584*a^3*b^2*c^7 + 4472*a^3*b^3*c^6 + 2236*a^
3*b^4*c^5 - 2436*a^3*b^5*c^4 - 744*a^3*b^6*c^3 + 464*a^3*b^7*c^2 - 4336*a^4
*b^2*c^6 + 3040*a^4*b^3*c^5 + 2390*a^4*b^4*c^4 - 964*a^4*b^5*c^3 - 592*a^4*
b^6*c^2 - 3284*a^5*b^2*c^5 + 908*a^5*b^3*c^4 + 1364*a^5*b^4*c^3 - 40*a^5*b^
5*c^2 - 1500*a^6*b^2*c^4 - 104*a^6*b^3*c^3 + 384*a^6*b^4*c^2 - 360*a^7*b^2*
c^3 - 144*a^7*b^3*c^2 - 24*a^8*b^2*c^2 - 544*a*b*c^10 + 20*a*b^10*c))/c^8 +
(((((2048*(236*a*c^13 - 32*b*c^13 + 12*c^14 + 1084*a^2*c^12 + 2328*a^3*c^1
1 + 2784*a^4*c^10 + 1948*a^5*c^9 + 780*a^6*c^8 + 160*a^7*c^7 + 12*a^8*c^6 -
39*b^2*c^12 + 121*b^3*c^11 + 61*b^4*c^10 - 220*b^5*c^9 - 36*b^6*c^8 + 232*
b^7*c^7 - 28*b^8*c^6 - 127*b^9*c^5 + 42*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^
2 - 635*a*b^2*c^11 + 1300*a*b^3*c^10 + 608*a*b^4*c^9 - 1792*a*b^5*c^8 - 60*
a*b^6*c^7 + 1218*a*b^7*c^6 - 249*a*b^8*c^5 - 340*a*b^9*c^4 + 98*a*b^10*c^3
+ 20*a*b^11*c^2 - 1616*a^2*b*c^11 - 3160*a^3*b*c^10 - 3440*a^4*b*c^9 - 2132
*a^5*b*c^8 - 704*a^6*b*c^7 - 96*a^7*b*c^6 - 2242*a^2*b^2*c^10 + 4146*a^2*b^
3*c^9 + 1420*a^2*b^4*c^8 - 4158*a^2*b^5*c^7 + 77*a^2*b^6*c^6 + 1735*a^2*b^7
*c^5 - 234*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 3714*a^3*b^2*c^
9 + 6252*a^3*b^3*c^8 + 1730*a^3*b^4*c^7 - 4300*a^3*b^5*c^6 - 79*a^3*b^6*c^5
+ 968*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 3523*a^4*b^2*c^8 + 50
25*a^4*b^3*c^7 + 1339*a^4*b^4*c^6 - 2082*a^4*b^5*c^5 - 192*a^4*b^6*c^4 + 15
6*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 2031*a^5*b^2*c^7 + 2104*a^5*b^3*c^6 + 634*a
^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 676*a^6*b^2*c^6 + 364*a^6*b
^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^2*c^5 - 404*a*b*c^12))/c^8 + (((2048*t
an(x/2)*(24*b*c^14 - 96*a*c^14 - 8*c^15 + 152*a^2*c^13 + 952*a^3*c^12 + 109
6*a^4*c^11 + 304*a^5*c^10 - 152*a^6*c^9 - 72*a^7*c^8 + 2*b^2*c^13 - 38*b^3*
c^12 - 7*b^4*c^11 + 39*b^5*c^10 - 15*b^6*c^9 + 35*b^7*c^8 - 44*b^8*c^7 - 4*
b^9*c^6 + 24*b^10*c^5 - 8*b^11*c^4 + 68*a*b^2*c^12 + 42*a*b^3*c^11 - 159*a*
b^4*c^10 - 400*a*b^5*c^9 + 537*a*b^6*c^8 + 68*a*b^7*c^7 - 276*a*b^8*c^6 + 7
2*a*b^9*c^5 + 8*a*b^10*c^4 - 944*a^2*b*c^12 - 2520*a^3*b*c^11 - 1824*a^4*b*
c^10 - 272*a^5*b*c^9 + 88*a^6*b*c^8 + 584*a^2*b^2*c^11 + 1742*a^2*b^3*c^10
- 1645*a^2*b^4*c^9 - 795*a^2*b^5*c^8 + 1132*a^2*b^6*c^7 - 112*a^2*b^7*c^6 -
112*a^2*b^8*c^5 + 8*a^2*b^9*c^4 + 476*a^3*b^2*c^10 + 2766*a^3*b^3*c^9 - 17

05*a^3*b^4*c^8 - 396*a^3*b^5*c^7 + 456*a^3*b^6*c^6 - 56*a^3*b^7*c^5 - 8*a^3
*b^8*c^4 + 230*a^4*b^2*c^9 + 880*a^4*b^3*c^8 - 656*a^4*b^4*c^7 + 140*a^4*b^
5*c^6 + 72*a^4*b^6*c^5 + 464*a^5*b^2*c^8 - 192*a^5*b^3*c^7 - 220*a^5*b^4*c^
6 + 256*a^6*b^2*c^7 + 136*a*b*c^13))/c^8 + (((2048*(48*a*c^15 + 272*a^2*c^1
4 + 576*a^3*c^13 + 576*a^4*c^12 + 272*a^5*c^11 + 48*a^6*c^10 - 12*b^2*c^14
+ 20*b^3*c^13 + 18*b^4*c^12 - 46*b^5*c^11 + 6*b^6*c^10 + 26*b^7*c^9 - 12*b^
8*c^8 - 140*a*b^2*c^13 + 288*a*b^3*c^12 + 30*a*b^4*c^11 - 240*a*b^5*c^10 +
74*a*b^6*c^9 + 20*a*b^7*c^8 - 416*a^2*b*c^13 - 736*a^3*b*c^12 - 544*a^4*b*c
^11 - 144*a^5*b*c^10 - 360*a^2*b^2*c^12 + 728*a^2*b^3*c^11 - 50*a^2*b^4*c^1
0 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 360*a^3*b^2*c^11 + 544*a^3*b^3*c^10 +
10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 172*a^4*b^2*c^10 + 116*a^4*b^3*c^9 + 8*a
^4*b^4*c^8 - 44*a^5*b^2*c^9 - 80*a*b*c^14))/c^8 - (2048*tan(x/2)*(a*c*1i -
b^2*1i + (c^2*3i)/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13
+ 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b
^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 5
6*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^
12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 -
272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 96
*a*b*c^15))/c^11)*(a*c*1i - b^2*1i + (c^2*3i)/2))/c^3)*(a*c*1i - b^2*1i + (
c^2*3i)/2))/c^3)*(a*c*1i - b^2*1i + (c^2*3i)/2))/c^3 - (2048*tan(x/2)*(20*a
*b^12 + 42*a*c^12 - 58*b*c^12 + 4*b^12*c - 4*b^13 + 22*c^13 - 40*a^2*b^11 +
40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 - 214*a^2*c^11 - 938*a^3*c^10 - 1538*
a^4*c^9 - 1278*a^5*c^8 - 498*a^6*c^7 - 14*a^7*c^6 + 52*a^8*c^5 + 12*a^9*c^4
+ 14*b^2*c^11 + 34*b^3*c^10 + 59*b^4*c^9 - 39*b^5*c^8 - 160*b^6*c^7 + 112*
b^7*c^6 + 105*b^8*c^5 - 89*b^9*c^4 - 28*b^10*c^3 + 28*b^11*c^2 - 518*a*b^2*
c^10 - 264*a*b^3*c^9 + 1339*a*b^4*c^8 - 92*a*b^5*c^7 - 1312*a*b^6*c^6 + 268
*a*b^7*c^5 + 649*a*b^8*c^4 - 124*a*b^9*c^3 - 180*a*b^10*c^2 + 1550*a^2*b*c^
10 - 160*a^2*b^10*c + 3488*a^3*b*c^9 + 320*a^3*b^9*c + 3350*a^4*b*c^8 - 300
*a^4*b^8*c + 1092*a^5*b*c^7 + 136*a^5*b^7*c - 462*a^6*b*c^6 - 24*a^6*b^6*c
- 440*a^7*b*c^5 - 92*a^8*b*c^4 - 1568*a^2*b^2*c^9 - 2708*a^2*b^3*c^8 + 3564
*a^2*b^4*c^7 + 1964*a^2*b^5*c^6 - 2790*a^2*b^6*c^5 - 922*a^2*b^7*c^4 + 1048
*a^2*b^8*c^3 + 276*a^2*b^9*c^2 - 652*a^3*b^2*c^8 - 6280*a^3*b^3*c^7 + 2020*
a^3*b^4*c^6 + 4988*a^3*b^5*c^5 - 1118*a^3*b^6*c^4 - 2008*a^3*b^7*c^3 + 140*
a^3*b^8*c^2 + 2350*a^4*b^2*c^7 - 5630*a^4*b^3*c^6 - 2295*a^4*b^4*c^5 + 3563
*a^4*b^5*c^4 + 1260*a^4*b^6*c^3 - 740*a^4*b^7*c^2 + 3314*a^5*b^2*c^6 - 1456
*a^5*b^3*c^5 - 2771*a^5*b^4*c^4 + 308*a^5*b^5*c^3 + 732*a^5*b^6*c^2 + 1572*
a^6*b^2*c^5 + 576*a^6*b^3*c^4 - 696*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 192*a^7
*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 148*a*b*c^11
+ 24*a*b^11*c))/c^8)*(a*c*1i - b^2*1i + (c^2*3i)/2))/c^3 + (((((2048*(236*
a*c^13 - 32*b*c^13 + 12*c^14 + 1084*a^2*c^12 + 2328*a^3*c^11 + 2784*a^4*c^1
0 + 1948*a^5*c^9 + 780*a^6*c^8 + 160*a^7*c^7 + 12*a^8*c^6 - 39*b^2*c^12 + 1
21*b^3*c^11 + 61*b^4*c^10 - 220*b^5*c^9 - 36*b^6*c^8 + 232*b^7*c^7 - 28*b^8
*c^6 - 127*b^9*c^5 + 42*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 635*a*b^2*c^
11 + 1300*a*b^3*c^10 + 608*a*b^4*c^9 - 1792*a*b^5*c^8 - 60*a*b^6*c^7 + 1218
*a*b^7*c^6 - 249*a*b^8*c^5 - 340*a*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^11*c^2

- 1616*a^2*b*c^11 - 3160*a^3*b*c^10 - 3440*a^4*b*c^9 - 2132*a^5*b*c^8 - 704
*a^6*b*c^7 - 96*a^7*b*c^6 - 2242*a^2*b^2*c^10 + 4146*a^2*b^3*c^9 + 1420*a^2
*b^4*c^8 - 4158*a^2*b^5*c^7 + 77*a^2*b^6*c^6 + 1735*a^2*b^7*c^5 - 234*a^2*b
^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 3714*a^3*b^2*c^9 + 6252*a^3*b^3
*c^8 + 1730*a^3*b^4*c^7 - 4300*a^3*b^5*c^6 - 79*a^3*b^6*c^5 + 968*a^3*b^7*c
^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 3523*a^4*b^2*c^8 + 5025*a^4*b^3*c^7 +
1339*a^4*b^4*c^6 - 2082*a^4*b^5*c^5 - 192*a^4*b^6*c^4 + 156*a^4*b^7*c^3 +
8*a^4*b^8*c^2 - 2031*a^5*b^2*c^7 + 2104*a^5*b^3*c^6 + 634*a^5*b^4*c^5 - 388
*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 676*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a^6
*b^4*c^4 - 100*a^7*b^2*c^5 - 404*a*b*c^12))/c^8 - (((2048*tan(x/2)*(24*b*c^
14 - 96*a*c^14 - 8*c^15 + 152*a^2*c^13 + 952*a^3*c^12 + 1096*a^4*c^11 + 304
*a^5*c^10 - 152*a^6*c^9 - 72*a^7*c^8 + 2*b^2*c^13 - 38*b^3*c^12 - 7*b^4*c^1
1 + 39*b^5*c^10 - 15*b^6*c^9 + 35*b^7*c^8 - 44*b^8*c^7 - 4*b^9*c^6 + 24*b^1
0*c^5 - 8*b^11*c^4 + 68*a*b^2*c^12 + 42*a*b^3*c^11 - 159*a*b^4*c^10 - 400*a
*b^5*c^9 + 537*a*b^6*c^8 + 68*a*b^7*c^7 - 276*a*b^8*c^6 + 72*a*b^9*c^5 + 8*
a*b^10*c^4 - 944*a^2*b*c^12 - 2520*a^3*b*c^11 - 1824*a^4*b*c^10 - 272*a^5*b
*c^9 + 88*a^6*b*c^8 + 584*a^2*b^2*c^11 + 1742*a^2*b^3*c^10 - 1645*a^2*b^4*c
^9 - 795*a^2*b^5*c^8 + 1132*a^2*b^6*c^7 - 112*a^2*b^7*c^6 - 112*a^2*b^8*c^5
+ 8*a^2*b^9*c^4 + 476*a^3*b^2*c^10 + 2766*a^3*b^3*c^9 - 1705*a^3*b^4*c^8 -
396*a^3*b^5*c^7 + 456*a^3*b^6*c^6 - 56*a^3*b^7*c^5 - 8*a^3*b^8*c^4 + 230*a
^4*b^2*c^9 + 880*a^4*b^3*c^8 - 656*a^4*b^4*c^7 + 140*a^4*b^5*c^6 + 72*a^4*b
^6*c^5 + 464*a^5*b^2*c^8 - 192*a^5*b^3*c^7 - 220*a^5*b^4*c^6 + 256*a^6*b^2*
c^7 + 136*a*b*c^13))/c^8 - (((2048*(48*a*c^15 + 272*a^2*c^14 + 576*a^3*c^13
+ 576*a^4*c^12 + 272*a^5*c^11 + 48*a^6*c^10 - 12*b^2*c^14 + 20*b^3*c^13 +
18*b^4*c^12 - 46*b^5*c^11 + 6*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 - 140*a*b^
2*c^13 + 288*a*b^3*c^12 + 30*a*b^4*c^11 - 240*a*b^5*c^10 + 74*a*b^6*c^9 + 2
0*a*b^7*c^8 - 416*a^2*b*c^13 - 736*a^3*b*c^12 - 544*a^4*b*c^11 - 144*a^5*b*
c^10 - 360*a^2*b^2*c^12 + 728*a^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*
c^9 + 4*a^2*b^6*c^8 - 360*a^3*b^2*c^11 + 544*a^3*b^3*c^10 + 10*a^3*b^4*c^9
- 20*a^3*b^5*c^8 - 172*a^4*b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*
a^5*b^2*c^9 - 80*a*b*c^14))/c^8 + (2048*tan(x/2)*(a*c*1i - b^2*1i + (c^2*3i
)/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 -
8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*
c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8
*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2
*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^1
2 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^11
)*(a*c*1i - b^2*1i + (c^2*3i)/2))/c^3)*(a*c*1i - b^2*1i + (c^2*3i)/2))/c^3)
*(a*c*1i - b^2*1i + (c^2*3i)/2))/c^3 + (2048*tan(x/2)*(20*a*b^12 + 42*a*c^1
2 - 58*b*c^12 + 4*b^12*c - 4*b^13 + 22*c^13 - 40*a^2*b^11 + 40*a^3*b^10 - 2
0*a^4*b^9 + 4*a^5*b^8 - 214*a^2*c^11 - 938*a^3*c^10 - 1538*a^4*c^9 - 1278*a
^5*c^8 - 498*a^6*c^7 - 14*a^7*c^6 + 52*a^8*c^5 + 12*a^9*c^4 + 14*b^2*c^11 +
34*b^3*c^10 + 59*b^4*c^9 - 39*b^5*c^8 - 160*b^6*c^7 + 112*b^7*c^6 + 105*b^
8*c^5 - 89*b^9*c^4 - 28*b^10*c^3 + 28*b^11*c^2 - 518*a*b^2*c^10 - 264*a*b^3
*c^9 + 1339*a*b^4*c^8 - 92*a*b^5*c^7 - 1312*a*b^6*c^6 + 268*a*b^7*c^5 + 649

```
*a*b^8*c^4 - 124*a*b^9*c^3 - 180*a*b^10*c^2 + 1550*a^2*b*c^10 - 160*a^2*b^1
0*c + 3488*a^3*b*c^9 + 320*a^3*b^9*c + 3350*a^4*b*c^8 - 300*a^4*b^8*c + 109
2*a^5*b*c^7 + 136*a^5*b^7*c - 462*a^6*b*c^6 - 24*a^6*b^6*c - 440*a^7*b*c^5
- 92*a^8*b*c^4 - 1568*a^2*b^2*c^9 - 2708*a^2*b^3*c^8 + 3564*a^2*b^4*c^7 + 1
964*a^2*b^5*c^6 - 2790*a^2*b^6*c^5 - 922*a^2*b^7*c^4 + 1048*a^2*b^8*c^3 + 2
76*a^2*b^9*c^2 - 652*a^3*b^2*c^8 - 6280*a^3*b^3*c^7 + 2020*a^3*b^4*c^6 + 49
88*a^3*b^5*c^5 - 1118*a^3*b^6*c^4 - 2008*a^3*b^7*c^3 + 140*a^3*b^8*c^2 + 23
50*a^4*b^2*c^7 - 5630*a^4*b^3*c^6 - 2295*a^4*b^4*c^5 + 3563*a^4*b^5*c^4 + 1
260*a^4*b^6*c^3 - 740*a^4*b^7*c^2 + 3314*a^5*b^2*c^6 - 1456*a^5*b^3*c^5 - 2
771*a^5*b^4*c^4 + 308*a^5*b^5*c^3 + 732*a^5*b^6*c^2 + 1572*a^6*b^2*c^5 + 57
6*a^6*b^3*c^4 - 696*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 192*a^7*b^2*c^4 + 272*a
^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 148*a*b*c^11 + 24*a*b^11*c))
/c^8)*(a*c*1i - b^2*1i + (c^2*3i)/2))/c^3))*(a*c*1i - b^2*1i + (c^2*3i)/2)*
2i)/c^3
```

3.7 $\int \frac{\sin^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 19, antiderivative size = 260

$$\int \frac{\sin^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{x}{c}+\frac{2\left(b-\frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{c\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}}+\frac{2\left(b+\frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{c\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}$$

```
[Out] -x/c+2*arctan((b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c-(-4*a*c+b^
2)^(1/2))^(1/2))*(b+(-b^2+2*c*(a+c))/(-4*a*c+b^2)^(1/2))/c/(b-2*c-(-4*a*c+b
^2)^(1/2))^(1/2)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2)+2*arctan((b-2*c+(-4*a*c+b
^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2))*(b+(b^2-2*c*(
a+c))/(-4*a*c+b^2)^(1/2))/c/(b-2*c+(-4*a*c+b^2)^(1/2))^(1/2)/(b+2*c+(-4*a*c
+b^2)^(1/2))^(1/2)
```

Rubi [A] (verified)

Time = 1.45 (sec) , antiderivative size = 260, normalized size of antiderivative = 1.00, number of steps used = 7, number of rules used = 4, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.211, Rules used = {3348, 3374, 2738, 211}

$$\int \frac{\sin^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{2\left(b-\frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{-\sqrt{b^2-4ac}+b-2c}}{\sqrt{-\sqrt{b^2-4ac}+b+2c}}\right)}{c\sqrt{-\sqrt{b^2-4ac}+b-2c}\sqrt{-\sqrt{b^2-4ac}+b+2c}}+\frac{2\left(\frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}+b\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{\sqrt{b^2-4ac}+b-2c}}{\sqrt{\sqrt{b^2-4ac}+b+2c}}\right)}{c\sqrt{\sqrt{b^2-4ac}+b-2c}\sqrt{\sqrt{b^2-4ac}+b+2c}}-\frac{x}{c}$$

```
[In] Int[Sin[x]^2/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] -(x/c) + (2*(b - (b^2 - 2*c*(a + c))/Sqrt[b^2 - 4*a*c])*ArcTan[(Sqrt[b - 2*
c - Sqrt[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]])/(c*Sqr
t[b - 2*c - Sqrt[b^2 - 4*a*c]]*Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]) + (2*(b +
 (b^2 - 2*c*(a + c))/Sqrt[b^2 - 4*a*c])*ArcTan[(Sqrt[b - 2*c + Sqrt[b^2 - 4
*a*c]]*Tan[x/2])/Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]]])/(c*Sqrt[b - 2*c + Sqrt
[b^2 - 4*a*c]]*Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]])
```

Rule 211

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(Rt[a/b, 2]/a)*ArcTan[x/R
t[a/b, 2]], x] /; FreeQ[{a, b}, x] && PosQ[a/b]
```

Rule 2738

```
Int[((a_) + (b_.)*sin[Pi/2 + (c_.) + (d_.)*(x_)])^(-1), x_Symbol] :> With[{
e = FreeFactors[Tan[(c + d*x)/2], x]}, Dist[2*(e/d), Subst[Int[1/(a + b + (
a - b)*e^2*x^2), x], x, Tan[(c + d*x)/2]/e], x]] /; FreeQ[{a, b, c, d}, x]
&& NeQ[a^2 - b^2, 0]
```

Rule 3348

```
Int[((a_.) + cos[(d_.) + (e_.)*(x_)]^(n_.)*(b_.) + cos[(d_.) + (e_.)*(x_)]^
(n2_.)*(c_.))^(p_.)*sin[(d_.) + (e_.)*(x_)]^(m_.), x_Symbol] :> Int[ExpandT
rig[(1 - cos[d + e*x]^2)^(m/2)*(a + b*cos[d + e*x]^n + c*cos[d + e*x]^(2*n)
)^p, x], x] /; FreeQ[{a, b, c, d, e}, x] && EqQ[n2, 2*n] && IntegerQ[m/2] &
& NeQ[b^2 - 4*a*c, 0] && IntegersQ[n, p]
```

Rule 3374

```
Int[(cos[(d_.) + (e_.)*(x_)]*(B_.) + (A_))/((a_.) + cos[(d_.) + (e_.)*(x_)]
*(b_.) + cos[(d_.) + (e_.)*(x_)]^2*(c_.)), x_Symbol] :> Module[{q = Rt[b^2
- 4*a*c, 2]}, Dist[B + (b*B - 2*A*c)/q, Int[1/(b + q + 2*c*Cos[d + e*x]), x
], x] + Dist[B - (b*B - 2*A*c)/q, Int[1/(b - q + 2*c*Cos[d + e*x]), x], x]]
 /; FreeQ[{a, b, c, d, e, A, B}, x] && NeQ[b^2 - 4*a*c, 0]
```

Rubi steps

$$\text{integral} = \int \left(-\frac{1}{c} + \frac{a\left(1+\frac{c}{a}\right) + b\cos(x)}{c\left(a+b\cos(x)+c\cos^2(x)\right)}\right) dx$$

$$= -\frac{x}{c} + \frac{\int \frac{a\left(1+\frac{c}{a}\right)+b\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx}{c}$$

$$= -\frac{x}{c} + \frac{\left(b-\frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}\right)\int \frac{1}{b-\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{c} + \frac{\left(b+\frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}\right)\int \frac{1}{b+\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{c}$$

$$= -\frac{x}{c} + \frac{\left(2\left(b - \frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}\right)\right)\text{Subst}\left(\int \frac{1}{b+2c-\sqrt{b^2-4ac}+\left(b-2c-\sqrt{b^2-4ac}\right)x^2}dx, x, \tan\left(\frac{x}{2}\right)\right)}{c}$$
$$+ \frac{\left(2\left(b + \frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}\right)\right)\text{Subst}\left(\int \frac{1}{b+2c+\sqrt{b^2-4ac}+\left(b-2c+\sqrt{b^2-4ac}\right)x^2}dx, x, \tan\left(\frac{x}{2}\right)\right)}{c}$$
$$= -\frac{x}{c} + \frac{2\left(b - \frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{c\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}}$$
$$+ \frac{2\left(b + \frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{c\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}$$

Mathematica [A] (verified)

Time = 0.46 (sec) , antiderivative size = 238, normalized size of antiderivative = 0.92

$$\int \frac{\sin^2(x)}{a+b\cos(x)+c\cos^2(x)}dx$$
$$= \frac{-\sqrt{b^2-4ac}x - \frac{\left(b^2-2c(a+c)+b\sqrt{b^2-4ac}\right)\text{arctanh}\left(\frac{\left(b-2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)-2b\sqrt{b^2-4ac}}}\right)}{\sqrt{-\frac{b^2}{2}+c(a+c)-\frac{1}{2}b\sqrt{b^2-4ac}}} + \sqrt{-2b^2+4c(a+c)+2b\sqrt{b^2-4ac}}a}{c\sqrt{b^2-4ac}}$$

```
[In] Integrate[Sin[x]^2/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (-(Sqrt[b^2 - 4*a*c]*x) - ((b^2 - 2*c*(a + c) + b*Sqrt[b^2 - 4*a*c])*ArcTan
h[((b - 2*c + Sqrt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) - 2*b*
Sqrt[b^2 - 4*a*c]]])/Sqrt[-1/2*b^2 + c*(a + c) - (b*Sqrt[b^2 - 4*a*c])/2] +
 Sqrt[-2*b^2 + 4*c*(a + c) + 2*b*Sqrt[b^2 - 4*a*c]]*ArcTanh[((-b + 2*c + Sq
rt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) + 2*b*Sqrt[b^2 - 4*a*c
]]])/(c*Sqrt[b^2 - 4*a*c])
```

Maple [C] (verified)

Result contains higher order function than in optimal. Order 9 vs. order 3.

Time = 0.71 (sec) , antiderivative size = 185, normalized size of antiderivative = 0.71

method	result
risch	$-\frac{x}{c}-\frac{\left(\sum_{\_R=\mathrm{RootOf}\left(\left(16a^2c^4-8a\,b^2c^3+b^4c^2\right)\_Z^4+\left(128a^2c^2-96a\,b^2c+128a\,c^3+16b^4-32b^2c^2\right)\_Z^2+256a^2+512ac-256b^2+256c^2\right)}\_R\ln\right.}{4}$
default	$\frac{2(a-b+c)\left(\frac{\left(b+2c+\sqrt{-4ac+b^2}\right)\operatorname{arctanh}\left(\frac{(-a+b-c)\tan\left(\frac{x}{2}\right)}{\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}\right)}{2\sqrt{-4ac+b^2}\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}+\frac{\left(\sqrt{-4ac+b^2}-b-2c\right)\arctan\left(\frac{(a-b+c)\tan\left(\frac{x}{2}\right)}{\sqrt{\left(\sqrt{-4ac+b^2}+a-c\right)(a-b+c)}}\right)}{2\sqrt{-4ac+b^2}\sqrt{\left(\sqrt{-4ac+b^2}+a-c\right)(a-b+c)}}\right)}{c}$

```
[In] int(sin(x)^2/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] -x/c-1/4*sum(_R*ln(exp(I*x)+(1/8*I/b*c^3*a-1/32*I*b*c^2)*_R^3+(1/4/b*c^2*a-
1/16*c*b)*_R^2+(1/2*I/b*a*c-1/4*I*b+1/2*I/b*c^2)*_R+1/b*a+1/b*c),_R=RootOf(
(16*a^2*c^4-8*a*b^2*c^3+b^4*c^2)*_Z^4+(128*a^2*c^2-96*a*b^2*c+128*a*c^3+16*
b^4-32*b^2*c^2)*_Z^2+256*a^2+512*a*c-256*b^2+256*c^2))
```

Fricas [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 971 vs. $2(220) = 440$.

Time = 0.34 (sec) , antiderivative size = 971, normalized size of antiderivative = 3.73

$$\int \frac{\sin^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx =$$

$$-\frac{\sqrt{2}c\sqrt{-\frac{b^2-2\,ac-2\,c^2+(b^2c^2-4\,ac^3)\sqrt{\frac{b^2}{b^2c^4-4\,ac^5}}}{b^2c^2-4\,ac^3}}\log\left(\sqrt{2}(b^2c^3-4\,ac^4)\sqrt{\frac{b^2}{b^2c^4-4\,ac^5}}\sqrt{-\frac{b^2-2\,ac-2\,c^2+(b^2c^2-4\,ac^3)\sqrt{\frac{b^2}{b^2c^4}}}{b^2c^2-4\,ac^3}}\right.}{}$$

```
[In] integrate(sin(x)^2/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
```

```
[Out] -1/4*(sqrt(2)*c*sqrt(-(b^2 - 2*a*c - 2*c^2 + (b^2*c^2 - 4*a*c^3)*sqrt(b^2/(
b^2*c^4 - 4*a*c^5)))/(b^2*c^2 - 4*a*c^3))*log(sqrt(2)*(b^2*c^3 - 4*a*c^4)*s
qrt(b^2/(b^2*c^4 - 4*a*c^5))*sqrt(-(b^2 - 2*a*c - 2*c^2 + (b^2*c^2 - 4*a*c^
3)*sqrt(b^2/(b^2*c^4 - 4*a*c^5)))/(b^2*c^2 - 4*a*c^3))*sin(x) + b^2*cos(x)
+ (b^2*c^2 - 4*a*c^3)*sqrt(b^2/(b^2*c^4 - 4*a*c^5))*cos(x) + 2*b*c) - sqrt(
2)*c*sqrt(-(b^2 - 2*a*c - 2*c^2 + (b^2*c^2 - 4*a*c^3)*sqrt(b^2/(b^2*c^4 - 4
*a*c^5)))/(b^2*c^2 - 4*a*c^3))*log(-sqrt(2)*(b^2*c^3 - 4*a*c^4)*sqrt(b^2/(b
^2*c^4 - 4*a*c^5))*sqrt(-(b^2 - 2*a*c - 2*c^2 + (b^2*c^2 - 4*a*c^3)*sqrt(b^
2/(b^2*c^4 - 4*a*c^5)))/(b^2*c^2 - 4*a*c^3))*sin(x) + b^2*cos(x) + (b^2*c^2
 - 4*a*c^3)*sqrt(b^2/(b^2*c^4 - 4*a*c^5))*cos(x) + 2*b*c) + sqrt(2)*c*sqrt(
-(b^2 - 2*a*c - 2*c^2 - (b^2*c^2 - 4*a*c^3)*sqrt(b^2/(b^2*c^4 - 4*a*c^5)))/
(b^2*c^2 - 4*a*c^3))*log(sqrt(2)*(b^2*c^3 - 4*a*c^4)*sqrt(b^2/(b^2*c^4 - 4*
a*c^5))*sqrt(-(b^2 - 2*a*c - 2*c^2 - (b^2*c^2 - 4*a*c^3)*sqrt(b^2/(b^2*c^4
```

```
- 4*a*c^5)))/(b^2*c^2 - 4*a*c^3))*sin(x) - b^2*cos(x) + (b^2*c^2 - 4*a*c^3)
*sqrt(b^2/(b^2*c^4 - 4*a*c^5))*cos(x) - 2*b*c) - sqrt(2)*c*sqrt(-(b^2 - 2*a
*c - 2*c^2 - (b^2*c^2 - 4*a*c^3)*sqrt(b^2/(b^2*c^4 - 4*a*c^5)))/(b^2*c^2 -
4*a*c^3))*log(-sqrt(2)*(b^2*c^3 - 4*a*c^4)*sqrt(b^2/(b^2*c^4 - 4*a*c^5))*sq
rt(-(b^2 - 2*a*c - 2*c^2 - (b^2*c^2 - 4*a*c^3)*sqrt(b^2/(b^2*c^4 - 4*a*c^5)
))/(b^2*c^2 - 4*a*c^3))*sin(x) - b^2*cos(x) + (b^2*c^2 - 4*a*c^3)*sqrt(b^2/
(b^2*c^4 - 4*a*c^5))*cos(x) - 2*b*c) + 4*x)/c
```

Sympy [F(-1)]

Timed out.

$$\int \frac{\sin^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Timed out}$$

```
[In] integrate(sin(x)**2/(a+b*cos(x)+c*cos(x)**2),x)
```

```
[Out] Timed out
```

Maxima [F]

$$\int \frac{\sin^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \int \frac{\sin\left(x\right)^2}{c\cos\left(x\right)^2+b\cos\left(x\right)+a}\,dx$$

```
[In] integrate(sin(x)^2/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
```

```
[Out] (c*integrate(2*(2*b^2*cos(3*x)^2 + 2*b^2*cos(x)^2 + 2*b^2*sin(3*x)^2 + 2*b^
2*sin(x)^2 + 4*(2*a^2 + 3*a*c + c^2)*cos(2*x)^2 + b*c*cos(x) + 4*(2*a^2 + 3
*a*c + c^2)*sin(2*x)^2 + 2*(4*a*b + 3*b*c)*sin(2*x)*sin(x) + (b*c*cos(3*x)
+ b*c*cos(x) + 2*(a*c + c^2)*cos(2*x))*cos(4*x) + (4*b^2*cos(x) + b*c + 2*(
4*a*b + 3*b*c)*cos(2*x))*cos(3*x) + 2*(a*c + c^2 + (4*a*b + 3*b*c)*cos(x))*
cos(2*x) + (b*c*sin(3*x) + b*c*sin(x) + 2*(a*c + c^2)*sin(2*x))*sin(4*x) +
2*(2*b^2*sin(x) + (4*a*b + 3*b*c)*sin(2*x))*sin(3*x))/(c^3*cos(4*x)^2 + 4*b
^2*c*cos(3*x)^2 + 4*b^2*c*cos(x)^2 + c^3*sin(4*x)^2 + 4*b^2*c*sin(3*x)^2 +
4*b^2*c*sin(x)^2 + 4*b*c^2*cos(x) + c^3 + 4*(4*a^2*c + 4*a*c^2 + c^3)*cos(2
*x)^2 + 4*(4*a^2*c + 4*a*c^2 + c^3)*sin(2*x)^2 + 8*(2*a*b*c + b*c^2)*sin(2*
x)*sin(x) + 2*(2*b*c^2*cos(3*x) + 2*b*c^2*cos(x) + c^3 + 2*(2*a*c^2 + c^3)*
cos(2*x))*cos(4*x) + 4*(2*b^2*c*cos(x) + b*c^2 + 2*(2*a*b*c + b*c^2)*cos(2*
x))*cos(3*x) + 4*(2*a*c^2 + c^3 + 2*(2*a*b*c + b*c^2)*cos(x))*cos(2*x) + 4*
(b*c^2*sin(3*x) + b*c^2*sin(x) + (2*a*c^2 + c^3)*sin(2*x))*sin(4*x) + 8*(b^
2*c*sin(x) + (2*a*b*c + b*c^2)*sin(2*x))*sin(3*x)), x) - x)/c
```

Giac [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 6566 vs. 2(220) = 440.

Time = 2.51 (sec) , antiderivative size = 6566, normalized size of antiderivative = 25.25

$$\int \frac{\sin^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(sin(x)^2/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
[Out] -x/c - ((2*a^2*b^4 - 4*a*b^5 + 2*b^6 - 16*a^3*b^2*c + 32*a^2*b^3*c - 12*a*b
^4*c - 4*b^5*c + 32*a^4*c^2 - 64*a^3*b*c^2 + 32*a*b^3*c^2 + 2*b^4*c^2 + 64*
a^3*c^3 - 64*a^2*b*c^3 - 16*a*b^2*c^3 + 32*a^2*c^4 + 3*sqrt(a^2 - a*b + b*c
 - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2 - 2*(b^2
- 4*a*c)*a^2*b^2 - 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*sqrt(b^2 - 4*a*c)*a*b^3 + 4*(b^2 - 4*a*c)*a*b^3 - 5*sqrt(a^2 - a*b +
b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4 - 2*(b^2 -
 4*a*c)*b^4 - 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c)
)*sqrt(b^2 - 4*a*c)*a^3*c + 8*(b^2 - 4*a*c)*a^3*c + 8*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c - 16*(b^2
- 4*a*c)*a^2*b*c + 34*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
 + c))*sqrt(b^2 - 4*a*c)*a*b^2*c + 4*(b^2 - 4*a*c)*a*b^2*c + 6*sqrt(a^2 - a
*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^3*c + 4
*(b^2 - 4*a*c)*b^3*c - 56*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
 - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^2 + 16*(b^2 - 4*a*c)*a^2*c^2 - 24*sqrt(a
^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b
*c^2 - 16*(b^2 - 4*a*c)*a*b*c^2 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c^2 - 2*(b^2 - 4*a*c)*b^2*c^2 +
20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4
*a*c)*a*c^3 + 8*(b^2 - 4*a*c)*a*c^3)*c^2*abs(a - b + c) - (4*a^2*b^4*c - 4*
b^6*c - 32*a^3*b^2*c^2 + 40*a*b^4*c^2 + 64*a^4*c^3 - 128*a^2*b^2*c^3 + 4*b^
4*c^3 + 128*a^3*c^4 - 32*a*b^2*c^4 + 64*a^2*c^5 + 3*sqrt(a^2 - a*b + b*c -
c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^2*c + sqrt(a^2 - a*b + b*c - c^2
 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^3*c - 7*sqrt(a^2 - a*b + b*c - c^2
+ sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^4*c - 5*sqrt(a^2 - a*b + b*c - c^2 + s
qrt(b^2 - 4*a*c)*(a - b + c))*b^5*c - 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(
b^2 - 4*a*c)*(a - b + c))*a^4*c^2 - 4*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
 - 4*a*c)*(a - b + c))*a^3*b*c^2 + 45*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
 - 4*a*c)*(a - b + c))*a^2*b^2*c^2 + 38*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b
^2 - 4*a*c)*(a - b + c))*a*b^3*c^2 + sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
- 4*a*c)*(a - b + c))*b^4*c^2 - 68*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
4*a*c)*(a - b + c))*a^3*c^3 - 72*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*
a*c)*(a - b + c))*a^2*b*c^3 + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a
*c)*(a - b + c))*a*b^2*c^3 + sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)
*(a - b + c))*b^3*c^3 - 36*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(
```

a - b + c))*a^2*c^4 - 4*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
 b + c))*a*b*c^4 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*b^2*c^4 + 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*a*c^5 - 4*(b^2 - 4*a*c)*a^2*b^2*c + 4*(b^2 - 4*a*c)*b^4*c + 16*(b^2 - 4
*a*c)*a^3*c^2 - 24*(b^2 - 4*a*c)*a*b^2*c^2 + 32*(b^2 - 4*a*c)*a^2*c^3 - 4*(
b^2 - 4*a*c)*b^2*c^3 + 16*(b^2 - 4*a*c)*a*c^4)*abs(a - b + c)*abs(c) + (2*a
^3*b^3*c^2 - 4*a^2*b^4*c^2 + 2*a*b^5*c^2 - 8*a^4*b*c^3 + 20*a^3*b^2*c^3 - 1
4*a^2*b^3*c^3 + 4*a*b^4*c^3 - 2*b^5*c^3 - 16*a^4*c^4 + 24*a^3*b*c^4 - 12*a^
2*b^2*c^4 + 6*a*b^3*c^4 - 16*a^3*c^5 + 8*a^2*b*c^5 - 4*a*b^2*c^5 + 6*b^3*c^
5 + 16*a^2*c^6 - 24*a*b*c^6 - 4*b^2*c^6 + 16*a*c^7 + 3*sqrt(a^2 - a*b + b*c
 - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^2 - 2*(b^
2 - 4*a*c)*a^3*b*c^2 - 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
- b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c^2 + 4*(b^2 - 4*a*c)*a^2*b^2*c^2 - 5*s
qrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c
)*a*b^3*c^2 - 2*(b^2 - 4*a*c)*a*b^3*c^2 + 6*sqrt(a^2 - a*b + b*c - c^2 + sq
rt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^3 - 4*(b^2 - 4*a*c)*a^
3*c^3 + 7*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(
b^2 - 4*a*c)*a^2*b*c^3 + 6*(b^2 - 4*a*c)*a^2*b*c^3 - 2*sqrt(a^2 - a*b + b*c
 - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^3 - 4*(b^
2 - 4*a*c)*a*b^2*c^3 + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
- b + c))*sqrt(b^2 - 4*a*c)*b^3*c^3 + 2*(b^2 - 4*a*c)*b^3*c^3 + 22*sqrt(a^2
 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c
^4 - 4*(b^2 - 4*a*c)*a^2*c^4 - 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*c^4 + 2*(b^2 - 4*a*c)*a*b*c^4 + 4*s
qrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c
)*b^2*c^4 - 38*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*
sqrt(b^2 - 4*a*c)*a*c^5 + 4*(b^2 - 4*a*c)*a*c^5 - 7*sqrt(a^2 - a*b + b*c -
c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b*c^5 - 6*(b^2 - 4*a
*c)*b*c^5 + 10*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*
sqrt(b^2 - 4*a*c)*c^6 + 4*(b^2 - 4*a*c)*c^6)*abs(a - b + c))*(pi*floor(1/2*
x/pi + 1/2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a*c - 2*c^2 + sqrt(-4*(
a*c + b*c + c^2)*(a*c - b*c + c^2) + 4*(a*c - c^2)^2))/(a*c - b*c + c^2))))
/((3*a^5*b^2*c^2 - 5*a^4*b^3*c^2 - 6*a^3*b^4*c^2 + 10*a^2*b^5*c^2 + 3*a*b^6
*c^2 - 5*b^7*c^2 - 12*a^6*c^3 + 20*a^5*b*c^3 + 47*a^4*b^2*c^3 - 60*a^3*b^3*
c^3 - 46*a^2*b^4*c^3 + 40*a*b^5*c^3 + 11*b^6*c^3 - 92*a^5*c^4 + 80*a^4*b*c^
4 + 182*a^3*b^2*c^4 - 94*a^2*b^3*c^4 - 78*a*b^4*c^4 - 6*b^5*c^4 - 184*a^4*c
^5 + 56*a^3*b*c^5 + 166*a^2*b^2*c^5 + 36*a*b^3*c^5 - 6*b^4*c^5 - 120*a^3*c^
6 - 48*a^2*b*c^6 + 23*a*b^2*c^6 + 11*b^3*c^6 + 4*a^2*c^7 - 44*a*b*c^7 - 5*b
^2*c^7 + 20*a*c^8)*abs(c)) + ((2*a^2*b^4 - 4*a*b^5 + 2*b^6 - 16*a^3*b^2*c +
 32*a^2*b^3*c - 12*a*b^4*c - 4*b^5*c + 32*a^4*c^2 - 64*a^3*b*c^2 + 32*a*b^3
*c^2 + 2*b^4*c^2 + 64*a^3*c^3 - 64*a^2*b*c^3 - 16*a*b^2*c^3 + 32*a^2*c^4 +
3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*a^2*b^2 - 2*(b^2 - 4*a*c)*a^2*b^2 - 2*sqrt(a^2 - a*b + b*c - c^2 - sqr
t(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3 + 4*(b^2 - 4*a*c)*a*b^3
 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -

4*a*c)*b^4 - 2*(b^2 - 4*a*c)*b^4 - 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^
2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c + 8*(b^2 - 4*a*c)*a^3*c + 8
*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a
*c)*a^2*b*c - 16*(b^2 - 4*a*c)*a^2*b*c + 34*sqrt(a^2 - a*b + b*c - c^2 - sq
rt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c + 4*(b^2 - 4*a*c)*a*
b^2*c + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(
b^2 - 4*a*c)*b^3*c + 4*(b^2 - 4*a*c)*b^3*c - 56*sqrt(a^2 - a*b + b*c - c^2
- sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^2 + 16*(b^2 - 4*a*
c)*a^2*c^2 - 24*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))
*sqrt(b^2 - 4*a*c)*a*b*c^2 - 16*(b^2 - 4*a*c)*a*b*c^2 - 5*sqrt(a^2 - a*b +
b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c^2 - 2*(b
^2 - 4*a*c)*b^2*c^2 + 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*sqrt(b^2 - 4*a*c)*a*c^3 + 8*(b^2 - 4*a*c)*a*c^3)*c^2*abs(a - b +
c) - (4*a^2*b^4*c - 4*b^6*c - 32*a^3*b^2*c^2 + 40*a*b^4*c^2 + 64*a^4*c^3 -
128*a^2*b^2*c^3 + 4*b^4*c^3 + 128*a^3*c^4 - 32*a*b^2*c^4 + 64*a^2*c^5 - 3*s
qrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^2*c - sqrt
(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^3*c + 7*sqrt(
a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^4*c + 5*sqrt(a^2
- a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*b^5*c + 12*sqrt(a^2 - a
*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*c^2 + 4*sqrt(a^2 - a*b
+ b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b*c^2 - 45*sqrt(a^2 - a*b
+ b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^2*c^2 - 38*sqrt(a^2 - a*
b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^3*c^2 - sqrt(a^2 - a*b +
b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*b^4*c^2 + 68*sqrt(a^2 - a*b + b
*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*c^3 + 72*sqrt(a^2 - a*b + b*c
- c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b*c^3 - 5*sqrt(a^2 - a*b + b*c
- c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^2*c^3 - sqrt(a^2 - a*b + b*c - c
^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*b^3*c^3 + 36*sqrt(a^2 - a*b + b*c - c^2
- sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*c^4 + 4*sqrt(a^2 - a*b + b*c - c^2 -
sqrt(b^2 - 4*a*c)*(a - b + c))*a*b*c^4 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqr
t(b^2 - 4*a*c)*(a - b + c))*b^2*c^4 - 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(
b^2 - 4*a*c)*(a - b + c))*a*c^5 - 4*(b^2 - 4*a*c)*a^2*b^2*c + 4*(b^2 - 4*a*
c)*b^4*c + 16*(b^2 - 4*a*c)*a^3*c^2 - 24*(b^2 - 4*a*c)*a*b^2*c^2 + 32*(b^2
- 4*a*c)*a^2*c^3 - 4*(b^2 - 4*a*c)*b^2*c^3 + 16*(b^2 - 4*a*c)*a*c^4)*abs(a
- b + c)*abs(c) + (2*a^3*b^3*c^2 - 4*a^2*b^4*c^2 + 2*a*b^5*c^2 - 8*a^4*b*c^
3 + 20*a^3*b^2*c^3 - 14*a^2*b^3*c^3 + 4*a*b^4*c^3 - 2*b^5*c^3 - 16*a^4*c^4
+ 24*a^3*b*c^4 - 12*a^2*b^2*c^4 + 6*a*b^3*c^4 - 16*a^3*c^5 + 8*a^2*b*c^5 -
4*a*b^2*c^5 + 6*b^3*c^5 + 16*a^2*c^6 - 24*a*b*c^6 - 4*b^2*c^6 + 16*a*c^7 +
3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*a^3*b*c^2 - 2*(b^2 - 4*a*c)*a^3*b*c^2 - 2*sqrt(a^2 - a*b + b*c - c^2 -
sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c^2 + 4*(b^2 - 4*
a*c)*a^2*b^2*c^2 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*sqrt(b^2 - 4*a*c)*a*b^3*c^2 - 2*(b^2 - 4*a*c)*a*b^3*c^2 + 6*sqrt(a^2
- a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^
3 - 4*(b^2 - 4*a*c)*a^3*c^3 + 7*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a

```
*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^3 + 6*(b^2 - 4*a*c)*a^2*b*c^3 -
2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*a*b^2*c^3 - 4*(b^2 - 4*a*c)*a*b^2*c^3 + 5*sqrt(a^2 - a*b + b*c - c^2 -
 sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^3*c^3 + 2*(b^2 - 4*a*c)
*b^3*c^3 + 22*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*s
qrt(b^2 - 4*a*c)*a^2*c^4 - 4*(b^2 - 4*a*c)*a^2*c^4 - 3*sqrt(a^2 - a*b + b*c
 - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*c^4 + 2*(b^2
- 4*a*c)*a*b*c^4 + 4*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*sqrt(b^2 - 4*a*c)*b^2*c^4 - 38*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
- 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*c^5 + 4*(b^2 - 4*a*c)*a*c^5 - 7*s
qrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c
)*b*c^5 - 6*(b^2 - 4*a*c)*b*c^5 + 10*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
- 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*c^6 + 4*(b^2 - 4*a*c)*c^6)*abs(a -
b + c))*(pi*floor(1/2*x/pi + 1/2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a
*c - 2*c^2 - sqrt(-4*(a*c + b*c + c^2)*(a*c - b*c + c^2) + 4*(a*c - c^2)^2)
)/(a*c - b*c + c^2))))/((3*a^5*b^2*c^2 - 5*a^4*b^3*c^2 - 6*a^3*b^4*c^2 + 10
*a^2*b^5*c^2 + 3*a*b^6*c^2 - 5*b^7*c^2 - 12*a^6*c^3 + 20*a^5*b*c^3 + 47*a^4
*b^2*c^3 - 60*a^3*b^3*c^3 - 46*a^2*b^4*c^3 + 40*a*b^5*c^3 + 11*b^6*c^3 - 92
*a^5*c^4 + 80*a^4*b*c^4 + 182*a^3*b^2*c^4 - 94*a^2*b^3*c^4 - 78*a*b^4*c^4 -
 6*b^5*c^4 - 184*a^4*c^5 + 56*a^3*b*c^5 + 166*a^2*b^2*c^5 + 36*a*b^3*c^5 -
6*b^4*c^5 - 120*a^3*c^6 - 48*a^2*b*c^6 + 23*a*b^2*c^6 + 11*b^3*c^6 + 4*a^2*
c^7 - 44*a*b*c^7 - 5*b^2*c^7 + 20*a*c^8)*abs(c))
```

Mupad [B] (verification not implemented)

Time = 14.83 (sec) , antiderivative size = 16390, normalized size of antiderivative = 63.04

$$\int \frac{\sin^2(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] int(sin(x)^2/(a + b*cos(x) + c*cos(x)^2),x)
```

```
[Out] atan(((tan(x/2)*(57344*a^4*b - 57344*a*b^4 + 8192*a*c^4 + 8192*a^4*c + 5734
4*b*c^4 - 57344*b^4*c - 24576*a^5 + 24576*b^5 - 24576*c^5 + 49152*a^2*b^3 -
 49152*a^3*b^2 + 147456*a^2*c^3 + 147456*a^3*c^2 - 49152*b^2*c^3 + 49152*b^
3*c^2 + 245760*a*b^2*c^2 - 442368*a^2*b*c^2 + 245760*a^2*b^2*c - 163840*a*b
*c^3 - 32768*a*b^3*c - 163840*a^3*b*c) + (-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^
(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 -
 8*a*b^2*c^3)))^(1/2)*(32768*a*b^5 - 253952*a*c^5 - 24576*a^5*c + 57344*b*c
^5 + 57344*b^5*c - 24576*b^6 - 24576*c^6 + 16384*a^2*b^4 - 32768*a^3*b^3 +
8192*a^4*b^2 - 638976*a^2*c^4 - 638976*a^3*c^3 - 253952*a^4*c^2 + 24576*b^2
*c^4 - 114688*b^3*c^3 + 24576*b^4*c^2 + (tan(x/2)*(16384*a*b^6 - 81920*a*c^
6 + 49152*b*c^6 + 49152*b^6*c - 16384*b^7 - 16384*c^7 + 16384*a^2*b^5 - 163
84*a^3*b^4 + 229376*a^2*c^5 + 491520*a^3*c^4 + 49152*a^4*c^3 - 147456*a^5*c
^2 - 32768*b^2*c^5 - 32768*b^5*c^2 + 327680*a*b^3*c^3 - 425984*a*b^4*c^2 -
```

1015808*a^2*b*c^4 - 180224*a^2*b^4*c - 983040*a^3*b*c^3 - 65536*a^3*b^3*c +
 49152*a^4*b*c^2 + 98304*a^4*b^2*c + 851968*a^2*b^2*c^3 + 131072*a^2*b^3*c^
2 + 393216*a^3*b^2*c^2 + 65536*a*b*c^5 + 98304*a*b^5*c) + (-(8*a*c^3 + b*(-
(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^
2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(24576*b^2*c^6 - 393216*a^2*c^6 - 58
9824*a^3*c^5 - 393216*a^4*c^4 - 98304*a^5*c^3 - 98304*a*c^7 - 49152*b^3*c^5
 + 49152*b^5*c^3 - 24576*b^6*c^2 + 98304*a*b^2*c^5 - 344064*a*b^3*c^4 + 983
04*a*b^4*c^3 + 49152*a*b^5*c^2 + 589824*a^2*b*c^5 + 589824*a^3*b*c^4 + 1966
08*a^4*b*c^3 + 147456*a^2*b^2*c^4 - 344064*a^2*b^3*c^3 + 98304*a^3*b^2*c^3
 - 49152*a^3*b^3*c^2 + 24576*a^4*b^2*c^2 + 196608*a*b*c^6 - tan(x/2)*(-(8*a*
c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)
/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(65536*a*c^8 - 131072*a^2*
c^7 - 262144*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c^4 - 16384*b^2*c^7 + 49
152*b^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c^3 + 16384*b^7*c^2
 + 294912*a*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^4*c^4 - 114688*a*b^5*c^
3 - 16384*a*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*b*c^5 - 65536*a^4*b*c^4
 - 655360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^4*c^3 - 16384*a^2*
b^5*c^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^4*c^2 - 1146
88*a^4*b^2*c^3 - 196608*a*b*c^7)))*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2)
 + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b
^2*c^3)))^(1/2) + 147456*a*b^2*c^3 - 458752*a*b^3*c^2 + 802816*a^2*b*c^3 -
245760*a^2*b^3*c + 557056*a^3*b*c^2 - 16384*a^3*b^2*c + 98304*a^2*b^2*c^2 +
 425984*a*b*c^4 + 106496*a*b^4*c + 122880*a^4*b*c))*(-(8*a*c^3 + b*(-(4*a*c
 - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4
 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*1i + (tan(x/2)*(57344*a^4*b - 57344*a*b^4
 + 8192*a*c^4 + 8192*a^4*c + 57344*b*c^4 - 57344*b^4*c - 24576*a^5 + 24576*b
^5 - 24576*c^5 + 49152*a^2*b^3 - 49152*a^3*b^2 + 147456*a^2*c^3 + 147456*a^
3*c^2 - 49152*b^2*c^3 + 49152*b^3*c^2 + 245760*a*b^2*c^2 - 442368*a^2*b*c^2
 + 245760*a^2*b^2*c - 163840*a*b*c^3 - 32768*a*b^3*c - 163840*a^3*b*c) - (-
(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b
^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(32768*a*b^5 - 253952
*a*c^5 - 24576*a^5*c + 57344*b*c^5 + 57344*b^5*c - 24576*b^6 - 24576*c^6 +
16384*a^2*b^4 - 32768*a^3*b^3 + 8192*a^4*b^2 - 638976*a^2*c^4 - 638976*a^3*
c^3 - 253952*a^4*c^2 + 24576*b^2*c^4 - 114688*b^3*c^3 + 24576*b^4*c^2 - (ta
n(x/2)*(16384*a*b^6 - 81920*a*c^6 + 49152*b*c^6 + 49152*b^6*c - 16384*b^7 -
 16384*c^7 + 16384*a^2*b^5 - 16384*a^3*b^4 + 229376*a^2*c^5 + 491520*a^3*c^
4 + 49152*a^4*c^3 - 147456*a^5*c^2 - 32768*b^2*c^5 - 32768*b^5*c^2 + 327680
*a*b^3*c^3 - 425984*a*b^4*c^2 - 1015808*a^2*b*c^4 - 180224*a^2*b^4*c - 9830
40*a^3*b*c^3 - 65536*a^3*b^3*c + 49152*a^4*b*c^2 + 98304*a^4*b^2*c + 851968
*a^2*b^2*c^3 + 131072*a^2*b^3*c^2 + 393216*a^3*b^2*c^2 + 65536*a*b*c^5 + 98
304*a*b^5*c) - (-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 -
2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(245
76*b^2*c^6 - 393216*a^2*c^6 - 589824*a^3*c^5 - 393216*a^4*c^4 - 98304*a^5*c
^3 - 98304*a*c^7 - 49152*b^3*c^5 + 49152*b^5*c^3 - 24576*b^6*c^2 + 98304*a*
b^2*c^5 - 344064*a*b^3*c^4 + 98304*a*b^4*c^3 + 49152*a*b^5*c^2 + 589824*a^2

*b*c^5 + 589824*a^3*b*c^4 + 196608*a^4*b*c^3 + 147456*a^2*b^2*c^4 - 344064*
a^2*b^3*c^3 + 98304*a^3*b^2*c^3 - 49152*a^3*b^3*c^2 + 24576*a^4*b^2*c^2 + 1
96608*a*b*c^6 + tan(x/2)*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*
a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^
(1/2)*(65536*a*c^8 - 131072*a^2*c^7 - 262144*a^3*c^6 + 131072*a^4*c^5 + 196
608*a^5*c^4 - 16384*b^2*c^7 + 49152*b^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4
 - 49152*b^6*c^3 + 16384*b^7*c^2 + 294912*a*b^2*c^6 - 409600*a*b^3*c^5 + 37
6832*a*b^4*c^4 - 114688*a*b^5*c^3 - 16384*a*b^6*c^2 + 589824*a^2*b*c^6 + 72
0896*a^3*b*c^5 - 65536*a^4*b*c^4 - 655360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 +
 196608*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^
3*c^3 + 16384*a^3*b^4*c^2 - 114688*a^4*b^2*c^3 - 196608*a*b*c^7)))*(-(8*a*c
^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/
(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2) + 147456*a*b^2*c^3 - 458752
*a*b^3*c^2 + 802816*a^2*b*c^3 - 245760*a^2*b^3*c + 557056*a^3*b*c^2 - 16384
*a^3*b^2*c + 98304*a^2*b^2*c^2 + 425984*a*b*c^4 + 106496*a*b^4*c + 122880*a
^4*b*c))*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*
c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*1i)/((tan(
x/2)*(57344*a^4*b - 57344*a*b^4 + 8192*a*c^4 + 8192*a^4*c + 57344*b*c^4 - 5
7344*b^4*c - 24576*a^5 + 24576*b^5 - 24576*c^5 + 49152*a^2*b^3 - 49152*a^3*
b^2 + 147456*a^2*c^3 + 147456*a^3*c^2 - 49152*b^2*c^3 + 49152*b^3*c^2 + 245
760*a*b^2*c^2 - 442368*a^2*b*c^2 + 245760*a^2*b^2*c - 163840*a*b*c^3 - 3276
8*a*b^3*c - 163840*a^3*b*c) - (-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4
 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^
3)))^(1/2)*(32768*a*b^5 - 253952*a*c^5 - 24576*a^5*c + 57344*b*c^5 + 57344*
b^5*c - 24576*b^6 - 24576*c^6 + 16384*a^2*b^4 - 32768*a^3*b^3 + 8192*a^4*b^
2 - 638976*a^2*c^4 - 638976*a^3*c^3 - 253952*a^4*c^2 + 24576*b^2*c^4 - 1146
88*b^3*c^3 + 24576*b^4*c^2 - (tan(x/2)*(16384*a*b^6 - 81920*a*c^6 + 49152*b
*c^6 + 49152*b^6*c - 16384*b^7 - 16384*c^7 + 16384*a^2*b^5 - 16384*a^3*b^4
+ 229376*a^2*c^5 + 491520*a^3*c^4 + 49152*a^4*c^3 - 147456*a^5*c^2 - 32768*
b^2*c^5 - 32768*b^5*c^2 + 327680*a*b^3*c^3 - 425984*a*b^4*c^2 - 1015808*a^2
*b*c^4 - 180224*a^2*b^4*c - 983040*a^3*b*c^3 - 65536*a^3*b^3*c + 49152*a^4*
b*c^2 + 98304*a^4*b^2*c + 851968*a^2*b^2*c^3 + 131072*a^2*b^3*c^2 + 393216*
a^3*b^2*c^2 + 65536*a*b*c^5 + 98304*a*b^5*c) - (-(8*a*c^3 + b*(-(4*a*c - b^
2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4
*c^2 - 8*a*b^2*c^3)))^(1/2)*(24576*b^2*c^6 - 393216*a^2*c^6 - 589824*a^3*c^
5 - 393216*a^4*c^4 - 98304*a^5*c^3 - 98304*a*c^7 - 49152*b^3*c^5 + 49152*b^
5*c^3 - 24576*b^6*c^2 + 98304*a*b^2*c^5 - 344064*a*b^3*c^4 + 98304*a*b^4*c^
3 + 49152*a*b^5*c^2 + 589824*a^2*b*c^5 + 589824*a^3*b*c^4 + 196608*a^4*b*c^
3 + 147456*a^2*b^2*c^4 - 344064*a^2*b^3*c^3 + 98304*a^3*b^2*c^3 - 49152*a^3
*b^3*c^2 + 24576*a^4*b^2*c^2 + 196608*a*b*c^6 + tan(x/2)*(-(8*a*c^3 + b*(-(
4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2
*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(65536*a*c^8 - 131072*a^2*c^7 - 26214
4*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c^4 - 16384*b^2*c^7 + 49152*b^3*c^6
 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c^3 + 16384*b^7*c^2 + 294912*a
*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^4*c^4 - 114688*a*b^5*c^3 - 16384*a

*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*b*c^5 - 65536*a^4*b*c^4 - 655360*a
^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 5
57056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^4*c^2 - 114688*a^4*b^2*
c^3 - 196608*a*b*c^7)))*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a
^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(
1/2) + 147456*a*b^2*c^3 - 458752*a*b^3*c^2 + 802816*a^2*b*c^3 - 245760*a^2*
b^3*c + 557056*a^3*b*c^2 - 16384*a^3*b^2*c + 98304*a^2*b^2*c^2 + 425984*a*b
*c^4 + 106496*a*b^4*c + 122880*a^4*b*c))*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^
(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 -
8*a*b^2*c^3)))^(1/2) - (tan(x/2)*(57344*a^4*b - 57344*a*b^4 + 8192*a*c^4 +
8192*a^4*c + 57344*b*c^4 - 57344*b^4*c - 24576*a^5 + 24576*b^5 - 24576*c^5
+ 49152*a^2*b^3 - 49152*a^3*b^2 + 147456*a^2*c^3 + 147456*a^3*c^2 - 49152*
b^2*c^3 + 49152*b^3*c^2 + 245760*a*b^2*c^2 - 442368*a^2*b*c^2 + 245760*a^2*
b^2*c - 163840*a*b*c^3 - 32768*a*b^3*c - 163840*a^3*b*c) + (-(8*a*c^3 + b*(
-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a
^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(32768*a*b^5 - 253952*a*c^5 - 24576
*a^5*c + 57344*b*c^5 + 57344*b^5*c - 24576*b^6 - 24576*c^6 + 16384*a^2*b^4
- 32768*a^3*b^3 + 8192*a^4*b^2 - 638976*a^2*c^4 - 638976*a^3*c^3 - 253952*a
^4*c^2 + 24576*b^2*c^4 - 114688*b^3*c^3 + 24576*b^4*c^2 + (tan(x/2)*(16384*
a*b^6 - 81920*a*c^6 + 49152*b*c^6 + 49152*b^6*c - 16384*b^7 - 16384*c^7 + 1
6384*a^2*b^5 - 16384*a^3*b^4 + 229376*a^2*c^5 + 491520*a^3*c^4 + 49152*a^4*
c^3 - 147456*a^5*c^2 - 32768*b^2*c^5 - 32768*b^5*c^2 + 327680*a*b^3*c^3 - 4
25984*a*b^4*c^2 - 1015808*a^2*b*c^4 - 180224*a^2*b^4*c - 983040*a^3*b*c^3 -
65536*a^3*b^3*c + 49152*a^4*b*c^2 + 98304*a^4*b^2*c + 851968*a^2*b^2*c^3 +
131072*a^2*b^3*c^2 + 393216*a^3*b^2*c^2 + 65536*a*b*c^5 + 98304*a*b^5*c) +
(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*
a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(24576*b^2*c^6 - 3
93216*a^2*c^6 - 589824*a^3*c^5 - 393216*a^4*c^4 - 98304*a^5*c^3 - 98304*a*c
^7 - 49152*b^3*c^5 + 49152*b^5*c^3 - 24576*b^6*c^2 + 98304*a*b^2*c^5 - 3440
64*a*b^3*c^4 + 98304*a*b^4*c^3 + 49152*a*b^5*c^2 + 589824*a^2*b*c^5 + 58982
4*a^3*b*c^4 + 196608*a^4*b*c^3 + 147456*a^2*b^2*c^4 - 344064*a^2*b^3*c^3 +
98304*a^3*b^2*c^3 - 49152*a^3*b^3*c^2 + 24576*a^4*b^2*c^2 + 196608*a*b*c^6
- tan(x/2)*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^
2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(65536*a
*c^8 - 131072*a^2*c^7 - 262144*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c^4 -
16384*b^2*c^7 + 49152*b^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c
^3 + 16384*b^7*c^2 + 294912*a*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^4*c^4
- 114688*a*b^5*c^3 - 16384*a*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*b*c^5
- 65536*a^4*b*c^4 - 655360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^
4*c^3 - 16384*a^2*b^5*c^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*
a^3*b^4*c^2 - 114688*a^4*b^2*c^3 - 196608*a*b*c^7)))*(-(8*a*c^3 + b*(-(4*a*
c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4
+ b^4*c^2 - 8*a*b^2*c^3)))^(1/2) + 147456*a*b^2*c^3 - 458752*a*b^3*c^2 + 8
02816*a^2*b*c^3 - 245760*a^2*b^3*c + 557056*a^3*b*c^2 - 16384*a^3*b^2*c + 9
8304*a^2*b^2*c^2 + 425984*a*b*c^4 + 106496*a*b^4*c + 122880*a^4*b*c))*(-(8*

a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2) + 131072*a*b^3 - 131072*a^3*b + 262144*a*c^3 + 262144*a^3*c - 131072*b*c^3 + 131072*b^3*c + 65536*a^4 - 65536*b^4 + 65536*c^4 + 393216*a^2*c^2 - 393216*a*b*c^2 - 393216*a^2*b*c))*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*2i + atan(((tan(x/2)*(57344*a^4*b - 57344*a*b^4 + 8192*a*c^4 + 8192*a^4*c + 57344*b*c^4 - 57344*b^4*c - 24576*a^5 + 24576*b^5 - 24576*c^5 + 49152*a^2*b^3 - 49152*a^3*b^2 + 147456*a^2*c^3 + 147456*a^3*c^2 - 49152*b^2*c^3 + 49152*b^3*c^2 + 245760*a*b^2*c^2 - 442368*a^2*b*c^2 + 245760*a^2*b^2*c - 163840*a*b*c^3 - 32768*a*b^3*c - 163840*a^3*b*c) + (-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(32768*a*b^5 - 253952*a*c^5 - 24576*a^5*c + 57344*b*c^5 + 57344*b^5*c - 24576*b^6 - 24576*c^6 + 16384*a^2*b^4 - 32768*a^3*b^3 + 8192*a^4*b^2 - 638976*a^2*c^4 - 638976*a^3*c^3 - 253952*a^4*c^2 + 24576*b^2*c^4 - 114688*b^3*c^3 + 24576*b^4*c^2 + (tan(x/2)*(16384*a*b^6 - 81920*a*c^6 + 49152*b*c^6 + 49152*b^6*c - 16384*b^7 - 16384*c^7 + 16384*a^2*b^5 - 16384*a^3*b^4 + 229376*a^2*c^5 + 491520*a^3*c^4 + 49152*a^4*c^3 - 147456*a^5*c^2 - 32768*b^2*c^5 - 32768*b^5*c^2 + 327680*a*b^3*c^3 - 425984*a*b^4*c^2 - 1015808*a^2*b*c^4 - 180224*a^2*b^4*c - 983040*a^3*b*c^3 - 65536*a^3*b^3*c + 49152*a^4*b*c^2 + 98304*a^4*b^2*c + 851968*a^2*b^2*c^3 + 131072*a^2*b^3*c^2 + 393216*a^3*b^2*c^2 + 65536*a*b*c^5 + 98304*a*b^5*c) + (-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(24576*b^2*c^6 - 393216*a^2*c^6 - 589824*a^3*c^5 - 393216*a^4*c^4 - 98304*a^5*c^3 - 98304*a*c^7 - 49152*b^3*c^5 + 49152*b^5*c^3 - 24576*b^6*c^2 + 98304*a*b^2*c^5 - 344064*a*b^3*c^4 + 98304*a*b^4*c^3 + 49152*a*b^5*c^2 + 589824*a^2*b*c^5 + 589824*a^3*b*c^4 + 196608*a^4*b*c^3 + 147456*a^2*b^2*c^4 - 344064*a^2*b^3*c^3 + 98304*a^3*b^2*c^3 - 49152*a^3*b^3*c^2 + 24576*a^4*b^2*c^2 + 196608*a*b*c^6 - tan(x/2)*(-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(65536*a*c^8 - 131072*a^2*c^7 - 262144*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c^4 - 16384*b^2*c^7 + 49152*b^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c^3 + 16384*b^7*c^2 + 294912*a*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^4*c^4 - 114688*a*b^5*c^3 - 16384*a*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*b*c^5 - 65536*a^4*b*c^4 - 655360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^4*c^2 - 114688*a^4*b^2*c^3 - 196608*a*b*c^7)))*(-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2) + 147456*a*b^2*c^3 - 458752*a*b^3*c^2 + 802816*a^2*b*c^3 - 245760*a^2*b^3*c + 557056*a^3*b*c^2 - 16384*a^3*b^2*c + 98304*a^2*b^2*c^2 + 425984*a*b*c^4 + 106496*a*b^4*c + 122880*a^4*b*c))*(-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*1i + (tan(x/2)*(57344*a^4*b - 57344*a*b^4 + 8192*a*c^4 + 8192*a^4*c + 57344*b*c^4 - 57344*b^4*c - 24576*a^5 + 24576*b^5 - 2457

6*c^5 + 49152*a^2*b^3 - 49152*a^3*b^2 + 147456*a^2*c^3 + 147456*a^3*c^2 - 4
9152*b^2*c^3 + 49152*b^3*c^2 + 245760*a*b^2*c^2 - 442368*a^2*b*c^2 + 245760
*a^2*b^2*c - 163840*a*b*c^3 - 32768*a*b^3*c - 163840*a^3*b*c) - (-(8*a*c^3
- b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*
(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(32768*a*b^5 - 253952*a*c^5 -
24576*a^5*c + 57344*b*c^5 + 57344*b^5*c - 24576*b^6 - 24576*c^6 + 16384*a^2
*b^4 - 32768*a^3*b^3 + 8192*a^4*b^2 - 638976*a^2*c^4 - 638976*a^3*c^3 - 253
952*a^4*c^2 + 24576*b^2*c^4 - 114688*b^3*c^3 + 24576*b^4*c^2 - (tan(x/2)*(1
6384*a*b^6 - 81920*a*c^6 + 49152*b*c^6 + 49152*b^6*c - 16384*b^7 - 16384*c^
7 + 16384*a^2*b^5 - 16384*a^3*b^4 + 229376*a^2*c^5 + 491520*a^3*c^4 + 49152
*a^4*c^3 - 147456*a^5*c^2 - 32768*b^2*c^5 - 32768*b^5*c^2 + 327680*a*b^3*c^
3 - 425984*a*b^4*c^2 - 1015808*a^2*b*c^4 - 180224*a^2*b^4*c - 983040*a^3*b*
c^3 - 65536*a^3*b^3*c + 49152*a^4*b*c^2 + 98304*a^4*b^2*c + 851968*a^2*b^2*
c^3 + 131072*a^2*b^3*c^2 + 393216*a^3*b^2*c^2 + 65536*a*b*c^5 + 98304*a*b^5
*c) - (-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2
 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(24576*b^2*c^
6 - 393216*a^2*c^6 - 589824*a^3*c^5 - 393216*a^4*c^4 - 98304*a^5*c^3 - 9830
4*a*c^7 - 49152*b^3*c^5 + 49152*b^5*c^3 - 24576*b^6*c^2 + 98304*a*b^2*c^5 -
 344064*a*b^3*c^4 + 98304*a*b^4*c^3 + 49152*a*b^5*c^2 + 589824*a^2*b*c^5 +
589824*a^3*b*c^4 + 196608*a^4*b*c^3 + 147456*a^2*b^2*c^4 - 344064*a^2*b^3*c
^3 + 98304*a^3*b^2*c^3 - 49152*a^3*b^3*c^2 + 24576*a^4*b^2*c^2 + 196608*a*b
c^6 + tan(x/2)(-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 -
 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(65
536*a*c^8 - 131072*a^2*c^7 - 262144*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c
^4 - 16384*b^2*c^7 + 49152*b^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*
b^6*c^3 + 16384*b^7*c^2 + 294912*a*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^
4*c^4 - 114688*a*b^5*c^3 - 16384*a*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*
b*c^5 - 65536*a^4*b*c^4 - 655360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a
^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 1
6384*a^3*b^4*c^2 - 114688*a^4*b^2*c^3 - 196608*a*b*c^7)))*(-(8*a*c^3 - b*(-
(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^
2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2) + 147456*a*b^2*c^3 - 458752*a*b^3*c^
2 + 802816*a^2*b*c^3 - 245760*a^2*b^3*c + 557056*a^3*b*c^2 - 16384*a^3*b^2*
c + 98304*a^2*b^2*c^2 + 425984*a*b*c^4 + 106496*a*b^4*c + 122880*a^4*b*c))*
(-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a
*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*1i)/((tan(x/2)*(573
44*a^4*b - 57344*a*b^4 + 8192*a*c^4 + 8192*a^4*c + 57344*b*c^4 - 57344*b^4*
c - 24576*a^5 + 24576*b^5 - 24576*c^5 + 49152*a^2*b^3 - 49152*a^3*b^2 + 147
456*a^2*c^3 + 147456*a^3*c^2 - 49152*b^2*c^3 + 49152*b^3*c^2 + 245760*a*b^2
*c^2 - 442368*a^2*b*c^2 + 245760*a^2*b^2*c - 163840*a*b*c^3 - 32768*a*b^3*c
 - 163840*a^3*b*c) - (-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*
c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2
)*(32768*a*b^5 - 253952*a*c^5 - 24576*a^5*c + 57344*b*c^5 + 57344*b^5*c - 2
4576*b^6 - 24576*c^6 + 16384*a^2*b^4 - 32768*a^3*b^3 + 8192*a^4*b^2 - 63897
6*a^2*c^4 - 638976*a^3*c^3 - 253952*a^4*c^2 + 24576*b^2*c^4 - 114688*b^3*c^

3 + 24576*b^4*c^2 - (tan(x/2)*(16384*a*b^6 - 81920*a*c^6 + 49152*b*c^6 + 49
152*b^6*c - 16384*b^7 - 16384*c^7 + 16384*a^2*b^5 - 16384*a^3*b^4 + 229376*
a^2*c^5 + 491520*a^3*c^4 + 49152*a^4*c^3 - 147456*a^5*c^2 - 32768*b^2*c^5 -
 32768*b^5*c^2 + 327680*a*b^3*c^3 - 425984*a*b^4*c^2 - 1015808*a^2*b*c^4 -
180224*a^2*b^4*c - 983040*a^3*b*c^3 - 65536*a^3*b^3*c + 49152*a^4*b*c^2 + 9
8304*a^4*b^2*c + 851968*a^2*b^2*c^3 + 131072*a^2*b^3*c^2 + 393216*a^3*b^2*c
^2 + 65536*a*b*c^5 + 98304*a*b^5*c) - (-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/
2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*
a*b^2*c^3)))^(1/2)*(24576*b^2*c^6 - 393216*a^2*c^6 - 589824*a^3*c^5 - 39321
6*a^4*c^4 - 98304*a^5*c^3 - 98304*a*c^7 - 49152*b^3*c^5 + 49152*b^5*c^3 - 2
4576*b^6*c^2 + 98304*a*b^2*c^5 - 344064*a*b^3*c^4 + 98304*a*b^4*c^3 + 49152
*a*b^5*c^2 + 589824*a^2*b*c^5 + 589824*a^3*b*c^4 + 196608*a^4*b*c^3 + 14745
6*a^2*b^2*c^4 - 344064*a^2*b^3*c^3 + 98304*a^3*b^2*c^3 - 49152*a^3*b^3*c^2
+ 24576*a^4*b^2*c^2 + 196608*a*b*c^6 + tan(x/2)*(-(8*a*c^3 - b*(-(4*a*c - b
^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^
4*c^2 - 8*a*b^2*c^3)))^(1/2)*(65536*a*c^8 - 131072*a^2*c^7 - 262144*a^3*c^6
 + 131072*a^4*c^5 + 196608*a^5*c^4 - 16384*b^2*c^7 + 49152*b^3*c^6 - 65536*
b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c^3 + 16384*b^7*c^2 + 294912*a*b^2*c^6
- 409600*a*b^3*c^5 + 376832*a*b^4*c^4 - 114688*a*b^5*c^3 - 16384*a*b^6*c^2
+ 589824*a^2*b*c^6 + 720896*a^3*b*c^5 - 65536*a^4*b*c^4 - 655360*a^2*b^2*c^
5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 557056*a^3
*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^4*c^2 - 114688*a^4*b^2*c^3 - 196
608*a*b*c^7)))*(-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 -
2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2) + 14
7456*a*b^2*c^3 - 458752*a*b^3*c^2 + 802816*a^2*b*c^3 - 245760*a^2*b^3*c + 5
57056*a^3*b*c^2 - 16384*a^3*b^2*c + 98304*a^2*b^2*c^2 + 425984*a*b*c^4 + 10
6496*a*b^4*c + 122880*a^4*b*c))*(-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b
^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*
c^3)))^(1/2) - (tan(x/2)*(57344*a^4*b - 57344*a*b^4 + 8192*a*c^4 + 8192*a^4
*c + 57344*b*c^4 - 57344*b^4*c - 24576*a^5 + 24576*b^5 - 24576*c^5 + 49152*
a^2*b^3 - 49152*a^3*b^2 + 147456*a^2*c^3 + 147456*a^3*c^2 - 49152*b^2*c^3 +
 49152*b^3*c^2 + 245760*a*b^2*c^2 - 442368*a^2*b*c^2 + 245760*a^2*b^2*c - 1
63840*a*b*c^3 - 32768*a*b^3*c - 163840*a^3*b*c) + (-(8*a*c^3 - b*(-(4*a*c -
 b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 +
b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(32768*a*b^5 - 253952*a*c^5 - 24576*a^5*c +
57344*b*c^5 + 57344*b^5*c - 24576*b^6 - 24576*c^6 + 16384*a^2*b^4 - 32768*a
^3*b^3 + 8192*a^4*b^2 - 638976*a^2*c^4 - 638976*a^3*c^3 - 253952*a^4*c^2 +
24576*b^2*c^4 - 114688*b^3*c^3 + 24576*b^4*c^2 + (tan(x/2)*(16384*a*b^6 - 8
1920*a*c^6 + 49152*b*c^6 + 49152*b^6*c - 16384*b^7 - 16384*c^7 + 16384*a^2*
b^5 - 16384*a^3*b^4 + 229376*a^2*c^5 + 491520*a^3*c^4 + 49152*a^4*c^3 - 147
456*a^5*c^2 - 32768*b^2*c^5 - 32768*b^5*c^2 + 327680*a*b^3*c^3 - 425984*a*b
^4*c^2 - 1015808*a^2*b*c^4 - 180224*a^2*b^4*c - 983040*a^3*b*c^3 - 65536*a^
3*b^3*c + 49152*a^4*b*c^2 + 98304*a^4*b^2*c + 851968*a^2*b^2*c^3 + 131072*a
^2*b^3*c^2 + 393216*a^3*b^2*c^2 + 65536*a*b*c^5 + 98304*a*b^5*c) + (-(8*a*c
^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/

(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(24576*b^2*c^6 - 393216*a^2
*c^6 - 589824*a^3*c^5 - 393216*a^4*c^4 - 98304*a^5*c^3 - 98304*a*c^7 - 4915
2*b^3*c^5 + 49152*b^5*c^3 - 24576*b^6*c^2 + 98304*a*b^2*c^5 - 344064*a*b^3*
c^4 + 98304*a*b^4*c^3 + 49152*a*b^5*c^2 + 589824*a^2*b*c^5 + 589824*a^3*b*c
^4 + 196608*a^4*b*c^3 + 147456*a^2*b^2*c^4 - 344064*a^2*b^3*c^3 + 98304*a^3
*b^2*c^3 - 49152*a^3*b^3*c^2 + 24576*a^4*b^2*c^2 + 196608*a*b*c^6 - tan(x/2
)*(-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6
*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*(65536*a*c^8 - 13
1072*a^2*c^7 - 262144*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c^4 - 16384*b^2
*c^7 + 49152*b^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c^3 + 1638
4*b^7*c^2 + 294912*a*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^4*c^4 - 114688
*a*b^5*c^3 - 16384*a*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*b*c^5 - 65536*
a^4*b*c^4 - 655360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^4*c^3 - 1
6384*a^2*b^5*c^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^4*c
^2 - 114688*a^4*b^2*c^3 - 196608*a*b*c^7))*(-(8*a*c^3 - b*(-(4*a*c - b^2)^
3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16*a^2*c^4 + b^4*c^
2 - 8*a*b^2*c^3)))^(1/2) + 147456*a*b^2*c^3 - 458752*a*b^3*c^2 + 802816*a^2
*b*c^3 - 245760*a^2*b^3*c + 557056*a^3*b*c^2 - 16384*a^3*b^2*c + 98304*a^2*
b^2*c^2 + 425984*a*b*c^4 + 106496*a*b^4*c + 122880*a^4*b*c))*(-(8*a*c^3 - b
*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(16
*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2) + 131072*a*b^3 - 131072*a^3*b + 2
62144*a*c^3 + 262144*a^3*c - 131072*b*c^3 + 131072*b^3*c + 65536*a^4 - 6553
6*b^4 + 65536*c^4 + 393216*a^2*c^2 - 393216*a*b*c^2 - 393216*a^2*b*c))*(-(8
*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2
c)/(2(16*a^2*c^4 + b^4*c^2 - 8*a*b^2*c^3)))^(1/2)*2i - (2*atan((344064*a^
4*tan(x/2))/(163840*a^3*c - 196608*a^3*b - 49152*a*c^3 - 98304*a*b^3 - 1638
4*b*c^3 + 344064*a^4 - 16384*b^4 + 16384*c^4 - 98304*a^2*b^2 - 98304*a^2*c^
2 + (147456*a^5)/c + (16384*b^5)/c + (16384*a*b^4)/c - (147456*a^4*b)/c - (
32768*a*b^5)/c^2 + (196608*a^2*b^3)/c - (229376*a^3*b^2)/c + (32768*a^2*b^4
)/c^2 + (32768*a^3*b^3)/c^2 - (32768*a^4*b^2)/c^2 + 65536*a*b*c^2 + 98304*a
*b^2*c + 32768*a^2*b*c) - (16384*b^4*tan(x/2))/(163840*a^3*c - 196608*a^3*b
 - 49152*a*c^3 - 98304*a*b^3 - 16384*b*c^3 + 344064*a^4 - 16384*b^4 + 16384
*c^4 - 98304*a^2*b^2 - 98304*a^2*c^2 + (147456*a^5)/c + (16384*b^5)/c + (16
384*a*b^4)/c - (147456*a^4*b)/c - (32768*a*b^5)/c^2 + (196608*a^2*b^3)/c -
(229376*a^3*b^2)/c + (32768*a^2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32768*a^4
*b^2)/c^2 + 65536*a*b*c^2 + 98304*a*b^2*c + 32768*a^2*b*c) + (16384*c^4*tan
(x/2))/(163840*a^3*c - 196608*a^3*b - 49152*a*c^3 - 98304*a*b^3 - 16384*b*c
^3 + 344064*a^4 - 16384*b^4 + 16384*c^4 - 98304*a^2*b^2 - 98304*a^2*c^2 + (
147456*a^5)/c + (16384*b^5)/c + (16384*a*b^4)/c - (147456*a^4*b)/c - (32768
*a*b^5)/c^2 + (196608*a^2*b^3)/c - (229376*a^3*b^2)/c + (32768*a^2*b^4)/c^2
 + (32768*a^3*b^3)/c^2 - (32768*a^4*b^2)/c^2 + 65536*a*b*c^2 + 98304*a*b^2*
c + 32768*a^2*b*c) + (147456*a^5*tan(x/2))/(16384*a*b^4 - 147456*a^4*b - 49
152*a*c^4 + 344064*a^4*c - 16384*b*c^4 - 16384*b^4*c + 147456*a^5 + 16384*b
^5 + 16384*c^5 + 196608*a^2*b^3 - 229376*a^3*b^2 - 98304*a^2*c^3 + 163840*a
^3*c^2 + 98304*a*b^2*c^2 + 32768*a^2*b*c^2 - 98304*a^2*b^2*c - (32768*a*b^5

)/c + (32768*a^2*b^4)/c + (32768*a^3*b^3)/c - (32768*a^4*b^2)/c + 65536*a*b
*c^3 - 98304*a*b^3*c - 196608*a^3*b*c) + (16384*b^5*tan(x/2))/(16384*a*b^4
- 147456*a^4*b - 49152*a*c^4 + 344064*a^4*c - 16384*b*c^4 - 16384*b^4*c + 1
47456*a^5 + 16384*b^5 + 16384*c^5 + 196608*a^2*b^3 - 229376*a^3*b^2 - 98304
*a^2*c^3 + 163840*a^3*c^2 + 98304*a*b^2*c^2 + 32768*a^2*b*c^2 - 98304*a^2*b
^2*c - (32768*a*b^5)/c + (32768*a^2*b^4)/c + (32768*a^3*b^3)/c - (32768*a^4
*b^2)/c + 65536*a*b*c^3 - 98304*a*b^3*c - 196608*a^3*b*c) - (98304*a^2*b^2*
tan(x/2))/(163840*a^3*c - 196608*a^3*b - 49152*a*c^3 - 98304*a*b^3 - 16384*
b*c^3 + 344064*a^4 - 16384*b^4 + 16384*c^4 - 98304*a^2*b^2 - 98304*a^2*c^2
+ (147456*a^5)/c + (16384*b^5)/c + (16384*a*b^4)/c - (147456*a^4*b)/c - (32
768*a*b^5)/c^2 + (196608*a^2*b^3)/c - (229376*a^3*b^2)/c + (32768*a^2*b^4)/
c^2 + (32768*a^3*b^3)/c^2 - (32768*a^4*b^2)/c^2 + 65536*a*b*c^2 + 98304*a*b
^2*c + 32768*a^2*b*c) - (98304*a^2*c^2*tan(x/2))/(163840*a^3*c - 196608*a^3
*b - 49152*a*c^3 - 98304*a*b^3 - 16384*b*c^3 + 344064*a^4 - 16384*b^4 + 163
84*c^4 - 98304*a^2*b^2 - 98304*a^2*c^2 + (147456*a^5)/c + (16384*b^5)/c + (
16384*a*b^4)/c - (147456*a^4*b)/c - (32768*a*b^5)/c^2 + (196608*a^2*b^3)/c
- (229376*a^3*b^2)/c + (32768*a^2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32768*a
^4*b^2)/c^2 + 65536*a*b*c^2 + 98304*a*b^2*c + 32768*a^2*b*c) + (16384*a*b^4
*tan(x/2))/(16384*a*b^4 - 147456*a^4*b - 49152*a*c^4 + 344064*a^4*c - 16384
*b*c^4 - 16384*b^4*c + 147456*a^5 + 16384*b^5 + 16384*c^5 + 196608*a^2*b^3
- 229376*a^3*b^2 - 98304*a^2*c^3 + 163840*a^3*c^2 + 98304*a*b^2*c^2 + 32768
*a^2*b*c^2 - 98304*a^2*b^2*c - (32768*a*b^5)/c + (32768*a^2*b^4)/c + (32768
*a^3*b^3)/c - (32768*a^4*b^2)/c + 65536*a*b*c^3 - 98304*a*b^3*c - 196608*a^
3*b*c) - (147456*a^4*b*tan(x/2))/(16384*a*b^4 - 147456*a^4*b - 49152*a*c^4
+ 344064*a^4*c - 16384*b*c^4 - 16384*b^4*c + 147456*a^5 + 16384*b^5 + 16384
*c^5 + 196608*a^2*b^3 - 229376*a^3*b^2 - 98304*a^2*c^3 + 163840*a^3*c^2 + 9
8304*a*b^2*c^2 + 32768*a^2*b*c^2 - 98304*a^2*b^2*c - (32768*a*b^5)/c + (327
68*a^2*b^4)/c + (32768*a^3*b^3)/c - (32768*a^4*b^2)/c + 65536*a*b*c^3 - 983
04*a*b^3*c - 196608*a^3*b*c) - (32768*a*b^5*tan(x/2))/(147456*a^5*c - 49152
*a*c^5 - 32768*a*b^5 - 16384*b*c^5 + 16384*b^5*c + 16384*c^6 + 32768*a^2*b^
4 + 32768*a^3*b^3 - 32768*a^4*b^2 - 98304*a^2*c^4 + 163840*a^3*c^3 + 344064
*a^4*c^2 - 16384*b^4*c^2 + 98304*a*b^2*c^3 - 98304*a*b^3*c^2 + 32768*a^2*b*
c^3 + 196608*a^2*b^3*c - 196608*a^3*b*c^2 - 229376*a^3*b^2*c - 98304*a^2*b^
2*c^2 + 65536*a*b*c^4 + 16384*a*b^4*c - 147456*a^4*b*c) + (196608*a^2*b^3*t
an(x/2))/(16384*a*b^4 - 147456*a^4*b - 49152*a*c^4 + 344064*a^4*c - 16384*b
*c^4 - 16384*b^4*c + 147456*a^5 + 16384*b^5 + 16384*c^5 + 196608*a^2*b^3 -
229376*a^3*b^2 - 98304*a^2*c^3 + 163840*a^3*c^2 + 98304*a*b^2*c^2 + 32768*a
^2*b*c^2 - 98304*a^2*b^2*c - (32768*a*b^5)/c + (32768*a^2*b^4)/c + (32768*a
^3*b^3)/c - (32768*a^4*b^2)/c + 65536*a*b*c^3 - 98304*a*b^3*c - 196608*a^3*
b*c) - (229376*a^3*b^2*tan(x/2))/(16384*a*b^4 - 147456*a^4*b - 49152*a*c^4
+ 344064*a^4*c - 16384*b*c^4 - 16384*b^4*c + 147456*a^5 + 16384*b^5 + 16384
*c^5 + 196608*a^2*b^3 - 229376*a^3*b^2 - 98304*a^2*c^3 + 163840*a^3*c^2 + 9
8304*a*b^2*c^2 + 32768*a^2*b*c^2 - 98304*a^2*b^2*c - (32768*a*b^5)/c + (327
68*a^2*b^4)/c + (32768*a^3*b^3)/c - (32768*a^4*b^2)/c + 65536*a*b*c^3 - 983
04*a*b^3*c - 196608*a^3*b*c) - (98304*a*b^3*tan(x/2))/(163840*a^3*c - 19660

8*a^3*b - 49152*a*c^3 - 98304*a*b^3 - 16384*b*c^3 + 344064*a^4 - 16384*b^4
+ 16384*c^4 - 98304*a^2*b^2 - 98304*a^2*c^2 + (147456*a^5)/c + (16384*b^5)/
c + (16384*a*b^4)/c - (147456*a^4*b)/c - (32768*a*b^5)/c^2 + (196608*a^2*b^
3)/c - (229376*a^3*b^2)/c + (32768*a^2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32
768*a^4*b^2)/c^2 + 65536*a*b*c^2 + 98304*a*b^2*c + 32768*a^2*b*c) - (196608
*a^3*b*tan(x/2))/(163840*a^3*c - 196608*a^3*b - 49152*a*c^3 - 98304*a*b^3 -
 16384*b*c^3 + 344064*a^4 - 16384*b^4 + 16384*c^4 - 98304*a^2*b^2 - 98304*a
^2*c^2 + (147456*a^5)/c + (16384*b^5)/c + (16384*a*b^4)/c - (147456*a^4*b)/
c - (32768*a*b^5)/c^2 + (196608*a^2*b^3)/c - (229376*a^3*b^2)/c + (32768*a^
2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32768*a^4*b^2)/c^2 + 65536*a*b*c^2 + 98
304*a*b^2*c + 32768*a^2*b*c) - (49152*a*c^3*tan(x/2))/(163840*a^3*c - 19660
8*a^3*b - 49152*a*c^3 - 98304*a*b^3 - 16384*b*c^3 + 344064*a^4 - 16384*b^4
+ 16384*c^4 - 98304*a^2*b^2 - 98304*a^2*c^2 + (147456*a^5)/c + (16384*b^5)/
c + (16384*a*b^4)/c - (147456*a^4*b)/c - (32768*a*b^5)/c^2 + (196608*a^2*b^
3)/c - (229376*a^3*b^2)/c + (32768*a^2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32
768*a^4*b^2)/c^2 + 65536*a*b*c^2 + 98304*a*b^2*c + 32768*a^2*b*c) + (163840
*a^3*c*tan(x/2))/(163840*a^3*c - 196608*a^3*b - 49152*a*c^3 - 98304*a*b^3 -
 16384*b*c^3 + 344064*a^4 - 16384*b^4 + 16384*c^4 - 98304*a^2*b^2 - 98304*a
^2*c^2 + (147456*a^5)/c + (16384*b^5)/c + (16384*a*b^4)/c - (147456*a^4*b)/
c - (32768*a*b^5)/c^2 + (196608*a^2*b^3)/c - (229376*a^3*b^2)/c + (32768*a^
2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32768*a^4*b^2)/c^2 + 65536*a*b*c^2 + 98
304*a*b^2*c + 32768*a^2*b*c) - (16384*b*c^3*tan(x/2))/(163840*a^3*c - 19660
8*a^3*b - 49152*a*c^3 - 98304*a*b^3 - 16384*b*c^3 + 344064*a^4 - 16384*b^4
+ 16384*c^4 - 98304*a^2*b^2 - 98304*a^2*c^2 + (147456*a^5)/c + (16384*b^5)/
c + (16384*a*b^4)/c - (147456*a^4*b)/c - (32768*a*b^5)/c^2 + (196608*a^2*b^
3)/c - (229376*a^3*b^2)/c + (32768*a^2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32
768*a^4*b^2)/c^2 + 65536*a*b*c^2 + 98304*a*b^2*c + 32768*a^2*b*c) + (32768*
a^2*b^4*tan(x/2))/(147456*a^5*c - 49152*a*c^5 - 32768*a*b^5 - 16384*b*c^5 +
 16384*b^5*c + 16384*c^6 + 32768*a^2*b^4 + 32768*a^3*b^3 - 32768*a^4*b^2 -
98304*a^2*c^4 + 163840*a^3*c^3 + 344064*a^4*c^2 - 16384*b^4*c^2 + 98304*a*b
^2*c^3 - 98304*a*b^3*c^2 + 32768*a^2*b*c^3 + 196608*a^2*b^3*c - 196608*a^3*
b*c^2 - 229376*a^3*b^2*c - 98304*a^2*b^2*c^2 + 65536*a*b*c^4 + 16384*a*b^4*
c - 147456*a^4*b*c) + (32768*a^3*b^3*tan(x/2))/(147456*a^5*c - 49152*a*c^5
- 32768*a*b^5 - 16384*b*c^5 + 16384*b^5*c + 16384*c^6 + 32768*a^2*b^4 + 327
68*a^3*b^3 - 32768*a^4*b^2 - 98304*a^2*c^4 + 163840*a^3*c^3 + 344064*a^4*c^
2 - 16384*b^4*c^2 + 98304*a*b^2*c^3 - 98304*a*b^3*c^2 + 32768*a^2*b*c^3 + 1
96608*a^2*b^3*c - 196608*a^3*b*c^2 - 229376*a^3*b^2*c - 98304*a^2*b^2*c^2 +
 65536*a*b*c^4 + 16384*a*b^4*c - 147456*a^4*b*c) - (32768*a^4*b^2*tan(x/2))
/(147456*a^5*c - 49152*a*c^5 - 32768*a*b^5 - 16384*b*c^5 + 16384*b^5*c + 16
384*c^6 + 32768*a^2*b^4 + 32768*a^3*b^3 - 32768*a^4*b^2 - 98304*a^2*c^4 + 1
63840*a^3*c^3 + 344064*a^4*c^2 - 16384*b^4*c^2 + 98304*a*b^2*c^3 - 98304*a*
b^3*c^2 + 32768*a^2*b*c^3 + 196608*a^2*b^3*c - 196608*a^3*b*c^2 - 229376*a^
3*b^2*c - 98304*a^2*b^2*c^2 + 65536*a*b*c^4 + 16384*a*b^4*c - 147456*a^4*b*
c) + (65536*a*b*c^2*tan(x/2))/(163840*a^3*c - 196608*a^3*b - 49152*a*c^3 -
98304*a*b^3 - 16384*b*c^3 + 344064*a^4 - 16384*b^4 + 16384*c^4 - 98304*a^2*

```
b^2 - 98304*a^2*c^2 + (147456*a^5)/c + (16384*b^5)/c + (16384*a*b^4)/c - (1
47456*a^4*b)/c - (32768*a*b^5)/c^2 + (196608*a^2*b^3)/c - (229376*a^3*b^2)/
c + (32768*a^2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32768*a^4*b^2)/c^2 + 65536
*a*b*c^2 + 98304*a*b^2*c + 32768*a^2*b*c) + (98304*a*b^2*c*tan(x/2))/(16384
0*a^3*c - 196608*a^3*b - 49152*a*c^3 - 98304*a*b^3 - 16384*b*c^3 + 344064*a
^4 - 16384*b^4 + 16384*c^4 - 98304*a^2*b^2 - 98304*a^2*c^2 + (147456*a^5)/c
 + (16384*b^5)/c + (16384*a*b^4)/c - (147456*a^4*b)/c - (32768*a*b^5)/c^2 +
 (196608*a^2*b^3)/c - (229376*a^3*b^2)/c + (32768*a^2*b^4)/c^2 + (32768*a^3
*b^3)/c^2 - (32768*a^4*b^2)/c^2 + 65536*a*b*c^2 + 98304*a*b^2*c + 32768*a^2
*b*c) + (32768*a^2*b*c*tan(x/2))/(163840*a^3*c - 196608*a^3*b - 49152*a*c^3
 - 98304*a*b^3 - 16384*b*c^3 + 344064*a^4 - 16384*b^4 + 16384*c^4 - 98304*a
^2*b^2 - 98304*a^2*c^2 + (147456*a^5)/c + (16384*b^5)/c + (16384*a*b^4)/c -
 (147456*a^4*b)/c - (32768*a*b^5)/c^2 + (196608*a^2*b^3)/c - (229376*a^3*b^
2)/c + (32768*a^2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32768*a^4*b^2)/c^2 + 65
536*a*b*c^2 + 98304*a*b^2*c + 32768*a^2*b*c)))/c
```

3.8 $\int \frac{\csc^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 19, antiderivative size = 326

$$\begin{aligned}
\int \frac{\csc^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx &= -\frac{2bc\left(1+\frac{b^2-2c(a+c)}{b\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{(a-b+c)(a+b+c)\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} \\
&\quad -\frac{2bc\left(1-\frac{b^2-2c(a+c)}{b\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{(a-b+c)(a+b+c)\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}} \\
&\quad -\frac{\sin(x)}{2(a+b+c)(1-\cos(x))}+\frac{\sin(x)}{2(a-b+c)(1+\cos(x))}
\end{aligned}$$

```
[Out] -1/2*sin(x)/(a+b+c)/(1-cos(x))+1/2*sin(x)/(a-b+c)/(1+cos(x))-2*b*c*arctan((
b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2)
)*(1+(b^2-2*c*(a+c))/b/(-4*a*c+b^2)^(1/2))/(a-b+c)/(a+b+c)/(b-2*c-(-4*a*c+b
^2)^(1/2))^(1/2)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2)-2*b*c*arctan((b-2*c+(-4*a
*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2))*(1+(-b^2+
2*c*(a+c))/b/(-4*a*c+b^2)^(1/2))/(a-b+c)/(a+b+c)/(b-2*c+(-4*a*c+b^2)^(1/2))
^(1/2)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2)
```

Rubi [A] (verified)

Time = 3.46 (sec) , antiderivative size = 326, normalized size of antiderivative = 1.00, number of steps used = 9, number of rules used = 5, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.263, Rules used = {3348, 2727, 3374, 2738, 211}

$$\int \frac{\csc^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{2bc\left(\frac{b^2-2c(a+c)}{b\sqrt{b^2-4ac}}+1\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{-\sqrt{b^2-4ac}+b-2c}}{\sqrt{-\sqrt{b^2-4ac}+b+2c}}\right)}{(a-b+c)(a+b+c)\sqrt{-\sqrt{b^2-4ac}+b-2c}\sqrt{-\sqrt{b^2-4ac}+b+2c}} - \frac{2bc\left(1-\frac{b^2-2c(a+c)}{b\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{\sqrt{b^2-4ac}+b-2c}}{\sqrt{\sqrt{b^2-4ac}+b+2c}}\right)}{(a-b+c)(a+b+c)\sqrt{\sqrt{b^2-4ac}+b-2c}\sqrt{\sqrt{b^2-4ac}+b+2c}} - \frac{\sin(x)}{2(1-\cos(x))(a+b+c)} + \frac{\sin(x)}{2(\cos(x)+1)(a-b+c)}$$

```
[In] Int[Csc[x]^2/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (-2*b*c*(1 + (b^2 - 2*c*(a + c))/(b*Sqrt[b^2 - 4*a*c]))*ArcTan[(Sqrt[b - 2*
c - Sqrt[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]])/((a -
b + c)*(a + b + c)*Sqrt[b - 2*c - Sqrt[b^2 - 4*a*c]]*Sqrt[b + 2*c - Sqrt[b^
2 - 4*a*c]]) - (2*b*c*(1 - (b^2 - 2*c*(a + c))/(b*Sqrt[b^2 - 4*a*c]))*ArcTa
n[(Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c + Sqrt[b^2 - 4*
a*c]]])/((a - b + c)*(a + b + c)*Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]*Sqrt[b +
 2*c + Sqrt[b^2 - 4*a*c]]) - Sin[x]/(2*(a + b + c)*(1 - Cos[x])) + Sin[x]/(
2*(a - b + c)*(1 + Cos[x]))
```

Rule 211

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(Rt[a/b, 2]/a)*ArcTan[x/R
t[a/b, 2]], x] /; FreeQ[{a, b}, x] && PosQ[a/b]
```

Rule 2727

```
Int[((a_) + (b_.)*sin[(c_.) + (d_.)*(x_)])^(-1), x_Symbol] :> Simp[-Cos[c +
 d*x]/(d*(b + a*Sin[c + d*x])), x] /; FreeQ[{a, b, c, d}, x] && EqQ[a^2 - b
^2, 0]
```

Rule 2738

```
Int[((a_) + (b_.)*sin[Pi/2 + (c_.) + (d_.)*(x_)])^(-1), x_Symbol] :> With[{
e = FreeFactors[Tan[(c + d*x)/2], x]}, Dist[2*(e/d), Subst[Int[1/(a + b + (
a - b)*e^2*x^2), x], x, Tan[(c + d*x)/2]/e], x]] /; FreeQ[{a, b, c, d}, x]
&& NeQ[a^2 - b^2, 0]
```

Rule 3348

```
Int[((a_.) + cos[(d_.) + (e_.)*(x_)]^(n_.)*(b_.) + cos[(d_.) + (e_.)*(x_)]^
(n2_.)*(c_.))^(p_.)*sin[(d_.) + (e_.)*(x_)]^(m_.), x_Symbol] :> Int[ExpandT
rig[(1 - cos[d + e*x]^2)^(m/2)*(a + b*cos[d + e*x]^n + c*cos[d + e*x]^(2*n)
)^p, x], x] /; FreeQ[{a, b, c, d, e}, x] && EqQ[n2, 2*n] && IntegerQ[m/2] &
& NeQ[b^2 - 4*a*c, 0] && IntegersQ[n, p]
```

Rule 3374

```
Int[(cos[(d_.) + (e_.)*(x_)]*(B_.) + (A_))/((a_.) + cos[(d_.) + (e_.)*(x_)]
*(b_.) + cos[(d_.) + (e_.)*(x_)]^2*(c_.)), x_Symbol] :> Module[{q = Rt[b^2
- 4*a*c, 2]}, Dist[B + (b*B - 2*A*c)/q, Int[1/(b + q + 2*c*Cos[d + e*x]), x
], x] + Dist[B - (b*B - 2*A*c)/q, Int[1/(b - q + 2*c*Cos[d + e*x]), x], x]]
 /; FreeQ[{a, b, c, d, e, A, B}, x] && NeQ[b^2 - 4*a*c, 0]
```

Rubi steps

$$\text{integral} = \int \left(-\frac{1}{2(a+b+c)(-1+\cos(x))} + \frac{1}{2(a-b+c)(1+\cos(x))} + \frac{-b^2\left(1-\frac{c(a+c)}{b^2}\right) - bc\cos(x)}{(a-b+c)(a+b+c)\left(a+b\cos(x)+c\cos^2(x)\right)}\right) dx$$

$$= \frac{\int \frac{1}{1+\cos(x)}\,dx}{2(a-b+c)} - \frac{\int \frac{1}{-1+\cos(x)}\,dx}{2(a+b+c)} + \frac{\int \frac{-b^2\left(1-\frac{c(a+c)}{b^2}\right)-bc\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx}{(a-b+c)(a+b+c)}$$

$$= -\frac{\sin(x)}{2(a+b+c)(1-\cos(x))} + \frac{\sin(x)}{2(a-b+c)(1+\cos(x))} - \frac{\left(c\left(b+\frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}\right)\right)\int \frac{1}{b-\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{(a-b+c)(a+b+c)} - \frac{\left(bc\left(1-\frac{b^2-2c(a+c)}{b\sqrt{b^2-4ac}}\right)\right)\int \frac{1}{b+\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{(a-b+c)(a+b+c)}$$

$$= -\frac{\sin(x)}{2(a+b+c)(1-\cos(x))} + \frac{\sin(x)}{2(a-b+c)(1+\cos(x))} - \frac{\left(2c\left(b+\frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}\right)\right)\text{Subst}\left(\int \frac{1}{b+2c-\sqrt{b^2-4ac}+\left(b-2c-\sqrt{b^2-4ac}\right)x^2}\,dx, x, \tan\left(\frac{x}{2}\right)\right)}{(a-b+c)(a+b+c)} - \frac{\left(2bc\left(1-\frac{b^2-2c(a+c)}{b\sqrt{b^2-4ac}}\right)\right)\text{Subst}\left(\int \frac{1}{b+2c+\sqrt{b^2-4ac}+\left(b-2c+\sqrt{b^2-4ac}\right)x^2}\,dx, x, \tan\left(\frac{x}{2}\right)\right)}{(a-b+c)(a+b+c)}$$

$$= -\frac{2c\left(b+\frac{b^2-2c(a+c)}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{(a-b+c)(a+b+c)\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} - \frac{2bc\left(1-\frac{b^2-2c(a+c)}{b\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{(a-b+c)(a+b+c)\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}} - \frac{\sin(x)}{2(a+b+c)(1-\cos(x))} + \frac{\sin(x)}{2(a-b+c)(1+\cos(x))}$$

Mathematica [A] (verified)

Time = 0.70 (sec) , antiderivative size = 335, normalized size of antiderivative = 1.03

$$\int \frac{\csc^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{\sqrt{2}c\left(-b^2+2c(a+c)+b\sqrt{b^2-4ac}\right)\text{arctanh}\left(\frac{\left(b-2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)-2b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\,(a^2-b^2+2ac+c^2)\sqrt{-b^2+2c(a+c)-b\sqrt{b^2-4ac}}} - \frac{\sqrt{2}c\left(b^2-2c(a+c)+b\sqrt{b^2-4ac}\right)\text{arctanh}\left(\frac{\left(-b+2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)+2b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\,(a^2-b^2+2ac+c^2)\sqrt{-b^2+2c(a+c)+b\sqrt{b^2-4ac}}} - \frac{\cot\left(\frac{x}{2}\right)}{2(a+b+c)} + \frac{\tan\left(\frac{x}{2}\right)}{2(a-b+c)}$$

```
[In] Integrate[Csc[x]^2/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (Sqrt[2]*c*(-b^2 + 2*c*(a + c) + b*Sqrt[b^2 - 4*a*c])*ArcTanh[((b - 2*c + S
qrt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) - 2*b*Sqrt[b^2 - 4*a*
c]]])/(Sqrt[b^2 - 4*a*c]*(a^2 - b^2 + 2*a*c + c^2)*Sqrt[-b^2 + 2*c*(a + c)
- b*Sqrt[b^2 - 4*a*c]]) - (Sqrt[2]*c*(b^2 - 2*c*(a + c) + b*Sqrt[b^2 - 4*a*
c])*ArcTanh[((-b + 2*c + Sqrt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a
+ c) + 2*b*Sqrt[b^2 - 4*a*c]]])/(Sqrt[b^2 - 4*a*c]*(a^2 - b^2 + 2*a*c + c^2
)*Sqrt[-b^2 + 2*c*(a + c) + b*Sqrt[b^2 - 4*a*c]]) - Cot[x/2]/(2*(a + b + c)
) + Tan[x/2]/(2*(a - b + c))
```

Maple [A] (verified)

Time = 11.98 (sec) , antiderivative size = 393, normalized size of antiderivative = 1.21

method	result
default	$\frac{\tan\left(\frac{x}{2}\right)}{2a-2b+2c}+\frac{(4a-4b+4c)\left(\frac{\left(\sqrt{-4ac+b^2}\,ac-\sqrt{-4ac+b^2}\,b^2+\sqrt{-4ac+b^2}\,bc+\sqrt{-4ac+b^2}\,c^2-3cab+2a\,c^2+b^3-b^2c-b\,c^2+2c^3\right)\operatorname{arctanh}\left(\frac{(}{\sqrt{(\sqrt{-}}}\right.}{2\sqrt{-4ac+b^2}\,(a-b+c)\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}\right.}{}$
risch	Expression too large to display

```
[In] int(csc(x)^2/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] 1/2/(a-b+c)*tan(1/2*x)+1/2/(a+b+c)/(a-b+c)*(4*a-4*b+4*c)*(1/2*((-4*a*c+b^2)
^(1/2)*a*c-(-4*a*c+b^2)^(1/2)*b^2+(-4*a*c+b^2)^(1/2)*b*c+(-4*a*c+b^2)^(1/2)
*c^2-3*c*a*b+2*a*c^2+b^3-b^2*c-b*c^2+2*c^3)/(-4*a*c+b^2)^(1/2)/(a-b+c)/(((-
4*a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2)*arctanh((-a+b-c)*tan(1/2*x)/(((-4*a*c+
b^2)^(1/2)-a+c)*(a-b+c))^(1/2))+1/2*((-4*a*c+b^2)^(1/2)*a*c-(-4*a*c+b^2)^(1
/2)*b^2+(-4*a*c+b^2)^(1/2)*b*c+(-4*a*c+b^2)^(1/2)*c^2+3*c*a*b-2*a*c^2-b^3+b
^2*c+b*c^2-2*c^3)/(-4*a*c+b^2)^(1/2)/(a-b+c)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b
+c))^(1/2)*arctan((a-b+c)*tan(1/2*x)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/
2)))-1/2/(a+b+c)/tan(1/2*x)
```

Fricas [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 16741 vs. 2(279) = 558.

Time = 3.38 (sec) , antiderivative size = 16741, normalized size of antiderivative = 51.35

$$\int \frac{\csc^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx=\text{Too large to display}$$

```
[In] integrate(csc(x)^2/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
```

```
[Out] Too large to include
```

Sympy [F]

$$\int \frac{\csc^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \int \frac{\csc^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$$

```
[In] integrate(csc(x)**2/(a+b*cos(x)+c*cos(x)**2),x)
```

```
[Out] Integral(csc(x)**2/(a + b*cos(x) + c*cos(x)**2), x)
```

Maxima [F]

$$\int \frac{\csc^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \int \frac{\csc(x)^2}{c\cos(x)^2+b\cos(x)+a}\,dx$$

```
[In] integrate(csc(x)^2/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
```

```
[Out] -(2*b*cos(2*x)*sin(x) + ((a^2 - b^2 + 2*a*c + c^2)*cos(2*x)^2 + (a^2 - b^2
+ 2*a*c + c^2)*sin(2*x)^2 + a^2 - b^2 + 2*a*c + c^2 - 2*(a^2 - b^2 + 2*a*c
+ c^2)*cos(2*x))*integrate(2*(2*b^2*c*cos(3*x)^2 + 2*b^2*c*cos(x)^2 + 2*b^2
*c*sin(3*x)^2 + 2*b^2*c*sin(x)^2 + b*c^2*cos(x) + 4*(2*a*b^2 - 3*a*c^2 - c^
3 - (2*a^2 - b^2)*c)*cos(2*x)^2 + 4*(2*a*b^2 - 3*a*c^2 - c^3 - (2*a^2 - b^2
)*c)*sin(2*x)^2 + 2*(2*b^3 - b*c^2)*sin(2*x)*sin(x) + (b*c^2*cos(3*x) + b*c
^2*cos(x) + 2*(b^2*c - a*c^2 - c^3)*cos(2*x))*cos(4*x) + (4*b^2*c*cos(x) +
b*c^2 + 2*(2*b^3 - b*c^2)*cos(2*x))*cos(3*x) + 2*(b^2*c - a*c^2 - c^3 + (2*
b^3 - b*c^2)*cos(x))*cos(2*x) + (b*c^2*sin(3*x) + b*c^2*sin(x) + 2*(b^2*c -
 a*c^2 - c^3)*sin(2*x))*sin(4*x) + 2*(2*b^2*c*sin(x) + (2*b^3 - b*c^2)*sin(
2*x))*sin(3*x))/(2*a*c^3 + c^4 + (a^2 - b^2)*c^2 + (2*a*c^3 + c^4 + (a^2 -
b^2)*c^2)*cos(4*x)^2 + 4*(a^2*b^2 - b^4 + 2*a*b^2*c + b^2*c^2)*cos(3*x)^2 +
 4*(4*a^4 - 4*a^2*b^2 + 6*a*c^3 + c^4 + (13*a^2 - b^2)*c^2 + 4*(3*a^3 - a*b
^2)*c)*cos(2*x)^2 + 4*(a^2*b^2 - b^4 + 2*a*b^2*c + b^2*c^2)*cos(x)^2 + (2*a
*c^3 + c^4 + (a^2 - b^2)*c^2)*sin(4*x)^2 + 4*(a^2*b^2 - b^4 + 2*a*b^2*c + b
^2*c^2)*sin(3*x)^2 + 4*(4*a^4 - 4*a^2*b^2 + 6*a*c^3 + c^4 + (13*a^2 - b^2)*
c^2 + 4*(3*a^3 - a*b^2)*c)*sin(2*x)^2 + 8*(2*a^3*b - 2*a*b^3 + 4*a*b*c^2 +
b*c^3 + (5*a^2*b - b^3)*c)*sin(2*x)*sin(x) + 4*(a^2*b^2 - b^4 + 2*a*b^2*c +
 b^2*c^2)*sin(x)^2 + 2*(2*a*c^3 + c^4 + (a^2 - b^2)*c^2 + 2*(2*a*b*c^2 + b*
c^3 + (a^2*b - b^3)*c)*cos(3*x) + 2*(4*a*c^3 + c^4 + (5*a^2 - b^2)*c^2 + 2*
(a^3 - a*b^2)*c)*cos(2*x) + 2*(2*a*b*c^2 + b*c^3 + (a^2*b - b^3)*c)*cos(x))
*cos(4*x) + 4*(2*a*b*c^2 + b*c^3 + (a^2*b - b^3)*c + 2*(2*a^3*b - 2*a*b^3 +
 4*a*b*c^2 + b*c^3 + (5*a^2*b - b^3)*c)*cos(2*x) + 2*(a^2*b^2 - b^4 + 2*a*b
^2*c + b^2*c^2)*cos(x))*cos(3*x) + 4*(4*a*c^3 + c^4 + (5*a^2 - b^2)*c^2 + 2
*(a^3 - a*b^2)*c + 2*(2*a^3*b - 2*a*b^3 + 4*a*b*c^2 + b*c^3 + (5*a^2*b - b^
3)*c)*cos(x))*cos(2*x) + 4*(2*a*b*c^2 + b*c^3 + (a^2*b - b^3)*c)*cos(x) + 4
*((2*a*b*c^2 + b*c^3 + (a^2*b - b^3)*c)*sin(3*x) + (4*a*c^3 + c^4 + (5*a^2
```

```
- b^2)*c^2 + 2*(a^3 - a*b^2)*c)*sin(2*x) + (2*a*b*c^2 + b*c^3 + (a^2*b - b^
3)*c)*sin(x))*sin(4*x) + 8*((2*a^3*b - 2*a*b^3 + 4*a*b*c^2 + b*c^3 + (5*a^2
*b - b^3)*c)*sin(2*x) + (a^2*b^2 - b^4 + 2*a*b^2*c + b^2*c^2)*sin(x))*sin(3
*x)), x) - 2*(b*cos(x) - a - c)*sin(2*x) - 2*b*sin(x))/((a^2 - b^2 + 2*a*c
+ c^2)*cos(2*x)^2 + (a^2 - b^2 + 2*a*c + c^2)*sin(2*x)^2 + a^2 - b^2 + 2*a*
c + c^2 - 2*(a^2 - b^2 + 2*a*c + c^2)*cos(2*x))
```

Giac [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 21750 vs. $2(279) = 558$.

Time = 2.63 (sec) , antiderivative size = 21750, normalized size of antiderivative = 66.72

$$\int \frac{\csc^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(csc(x)^2/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] ((2*a^2*b^6 - 4*a*b^7 + 2*b^8 - 18*a^3*b^4*c + 34*a^2*b^5*c - 10*a*b^6*c -
6*b^7*c + 48*a^4*b^2*c^2 - 80*a^3*b^3*c^2 - 22*a^2*b^4*c^2 + 52*a*b^5*c^2 +
 4*b^6*c^2 - 32*a^5*c^3 + 32*a^4*b*c^3 + 144*a^3*b^2*c^3 - 128*a^2*b^3*c^3
- 38*a*b^4*c^3 + 2*b^5*c^3 - 96*a^4*c^4 + 64*a^3*b*c^4 + 112*a^2*b^2*c^4 -
16*a*b^3*c^4 - 2*b^4*c^4 - 96*a^3*c^5 + 32*a^2*b*c^5 + 16*a*b^2*c^5 - 32*a^
2*c^6 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(
b^2 - 4*a*c)*a^2*b^4 - 2*(b^2 - 4*a*c)*a^2*b^4 - 2*sqrt(a^2 - a*b + b*c - c
^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^5 + 4*(b^2 - 4*a*
c)*a*b^5 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sq
rt(b^2 - 4*a*c)*b^6 - 2*(b^2 - 4*a*c)*b^6 - 15*sqrt(a^2 - a*b + b*c - c^2 -
 sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c + 10*(b^2 - 4*a
*c)*a^3*b^2*c + 7*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c
))*sqrt(b^2 - 4*a*c)*a^2*b^3*c - 18*(b^2 - 4*a*c)*a^2*b^3*c + 41*sqrt(a^2 -
 a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4*c
 + 2*(b^2 - 4*a*c)*a*b^4*c + 11*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a
*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5*c + 6*(b^2 - 4*a*c)*b^5*c + 12*sqrt(
a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^
4*c^2 - 8*(b^2 - 4*a*c)*a^4*c^2 + 4*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^2 + 8*(b^2 - 4*a*c)*a^3*b*c^
2 - 101*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^
2 - 4*a*c)*a^2*b^2*c^2 + 30*(b^2 - 4*a*c)*a^2*b^2*c^2 - 62*sqrt(a^2 - a*b +
 b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c^2 - 2
8*(b^2 - 4*a*c)*a*b^3*c^2 - 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c^2 - 4*(b^2 - 4*a*c)*b^4*c^2 + 68*sqr
t(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*
a^3*c^3 - 24*(b^2 - 4*a*c)*a^3*c^3 + 72*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b
^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^3 + 16*(b^2 - 4*a*c)*a^2
*b*c^3 + 15*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
```

t(b^2 - 4*a*c)*a*b^2*c^3 + 22*(b^2 - 4*a*c)*a*b^2*c^3 - sqrt(a^2 - a*b + b*
c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^3*c^3 - 2*(b^2
 - 4*a*c)*b^3*c^3 + 36*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*sqrt(b^2 - 4*a*c)*a^2*c^4 - 24*(b^2 - 4*a*c)*a^2*c^4 + 4*sqrt(a^2 -
 a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*c^4
 + 8*(b^2 - 4*a*c)*a*b*c^4 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c^4 + 2*(b^2 - 4*a*c)*b^2*c^4 - 20*sq
rt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)
*a*c^5 - 8*(b^2 - 4*a*c)*a*c^5)*(a^2 - b^2 + 2*a*c + c^2)^2*abs(-a + b - c)
 + (4*a^4*b^6 - 8*a^3*b^7 + 8*a*b^9 - 4*b^10 - 36*a^5*b^4*c + 76*a^4*b^5*c
+ 8*a^3*b^6*c - 96*a^2*b^7*c + 44*a*b^8*c + 4*b^9*c + 96*a^6*b^2*c^2 - 224*
a^5*b^3*c^2 - 84*a^4*b^4*c^2 + 432*a^3*b^5*c^2 - 160*a^2*b^6*c^2 - 72*a*b^7
*c^2 + 12*b^8*c^2 - 64*a^7*c^3 + 192*a^6*b*c^3 + 288*a^5*b^2*c^3 - 896*a^4*
b^3*c^3 + 152*a^3*b^4*c^3 + 456*a^2*b^5*c^3 - 104*a*b^6*c^3 - 16*b^7*c^3 -
320*a^6*c^4 + 768*a^5*b*c^4 + 320*a^4*b^2*c^4 - 1216*a^3*b^3*c^4 + 216*a^2*
b^4*c^4 + 176*a*b^5*c^4 - 4*b^6*c^4 - 640*a^5*c^5 + 1152*a^4*b*c^5 + 192*a^
3*b^2*c^5 - 640*a^2*b^3*c^5 + 12*a*b^4*c^5 + 12*b^5*c^5 - 640*a^4*c^6 + 768
*a^3*b*c^6 + 96*a^2*b^2*c^6 - 96*a*b^3*c^6 - 4*b^4*c^6 - 320*a^3*c^7 + 192*
a^2*b*c^7 + 32*a*b^2*c^7 - 64*a^2*c^8 - 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt
(b^2 - 4*a*c)*(a - b + c))*a^5*b^4 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^
2 - 4*a*c)*(a - b + c))*a^4*b^5 + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
 4*a*c)*(a - b + c))*a^3*b^6 - 10*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4
*a*c)*(a - b + c))*a^2*b^7 - 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*a*b^8 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*b^9 + 15*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
 c))*a^6*b^2*c - 28*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
 c))*a^5*b^3*c - 48*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
 c))*a^4*b^4*c + 76*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
 c))*a^3*b^5*c + 39*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
 c))*a^2*b^6*c - 48*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
 c))*a*b^7*c - 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c)
)*b^8*c - 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^
7*c^2 + 32*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^6*
b*c^2 + 122*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^5
*b^2*c^2 - 192*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*
a^4*b^3*c^2 - 174*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c
))*a^3*b^4*c^2 + 190*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^2*b^5*c^2 + 52*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*a*b^6*c^2 - 10*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*b^7*c^2 - 104*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
 + c))*a^6*c^3 + 192*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^5*b*c^3 + 333*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
 + c))*a^4*b^2*c^3 - 392*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*a^3*b^3*c^3 - 196*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*
(a - b + c))*a^2*b^4*c^3 + 76*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c

```
)*(a - b + c))*a*b^5*c^3 + 23*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*b^6*c^3 - 276*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)
*(a - b + c))*a^5*c^4 + 320*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*
(a - b + c))*a^4*b*c^4 + 412*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)
*(a - b + c))*a^3*b^2*c^4 - 176*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a
*c)*(a - b + c))*a^2*b^3*c^4 - 143*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
4*a*c)*(a - b + c))*a*b^4*c^4 - 11*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
4*a*c)*(a - b + c))*b^5*c^4 - 304*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4
*a*c)*(a - b + c))*a^4*c^5 + 128*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*
a*c)*(a - b + c))*a^3*b*c^5 + 233*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4
*a*c)*(a - b + c))*a^2*b^2*c^5 + 68*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
 4*a*c)*(a - b + c))*a*b^3*c^5 - 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
 4*a*c)*(a - b + c))*b^4*c^5 - 116*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
4*a*c)*(a - b + c))*a^3*c^6 - 96*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*
a*c)*(a - b + c))*a^2*b*c^6 + 42*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*
a*c)*(a - b + c))*a*b^2*c^6 + 16*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*
a*c)*(a - b + c))*b^3*c^6 + 24*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*a^2*c^7 - 64*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)
*(a - b + c))*a*b*c^7 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*b^2*c^7 + 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
 b + c))*a*c^8 - 4*(b^2 - 4*a*c)*a^4*b^4 + 8*(b^2 - 4*a*c)*a^3*b^5 - 8*(b^2
 - 4*a*c)*a*b^7 + 4*(b^2 - 4*a*c)*b^8 + 20*(b^2 - 4*a*c)*a^5*b^2*c - 44*(b^
2 - 4*a*c)*a^4*b^3*c - 8*(b^2 - 4*a*c)*a^3*b^4*c + 64*(b^2 - 4*a*c)*a^2*b^5
*c - 28*(b^2 - 4*a*c)*a*b^6*c - 4*(b^2 - 4*a*c)*b^7*c - 16*(b^2 - 4*a*c)*a^
6*c^2 + 48*(b^2 - 4*a*c)*a^5*b*c^2 + 52*(b^2 - 4*a*c)*a^4*b^2*c^2 - 176*(b^
2 - 4*a*c)*a^3*b^3*c^2 + 48*(b^2 - 4*a*c)*a^2*b^4*c^2 + 56*(b^2 - 4*a*c)*a*
b^5*c^2 - 12*(b^2 - 4*a*c)*b^6*c^2 - 80*(b^2 - 4*a*c)*a^5*c^3 + 192*(b^2 -
4*a*c)*a^4*b*c^3 + 40*(b^2 - 4*a*c)*a^3*b^2*c^3 - 232*(b^2 - 4*a*c)*a^2*b^3
*c^3 + 56*(b^2 - 4*a*c)*a*b^4*c^3 + 16*(b^2 - 4*a*c)*b^5*c^3 - 160*(b^2 - 4
*a*c)*a^4*c^4 + 288*(b^2 - 4*a*c)*a^3*b*c^4 + 8*(b^2 - 4*a*c)*a^2*b^2*c^4 -
 112*(b^2 - 4*a*c)*a*b^3*c^4 + 4*(b^2 - 4*a*c)*b^4*c^4 - 160*(b^2 - 4*a*c)*
a^3*c^5 + 192*(b^2 - 4*a*c)*a^2*b*c^5 + 4*(b^2 - 4*a*c)*a*b^2*c^5 - 12*(b^2
 - 4*a*c)*b^3*c^5 - 80*(b^2 - 4*a*c)*a^2*c^6 + 48*(b^2 - 4*a*c)*a*b*c^6 + 4
*(b^2 - 4*a*c)*b^2*c^6 - 16*(b^2 - 4*a*c)*a*c^7)*abs(-a^2 + b^2 - 2*a*c - c
^2)*abs(-a + b - c) - (2*a^7*b^5 - 4*a^6*b^6 - 2*a^5*b^7 + 8*a^4*b^8 - 2*a^
3*b^9 - 4*a^2*b^10 + 2*a*b^11 - 14*a^8*b^3*c + 26*a^7*b^4*c + 28*a^6*b^5*c
 - 70*a^5*b^6*c + 46*a^3*b^8*c - 12*a^2*b^9*c - 2*a*b^10*c - 2*b^11*c + 24*a
^9*b*c^2 - 36*a^8*b^2*c^2 - 120*a^7*b^3*c^2 + 190*a^6*b^4*c^2 + 102*a^5*b^5
*c^2 - 194*a^4*b^6*c^2 - 4*a^3*b^7*c^2 + 18*a^2*b^8*c^2 + 14*a*b^9*c^2 + 6*
b^10*c^2 - 16*a^9*c^3 + 160*a^8*b*c^3 - 128*a^7*b^2*c^3 - 384*a^6*b^3*c^3 +
 338*a^5*b^4*c^3 + 176*a^4*b^5*c^3 - 60*a^3*b^6*c^3 - 32*a^2*b^7*c^3 - 54*a
*b^8*c^3 - 96*a^8*c^4 + 416*a^7*b*c^4 - 112*a^6*b^2*c^4 - 504*a^5*b^3*c^4 +
 70*a^4*b^4*c^4 + 46*a^3*b^5*c^4 + 184*a^2*b^6*c^4 + 22*a*b^7*c^4 - 18*b^8*
c^4 - 224*a^7*c^5 + 480*a^6*b*c^5 + 96*a^5*b^2*c^5 - 76*a^4*b^3*c^5 - 306*a
^3*b^4*c^5 - 180*a^2*b^5*c^5 + 130*a*b^6*c^5 + 16*b^7*c^5 - 224*a^6*c^6 + 8
```

0*a^5*b*c^6 + 200*a^4*b^2*c^6 + 472*a^3*b^3*c^6 - 262*a^2*b^4*c^6 - 150*a*b
^5*c^6 + 14*b^6*c^6 - 416*a^4*b*c^7 + 64*a^3*b^2*c^7 + 464*a^2*b^3*c^7 - 58
*a*b^4*c^7 - 24*b^5*c^7 + 224*a^4*c^8 - 480*a^3*b*c^8 - 48*a^2*b^2*c^8 + 15
2*a*b^3*c^8 + 2*b^4*c^8 + 224*a^3*c^9 - 224*a^2*b*c^9 - 32*a*b^2*c^9 + 10*b
^3*c^9 + 96*a^2*c^10 - 40*a*b*c^10 - 4*b^2*c^10 + 16*a*c^11 + 3*sqrt(a^2 -
a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^7*b^3
- 2*(b^2 - 4*a*c)*a^7*b^3 - 2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^6*b^4 + 4*(b^2 - 4*a*c)*a^6*b^4 - 11*sqr
t(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*
a^5*b^5 + 2*(b^2 - 4*a*c)*a^5*b^5 + 4*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^6 - 8*(b^2 - 4*a*c)*a^4*b^6
+ 13*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
 4*a*c)*a^3*b^7 + 2*(b^2 - 4*a*c)*a^3*b^7 - 2*sqrt(a^2 - a*b + b*c - c^2 -
sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^8 + 4*(b^2 - 4*a*c)*
a^2*b^8 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*a*b^9 - 2*(b^2 - 4*a*c)*a*b^9 - 9*sqrt(a^2 - a*b + b*c - c^2
 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^8*b*c + 6*(b^2 - 4*a*
c)*a^8*b*c + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*
sqrt(b^2 - 4*a*c)*a^7*b^2*c - 10*(b^2 - 4*a*c)*a^7*b^2*c + 58*sqrt(a^2 - a*
b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^6*b^3*c
- 20*(b^2 - 4*a*c)*a^6*b^3*c - sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*b^4*c + 38*(b^2 - 4*a*c)*a^5*b^4*c -
92*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4
*a*c)*a^4*b^5*c + 8*(b^2 - 4*a*c)*a^4*b^5*c - 15*sqrt(a^2 - a*b + b*c - c^2
 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^6*c - 30*(b^2 - 4
*a*c)*a^3*b^6*c + 38*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*sqrt(b^2 - 4*a*c)*a^2*b^7*c + 4*(b^2 - 4*a*c)*a^2*b^7*c + 13*sqrt(a^2
 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^8
c + 2(b^2 - 4*a*c)*a*b^8*c + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^9*c + 2*(b^2 - 4*a*c)*b^9*c + 6*sqrt(
a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^
8*c^2 - 4*(b^2 - 4*a*c)*a^8*c^2 - 76*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
- 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^7*b*c^2 + 40*(b^2 - 4*a*c)*a^7*b*
c^2 - 43*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b
^2 - 4*a*c)*a^6*b^2*c^2 - 38*(b^2 - 4*a*c)*a^6*b^2*c^2 + 209*sqrt(a^2 - a*b
 + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*b^3*c^2
 - 70*(b^2 - 4*a*c)*a^5*b^3*c^2 + 109*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^4*c^2 + 74*(b^2 - 4*a*c)*a^4
*b^4*c^2 - 94*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*s
qrt(b^2 - 4*a*c)*a^3*b^5*c^2 + 20*(b^2 - 4*a*c)*a^3*b^5*c^2 - 69*sqrt(a^2 -
 a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^6
c^2 - 10(b^2 - 4*a*c)*a^2*b^6*c^2 - 47*sqrt(a^2 - a*b + b*c - c^2 - sqrt(
b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^7*c^2 - 6*(b^2 - 4*a*c)*a*b
^7*c^2 - 11*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*b^8*c^2 - 6*(b^2 - 4*a*c)*b^8*c^2 + 52*sqrt(a^2 - a*b + b*c

- c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^7*c^3 - 24*(b^2
- 4*a*c)*a^7*c^3 - 156*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*sqrt(b^2 - 4*a*c)*a^6*b*c^3 + 104*(b^2 - 4*a*c)*a^6*b*c^3 - 205*sqr
t(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*
a^5*b^2*c^3 - 42*(b^2 - 4*a*c)*a^5*b^2*c^3 + 100*sqrt(a^2 - a*b + b*c - c^2
- sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^3*c^3 - 96*(b^2 -
4*a*c)*a^4*b^3*c^3 + 162*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*sqrt(b^2 - 4*a*c)*a^3*b^4*c^3 + 20*(b^2 - 4*a*c)*a^3*b^4*c^3 + 1
48*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4
*a*c)*a^2*b^5*c^3 + 8*(b^2 - 4*a*c)*a^2*b^5*c^3 + 47*sqrt(a^2 - a*b + b*c -
c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^6*c^3 + 30*(b^2
- 4*a*c)*a*b^6*c^3 - 4*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*sqrt(b^2 - 4*a*c)*b^7*c^3 + 132*sqrt(a^2 - a*b + b*c - c^2 - sqrt(
b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^6*c^4 - 56*(b^2 - 4*a*c)*a^6*
c^4 - 36*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b
^2 - 4*a*c)*a^5*b*c^4 + 120*(b^2 - 4*a*c)*a^5*b*c^4 - 211*sqrt(a^2 - a*b +
b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^2*c^4 +
10*(b^2 - 4*a*c)*a^4*b^2*c^4 - 251*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^3*c^4 - 14*(b^2 - 4*a*c)*a^3*b^
3*c^4 - 84*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt
(b^2 - 4*a*c)*a^2*b^4*c^4 - 64*(b^2 - 4*a*c)*a^2*b^4*c^4 + 73*sqrt(a^2 - a*
b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^5*c^4
- 22*(b^2 - 4*a*c)*a*b^5*c^4 + 33*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4
*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^6*c^4 + 18*(b^2 - 4*a*c)*b^6*c^4 + 1
00*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4
*a*c)*a^5*c^5 - 56*(b^2 - 4*a*c)*a^5*c^5 + 194*sqrt(a^2 - a*b + b*c - c^2 -
sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b*c^5 + 20*(b^2 - 4*a
*c)*a^4*b*c^5 + 113*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c^5 + 50*(b^2 - 4*a*c)*a^3*b^2*c^5 - 198*sqr
t(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*
a^2*b^3*c^5 + 92*(b^2 - 4*a*c)*a^2*b^3*c^5 - 161*sqrt(a^2 - a*b + b*c - c^2
- sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4*c^5 - 58*(b^2 - 4
*a*c)*a*b^4*c^5 - 24*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*sqrt(b^2 - 4*a*c)*b^5*c^5 - 16*(b^2 - 4*a*c)*b^5*c^5 - 80*sqrt(a^2 -
a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*c^6
+ 172*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*a^3*b*c^6 - 104*(b^2 - 4*a*c)*a^3*b*c^6 + 263*sqrt(a^2 - a*b + b*c
- c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c^6 + 30*
(b^2 - 4*a*c)*a^2*b^2*c^6 + 39*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c^6 + 86*(b^2 - 4*a*c)*a*b^3*c^6 -
23*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4
*a*c)*b^4*c^6 - 14*(b^2 - 4*a*c)*b^4*c^6 - 164*sqrt(a^2 - a*b + b*c - c^2 -
sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^7 + 56*(b^2 - 4*a*c
)*a^3*c^7 - 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*
sqrt(b^2 - 4*a*c)*a^2*b*c^7 - 120*(b^2 - 4*a*c)*a^2*b*c^7 + 89*sqrt(a^2 - a

```
*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^7
 + 2*(b^2 - 4*a*c)*a*b^2*c^7 + 40*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4
*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^3*c^7 + 24*(b^2 - 4*a*c)*b^3*c^7 - 6
8*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*a^2*c^8 + 56*(b^2 - 4*a*c)*a^2*c^8 - 60*sqrt(a^2 - a*b + b*c - c^2 - s
qrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*c^8 - 56*(b^2 - 4*a*c)*
a*b*c^8 - 9*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*b^2*c^8 - 2*(b^2 - 4*a*c)*b^2*c^8 + 12*sqrt(a^2 - a*b + b*c
- c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*c^9 + 24*(b^2 -
4*a*c)*a*c^9 - 17*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c
))*sqrt(b^2 - 4*a*c)*b*c^9 - 10*(b^2 - 4*a*c)*b*c^9 + 10*sqrt(a^2 - a*b + b
*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*c^10 + 4*(b^2 -
 4*a*c)*c^10)*abs(-a + b - c))*(pi*floor(1/2*x/pi + 1/2) + arctan(2*sqrt(1/
2)*tan(1/2*x)/sqrt((2*a^3 - 2*a*b^2 + 2*a^2*c + 2*b^2*c - 2*a*c^2 - 2*c^3 +
 sqrt(-4*(a^3 + a^2*b - a*b^2 - b^3 + 3*a^2*c + 2*a*b*c - b^2*c + 3*a*c^2 +
 b*c^2 + c^3)*(a^3 - a^2*b - a*b^2 + b^3 + 3*a^2*c - 2*a*b*c - b^2*c + 3*a*
c^2 - b*c^2 + c^3) + 4*(a^3 - a*b^2 + a^2*c + b^2*c - a*c^2 - c^3)^2))/(a^3
 - a^2*b - a*b^2 + b^3 + 3*a^2*c - 2*a*b*c - b^2*c + 3*a*c^2 - b*c^2 + c^3)
)))/((3*a^10*b^2 - 8*a^9*b^3 - 7*a^8*b^4 + 32*a^7*b^5 - 2*a^6*b^6 - 48*a^5*
b^7 + 18*a^4*b^8 + 32*a^3*b^9 - 17*a^2*b^10 - 8*a*b^11 + 5*b^12 - 12*a^11*c
 + 32*a^10*b*c + 66*a^9*b^2*c - 208*a^8*b^3*c - 80*a^7*b^4*c + 448*a^6*b^5*
c - 36*a^5*b^6*c - 416*a^4*b^7*c + 108*a^3*b^8*c + 160*a^2*b^9*c - 46*a*b^1
0*c - 16*b^11*c - 152*a^10*c^2 + 320*a^9*b*c^2 + 543*a^8*b^2*c^2 - 1344*a^7
*b^3*c^2 - 532*a^6*b^4*c^2 + 1920*a^5*b^5*c^2 + 50*a^4*b^6*c^2 - 1088*a^3*b
^7*c^2 + 84*a^2*b^8*c^2 + 192*a*b^9*c^2 + 7*b^10*c^2 - 764*a^9*c^3 + 1280*a
^8*b*c^3 + 2072*a^7*b^2*c^3 - 3744*a^6*b^3*c^3 - 1712*a^5*b^4*c^3 + 3424*a^
4*b^5*c^3 + 520*a^3*b^6*c^3 - 992*a^2*b^7*c^3 - 116*a*b^8*c^3 + 32*b^9*c^3
- 2080*a^8*c^4 + 2688*a^7*b*c^4 + 4342*a^6*b^2*c^4 - 5264*a^5*b^3*c^4 - 289
0*a^4*b^4*c^4 + 2720*a^3*b^5*c^4 + 802*a^2*b^6*c^4 - 272*a*b^7*c^4 - 46*b^8
*c^4 - 3416*a^7*c^5 + 3136*a^6*b*c^5 + 5388*a^5*b^2*c^5 - 3648*a^4*b^3*c^5
- 2672*a^3*b^4*c^5 + 768*a^2*b^5*c^5 + 412*a*b^6*c^5 - 3472*a^6*c^6 + 1792*
a^5*b*c^6 + 4006*a^4*b^2*c^6 - 768*a^3*b^3*c^6 - 1300*a^2*b^4*c^6 - 64*a*b^
5*c^6 + 46*b^6*c^6 - 2072*a^5*c^7 + 1688*a^3*b^2*c^7 + 416*a^2*b^3*c^7 - 27
2*a*b^4*c^7 - 32*b^5*c^7 - 544*a^4*c^8 - 640*a^3*b*c^8 + 327*a^2*b^2*c^8 +
216*a*b^3*c^8 - 7*b^4*c^8 + 100*a^3*c^9 - 352*a^2*b*c^9 + 2*a*b^2*c^9 + 16*
b^3*c^9 + 104*a^2*c^10 - 64*a*b*c^10 - 5*b^2*c^10 + 20*a*c^11)*abs(-a^2 + b
^2 - 2*a*c - c^2)) - ((2*a^2*b^6 - 4*a*b^7 + 2*b^8 - 18*a^3*b^4*c + 34*a^2*
b^5*c - 10*a*b^6*c - 6*b^7*c + 48*a^4*b^2*c^2 - 80*a^3*b^3*c^2 - 22*a^2*b^4
*c^2 + 52*a*b^5*c^2 + 4*b^6*c^2 - 32*a^5*c^3 + 32*a^4*b*c^3 + 144*a^3*b^2*c
^3 - 128*a^2*b^3*c^3 - 38*a*b^4*c^3 + 2*b^5*c^3 - 96*a^4*c^4 + 64*a^3*b*c^4
 + 112*a^2*b^2*c^4 - 16*a*b^3*c^4 - 2*b^4*c^4 - 96*a^3*c^5 + 32*a^2*b*c^5 +
 16*a*b^2*c^5 - 32*a^2*c^6 + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*
c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^4 - 2*(b^2 - 4*a*c)*a^2*b^4 - 2*sqr
t(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*
a*b^5 + 4*(b^2 - 4*a*c)*a*b^5 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4
```

```
*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^6 - 2*(b^2 - 4*a*c)*b^6 - 15*sqrt(a^
2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*
b^2*c + 10*(b^2 - 4*a*c)*a^3*b^2*c + 7*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^
2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3*c - 18*(b^2 - 4*a*c)*a^2*
b^3*c + 41*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt
(b^2 - 4*a*c)*a*b^4*c + 2*(b^2 - 4*a*c)*a*b^4*c + 11*sqrt(a^2 - a*b + b*c -
 c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5*c + 6*(b^2 - 4*
a*c)*b^5*c + 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))
*sqrt(b^2 - 4*a*c)*a^4*c^2 - 8*(b^2 - 4*a*c)*a^4*c^2 + 4*sqrt(a^2 - a*b + b
*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^2 + 8*(
b^2 - 4*a*c)*a^3*b*c^2 - 101*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)
*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c^2 + 30*(b^2 - 4*a*c)*a^2*b^2*c^2
- 62*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
 4*a*c)*a*b^3*c^2 - 28*(b^2 - 4*a*c)*a*b^3*c^2 - 6*sqrt(a^2 - a*b + b*c - c
^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c^2 - 4*(b^2 - 4*
a*c)*b^4*c^2 + 68*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
))*sqrt(b^2 - 4*a*c)*a^3*c^3 - 24*(b^2 - 4*a*c)*a^3*c^3 + 72*sqrt(a^2 - a*b
 + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^3 +
 16*(b^2 - 4*a*c)*a^2*b*c^3 + 15*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^3 + 22*(b^2 - 4*a*c)*a*b^2*c^3
- sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*b^3*c^3 - 2*(b^2 - 4*a*c)*b^3*c^3 + 36*sqrt(a^2 - a*b + b*c - c^2 + sq
rt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^4 - 24*(b^2 - 4*a*c)*a
^2*c^4 + 4*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt
(b^2 - 4*a*c)*a*b*c^4 + 8*(b^2 - 4*a*c)*a*b*c^4 + 5*sqrt(a^2 - a*b + b*c -
c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c^4 + 2*(b^2 - 4
*a*c)*b^2*c^4 - 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*sqrt(b^2 - 4*a*c)*a*c^5 - 8*(b^2 - 4*a*c)*a*c^5)*(a^2 - b^2 + 2*a*c + c
^2)^2*abs(-a + b - c) + (4*a^4*b^6 - 8*a^3*b^7 + 8*a*b^9 - 4*b^10 - 36*a^5*
b^4*c + 76*a^4*b^5*c + 8*a^3*b^6*c - 96*a^2*b^7*c + 44*a*b^8*c + 4*b^9*c +
96*a^6*b^2*c^2 - 224*a^5*b^3*c^2 - 84*a^4*b^4*c^2 + 432*a^3*b^5*c^2 - 160*a
^2*b^6*c^2 - 72*a*b^7*c^2 + 12*b^8*c^2 - 64*a^7*c^3 + 192*a^6*b*c^3 + 288*a
^5*b^2*c^3 - 896*a^4*b^3*c^3 + 152*a^3*b^4*c^3 + 456*a^2*b^5*c^3 - 104*a*b^
6*c^3 - 16*b^7*c^3 - 320*a^6*c^4 + 768*a^5*b*c^4 + 320*a^4*b^2*c^4 - 1216*a
^3*b^3*c^4 + 216*a^2*b^4*c^4 + 176*a*b^5*c^4 - 4*b^6*c^4 - 640*a^5*c^5 + 11
52*a^4*b*c^5 + 192*a^3*b^2*c^5 - 640*a^2*b^3*c^5 + 12*a*b^4*c^5 + 12*b^5*c^
5 - 640*a^4*c^6 + 768*a^3*b*c^6 + 96*a^2*b^2*c^6 - 96*a*b^3*c^6 - 4*b^4*c^6
 - 320*a^3*c^7 + 192*a^2*b*c^7 + 32*a*b^2*c^7 - 64*a^2*c^8 + 3*sqrt(a^2 - a
*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^5*b^4 - 5*sqrt(a^2 - a*b
+ b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*b^5 - 6*sqrt(a^2 - a*b + b
*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^6 + 10*sqrt(a^2 - a*b + b*c
 - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^7 + 3*sqrt(a^2 - a*b + b*c -
c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^8 - 5*sqrt(a^2 - a*b + b*c - c^2 +
 sqrt(b^2 - 4*a*c)*(a - b + c))*b^9 - 15*sqrt(a^2 - a*b + b*c - c^2 + sqrt(
b^2 - 4*a*c)*(a - b + c))*a^6*b^2*c + 28*sqrt(a^2 - a*b + b*c - c^2 + sqrt(
```

b^2 - 4*a*c)*(a - b + c))*a^5*b^3*c + 48*sqrt(a^2 - a*b + b*c - c^2 + sqrt(
b^2 - 4*a*c)*(a - b + c))*a^4*b^4*c - 76*sqrt(a^2 - a*b + b*c - c^2 + sqrt(
b^2 - 4*a*c)*(a - b + c))*a^3*b^5*c - 39*sqrt(a^2 - a*b + b*c - c^2 + sqrt(
b^2 - 4*a*c)*(a - b + c))*a^2*b^6*c + 48*sqrt(a^2 - a*b + b*c - c^2 + sqrt(
b^2 - 4*a*c)*(a - b + c))*a*b^7*c + 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
 - 4*a*c)*(a - b + c))*b^8*c + 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4
*a*c)*(a - b + c))*a^7*c^2 - 32*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a
c)(a - b + c))*a^6*b*c^2 - 122*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*
a*c)*(a - b + c))*a^5*b^2*c^2 + 192*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
 4*a*c)*(a - b + c))*a^4*b^3*c^2 + 174*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^
2 - 4*a*c)*(a - b + c))*a^3*b^4*c^2 - 190*sqrt(a^2 - a*b + b*c - c^2 + sqrt
(b^2 - 4*a*c)*(a - b + c))*a^2*b^5*c^2 - 52*sqrt(a^2 - a*b + b*c - c^2 + sq
rt(b^2 - 4*a*c)*(a - b + c))*a*b^6*c^2 + 10*sqrt(a^2 - a*b + b*c - c^2 + sq
rt(b^2 - 4*a*c)*(a - b + c))*b^7*c^2 + 104*sqrt(a^2 - a*b + b*c - c^2 + sqr
t(b^2 - 4*a*c)*(a - b + c))*a^6*c^3 - 192*sqrt(a^2 - a*b + b*c - c^2 + sqrt
(b^2 - 4*a*c)*(a - b + c))*a^5*b*c^3 - 333*sqrt(a^2 - a*b + b*c - c^2 + sqr
t(b^2 - 4*a*c)*(a - b + c))*a^4*b^2*c^3 + 392*sqrt(a^2 - a*b + b*c - c^2 +
sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^3*c^3 + 196*sqrt(a^2 - a*b + b*c - c^2
 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^4*c^3 - 76*sqrt(a^2 - a*b + b*c - c
^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^5*c^3 - 23*sqrt(a^2 - a*b + b*c - c
^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^6*c^3 + 276*sqrt(a^2 - a*b + b*c - c^
2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^5*c^4 - 320*sqrt(a^2 - a*b + b*c - c^2
 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*b*c^4 - 412*sqrt(a^2 - a*b + b*c - c^
2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^2*c^4 + 176*sqrt(a^2 - a*b + b*c -
 c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^3*c^4 + 143*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^4*c^4 + 11*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^5*c^4 + 304*sqrt(a^2 - a*b + b*c
 - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*c^5 - 128*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b*c^5 - 233*sqrt(a^2 - a*b + b*c
 - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^2*c^5 - 68*sqrt(a^2 - a*b + b
*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^3*c^5 + 12*sqrt(a^2 - a*b + b
*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^4*c^5 + 116*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*c^6 + 96*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b*c^6 - 42*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^2*c^6 - 16*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^3*c^6 - 24*sqrt(a^2 - a*b + b*c -
c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*c^7 + 64*sqrt(a^2 - a*b + b*c - c^
2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b*c^7 + 5*sqrt(a^2 - a*b + b*c - c^2 +
 sqrt(b^2 - 4*a*c)*(a - b + c))*b^2*c^7 - 20*sqrt(a^2 - a*b + b*c - c^2 + s
qrt(b^2 - 4*a*c)*(a - b + c))*a*c^8 - 4*(b^2 - 4*a*c)*a^4*b^4 + 8*(b^2 - 4*
a*c)*a^3*b^5 - 8*(b^2 - 4*a*c)*a*b^7 + 4*(b^2 - 4*a*c)*b^8 + 20*(b^2 - 4*a*
c)*a^5*b^2*c - 44*(b^2 - 4*a*c)*a^4*b^3*c - 8*(b^2 - 4*a*c)*a^3*b^4*c + 64*
(b^2 - 4*a*c)*a^2*b^5*c - 28*(b^2 - 4*a*c)*a*b^6*c - 4*(b^2 - 4*a*c)*b^7*c
- 16*(b^2 - 4*a*c)*a^6*c^2 + 48*(b^2 - 4*a*c)*a^5*b*c^2 + 52*(b^2 - 4*a*c)*
a^4*b^2*c^2 - 176*(b^2 - 4*a*c)*a^3*b^3*c^2 + 48*(b^2 - 4*a*c)*a^2*b^4*c^2

```
+ 56*(b^2 - 4*a*c)*a*b^5*c^2 - 12*(b^2 - 4*a*c)*b^6*c^2 - 80*(b^2 - 4*a*c)*
a^5*c^3 + 192*(b^2 - 4*a*c)*a^4*b*c^3 + 40*(b^2 - 4*a*c)*a^3*b^2*c^3 - 232*
(b^2 - 4*a*c)*a^2*b^3*c^3 + 56*(b^2 - 4*a*c)*a*b^4*c^3 + 16*(b^2 - 4*a*c)*b
^5*c^3 - 160*(b^2 - 4*a*c)*a^4*c^4 + 288*(b^2 - 4*a*c)*a^3*b*c^4 + 8*(b^2 -
 4*a*c)*a^2*b^2*c^4 - 112*(b^2 - 4*a*c)*a*b^3*c^4 + 4*(b^2 - 4*a*c)*b^4*c^4
 - 160*(b^2 - 4*a*c)*a^3*c^5 + 192*(b^2 - 4*a*c)*a^2*b*c^5 + 4*(b^2 - 4*a*c
)*a*b^2*c^5 - 12*(b^2 - 4*a*c)*b^3*c^5 - 80*(b^2 - 4*a*c)*a^2*c^6 + 48*(b^2
 - 4*a*c)*a*b*c^6 + 4*(b^2 - 4*a*c)*b^2*c^6 - 16*(b^2 - 4*a*c)*a*c^7)*abs(-
a^2 + b^2 - 2*a*c - c^2)*abs(-a + b - c) - (2*a^7*b^5 - 4*a^6*b^6 - 2*a^5*b
^7 + 8*a^4*b^8 - 2*a^3*b^9 - 4*a^2*b^10 + 2*a*b^11 - 14*a^8*b^3*c + 26*a^7*
b^4*c + 28*a^6*b^5*c - 70*a^5*b^6*c + 46*a^3*b^8*c - 12*a^2*b^9*c - 2*a*b^1
0*c - 2*b^11*c + 24*a^9*b*c^2 - 36*a^8*b^2*c^2 - 120*a^7*b^3*c^2 + 190*a^6*
b^4*c^2 + 102*a^5*b^5*c^2 - 194*a^4*b^6*c^2 - 4*a^3*b^7*c^2 + 18*a^2*b^8*c^
2 + 14*a*b^9*c^2 + 6*b^10*c^2 - 16*a^9*c^3 + 160*a^8*b*c^3 - 128*a^7*b^2*c^
3 - 384*a^6*b^3*c^3 + 338*a^5*b^4*c^3 + 176*a^4*b^5*c^3 - 60*a^3*b^6*c^3 -
32*a^2*b^7*c^3 - 54*a*b^8*c^3 - 96*a^8*c^4 + 416*a^7*b*c^4 - 112*a^6*b^2*c^
4 - 504*a^5*b^3*c^4 + 70*a^4*b^4*c^4 + 46*a^3*b^5*c^4 + 184*a^2*b^6*c^4 + 2
2*a*b^7*c^4 - 18*b^8*c^4 - 224*a^7*c^5 + 480*a^6*b*c^5 + 96*a^5*b^2*c^5 - 7
6*a^4*b^3*c^5 - 306*a^3*b^4*c^5 - 180*a^2*b^5*c^5 + 130*a*b^6*c^5 + 16*b^7*
c^5 - 224*a^6*c^6 + 80*a^5*b*c^6 + 200*a^4*b^2*c^6 + 472*a^3*b^3*c^6 - 262*
a^2*b^4*c^6 - 150*a*b^5*c^6 + 14*b^6*c^6 - 416*a^4*b*c^7 + 64*a^3*b^2*c^7 +
 464*a^2*b^3*c^7 - 58*a*b^4*c^7 - 24*b^5*c^7 + 224*a^4*c^8 - 480*a^3*b*c^8
- 48*a^2*b^2*c^8 + 152*a*b^3*c^8 + 2*b^4*c^8 + 224*a^3*c^9 - 224*a^2*b*c^9
- 32*a*b^2*c^9 + 10*b^3*c^9 + 96*a^2*c^10 - 40*a*b*c^10 - 4*b^2*c^10 + 16*a
*c^11 + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(
b^2 - 4*a*c)*a^7*b^3 - 2*(b^2 - 4*a*c)*a^7*b^3 - 2*sqrt(a^2 - a*b + b*c - c
^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^6*b^4 + 4*(b^2 - 4*
a*c)*a^6*b^4 - 11*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
))*sqrt(b^2 - 4*a*c)*a^5*b^5 + 2*(b^2 - 4*a*c)*a^5*b^5 + 4*sqrt(a^2 - a*b +
 b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^6 - 8*(
b^2 - 4*a*c)*a^4*b^6 + 13*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
 - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^7 + 2*(b^2 - 4*a*c)*a^3*b^7 - 2*sqrt(a^2
 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b
^8 + 4*(b^2 - 4*a*c)*a^2*b^8 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^9 - 2*(b^2 - 4*a*c)*a*b^9 - 9*sqrt(
a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^
8*b*c + 6*(b^2 - 4*a*c)*a^8*b*c + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^7*b^2*c - 10*(b^2 - 4*a*c)*a^7*b^2
*c + 58*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^
2 - 4*a*c)*a^6*b^3*c - 20*(b^2 - 4*a*c)*a^6*b^3*c - sqrt(a^2 - a*b + b*c -
c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*b^4*c + 38*(b^2
- 4*a*c)*a^5*b^4*c - 92*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
 b + c))*sqrt(b^2 - 4*a*c)*a^4*b^5*c + 8*(b^2 - 4*a*c)*a^4*b^5*c - 15*sqrt(
a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^
3*b^6*c - 30*(b^2 - 4*a*c)*a^3*b^6*c + 38*sqrt(a^2 - a*b + b*c - c^2 + sqrt
```

(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^7*c + 4*(b^2 - 4*a*c)*a^
2*b^7*c + 13*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sq
rt(b^2 - 4*a*c)*a*b^8*c + 2*(b^2 - 4*a*c)*a*b^8*c + 5*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^9*c + 2*(b^2 - 4
*a*c)*b^9*c + 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))
*sqrt(b^2 - 4*a*c)*a^8*c^2 - 4*(b^2 - 4*a*c)*a^8*c^2 - 76*sqrt(a^2 - a*b +
b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^7*b*c^2 + 40
*(b^2 - 4*a*c)*a^7*b*c^2 - 43*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^6*b^2*c^2 - 38*(b^2 - 4*a*c)*a^6*b^2*c^2
+ 209*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*a^5*b^3*c^2 - 70*(b^2 - 4*a*c)*a^5*b^3*c^2 + 109*sqrt(a^2 - a*b +
b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^4*c^2 +
74*(b^2 - 4*a*c)*a^4*b^4*c^2 - 94*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^5*c^2 + 20*(b^2 - 4*a*c)*a^3*b^
5*c^2 - 69*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt
(b^2 - 4*a*c)*a^2*b^6*c^2 - 10*(b^2 - 4*a*c)*a^2*b^6*c^2 - 47*sqrt(a^2 - a*
b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^7*c^2
- 6*(b^2 - 4*a*c)*a*b^7*c^2 - 11*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^8*c^2 - 6*(b^2 - 4*a*c)*b^8*c^2 + 52*
sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*
c)*a^7*c^3 - 24*(b^2 - 4*a*c)*a^7*c^3 - 156*sqrt(a^2 - a*b + b*c - c^2 + sq
rt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^6*b*c^3 + 104*(b^2 - 4*a*c
)*a^6*b*c^3 - 205*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
))*sqrt(b^2 - 4*a*c)*a^5*b^2*c^3 - 42*(b^2 - 4*a*c)*a^5*b^2*c^3 + 100*sqrt(
a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^
4*b^3*c^3 - 96*(b^2 - 4*a*c)*a^4*b^3*c^3 + 162*sqrt(a^2 - a*b + b*c - c^2 +
sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^4*c^3 + 20*(b^2 - 4
*a*c)*a^3*b^4*c^3 + 148*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
b + c))*sqrt(b^2 - 4*a*c)*a^2*b^5*c^3 + 8*(b^2 - 4*a*c)*a^2*b^5*c^3 + 47*s
qrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c
)*a*b^6*c^3 + 30*(b^2 - 4*a*c)*a*b^6*c^3 - 4*sqrt(a^2 - a*b + b*c - c^2 + s
qrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^7*c^3 + 132*sqrt(a^2 - a*
b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^6*c^4 -
56*(b^2 - 4*a*c)*a^6*c^4 - 36*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*b*c^4 + 120*(b^2 - 4*a*c)*a^5*b*c^4 -
211*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
4*a*c)*a^4*b^2*c^4 + 10*(b^2 - 4*a*c)*a^4*b^2*c^4 - 251*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^3*c^4 - 14
*(b^2 - 4*a*c)*a^3*b^3*c^4 - 84*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a
c)(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^4*c^4 - 64*(b^2 - 4*a*c)*a^2*b^4*c
^4 + 73*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^
2 - 4*a*c)*a*b^5*c^4 - 22*(b^2 - 4*a*c)*a*b^5*c^4 + 33*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^6*c^4 + 18*(b^2
- 4*a*c)*b^6*c^4 + 100*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
b + c))*sqrt(b^2 - 4*a*c)*a^5*c^5 - 56*(b^2 - 4*a*c)*a^5*c^5 + 194*sqrt(a^

2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*
b*c^5 + 20*(b^2 - 4*a*c)*a^4*b*c^5 + 113*sqrt(a^2 - a*b + b*c - c^2 + sqrt(
b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c^5 + 50*(b^2 - 4*a*c)*
a^3*b^2*c^5 - 198*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
))*sqrt(b^2 - 4*a*c)*a^2*b^3*c^5 + 92*(b^2 - 4*a*c)*a^2*b^3*c^5 - 161*sqrt(
a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*
b^4*c^5 - 58*(b^2 - 4*a*c)*a*b^4*c^5 - 24*sqrt(a^2 - a*b + b*c - c^2 + sqrt
(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5*c^5 - 16*(b^2 - 4*a*c)*b^5
*c^5 - 80*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(
b^2 - 4*a*c)*a^4*c^6 + 172*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(
a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^6 - 104*(b^2 - 4*a*c)*a^3*b*c^6 + 263
*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a
*c)*a^2*b^2*c^6 + 30*(b^2 - 4*a*c)*a^2*b^2*c^6 + 39*sqrt(a^2 - a*b + b*c -
c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c^6 + 86*(b^2
- 4*a*c)*a*b^3*c^6 - 23*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
b + c))*sqrt(b^2 - 4*a*c)*b^4*c^6 - 14*(b^2 - 4*a*c)*b^4*c^6 - 164*sqrt(a^
2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*
c^7 + 56*(b^2 - 4*a*c)*a^3*c^7 - 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^7 - 120*(b^2 - 4*a*c)*a^2*b*
c^7 + 89*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b
^2 - 4*a*c)*a*b^2*c^7 + 2*(b^2 - 4*a*c)*a*b^2*c^7 + 40*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^3*c^7 + 24*(b^2
- 4*a*c)*b^3*c^7 - 68*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
b + c))*sqrt(b^2 - 4*a*c)*a^2*c^8 + 56*(b^2 - 4*a*c)*a^2*c^8 - 60*sqrt(a^2
- a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*c^
8 - 56*(b^2 - 4*a*c)*a*b*c^8 - 9*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c^8 - 2*(b^2 - 4*a*c)*b^2*c^8 + 12*
sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*
c)*a*c^9 + 24*(b^2 - 4*a*c)*a*c^9 - 17*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^
2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b*c^9 - 10*(b^2 - 4*a*c)*b*c^9 +
10*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4
*a*c)*c^10 + 4*(b^2 - 4*a*c)*c^10)*abs(-a + b - c))*(pi*floor(1/2*x/pi + 1/
2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a^3 - 2*a*b^2 + 2*a^2*c + 2*b^2*
c - 2*a*c^2 - 2*c^3 - sqrt(-4*(a^3 + a^2*b - a*b^2 - b^3 + 3*a^2*c + 2*a*b*
c - b^2*c + 3*a*c^2 + b*c^2 + c^3)*(a^3 - a^2*b - a*b^2 + b^3 + 3*a^2*c - 2
*a*b*c - b^2*c + 3*a*c^2 - b*c^2 + c^3) + 4*(a^3 - a*b^2 + a^2*c + b^2*c -
a*c^2 - c^3)^2))/(a^3 - a^2*b - a*b^2 + b^3 + 3*a^2*c - 2*a*b*c - b^2*c + 3
*a*c^2 - b*c^2 + c^3))))/((3*a^10*b^2 - 8*a^9*b^3 - 7*a^8*b^4 + 32*a^7*b^5
- 2*a^6*b^6 - 48*a^5*b^7 + 18*a^4*b^8 + 32*a^3*b^9 - 17*a^2*b^10 - 8*a*b^11
+ 5*b^12 - 12*a^11*c + 32*a^10*b*c + 66*a^9*b^2*c - 208*a^8*b^3*c - 80*a^7
*b^4*c + 448*a^6*b^5*c - 36*a^5*b^6*c - 416*a^4*b^7*c + 108*a^3*b^8*c + 160
*a^2*b^9*c - 46*a*b^10*c - 16*b^11*c - 152*a^10*c^2 + 320*a^9*b*c^2 + 543*a
^8*b^2*c^2 - 1344*a^7*b^3*c^2 - 532*a^6*b^4*c^2 + 1920*a^5*b^5*c^2 + 50*a^4
*b^6*c^2 - 1088*a^3*b^7*c^2 + 84*a^2*b^8*c^2 + 192*a*b^9*c^2 + 7*b^10*c^2 -
764*a^9*c^3 + 1280*a^8*b*c^3 + 2072*a^7*b^2*c^3 - 3744*a^6*b^3*c^3 - 1712*

```
a^5*b^4*c^3 + 3424*a^4*b^5*c^3 + 520*a^3*b^6*c^3 - 992*a^2*b^7*c^3 - 116*a*
b^8*c^3 + 32*b^9*c^3 - 2080*a^8*c^4 + 2688*a^7*b*c^4 + 4342*a^6*b^2*c^4 - 5
264*a^5*b^3*c^4 - 2890*a^4*b^4*c^4 + 2720*a^3*b^5*c^4 + 802*a^2*b^6*c^4 - 2
72*a*b^7*c^4 - 46*b^8*c^4 - 3416*a^7*c^5 + 3136*a^6*b*c^5 + 5388*a^5*b^2*c^
5 - 3648*a^4*b^3*c^5 - 2672*a^3*b^4*c^5 + 768*a^2*b^5*c^5 + 412*a*b^6*c^5 -
 3472*a^6*c^6 + 1792*a^5*b*c^6 + 4006*a^4*b^2*c^6 - 768*a^3*b^3*c^6 - 1300*
a^2*b^4*c^6 - 64*a*b^5*c^6 + 46*b^6*c^6 - 2072*a^5*c^7 + 1688*a^3*b^2*c^7 +
 416*a^2*b^3*c^7 - 272*a*b^4*c^7 - 32*b^5*c^7 - 544*a^4*c^8 - 640*a^3*b*c^8
 + 327*a^2*b^2*c^8 + 216*a*b^3*c^8 - 7*b^4*c^8 + 100*a^3*c^9 - 352*a^2*b*c^
9 + 2*a*b^2*c^9 + 16*b^3*c^9 + 104*a^2*c^10 - 64*a*b*c^10 - 5*b^2*c^10 + 20
*a*c^11)*abs(-a^2 + b^2 - 2*a*c - c^2)) + 1/2*tan(1/2*x)/(a - b + c) - 1/2/
((a + b + c)*tan(1/2*x))
```

Mupad [B] (verification not implemented)

Time = 15.47 (sec) , antiderivative size = 39229, normalized size of antiderivative = 120.33

$$\int \frac{\csc^2(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] int(1/(sin(x)^2*(a + b*cos(x) + c*cos(x)^2)),x)
```

```
[Out] atan((((-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*
c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a
*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c
^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a
^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*
a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^
4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*
a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a
*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^
2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^
4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^
2 + 14*a*b^8*c)))^(1/2)*(128*a*c^13 - 64*a*b^13 - 32*b^13*c + 32*b^14 - 96*
a^2*b^12 + 256*a^3*b^11 + 64*a^4*b^10 - 384*a^5*b^9 + 64*a^6*b^8 + 256*a^7*
b^7 - 96*a^8*b^6 - 64*a^9*b^5 + 32*a^10*b^4 + 1408*a^2*c^12 + 7040*a^3*c^11
 + 21120*a^4*c^10 + 42240*a^5*c^9 + 59136*a^6*c^8 + 59136*a^7*c^7 + 42240*a
^8*c^6 + 21120*a^9*c^5 + 7040*a^10*c^4 + 1408*a^11*c^3 + 128*a^12*c^2 - 32*
b^2*c^12 + 96*b^3*c^11 + 64*b^4*c^10 - 416*b^5*c^9 + 96*b^6*c^8 + 704*b^7*c
^7 - 384*b^8*c^6 - 576*b^9*c^5 + 416*b^10*c^4 + 224*b^11*c^3 - 192*b^12*c^2
 + tan(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-
(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 +
 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*
b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c
+ 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2)
```

```
 - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a
^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4
+ 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 +
 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c -
96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a
^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b
^2*c^2 + 14*a*b^8*c)))^(1/2)*(64*a*b^14 - 256*a*c^14 + 256*a^14*c - 64*b^14
*c - 128*a^2*b^13 - 256*a^3*b^12 + 640*a^4*b^11 + 320*a^5*b^10 - 1280*a^6*b
^9 + 1280*a^8*b^7 - 320*a^9*b^6 - 640*a^10*b^5 + 256*a^11*b^4 + 128*a^12*b^
3 - 64*a^13*b^2 - 2816*a^2*c^13 - 13824*a^3*c^12 - 39424*a^4*c^11 - 70400*a
^5*c^10 - 76032*a^6*c^9 - 33792*a^7*c^8 + 33792*a^8*c^7 + 76032*a^9*c^6 + 7
0400*a^10*c^5 + 39424*a^11*c^4 + 13824*a^12*c^3 + 2816*a^13*c^2 + 64*b^2*c^
13 - 128*b^3*c^12 - 256*b^4*c^11 + 640*b^5*c^10 + 320*b^6*c^9 - 1280*b^7*c^
8 + 1280*b^9*c^6 - 320*b^10*c^5 - 640*b^11*c^4 + 256*b^12*c^3 + 128*b^13*c^
2 + 1728*a*b^2*c^12 - 3840*a*b^3*c^11 - 3584*a*b^4*c^10 + 10240*a*b^5*c^9 +
 2240*a*b^6*c^8 - 12800*a*b^7*c^7 + 1280*a*b^8*c^6 + 7680*a*b^9*c^5 - 1984*
a*b^10*c^4 - 1792*a*b^11*c^3 + 512*a*b^12*c^2 + 5120*a^2*b*c^12 - 512*a^2*b
^12*c + 22528*a^3*b*c^11 + 1792*a^3*b^11*c + 56320*a^4*b*c^10 + 1984*a^4*b^
10*c + 84480*a^5*b*c^9 - 7680*a^5*b^9*c + 67584*a^6*b*c^8 - 1280*a^6*b^8*c
+ 12800*a^7*b^7*c - 67584*a^8*b*c^6 - 2240*a^8*b^6*c - 84480*a^9*b*c^5 - 10
240*a^9*b^5*c - 56320*a^10*b*c^4 + 3584*a^10*b^4*c - 22528*a^11*b*c^3 + 384
0*a^11*b^3*c - 5120*a^12*b*c^2 - 1728*a^12*b^2*c + 12672*a^2*b^2*c^11 - 261
12*a^2*b^3*c^10 - 17920*a^2*b^4*c^9 + 48000*a^2*b^5*c^8 + 6400*a^2*b^6*c^7
- 38400*a^2*b^7*c^6 + 3840*a^2*b^8*c^5 + 11520*a^2*b^9*c^4 - 1664*a^2*b^10*
c^3 + 45696*a^3*b^2*c^10 - 83200*a^3*b^3*c^9 - 44800*a^3*b^4*c^8 + 102400*a
^3*b^5*c^7 + 8960*a^3*b^6*c^6 - 43520*a^3*b^7*c^5 + 2560*a^3*b^8*c^4 + 1664
*a^3*b^10*c^2 + 94400*a^4*b^2*c^9 - 144000*a^4*b^3*c^8 - 58880*a^4*b^4*c^7
+ 98560*a^4*b^5*c^6 + 4480*a^4*b^6*c^5 - 2560*a^4*b^8*c^3 - 11520*a^4*b^9*c
^2 + 111168*a^5*b^2*c^8 - 124416*a^5*b^3*c^7 - 28672*a^5*b^4*c^6 - 4480*a^5
*b^6*c^4 + 43520*a^5*b^7*c^3 - 3840*a^5*b^8*c^2 + 51456*a^6*b^2*c^7 + 28672
*a^6*b^4*c^5 - 98560*a^6*b^5*c^4 - 8960*a^6*b^6*c^3 + 38400*a^6*b^7*c^2 - 5
1456*a^7*b^2*c^6 + 124416*a^7*b^3*c^5 + 58880*a^7*b^4*c^4 - 102400*a^7*b^5*
c^3 - 6400*a^7*b^6*c^2 - 111168*a^8*b^2*c^5 + 144000*a^8*b^3*c^4 + 44800*a^
8*b^4*c^3 - 48000*a^8*b^5*c^2 - 94400*a^9*b^2*c^4 + 83200*a^9*b^3*c^3 + 179
20*a^9*b^4*c^2 - 45696*a^10*b^2*c^3 + 26112*a^10*b^3*c^2 - 12672*a^11*b^2*c
^2 + 512*a*b*c^13 - 512*a^13*b*c) - 608*a*b^2*c^11 + 2624*a*b^3*c^10 + 224*
a*b^4*c^9 - 6208*a*b^5*c^8 + 2112*a*b^6*c^7 + 6784*a*b^7*c^6 - 3520*a*b^8*c
^5 - 3584*a*b^9*c^4 + 2080*a*b^10*c^3 + 832*a*b^11*c^2 - 3840*a^2*b*c^11 +
992*a^2*b^11*c - 17280*a^3*b*c^10 + 992*a^3*b^10*c - 46080*a^4*b*c^9 - 3136
*a^4*b^9*c - 80640*a^5*b*c^8 - 320*a^5*b^8*c - 96768*a^6*b*c^7 + 3776*a^6*b
^7*c - 80640*a^7*b*c^6 - 832*a^7*b^6*c - 46080*a^8*b*c^5 - 1952*a^8*b^5*c -
 17280*a^9*b*c^4 + 736*a^9*b^4*c - 3840*a^10*b*c^3 + 352*a^10*b^3*c - 384*a
^11*b*c^2 - 160*a^11*b^2*c - 4192*a^2*b^2*c^10 + 17888*a^2*b^3*c^9 + 288*a^
2*b^4*c^8 - 30080*a^2*b^5*c^7 + 8768*a^2*b^6*c^6 + 22848*a^2*b^7*c^5 - 8768
*a^2*b^8*c^4 - 7808*a^2*b^9*c^3 + 2592*a^2*b^10*c^2 - 15648*a^3*b^2*c^9 + 6
```

0160*a^3*b^3*c^8 + 1152*a^3*b^4*c^7 - 73472*a^3*b^5*c^6 + 15424*a^3*b^6*c^5
 + 37888*a^3*b^7*c^4 - 8960*a^3*b^8*c^3 - 7552*a^3*b^9*c^2 - 36672*a^4*b^2*
c^8 + 120512*a^4*b^3*c^7 + 5376*a^4*b^4*c^6 - 104384*a^4*b^5*c^5 + 12800*a^
4*b^6*c^4 + 34112*a^4*b^7*c^3 - 3712*a^4*b^8*c^2 - 57792*a^5*b^2*c^7 + 1550
08*a^5*b^3*c^6 + 12096*a^5*b^4*c^5 - 90496*a^5*b^5*c^4 + 3776*a^5*b^6*c^3 +
 16512*a^5*b^7*c^2 - 63168*a^6*b^2*c^6 + 131264*a^6*b^3*c^5 + 14784*a^6*b^4
*c^4 - 47488*a^6*b^5*c^3 - 1088*a^6*b^6*c^2 - 48192*a^7*b^2*c^5 + 72448*a^7
*b^3*c^4 + 10368*a^7*b^4*c^3 - 14080*a^7*b^5*c^2 - 25248*a^8*b^2*c^4 + 2480
0*a^8*b^3*c^3 + 4032*a^8*b^4*c^2 - 8672*a^9*b^2*c^3 + 4672*a^9*b^3*c^2 - 17
60*a^10*b^2*c^2 - 384*a*b*c^12 - 416*a*b^12*c) + tan(x/2)*(32*a*b^12 - 512*
a*c^12 + 128*b*c^12 + 96*b^12*c - 32*b^13 - 64*c^13 + 96*a^2*b^11 - 96*a^3*
b^10 - 96*a^4*b^9 + 96*a^5*b^8 + 32*a^6*b^7 - 32*a^7*b^6 - 1728*a^2*c^11 -
3072*a^3*c^10 - 2688*a^4*c^9 + 2688*a^6*c^7 + 3072*a^7*c^6 + 1728*a^8*c^5 +
 512*a^9*c^4 + 64*a^10*c^3 + 160*b^2*c^11 - 544*b^3*c^10 + 64*b^4*c^9 + 896
*b^5*c^8 - 608*b^6*c^7 - 672*b^7*c^6 + 800*b^8*c^5 + 160*b^9*c^4 - 448*b^10
*c^3 + 64*b^11*c^2 + 480*a*b^2*c^10 - 4352*a*b^3*c^9 + 2560*a*b^4*c^8 + 524
8*a*b^5*c^7 - 5664*a*b^6*c^6 - 2240*a*b^7*c^5 + 4320*a*b^8*c^4 - 256*a*b^9*
c^3 - 1216*a*b^10*c^2 + 5632*a^2*b*c^10 - 672*a^2*b^10*c + 14336*a^3*b*c^9
- 768*a^3*b^9*c + 23296*a^4*b*c^8 + 1248*a^4*b^8*c + 25088*a^5*b*c^7 + 576*
a^5*b^7*c + 17920*a^6*b*c^6 - 864*a^6*b^6*c + 8192*a^7*b*c^5 - 128*a^7*b^5*
c + 2176*a^8*b*c^4 + 192*a^8*b^4*c + 256*a^9*b*c^3 - 1408*a^2*b^2*c^9 - 147
20*a^2*b^3*c^8 + 13440*a^2*b^4*c^7 + 11904*a^2*b^5*c^6 - 16800*a^2*b^6*c^5
- 1696*a^2*b^7*c^4 + 7168*a^2*b^8*c^3 - 1216*a^2*b^9*c^2 - 9856*a^3*b^2*c^8
 - 27392*a^3*b^3*c^7 + 31232*a^3*b^4*c^6 + 12928*a^3*b^5*c^5 - 23264*a^3*b^
6*c^4 + 1152*a^3*b^7*c^3 + 4800*a^3*b^8*c^2 - 22848*a^4*b^2*c^7 - 30400*a^4
*b^3*c^6 + 39680*a^4*b^4*c^5 + 6272*a^4*b^5*c^4 - 16544*a^4*b^6*c^3 + 1824*
a^4*b^7*c^2 - 29120*a^5*b^2*c^6 - 20224*a^5*b^3*c^5 + 29184*a^5*b^4*c^4 + 3
84*a^5*b^5*c^3 - 5856*a^5*b^6*c^2 - 22400*a^6*b^2*c^5 - 7552*a^6*b^3*c^4 +
12160*a^6*b^4*c^3 - 640*a^6*b^5*c^2 - 10368*a^7*b^2*c^4 - 1280*a^7*b^3*c^3
+ 2560*a^7*b^4*c^2 - 2656*a^8*b^2*c^3 - 32*a^8*b^3*c^2 - 288*a^9*b^2*c^2 +
1280*a*b*c^11 + 320*a*b^11*c))*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 +
 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c
^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^
2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^
(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*
a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b
^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5
*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*
c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4
*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*
a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5
*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*1i - ((-(8*a*c^7 + b^8 + 24
*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^
6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c
- b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*

c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(128*a*c^13 - 64*a*b^13 - 32*b^13*c + 32*b^14 - 96*a^2*b^12 + 256*a^3*b^11 + 64*a^4*b^10 - 384*a^5*b^9 + 64*a^6*b^8 + 256*a^7*b^7 - 96*a^8*b^6 - 64*a^9*b^5 + 32*a^10*b^4 + 1408*a^2*c^12 + 7040*a^3*c^11 + 21120*a^4*c^10 + 42240*a^5*c^9 + 59136*a^6*c^8 + 59136*a^7*c^7 + 42240*a^8*c^6 + 21120*a^9*c^5 + 7040*a^10*c^4 + 1408*a^11*c^3 + 128*a^12*c^2 - 32*b^2*c^12 + 96*b^3*c^11 + 64*b^4*c^10 - 416*b^5*c^9 + 96*b^6*c^8 + 704*b^7*c^7 - 384*b^8*c^6 - 576*b^9*c^5 + 416*b^10*c^4 + 224*b^11*c^3 - 192*b^12*c^2 - tan(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(64*a*b^14 - 256*a*c^14 + 256*a^14*c - 64*b^14*c - 128*a^2*b^13 - 256*a^3*b^12 + 640*a^4*b^11 + 320*a^5*b^10 - 1280*a^6*b^9 + 1280*a^8*b^7 - 320*a^9*b^6 - 640*a^10*b^5 + 256*a^11*b^4 + 128*a^12*b^3 - 64*a^13*b^2 - 2816*a^2*c^13 - 13824*a^3*c^12 - 39424*a^4*c^11 - 70400*a^5*c^10 - 76032*a^6*c^9 - 33792*a^7*c^8 + 33792*a^8*c^7 + 76032*a^9*c^6 + 70400*a^10*c^5 + 39424*a^11*c^4 + 13824*a^12*c^3 + 2816*a^13*c^2 + 64*b^2*c^13 - 128*b^3*c^12 - 256*b^4*c^11 + 640*b^5*c^10 + 320*b^6*c^9 - 1280*b^7*c^8 + 1280*b^9*c^6 - 320*b^10*c^5 - 640*b^11*c^4 + 256*b^12*c^3 + 128*b^13*c^2 + 1728*a*b^2*c^12 - 3840*a*b^3*c^11 - 3584*a*b^4*c^10 + 10240*a*b^5*c^9 + 2240*a*b^6*c^8 - 12800*a*b^7*c^7 + 1280*a*b^8*c^6 + 7680*a*b^9*c^5 - 1984*a*b^10*c^4 - 1792*a*b^11*c^3 + 512*a*b^12*c^2 + 5120*a^2*b*c^12 - 512*a^2*b^12*c + 22528*a^3*b*c^11 + 1792*a^3*b^11*c + 56320*a^4*b*c^10 + 1984*a^4*b^10*c + 84480*a^5*b*c^9 - 7680*a^5*b^9*c + 67584*a^6*b*c^8 - 1280*a^6*b^8*c + 12800*a^7*b^7*c - 67584*a^8*b*c^6 - 2240*a^8*b^6*c - 84480*a^9*b*c^5 - 10240*a^9*b^5*c - 56320*a^10*b*c^4 + 3584*a^10*b^4*c - 22528*a^11*b*c^3 + 3840*a^11*b^3*c - 5120*a^12*b*c^2 - 1728*a^12*b^2*c + 12672*a^2*b^2*c^11 - 26112*a^2*b^3*c^10 - 17920*a^2*b^4*c^9 + 48000*a^2*b^5*c^8 + 6400*a^2*b^6*c^7 - 38400*a^2*b^7*c^6 + 3840*a^2*b^8*c^5 + 11520*a^2*b^9*c^4 - 1664*a^2*b^10*c^3 + 45696*a^3*b^2*c^10 - 83200*a^3*b^3*c^9 - 44800*a^3*b^4*c^8 + 102400*a^3*b^5*c^7 + 8960*a^3*b^6*c^

6 - 43520*a^3*b^7*c^5 + 2560*a^3*b^8*c^4 + 1664*a^3*b^10*c^2 + 94400*a^4*b^
2*c^9 - 144000*a^4*b^3*c^8 - 58880*a^4*b^4*c^7 + 98560*a^4*b^5*c^6 + 4480*a
^4*b^6*c^5 - 2560*a^4*b^8*c^3 - 11520*a^4*b^9*c^2 + 111168*a^5*b^2*c^8 - 12
4416*a^5*b^3*c^7 - 28672*a^5*b^4*c^6 - 4480*a^5*b^6*c^4 + 43520*a^5*b^7*c^3
 - 3840*a^5*b^8*c^2 + 51456*a^6*b^2*c^7 + 28672*a^6*b^4*c^5 - 98560*a^6*b^5
*c^4 - 8960*a^6*b^6*c^3 + 38400*a^6*b^7*c^2 - 51456*a^7*b^2*c^6 + 124416*a^
7*b^3*c^5 + 58880*a^7*b^4*c^4 - 102400*a^7*b^5*c^3 - 6400*a^7*b^6*c^2 - 111
168*a^8*b^2*c^5 + 144000*a^8*b^3*c^4 + 44800*a^8*b^4*c^3 - 48000*a^8*b^5*c^
2 - 94400*a^9*b^2*c^4 + 83200*a^9*b^3*c^3 + 17920*a^9*b^4*c^2 - 45696*a^10*
b^2*c^3 + 26112*a^10*b^3*c^2 - 12672*a^11*b^2*c^2 + 512*a*b*c^13 - 512*a^13
*b*c) - 608*a*b^2*c^11 + 2624*a*b^3*c^10 + 224*a*b^4*c^9 - 6208*a*b^5*c^8 +
 2112*a*b^6*c^7 + 6784*a*b^7*c^6 - 3520*a*b^8*c^5 - 3584*a*b^9*c^4 + 2080*a
*b^10*c^3 + 832*a*b^11*c^2 - 3840*a^2*b*c^11 + 992*a^2*b^11*c - 17280*a^3*b
*c^10 + 992*a^3*b^10*c - 46080*a^4*b*c^9 - 3136*a^4*b^9*c - 80640*a^5*b*c^8
 - 320*a^5*b^8*c - 96768*a^6*b*c^7 + 3776*a^6*b^7*c - 80640*a^7*b*c^6 - 832
*a^7*b^6*c - 46080*a^8*b*c^5 - 1952*a^8*b^5*c - 17280*a^9*b*c^4 + 736*a^9*b
^4*c - 3840*a^10*b*c^3 + 352*a^10*b^3*c - 384*a^11*b*c^2 - 160*a^11*b^2*c -
 4192*a^2*b^2*c^10 + 17888*a^2*b^3*c^9 + 288*a^2*b^4*c^8 - 30080*a^2*b^5*c^
7 + 8768*a^2*b^6*c^6 + 22848*a^2*b^7*c^5 - 8768*a^2*b^8*c^4 - 7808*a^2*b^9*
c^3 + 2592*a^2*b^10*c^2 - 15648*a^3*b^2*c^9 + 60160*a^3*b^3*c^8 + 1152*a^3*
b^4*c^7 - 73472*a^3*b^5*c^6 + 15424*a^3*b^6*c^5 + 37888*a^3*b^7*c^4 - 8960*
a^3*b^8*c^3 - 7552*a^3*b^9*c^2 - 36672*a^4*b^2*c^8 + 120512*a^4*b^3*c^7 + 5
376*a^4*b^4*c^6 - 104384*a^4*b^5*c^5 + 12800*a^4*b^6*c^4 + 34112*a^4*b^7*c^
3 - 3712*a^4*b^8*c^2 - 57792*a^5*b^2*c^7 + 155008*a^5*b^3*c^6 + 12096*a^5*b
^4*c^5 - 90496*a^5*b^5*c^4 + 3776*a^5*b^6*c^3 + 16512*a^5*b^7*c^2 - 63168*a
^6*b^2*c^6 + 131264*a^6*b^3*c^5 + 14784*a^6*b^4*c^4 - 47488*a^6*b^5*c^3 - 1
088*a^6*b^6*c^2 - 48192*a^7*b^2*c^5 + 72448*a^7*b^3*c^4 + 10368*a^7*b^4*c^3
 - 14080*a^7*b^5*c^2 - 25248*a^8*b^2*c^4 + 24800*a^8*b^3*c^3 + 4032*a^8*b^4
*c^2 - 8672*a^9*b^2*c^3 + 4672*a^9*b^3*c^2 - 1760*a^10*b^2*c^2 - 384*a*b*c^
12 - 416*a*b^12*c) - tan(x/2)*(32*a*b^12 - 512*a*c^12 + 128*b*c^12 + 96*b^1
2*c - 32*b^13 - 64*c^13 + 96*a^2*b^11 - 96*a^3*b^10 - 96*a^4*b^9 + 96*a^5*b
^8 + 32*a^6*b^7 - 32*a^7*b^6 - 1728*a^2*c^11 - 3072*a^3*c^10 - 2688*a^4*c^9
 + 2688*a^6*c^7 + 3072*a^7*c^6 + 1728*a^8*c^5 + 512*a^9*c^4 + 64*a^10*c^3 +
 160*b^2*c^11 - 544*b^3*c^10 + 64*b^4*c^9 + 896*b^5*c^8 - 608*b^6*c^7 - 672
*b^7*c^6 + 800*b^8*c^5 + 160*b^9*c^4 - 448*b^10*c^3 + 64*b^11*c^2 + 480*a*b
^2*c^10 - 4352*a*b^3*c^9 + 2560*a*b^4*c^8 + 5248*a*b^5*c^7 - 5664*a*b^6*c^6
 - 2240*a*b^7*c^5 + 4320*a*b^8*c^4 - 256*a*b^9*c^3 - 1216*a*b^10*c^2 + 5632
*a^2*b*c^10 - 672*a^2*b^10*c + 14336*a^3*b*c^9 - 768*a^3*b^9*c + 23296*a^4*
b*c^8 + 1248*a^4*b^8*c + 25088*a^5*b*c^7 + 576*a^5*b^7*c + 17920*a^6*b*c^6
 - 864*a^6*b^6*c + 8192*a^7*b*c^5 - 128*a^7*b^5*c + 2176*a^8*b*c^4 + 192*a^8
*b^4*c + 256*a^9*b*c^3 - 1408*a^2*b^2*c^9 - 14720*a^2*b^3*c^8 + 13440*a^2*b
^4*c^7 + 11904*a^2*b^5*c^6 - 16800*a^2*b^6*c^5 - 1696*a^2*b^7*c^4 + 7168*a^
2*b^8*c^3 - 1216*a^2*b^9*c^2 - 9856*a^3*b^2*c^8 - 27392*a^3*b^3*c^7 + 31232
*a^3*b^4*c^6 + 12928*a^3*b^5*c^5 - 23264*a^3*b^6*c^4 + 1152*a^3*b^7*c^3 + 4
800*a^3*b^8*c^2 - 22848*a^4*b^2*c^7 - 30400*a^4*b^3*c^6 + 39680*a^4*b^4*c^5

+ 6272*a^4*b^5*c^4 - 16544*a^4*b^6*c^3 + 1824*a^4*b^7*c^2 - 29120*a^5*b^2*
c^6 - 20224*a^5*b^3*c^5 + 29184*a^5*b^4*c^4 + 384*a^5*b^5*c^3 - 5856*a^5*b^
6*c^2 - 22400*a^6*b^2*c^5 - 7552*a^6*b^3*c^4 + 12160*a^6*b^4*c^3 - 640*a^6*
b^5*c^2 - 10368*a^7*b^2*c^4 - 1280*a^7*b^3*c^3 + 2560*a^7*b^4*c^2 - 2656*a^
8*b^2*c^3 - 32*a^8*b^3*c^2 - 288*a^9*b^2*c^2 + 1280*a*b*c^11 + 320*a*b^11*c
))*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c -
b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4
*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 -
38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b
c^2(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^
3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 +
16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*
c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4
*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^
2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^
3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 +
14*a*b^8*c)))^(1/2)*1i)/(512*a*c^11 + 64*c^12 + ((-(8*a*c^7 + b^8 + 24*a^2*
c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3
*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2
)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(
-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2)
+ 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))
/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240
*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 -
3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*
b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*
a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4
*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(128*a*c^
13 - 64*a*b^13 - 32*b^13*c + 32*b^14 - 96*a^2*b^12 + 256*a^3*b^11 + 64*a^4*
b^10 - 384*a^5*b^9 + 64*a^6*b^8 + 256*a^7*b^7 - 96*a^8*b^6 - 64*a^9*b^5 + 3
2*a^10*b^4 + 1408*a^2*c^12 + 7040*a^3*c^11 + 21120*a^4*c^10 + 42240*a^5*c^9
+ 59136*a^6*c^8 + 59136*a^7*c^7 + 42240*a^8*c^6 + 21120*a^9*c^5 + 7040*a^1
0*c^4 + 1408*a^11*c^3 + 128*a^12*c^2 - 32*b^2*c^12 + 96*b^3*c^11 + 64*b^4*c
^10 - 416*b^5*c^9 + 96*b^6*c^8 + 704*b^7*c^7 - 384*b^8*c^6 - 576*b^9*c^5 +
416*b^10*c^4 + 224*b^11*c^3 - 192*b^12*c^2 + tan(x/2)*(-(8*a*c^7 + b^8 + 24
*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^
6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c
- b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*
c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^
(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(
1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7
+ 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c
^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36
*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4
- 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 15

9*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(64*
a*b^14 - 256*a*c^14 + 256*a^14*c - 64*b^14*c - 128*a^2*b^13 - 256*a^3*b^12
+ 640*a^4*b^11 + 320*a^5*b^10 - 1280*a^6*b^9 + 1280*a^8*b^7 - 320*a^9*b^6 -
640*a^10*b^5 + 256*a^11*b^4 + 128*a^12*b^3 - 64*a^13*b^2 - 2816*a^2*c^13 -
13824*a^3*c^12 - 39424*a^4*c^11 - 70400*a^5*c^10 - 76032*a^6*c^9 - 33792*a
^7*c^8 + 33792*a^8*c^7 + 76032*a^9*c^6 + 70400*a^10*c^5 + 39424*a^11*c^4 +
13824*a^12*c^3 + 2816*a^13*c^2 + 64*b^2*c^13 - 128*b^3*c^12 - 256*b^4*c^11
+ 640*b^5*c^10 + 320*b^6*c^9 - 1280*b^7*c^8 + 1280*b^9*c^6 - 320*b^10*c^5 -
640*b^11*c^4 + 256*b^12*c^3 + 128*b^13*c^2 + 1728*a*b^2*c^12 - 3840*a*b^3*
c^11 - 3584*a*b^4*c^10 + 10240*a*b^5*c^9 + 2240*a*b^6*c^8 - 12800*a*b^7*c^7
+ 1280*a*b^8*c^6 + 7680*a*b^9*c^5 - 1984*a*b^10*c^4 - 1792*a*b^11*c^3 + 51
2*a*b^12*c^2 + 5120*a^2*b*c^12 - 512*a^2*b^12*c + 22528*a^3*b*c^11 + 1792*a
^3*b^11*c + 56320*a^4*b*c^10 + 1984*a^4*b^10*c + 84480*a^5*b*c^9 - 7680*a^5
*b^9*c + 67584*a^6*b*c^8 - 1280*a^6*b^8*c + 12800*a^7*b^7*c - 67584*a^8*b*c
^6 - 2240*a^8*b^6*c - 84480*a^9*b*c^5 - 10240*a^9*b^5*c - 56320*a^10*b*c^4
+ 3584*a^10*b^4*c - 22528*a^11*b*c^3 + 3840*a^11*b^3*c - 5120*a^12*b*c^2 -
1728*a^12*b^2*c + 12672*a^2*b^2*c^11 - 26112*a^2*b^3*c^10 - 17920*a^2*b^4*c
^9 + 48000*a^2*b^5*c^8 + 6400*a^2*b^6*c^7 - 38400*a^2*b^7*c^6 + 3840*a^2*b^
8*c^5 + 11520*a^2*b^9*c^4 - 1664*a^2*b^10*c^3 + 45696*a^3*b^2*c^10 - 83200*
a^3*b^3*c^9 - 44800*a^3*b^4*c^8 + 102400*a^3*b^5*c^7 + 8960*a^3*b^6*c^6 - 4
3520*a^3*b^7*c^5 + 2560*a^3*b^8*c^4 + 1664*a^3*b^10*c^2 + 94400*a^4*b^2*c^9
- 144000*a^4*b^3*c^8 - 58880*a^4*b^4*c^7 + 98560*a^4*b^5*c^6 + 4480*a^4*b^
6*c^5 - 2560*a^4*b^8*c^3 - 11520*a^4*b^9*c^2 + 111168*a^5*b^2*c^8 - 124416*
a^5*b^3*c^7 - 28672*a^5*b^4*c^6 - 4480*a^5*b^6*c^4 + 43520*a^5*b^7*c^3 - 38
40*a^5*b^8*c^2 + 51456*a^6*b^2*c^7 + 28672*a^6*b^4*c^5 - 98560*a^6*b^5*c^4
- 8960*a^6*b^6*c^3 + 38400*a^6*b^7*c^2 - 51456*a^7*b^2*c^6 + 124416*a^7*b^3
*c^5 + 58880*a^7*b^4*c^4 - 102400*a^7*b^5*c^3 - 6400*a^7*b^6*c^2 - 111168*a
^8*b^2*c^5 + 144000*a^8*b^3*c^4 + 44800*a^8*b^4*c^3 - 48000*a^8*b^5*c^2 - 9
4400*a^9*b^2*c^4 + 83200*a^9*b^3*c^3 + 17920*a^9*b^4*c^2 - 45696*a^10*b^2*c
^3 + 26112*a^10*b^3*c^2 - 12672*a^11*b^2*c^2 + 512*a*b*c^13 - 512*a^13*b*c)
- 608*a*b^2*c^11 + 2624*a*b^3*c^10 + 224*a*b^4*c^9 - 6208*a*b^5*c^8 + 2112
*a*b^6*c^7 + 6784*a*b^7*c^6 - 3520*a*b^8*c^5 - 3584*a*b^9*c^4 + 2080*a*b^10
*c^3 + 832*a*b^11*c^2 - 3840*a^2*b*c^11 + 992*a^2*b^11*c - 17280*a^3*b*c^10
+ 992*a^3*b^10*c - 46080*a^4*b*c^9 - 3136*a^4*b^9*c - 80640*a^5*b*c^8 - 32
0*a^5*b^8*c - 96768*a^6*b*c^7 + 3776*a^6*b^7*c - 80640*a^7*b*c^6 - 832*a^7*
b^6*c - 46080*a^8*b*c^5 - 1952*a^8*b^5*c - 17280*a^9*b*c^4 + 736*a^9*b^4*c
- 3840*a^10*b*c^3 + 352*a^10*b^3*c - 384*a^11*b*c^2 - 160*a^11*b^2*c - 4192
*a^2*b^2*c^10 + 17888*a^2*b^3*c^9 + 288*a^2*b^4*c^8 - 30080*a^2*b^5*c^7 + 8
768*a^2*b^6*c^6 + 22848*a^2*b^7*c^5 - 8768*a^2*b^8*c^4 - 7808*a^2*b^9*c^3 +
2592*a^2*b^10*c^2 - 15648*a^3*b^2*c^9 + 60160*a^3*b^3*c^8 + 1152*a^3*b^4*c
^7 - 73472*a^3*b^5*c^6 + 15424*a^3*b^6*c^5 + 37888*a^3*b^7*c^4 - 8960*a^3*b
^8*c^3 - 7552*a^3*b^9*c^2 - 36672*a^4*b^2*c^8 + 120512*a^4*b^3*c^7 + 5376*a
^4*b^4*c^6 - 104384*a^4*b^5*c^5 + 12800*a^4*b^6*c^4 + 34112*a^4*b^7*c^3 - 3
712*a^4*b^8*c^2 - 57792*a^5*b^2*c^7 + 155008*a^5*b^3*c^6 + 12096*a^5*b^4*c^
5 - 90496*a^5*b^5*c^4 + 3776*a^5*b^6*c^3 + 16512*a^5*b^7*c^2 - 63168*a^6*b^

```
2*c^6 + 131264*a^6*b^3*c^5 + 14784*a^6*b^4*c^4 - 47488*a^6*b^5*c^3 - 1088*a
^6*b^6*c^2 - 48192*a^7*b^2*c^5 + 72448*a^7*b^3*c^4 + 10368*a^7*b^4*c^3 - 14
080*a^7*b^5*c^2 - 25248*a^8*b^2*c^4 + 24800*a^8*b^3*c^3 + 4032*a^8*b^4*c^2
- 8672*a^9*b^2*c^3 + 4672*a^9*b^3*c^2 - 1760*a^10*b^2*c^2 - 384*a*b*c^12 -
416*a*b^12*c) + tan(x/2)*(32*a*b^12 - 512*a*c^12 + 128*b*c^12 + 96*b^12*c -
 32*b^13 - 64*c^13 + 96*a^2*b^11 - 96*a^3*b^10 - 96*a^4*b^9 + 96*a^5*b^8 +
32*a^6*b^7 - 32*a^7*b^6 - 1728*a^2*c^11 - 3072*a^3*c^10 - 2688*a^4*c^9 + 26
88*a^6*c^7 + 3072*a^7*c^6 + 1728*a^8*c^5 + 512*a^9*c^4 + 64*a^10*c^3 + 160*
b^2*c^11 - 544*b^3*c^10 + 64*b^4*c^9 + 896*b^5*c^8 - 608*b^6*c^7 - 672*b^7*
c^6 + 800*b^8*c^5 + 160*b^9*c^4 - 448*b^10*c^3 + 64*b^11*c^2 + 480*a*b^2*c^
10 - 4352*a*b^3*c^9 + 2560*a*b^4*c^8 + 5248*a*b^5*c^7 - 5664*a*b^6*c^6 - 22
40*a*b^7*c^5 + 4320*a*b^8*c^4 - 256*a*b^9*c^3 - 1216*a*b^10*c^2 + 5632*a^2*
b*c^10 - 672*a^2*b^10*c + 14336*a^3*b*c^9 - 768*a^3*b^9*c + 23296*a^4*b*c^8
 + 1248*a^4*b^8*c + 25088*a^5*b*c^7 + 576*a^5*b^7*c + 17920*a^6*b*c^6 - 864
*a^6*b^6*c + 8192*a^7*b*c^5 - 128*a^7*b^5*c + 2176*a^8*b*c^4 + 192*a^8*b^4*
c + 256*a^9*b*c^3 - 1408*a^2*b^2*c^9 - 14720*a^2*b^3*c^8 + 13440*a^2*b^4*c^
7 + 11904*a^2*b^5*c^6 - 16800*a^2*b^6*c^5 - 1696*a^2*b^7*c^4 + 7168*a^2*b^8
*c^3 - 1216*a^2*b^9*c^2 - 9856*a^3*b^2*c^8 - 27392*a^3*b^3*c^7 + 31232*a^3*
b^4*c^6 + 12928*a^3*b^5*c^5 - 23264*a^3*b^6*c^4 + 1152*a^3*b^7*c^3 + 4800*a
^3*b^8*c^2 - 22848*a^4*b^2*c^7 - 30400*a^4*b^3*c^6 + 39680*a^4*b^4*c^5 + 62
72*a^4*b^5*c^4 - 16544*a^4*b^6*c^3 + 1824*a^4*b^7*c^2 - 29120*a^5*b^2*c^6 -
 20224*a^5*b^3*c^5 + 29184*a^5*b^4*c^4 + 384*a^5*b^5*c^3 - 5856*a^5*b^6*c^2
 - 22400*a^6*b^2*c^5 - 7552*a^6*b^3*c^4 + 12160*a^6*b^4*c^3 - 640*a^6*b^5*c
^2 - 10368*a^7*b^2*c^4 - 1280*a^7*b^3*c^3 + 2560*a^7*b^4*c^2 - 2656*a^8*b^2
*c^3 - 32*a^8*b^3*c^2 - 288*a^9*b^2*c^2 + 1280*a*b*c^11 + 320*a*b^11*c))*(-
(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^
3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3
+ 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a
^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*
(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(
-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^
2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 +
 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5
- 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6
 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 4
48*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*
b^8*c)))^(1/2) + ((-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 +
b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2
*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 3
3*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*
b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^2)^3)
^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b
^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^
6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2
*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^
```

```
2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 +
 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96
*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(128*a*c^13 - 64*a*b^13 - 32*b^13*c + 32
*b^14 - 96*a^2*b^12 + 256*a^3*b^11 + 64*a^4*b^10 - 384*a^5*b^9 + 64*a^6*b^8
 + 256*a^7*b^7 - 96*a^8*b^6 - 64*a^9*b^5 + 32*a^10*b^4 + 1408*a^2*c^12 + 70
40*a^3*c^11 + 21120*a^4*c^10 + 42240*a^5*c^9 + 59136*a^6*c^8 + 59136*a^7*c^
7 + 42240*a^8*c^6 + 21120*a^9*c^5 + 7040*a^10*c^4 + 1408*a^11*c^3 + 128*a^1
2*c^2 - 32*b^2*c^12 + 96*b^3*c^11 + 64*b^4*c^10 - 416*b^5*c^9 + 96*b^6*c^8
+ 704*b^7*c^7 - 384*b^8*c^6 - 576*b^9*c^5 + 416*b^10*c^4 + 224*b^11*c^3 - 1
92*b^12*c^2 - tan(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c
^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*
a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^
4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) -
10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b^
2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*
a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 2
40*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*
a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a
^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*
c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3
 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(64*a*b^14 - 256*a*c^14 + 256*a^14*
c - 64*b^14*c - 128*a^2*b^13 - 256*a^3*b^12 + 640*a^4*b^11 + 320*a^5*b^10 -
 1280*a^6*b^9 + 1280*a^8*b^7 - 320*a^9*b^6 - 640*a^10*b^5 + 256*a^11*b^4 +
128*a^12*b^3 - 64*a^13*b^2 - 2816*a^2*c^13 - 13824*a^3*c^12 - 39424*a^4*c^1
1 - 70400*a^5*c^10 - 76032*a^6*c^9 - 33792*a^7*c^8 + 33792*a^8*c^7 + 76032*
a^9*c^6 + 70400*a^10*c^5 + 39424*a^11*c^4 + 13824*a^12*c^3 + 2816*a^13*c^2
+ 64*b^2*c^13 - 128*b^3*c^12 - 256*b^4*c^11 + 640*b^5*c^10 + 320*b^6*c^9 -
1280*b^7*c^8 + 1280*b^9*c^6 - 320*b^10*c^5 - 640*b^11*c^4 + 256*b^12*c^3 +
128*b^13*c^2 + 1728*a*b^2*c^12 - 3840*a*b^3*c^11 - 3584*a*b^4*c^10 + 10240*
a*b^5*c^9 + 2240*a*b^6*c^8 - 12800*a*b^7*c^7 + 1280*a*b^8*c^6 + 7680*a*b^9*
c^5 - 1984*a*b^10*c^4 - 1792*a*b^11*c^3 + 512*a*b^12*c^2 + 5120*a^2*b*c^12
- 512*a^2*b^12*c + 22528*a^3*b*c^11 + 1792*a^3*b^11*c + 56320*a^4*b*c^10 +
1984*a^4*b^10*c + 84480*a^5*b*c^9 - 7680*a^5*b^9*c + 67584*a^6*b*c^8 - 1280
*a^6*b^8*c + 12800*a^7*b^7*c - 67584*a^8*b*c^6 - 2240*a^8*b^6*c - 84480*a^9
*b*c^5 - 10240*a^9*b^5*c - 56320*a^10*b*c^4 + 3584*a^10*b^4*c - 22528*a^11*
b*c^3 + 3840*a^11*b^3*c - 5120*a^12*b*c^2 - 1728*a^12*b^2*c + 12672*a^2*b^2
*c^11 - 26112*a^2*b^3*c^10 - 17920*a^2*b^4*c^9 + 48000*a^2*b^5*c^8 + 6400*a
^2*b^6*c^7 - 38400*a^2*b^7*c^6 + 3840*a^2*b^8*c^5 + 11520*a^2*b^9*c^4 - 166
4*a^2*b^10*c^3 + 45696*a^3*b^2*c^10 - 83200*a^3*b^3*c^9 - 44800*a^3*b^4*c^8
 + 102400*a^3*b^5*c^7 + 8960*a^3*b^6*c^6 - 43520*a^3*b^7*c^5 + 2560*a^3*b^8
*c^4 + 1664*a^3*b^10*c^2 + 94400*a^4*b^2*c^9 - 144000*a^4*b^3*c^8 - 58880*a
^4*b^4*c^7 + 98560*a^4*b^5*c^6 + 4480*a^4*b^6*c^5 - 2560*a^4*b^8*c^3 - 1152
0*a^4*b^9*c^2 + 111168*a^5*b^2*c^8 - 124416*a^5*b^3*c^7 - 28672*a^5*b^4*c^6
 - 4480*a^5*b^6*c^4 + 43520*a^5*b^7*c^3 - 3840*a^5*b^8*c^2 + 51456*a^6*b^2*
c^7 + 28672*a^6*b^4*c^5 - 98560*a^6*b^5*c^4 - 8960*a^6*b^6*c^3 + 38400*a^6*
```

b^7*c^2 - 51456*a^7*b^2*c^6 + 124416*a^7*b^3*c^5 + 58880*a^7*b^4*c^4 - 1024
00*a^7*b^5*c^3 - 6400*a^7*b^6*c^2 - 111168*a^8*b^2*c^5 + 144000*a^8*b^3*c^4
 + 44800*a^8*b^4*c^3 - 48000*a^8*b^5*c^2 - 94400*a^9*b^2*c^4 + 83200*a^9*b^
3*c^3 + 17920*a^9*b^4*c^2 - 45696*a^10*b^2*c^3 + 26112*a^10*b^3*c^2 - 12672
*a^11*b^2*c^2 + 512*a*b*c^13 - 512*a^13*b*c) - 608*a*b^2*c^11 + 2624*a*b^3*
c^10 + 224*a*b^4*c^9 - 6208*a*b^5*c^8 + 2112*a*b^6*c^7 + 6784*a*b^7*c^6 - 3
520*a*b^8*c^5 - 3584*a*b^9*c^4 + 2080*a*b^10*c^3 + 832*a*b^11*c^2 - 3840*a^
2*b*c^11 + 992*a^2*b^11*c - 17280*a^3*b*c^10 + 992*a^3*b^10*c - 46080*a^4*b
*c^9 - 3136*a^4*b^9*c - 80640*a^5*b*c^8 - 320*a^5*b^8*c - 96768*a^6*b*c^7 +
 3776*a^6*b^7*c - 80640*a^7*b*c^6 - 832*a^7*b^6*c - 46080*a^8*b*c^5 - 1952*
a^8*b^5*c - 17280*a^9*b*c^4 + 736*a^9*b^4*c - 3840*a^10*b*c^3 + 352*a^10*b^
3*c - 384*a^11*b*c^2 - 160*a^11*b^2*c - 4192*a^2*b^2*c^10 + 17888*a^2*b^3*c
^9 + 288*a^2*b^4*c^8 - 30080*a^2*b^5*c^7 + 8768*a^2*b^6*c^6 + 22848*a^2*b^7
*c^5 - 8768*a^2*b^8*c^4 - 7808*a^2*b^9*c^3 + 2592*a^2*b^10*c^2 - 15648*a^3*
b^2*c^9 + 60160*a^3*b^3*c^8 + 1152*a^3*b^4*c^7 - 73472*a^3*b^5*c^6 + 15424*
a^3*b^6*c^5 + 37888*a^3*b^7*c^4 - 8960*a^3*b^8*c^3 - 7552*a^3*b^9*c^2 - 366
72*a^4*b^2*c^8 + 120512*a^4*b^3*c^7 + 5376*a^4*b^4*c^6 - 104384*a^4*b^5*c^5
 + 12800*a^4*b^6*c^4 + 34112*a^4*b^7*c^3 - 3712*a^4*b^8*c^2 - 57792*a^5*b^2
*c^7 + 155008*a^5*b^3*c^6 + 12096*a^5*b^4*c^5 - 90496*a^5*b^5*c^4 + 3776*a^
5*b^6*c^3 + 16512*a^5*b^7*c^2 - 63168*a^6*b^2*c^6 + 131264*a^6*b^3*c^5 + 14
784*a^6*b^4*c^4 - 47488*a^6*b^5*c^3 - 1088*a^6*b^6*c^2 - 48192*a^7*b^2*c^5
+ 72448*a^7*b^3*c^4 + 10368*a^7*b^4*c^3 - 14080*a^7*b^5*c^2 - 25248*a^8*b^2
*c^4 + 24800*a^8*b^3*c^3 + 4032*a^8*b^4*c^2 - 8672*a^9*b^2*c^3 + 4672*a^9*b
^3*c^2 - 1760*a^10*b^2*c^2 - 384*a*b*c^12 - 416*a*b^12*c) - tan(x/2)*(32*a*
b^12 - 512*a*c^12 + 128*b*c^12 + 96*b^12*c - 32*b^13 - 64*c^13 + 96*a^2*b^1
1 - 96*a^3*b^10 - 96*a^4*b^9 + 96*a^5*b^8 + 32*a^6*b^7 - 32*a^7*b^6 - 1728*
a^2*c^11 - 3072*a^3*c^10 - 2688*a^4*c^9 + 2688*a^6*c^7 + 3072*a^7*c^6 + 172
8*a^8*c^5 + 512*a^9*c^4 + 64*a^10*c^3 + 160*b^2*c^11 - 544*b^3*c^10 + 64*b^
4*c^9 + 896*b^5*c^8 - 608*b^6*c^7 - 672*b^7*c^6 + 800*b^8*c^5 + 160*b^9*c^4
 - 448*b^10*c^3 + 64*b^11*c^2 + 480*a*b^2*c^10 - 4352*a*b^3*c^9 + 2560*a*b^
4*c^8 + 5248*a*b^5*c^7 - 5664*a*b^6*c^6 - 2240*a*b^7*c^5 + 4320*a*b^8*c^4 -
 256*a*b^9*c^3 - 1216*a*b^10*c^2 + 5632*a^2*b*c^10 - 672*a^2*b^10*c + 14336
*a^3*b*c^9 - 768*a^3*b^9*c + 23296*a^4*b*c^8 + 1248*a^4*b^8*c + 25088*a^5*b
*c^7 + 576*a^5*b^7*c + 17920*a^6*b*c^6 - 864*a^6*b^6*c + 8192*a^7*b*c^5 - 1
28*a^7*b^5*c + 2176*a^8*b*c^4 + 192*a^8*b^4*c + 256*a^9*b*c^3 - 1408*a^2*b^
2*c^9 - 14720*a^2*b^3*c^8 + 13440*a^2*b^4*c^7 + 11904*a^2*b^5*c^6 - 16800*a
^2*b^6*c^5 - 1696*a^2*b^7*c^4 + 7168*a^2*b^8*c^3 - 1216*a^2*b^9*c^2 - 9856*
a^3*b^2*c^8 - 27392*a^3*b^3*c^7 + 31232*a^3*b^4*c^6 + 12928*a^3*b^5*c^5 - 2
3264*a^3*b^6*c^4 + 1152*a^3*b^7*c^3 + 4800*a^3*b^8*c^2 - 22848*a^4*b^2*c^7
- 30400*a^4*b^3*c^6 + 39680*a^4*b^4*c^5 + 6272*a^4*b^5*c^4 - 16544*a^4*b^6*
c^3 + 1824*a^4*b^7*c^2 - 29120*a^5*b^2*c^6 - 20224*a^5*b^3*c^5 + 29184*a^5*
b^4*c^4 + 384*a^5*b^5*c^3 - 5856*a^5*b^6*c^2 - 22400*a^6*b^2*c^5 - 7552*a^6
*b^3*c^4 + 12160*a^6*b^4*c^3 - 640*a^6*b^5*c^2 - 10368*a^7*b^2*c^4 - 1280*a
^7*b^3*c^3 + 2560*a^7*b^4*c^2 - 2656*a^8*b^2*c^3 - 32*a^8*b^3*c^2 - 288*a^9
*b^2*c^2 + 1280*a*b*c^11 + 320*a*b^11*c))*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 2

4*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^
4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1
/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c
 - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*
b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*
a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^
6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c
^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c +
 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6
*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^
2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2) + 1792*a^2*c^10
+ 3584*a^3*c^9 + 4480*a^4*c^8 + 3584*a^5*c^7 + 1792*a^6*c^6 + 512*a^7*c^5 +
 64*a^8*c^4 - 320*b^2*c^10 + 64*b^3*c^9 + 576*b^4*c^8 - 192*b^5*c^7 - 448*b
^6*c^6 + 192*b^7*c^5 + 128*b^8*c^4 - 64*b^9*c^3 - 1984*a*b^2*c^9 + 384*a*b^
3*c^8 + 2496*a*b^4*c^7 - 768*a*b^5*c^6 - 1088*a*b^6*c^5 + 384*a*b^7*c^4 + 6
4*a*b^8*c^3 - 5184*a^2*b^2*c^8 + 960*a^2*b^3*c^7 + 4224*a^2*b^4*c^6 - 1152*
a^2*b^5*c^5 - 832*a^2*b^6*c^4 + 192*a^2*b^7*c^3 - 7360*a^3*b^2*c^7 + 1280*a
^3*b^3*c^6 + 3456*a^3*b^4*c^5 - 768*a^3*b^5*c^4 - 192*a^3*b^6*c^3 - 6080*a^
4*b^2*c^6 + 960*a^4*b^3*c^5 + 1344*a^4*b^4*c^4 - 192*a^4*b^5*c^3 - 2880*a^5
*b^2*c^5 + 384*a^5*b^3*c^4 + 192*a^5*b^4*c^3 - 704*a^6*b^2*c^4 + 64*a^6*b^3
*c^3 - 64*a^7*b^2*c^3))*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*
c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18
*a*b^2*c^5 + 24*a*b^4*c^3 + 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c
^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) -
 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b*c^3*(-(4*a*c - b
^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3
*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 +
240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8
*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*
a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2
*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^
3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*2i + atan((((-(8*a*c^7 + b^8 + 24*
a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6
 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c -
 b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c
^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(
1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1
/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 +
 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^
6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*
a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 -
 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159
*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(128*
a*c^13 - 64*a*b^13 - 32*b^13*c + 32*b^14 - 96*a^2*b^12 + 256*a^3*b^11 + 64*
a^4*b^10 - 384*a^5*b^9 + 64*a^6*b^8 + 256*a^7*b^7 - 96*a^8*b^6 - 64*a^9*b^5

+ 32*a^10*b^4 + 1408*a^2*c^12 + 7040*a^3*c^11 + 21120*a^4*c^10 + 42240*a^5
*c^9 + 59136*a^6*c^8 + 59136*a^7*c^7 + 42240*a^8*c^6 + 21120*a^9*c^5 + 7040
*a^10*c^4 + 1408*a^11*c^3 + 128*a^12*c^2 - 32*b^2*c^12 + 96*b^3*c^11 + 64*b
^4*c^10 - 416*b^5*c^9 + 96*b^6*c^8 + 704*b^7*c^7 - 384*b^8*c^6 - 576*b^9*c^
5 + 416*b^10*c^4 + 224*b^11*c^3 - 192*b^12*c^2 + tan(x/2)*(-(8*a*c^7 + b^8
+ 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^
2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*
a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*
b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)
^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^
3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*
c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b
^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3
- 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*
c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4
+ 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*
(64*a*b^14 - 256*a*c^14 + 256*a^14*c - 64*b^14*c - 128*a^2*b^13 - 256*a^3*b
^12 + 640*a^4*b^11 + 320*a^5*b^10 - 1280*a^6*b^9 + 1280*a^8*b^7 - 320*a^9*b
^6 - 640*a^10*b^5 + 256*a^11*b^4 + 128*a^12*b^3 - 64*a^13*b^2 - 2816*a^2*c^
13 - 13824*a^3*c^12 - 39424*a^4*c^11 - 70400*a^5*c^10 - 76032*a^6*c^9 - 337
92*a^7*c^8 + 33792*a^8*c^7 + 76032*a^9*c^6 + 70400*a^10*c^5 + 39424*a^11*c^
4 + 13824*a^12*c^3 + 2816*a^13*c^2 + 64*b^2*c^13 - 128*b^3*c^12 - 256*b^4*c
^11 + 640*b^5*c^10 + 320*b^6*c^9 - 1280*b^7*c^8 + 1280*b^9*c^6 - 320*b^10*c
^5 - 640*b^11*c^4 + 256*b^12*c^3 + 128*b^13*c^2 + 1728*a*b^2*c^12 - 3840*a*
b^3*c^11 - 3584*a*b^4*c^10 + 10240*a*b^5*c^9 + 2240*a*b^6*c^8 - 12800*a*b^7
*c^7 + 1280*a*b^8*c^6 + 7680*a*b^9*c^5 - 1984*a*b^10*c^4 - 1792*a*b^11*c^3
+ 512*a*b^12*c^2 + 5120*a^2*b*c^12 - 512*a^2*b^12*c + 22528*a^3*b*c^11 + 17
92*a^3*b^11*c + 56320*a^4*b*c^10 + 1984*a^4*b^10*c + 84480*a^5*b*c^9 - 7680
*a^5*b^9*c + 67584*a^6*b*c^8 - 1280*a^6*b^8*c + 12800*a^7*b^7*c - 67584*a^8
*b*c^6 - 2240*a^8*b^6*c - 84480*a^9*b*c^5 - 10240*a^9*b^5*c - 56320*a^10*b*
c^4 + 3584*a^10*b^4*c - 22528*a^11*b*c^3 + 3840*a^11*b^3*c - 5120*a^12*b*c^
2 - 1728*a^12*b^2*c + 12672*a^2*b^2*c^11 - 26112*a^2*b^3*c^10 - 17920*a^2*b
^4*c^9 + 48000*a^2*b^5*c^8 + 6400*a^2*b^6*c^7 - 38400*a^2*b^7*c^6 + 3840*a^
2*b^8*c^5 + 11520*a^2*b^9*c^4 - 1664*a^2*b^10*c^3 + 45696*a^3*b^2*c^10 - 83
200*a^3*b^3*c^9 - 44800*a^3*b^4*c^8 + 102400*a^3*b^5*c^7 + 8960*a^3*b^6*c^6
- 43520*a^3*b^7*c^5 + 2560*a^3*b^8*c^4 + 1664*a^3*b^10*c^2 + 94400*a^4*b^2
*c^9 - 144000*a^4*b^3*c^8 - 58880*a^4*b^4*c^7 + 98560*a^4*b^5*c^6 + 4480*a^
4*b^6*c^5 - 2560*a^4*b^8*c^3 - 11520*a^4*b^9*c^2 + 111168*a^5*b^2*c^8 - 124
416*a^5*b^3*c^7 - 28672*a^5*b^4*c^6 - 4480*a^5*b^6*c^4 + 43520*a^5*b^7*c^3
- 3840*a^5*b^8*c^2 + 51456*a^6*b^2*c^7 + 28672*a^6*b^4*c^5 - 98560*a^6*b^5*
c^4 - 8960*a^6*b^6*c^3 + 38400*a^6*b^7*c^2 - 51456*a^7*b^2*c^6 + 124416*a^7
*b^3*c^5 + 58880*a^7*b^4*c^4 - 102400*a^7*b^5*c^3 - 6400*a^7*b^6*c^2 - 1111
68*a^8*b^2*c^5 + 144000*a^8*b^3*c^4 + 44800*a^8*b^4*c^3 - 48000*a^8*b^5*c^2
- 94400*a^9*b^2*c^4 + 83200*a^9*b^3*c^3 + 17920*a^9*b^4*c^2 - 45696*a^10*b
^2*c^3 + 26112*a^10*b^3*c^2 - 12672*a^11*b^2*c^2 + 512*a*b*c^13 - 512*a^13*

b*c) - 608*a*b^2*c^11 + 2624*a*b^3*c^10 + 224*a*b^4*c^9 - 6208*a*b^5*c^8 +
2112*a*b^6*c^7 + 6784*a*b^7*c^6 - 3520*a*b^8*c^5 - 3584*a*b^9*c^4 + 2080*a*
b^10*c^3 + 832*a*b^11*c^2 - 3840*a^2*b*c^11 + 992*a^2*b^11*c - 17280*a^3*b*
c^10 + 992*a^3*b^10*c - 46080*a^4*b*c^9 - 3136*a^4*b^9*c - 80640*a^5*b*c^8
- 320*a^5*b^8*c - 96768*a^6*b*c^7 + 3776*a^6*b^7*c - 80640*a^7*b*c^6 - 832*
a^7*b^6*c - 46080*a^8*b*c^5 - 1952*a^8*b^5*c - 17280*a^9*b*c^4 + 736*a^9*b^
4*c - 3840*a^10*b*c^3 + 352*a^10*b^3*c - 384*a^11*b*c^2 - 160*a^11*b^2*c -
4192*a^2*b^2*c^10 + 17888*a^2*b^3*c^9 + 288*a^2*b^4*c^8 - 30080*a^2*b^5*c^7
 + 8768*a^2*b^6*c^6 + 22848*a^2*b^7*c^5 - 8768*a^2*b^8*c^4 - 7808*a^2*b^9*c
^3 + 2592*a^2*b^10*c^2 - 15648*a^3*b^2*c^9 + 60160*a^3*b^3*c^8 + 1152*a^3*b
^4*c^7 - 73472*a^3*b^5*c^6 + 15424*a^3*b^6*c^5 + 37888*a^3*b^7*c^4 - 8960*a
^3*b^8*c^3 - 7552*a^3*b^9*c^2 - 36672*a^4*b^2*c^8 + 120512*a^4*b^3*c^7 + 53
76*a^4*b^4*c^6 - 104384*a^4*b^5*c^5 + 12800*a^4*b^6*c^4 + 34112*a^4*b^7*c^3
 - 3712*a^4*b^8*c^2 - 57792*a^5*b^2*c^7 + 155008*a^5*b^3*c^6 + 12096*a^5*b^
4*c^5 - 90496*a^5*b^5*c^4 + 3776*a^5*b^6*c^3 + 16512*a^5*b^7*c^2 - 63168*a^
6*b^2*c^6 + 131264*a^6*b^3*c^5 + 14784*a^6*b^4*c^4 - 47488*a^6*b^5*c^3 - 10
88*a^6*b^6*c^2 - 48192*a^7*b^2*c^5 + 72448*a^7*b^3*c^4 + 10368*a^7*b^4*c^3
- 14080*a^7*b^5*c^2 - 25248*a^8*b^2*c^4 + 24800*a^8*b^3*c^3 + 4032*a^8*b^4*
c^2 - 8672*a^9*b^2*c^3 + 4672*a^9*b^3*c^2 - 1760*a^10*b^2*c^2 - 384*a*b*c^1
2 - 416*a*b^12*c) + tan(x/2)*(32*a*b^12 - 512*a*c^12 + 128*b*c^12 + 96*b^12
*c - 32*b^13 - 64*c^13 + 96*a^2*b^11 - 96*a^3*b^10 - 96*a^4*b^9 + 96*a^5*b^
8 + 32*a^6*b^7 - 32*a^7*b^6 - 1728*a^2*c^11 - 3072*a^3*c^10 - 2688*a^4*c^9
+ 2688*a^6*c^7 + 3072*a^7*c^6 + 1728*a^8*c^5 + 512*a^9*c^4 + 64*a^10*c^3 +
160*b^2*c^11 - 544*b^3*c^10 + 64*b^4*c^9 + 896*b^5*c^8 - 608*b^6*c^7 - 672*
b^7*c^6 + 800*b^8*c^5 + 160*b^9*c^4 - 448*b^10*c^3 + 64*b^11*c^2 + 480*a*b^
2*c^10 - 4352*a*b^3*c^9 + 2560*a*b^4*c^8 + 5248*a*b^5*c^7 - 5664*a*b^6*c^6
- 2240*a*b^7*c^5 + 4320*a*b^8*c^4 - 256*a*b^9*c^3 - 1216*a*b^10*c^2 + 5632*
a^2*b*c^10 - 672*a^2*b^10*c + 14336*a^3*b*c^9 - 768*a^3*b^9*c + 23296*a^4*b
*c^8 + 1248*a^4*b^8*c + 25088*a^5*b*c^7 + 576*a^5*b^7*c + 17920*a^6*b*c^6 -
 864*a^6*b^6*c + 8192*a^7*b*c^5 - 128*a^7*b^5*c + 2176*a^8*b*c^4 + 192*a^8*
b^4*c + 256*a^9*b*c^3 - 1408*a^2*b^2*c^9 - 14720*a^2*b^3*c^8 + 13440*a^2*b^
4*c^7 + 11904*a^2*b^5*c^6 - 16800*a^2*b^6*c^5 - 1696*a^2*b^7*c^4 + 7168*a^2
*b^8*c^3 - 1216*a^2*b^9*c^2 - 9856*a^3*b^2*c^8 - 27392*a^3*b^3*c^7 + 31232*
a^3*b^4*c^6 + 12928*a^3*b^5*c^5 - 23264*a^3*b^6*c^4 + 1152*a^3*b^7*c^3 + 48
00*a^3*b^8*c^2 - 22848*a^4*b^2*c^7 - 30400*a^4*b^3*c^6 + 39680*a^4*b^4*c^5
+ 6272*a^4*b^5*c^4 - 16544*a^4*b^6*c^3 + 1824*a^4*b^7*c^2 - 29120*a^5*b^2*c
^6 - 20224*a^5*b^3*c^5 + 29184*a^5*b^4*c^4 + 384*a^5*b^5*c^3 - 5856*a^5*b^6
*c^2 - 22400*a^6*b^2*c^5 - 7552*a^6*b^3*c^4 + 12160*a^6*b^4*c^3 - 640*a^6*b
^5*c^2 - 10368*a^7*b^2*c^4 - 1280*a^7*b^3*c^3 + 2560*a^7*b^4*c^2 - 2656*a^8
*b^2*c^3 - 32*a^8*b^3*c^2 - 288*a^9*b^2*c^2 + 1280*a*b*c^11 + 320*a*b^11*c)
)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b
^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*
c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 -
38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*
c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3

```
*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 1
6*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c
^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*
c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2
*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3
 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 1
4*a*b^8*c)))^(1/2)*1i - ((-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4
*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 1
8*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*
c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2)
- 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c -
b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 -
3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 +
 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 -
8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8
*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^
2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c
^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(128*a*c^13 - 64*a*b^13 - 32*b^13
*c + 32*b^14 - 96*a^2*b^12 + 256*a^3*b^11 + 64*a^4*b^10 - 384*a^5*b^9 + 64*
a^6*b^8 + 256*a^7*b^7 - 96*a^8*b^6 - 64*a^9*b^5 + 32*a^10*b^4 + 1408*a^2*c^
12 + 7040*a^3*c^11 + 21120*a^4*c^10 + 42240*a^5*c^9 + 59136*a^6*c^8 + 59136
*a^7*c^7 + 42240*a^8*c^6 + 21120*a^9*c^5 + 7040*a^10*c^4 + 1408*a^11*c^3 +
128*a^12*c^2 - 32*b^2*c^12 + 96*b^3*c^11 + 64*b^4*c^10 - 416*b^5*c^9 + 96*b
^6*c^8 + 704*b^7*c^7 - 384*b^8*c^6 - 576*b^9*c^5 + 416*b^10*c^4 + 224*b^11*
c^3 - 192*b^12*c^2 - tan(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 +
8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^
2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2
*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(
1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a
*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^
10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*
c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c
^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*
c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a
^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*
b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(64*a*b^14 - 256*a*c^14 + 25
6*a^14*c - 64*b^14*c - 128*a^2*b^13 - 256*a^3*b^12 + 640*a^4*b^11 + 320*a^5
*b^10 - 1280*a^6*b^9 + 1280*a^8*b^7 - 320*a^9*b^6 - 640*a^10*b^5 + 256*a^11
*b^4 + 128*a^12*b^3 - 64*a^13*b^2 - 2816*a^2*c^13 - 13824*a^3*c^12 - 39424*
a^4*c^11 - 70400*a^5*c^10 - 76032*a^6*c^9 - 33792*a^7*c^8 + 33792*a^8*c^7 +
 76032*a^9*c^6 + 70400*a^10*c^5 + 39424*a^11*c^4 + 13824*a^12*c^3 + 2816*a^
13*c^2 + 64*b^2*c^13 - 128*b^3*c^12 - 256*b^4*c^11 + 640*b^5*c^10 + 320*b^6
*c^9 - 1280*b^7*c^8 + 1280*b^9*c^6 - 320*b^10*c^5 - 640*b^11*c^4 + 256*b^12
*c^3 + 128*b^13*c^2 + 1728*a*b^2*c^12 - 3840*a*b^3*c^11 - 3584*a*b^4*c^10 +
 10240*a*b^5*c^9 + 2240*a*b^6*c^8 - 12800*a*b^7*c^7 + 1280*a*b^8*c^6 + 7680
```

*a*b^9*c^5 - 1984*a*b^10*c^4 - 1792*a*b^11*c^3 + 512*a*b^12*c^2 + 5120*a^2*
b*c^12 - 512*a^2*b^12*c + 22528*a^3*b*c^11 + 1792*a^3*b^11*c + 56320*a^4*b*
c^10 + 1984*a^4*b^10*c + 84480*a^5*b*c^9 - 7680*a^5*b^9*c + 67584*a^6*b*c^8
 - 1280*a^6*b^8*c + 12800*a^7*b^7*c - 67584*a^8*b*c^6 - 2240*a^8*b^6*c - 84
480*a^9*b*c^5 - 10240*a^9*b^5*c - 56320*a^10*b*c^4 + 3584*a^10*b^4*c - 2252
8*a^11*b*c^3 + 3840*a^11*b^3*c - 5120*a^12*b*c^2 - 1728*a^12*b^2*c + 12672*
a^2*b^2*c^11 - 26112*a^2*b^3*c^10 - 17920*a^2*b^4*c^9 + 48000*a^2*b^5*c^8 +
 6400*a^2*b^6*c^7 - 38400*a^2*b^7*c^6 + 3840*a^2*b^8*c^5 + 11520*a^2*b^9*c^
4 - 1664*a^2*b^10*c^3 + 45696*a^3*b^2*c^10 - 83200*a^3*b^3*c^9 - 44800*a^3*
b^4*c^8 + 102400*a^3*b^5*c^7 + 8960*a^3*b^6*c^6 - 43520*a^3*b^7*c^5 + 2560*
a^3*b^8*c^4 + 1664*a^3*b^10*c^2 + 94400*a^4*b^2*c^9 - 144000*a^4*b^3*c^8 -
58880*a^4*b^4*c^7 + 98560*a^4*b^5*c^6 + 4480*a^4*b^6*c^5 - 2560*a^4*b^8*c^3
 - 11520*a^4*b^9*c^2 + 111168*a^5*b^2*c^8 - 124416*a^5*b^3*c^7 - 28672*a^5*
b^4*c^6 - 4480*a^5*b^6*c^4 + 43520*a^5*b^7*c^3 - 3840*a^5*b^8*c^2 + 51456*a
^6*b^2*c^7 + 28672*a^6*b^4*c^5 - 98560*a^6*b^5*c^4 - 8960*a^6*b^6*c^3 + 384
00*a^6*b^7*c^2 - 51456*a^7*b^2*c^6 + 124416*a^7*b^3*c^5 + 58880*a^7*b^4*c^4
 - 102400*a^7*b^5*c^3 - 6400*a^7*b^6*c^2 - 111168*a^8*b^2*c^5 + 144000*a^8*
b^3*c^4 + 44800*a^8*b^4*c^3 - 48000*a^8*b^5*c^2 - 94400*a^9*b^2*c^4 + 83200
*a^9*b^3*c^3 + 17920*a^9*b^4*c^2 - 45696*a^10*b^2*c^3 + 26112*a^10*b^3*c^2
 - 12672*a^11*b^2*c^2 + 512*a*b*c^13 - 512*a^13*b*c) - 608*a*b^2*c^11 + 2624
*a*b^3*c^10 + 224*a*b^4*c^9 - 6208*a*b^5*c^8 + 2112*a*b^6*c^7 + 6784*a*b^7*
c^6 - 3520*a*b^8*c^5 - 3584*a*b^9*c^4 + 2080*a*b^10*c^3 + 832*a*b^11*c^2 -
3840*a^2*b*c^11 + 992*a^2*b^11*c - 17280*a^3*b*c^10 + 992*a^3*b^10*c - 4608
0*a^4*b*c^9 - 3136*a^4*b^9*c - 80640*a^5*b*c^8 - 320*a^5*b^8*c - 96768*a^6*
b*c^7 + 3776*a^6*b^7*c - 80640*a^7*b*c^6 - 832*a^7*b^6*c - 46080*a^8*b*c^5
 - 1952*a^8*b^5*c - 17280*a^9*b*c^4 + 736*a^9*b^4*c - 3840*a^10*b*c^3 + 352*
a^10*b^3*c - 384*a^11*b*c^2 - 160*a^11*b^2*c - 4192*a^2*b^2*c^10 + 17888*a^
2*b^3*c^9 + 288*a^2*b^4*c^8 - 30080*a^2*b^5*c^7 + 8768*a^2*b^6*c^6 + 22848*
a^2*b^7*c^5 - 8768*a^2*b^8*c^4 - 7808*a^2*b^9*c^3 + 2592*a^2*b^10*c^2 - 156
48*a^3*b^2*c^9 + 60160*a^3*b^3*c^8 + 1152*a^3*b^4*c^7 - 73472*a^3*b^5*c^6 +
 15424*a^3*b^6*c^5 + 37888*a^3*b^7*c^4 - 8960*a^3*b^8*c^3 - 7552*a^3*b^9*c^
2 - 36672*a^4*b^2*c^8 + 120512*a^4*b^3*c^7 + 5376*a^4*b^4*c^6 - 104384*a^4*
b^5*c^5 + 12800*a^4*b^6*c^4 + 34112*a^4*b^7*c^3 - 3712*a^4*b^8*c^2 - 57792*
a^5*b^2*c^7 + 155008*a^5*b^3*c^6 + 12096*a^5*b^4*c^5 - 90496*a^5*b^5*c^4 +
3776*a^5*b^6*c^3 + 16512*a^5*b^7*c^2 - 63168*a^6*b^2*c^6 + 131264*a^6*b^3*c
^5 + 14784*a^6*b^4*c^4 - 47488*a^6*b^5*c^3 - 1088*a^6*b^6*c^2 - 48192*a^7*b
^2*c^5 + 72448*a^7*b^3*c^4 + 10368*a^7*b^4*c^3 - 14080*a^7*b^5*c^2 - 25248*
a^8*b^2*c^4 + 24800*a^8*b^3*c^3 + 4032*a^8*b^4*c^2 - 8672*a^9*b^2*c^3 + 467
2*a^9*b^3*c^2 - 1760*a^10*b^2*c^2 - 384*a*b*c^12 - 416*a*b^12*c) - tan(x/2)
*(32*a*b^12 - 512*a*c^12 + 128*b*c^12 + 96*b^12*c - 32*b^13 - 64*c^13 + 96*
a^2*b^11 - 96*a^3*b^10 - 96*a^4*b^9 + 96*a^5*b^8 + 32*a^6*b^7 - 32*a^7*b^6
 - 1728*a^2*c^11 - 3072*a^3*c^10 - 2688*a^4*c^9 + 2688*a^6*c^7 + 3072*a^7*c^
6 + 1728*a^8*c^5 + 512*a^9*c^4 + 64*a^10*c^3 + 160*b^2*c^11 - 544*b^3*c^10
 + 64*b^4*c^9 + 896*b^5*c^8 - 608*b^6*c^7 - 672*b^7*c^6 + 800*b^8*c^5 + 160*
b^9*c^4 - 448*b^10*c^3 + 64*b^11*c^2 + 480*a*b^2*c^10 - 4352*a*b^3*c^9 + 25

60*a*b^4*c^8 + 5248*a*b^5*c^7 - 5664*a*b^6*c^6 - 2240*a*b^7*c^5 + 4320*a*b^
8*c^4 - 256*a*b^9*c^3 - 1216*a*b^10*c^2 + 5632*a^2*b*c^10 - 672*a^2*b^10*c
+ 14336*a^3*b*c^9 - 768*a^3*b^9*c + 23296*a^4*b*c^8 + 1248*a^4*b^8*c + 2508
8*a^5*b*c^7 + 576*a^5*b^7*c + 17920*a^6*b*c^6 - 864*a^6*b^6*c + 8192*a^7*b*
c^5 - 128*a^7*b^5*c + 2176*a^8*b*c^4 + 192*a^8*b^4*c + 256*a^9*b*c^3 - 1408
*a^2*b^2*c^9 - 14720*a^2*b^3*c^8 + 13440*a^2*b^4*c^7 + 11904*a^2*b^5*c^6 -
16800*a^2*b^6*c^5 - 1696*a^2*b^7*c^4 + 7168*a^2*b^8*c^3 - 1216*a^2*b^9*c^2
- 9856*a^3*b^2*c^8 - 27392*a^3*b^3*c^7 + 31232*a^3*b^4*c^6 + 12928*a^3*b^5*
c^5 - 23264*a^3*b^6*c^4 + 1152*a^3*b^7*c^3 + 4800*a^3*b^8*c^2 - 22848*a^4*b
^2*c^7 - 30400*a^4*b^3*c^6 + 39680*a^4*b^4*c^5 + 6272*a^4*b^5*c^4 - 16544*a
^4*b^6*c^3 + 1824*a^4*b^7*c^2 - 29120*a^5*b^2*c^6 - 20224*a^5*b^3*c^5 + 291
84*a^5*b^4*c^4 + 384*a^5*b^5*c^3 - 5856*a^5*b^6*c^2 - 22400*a^6*b^2*c^5 - 7
552*a^6*b^3*c^4 + 12160*a^6*b^4*c^3 - 640*a^6*b^5*c^2 - 10368*a^7*b^2*c^4 -
1280*a^7*b^3*c^3 + 2560*a^7*b^4*c^2 - 2656*a^8*b^2*c^3 - 32*a^8*b^3*c^2 -
288*a^9*b^2*c^2 + 1280*a*b*c^11 + 320*a*b^11*c))*(-(8*a*c^7 + b^8 + 24*a^2*
c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3
*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2
)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(
-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2)
- 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))
/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240
*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 -
3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*
b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*
a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4
*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*1i)/(512*
a*c^11 + 64*c^12 + ((-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4
- b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b
^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 +
33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*
a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^
3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4
*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*
a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b
^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*
b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5
+ 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 -
96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(128*a*c^13 - 64*a*b^13 - 32*b^13*c +
32*b^14 - 96*a^2*b^12 + 256*a^3*b^11 + 64*a^4*b^10 - 384*a^5*b^9 + 64*a^6*b
^8 + 256*a^7*b^7 - 96*a^8*b^6 - 64*a^9*b^5 + 32*a^10*b^4 + 1408*a^2*c^12 +
7040*a^3*c^11 + 21120*a^4*c^10 + 42240*a^5*c^9 + 59136*a^6*c^8 + 59136*a^7*
c^7 + 42240*a^8*c^6 + 21120*a^9*c^5 + 7040*a^10*c^4 + 1408*a^11*c^3 + 128*a
^12*c^2 - 32*b^2*c^12 + 96*b^3*c^11 + 64*b^4*c^10 - 416*b^5*c^9 + 96*b^6*c^
8 + 704*b^7*c^7 - 384*b^8*c^6 - 576*b^9*c^5 + 416*b^10*c^4 + 224*b^11*c^3 -
192*b^12*c^2 + tan(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4

c^4 - b^5(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 1
8*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*
c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2)
- 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c -
b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 -
3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 +
240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 -
8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8
*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^
2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c
^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(64*a*b^14 - 256*a*c^14 + 256*a^1
4*c - 64*b^14*c - 128*a^2*b^13 - 256*a^3*b^12 + 640*a^4*b^11 + 320*a^5*b^10
- 1280*a^6*b^9 + 1280*a^8*b^7 - 320*a^9*b^6 - 640*a^10*b^5 + 256*a^11*b^4
+ 128*a^12*b^3 - 64*a^13*b^2 - 2816*a^2*c^13 - 13824*a^3*c^12 - 39424*a^4*c
^11 - 70400*a^5*c^10 - 76032*a^6*c^9 - 33792*a^7*c^8 + 33792*a^8*c^7 + 7603
2*a^9*c^6 + 70400*a^10*c^5 + 39424*a^11*c^4 + 13824*a^12*c^3 + 2816*a^13*c^
2 + 64*b^2*c^13 - 128*b^3*c^12 - 256*b^4*c^11 + 640*b^5*c^10 + 320*b^6*c^9
- 1280*b^7*c^8 + 1280*b^9*c^6 - 320*b^10*c^5 - 640*b^11*c^4 + 256*b^12*c^3
+ 128*b^13*c^2 + 1728*a*b^2*c^12 - 3840*a*b^3*c^11 - 3584*a*b^4*c^10 + 1024
0*a*b^5*c^9 + 2240*a*b^6*c^8 - 12800*a*b^7*c^7 + 1280*a*b^8*c^6 + 7680*a*b^
9*c^5 - 1984*a*b^10*c^4 - 1792*a*b^11*c^3 + 512*a*b^12*c^2 + 5120*a^2*b*c^1
2 - 512*a^2*b^12*c + 22528*a^3*b*c^11 + 1792*a^3*b^11*c + 56320*a^4*b*c^10
+ 1984*a^4*b^10*c + 84480*a^5*b*c^9 - 7680*a^5*b^9*c + 67584*a^6*b*c^8 - 12
80*a^6*b^8*c + 12800*a^7*b^7*c - 67584*a^8*b*c^6 - 2240*a^8*b^6*c - 84480*a
^9*b*c^5 - 10240*a^9*b^5*c - 56320*a^10*b*c^4 + 3584*a^10*b^4*c - 22528*a^1
1*b*c^3 + 3840*a^11*b^3*c - 5120*a^12*b*c^2 - 1728*a^12*b^2*c + 12672*a^2*b
^2*c^11 - 26112*a^2*b^3*c^10 - 17920*a^2*b^4*c^9 + 48000*a^2*b^5*c^8 + 6400
*a^2*b^6*c^7 - 38400*a^2*b^7*c^6 + 3840*a^2*b^8*c^5 + 11520*a^2*b^9*c^4 - 1
664*a^2*b^10*c^3 + 45696*a^3*b^2*c^10 - 83200*a^3*b^3*c^9 - 44800*a^3*b^4*c
^8 + 102400*a^3*b^5*c^7 + 8960*a^3*b^6*c^6 - 43520*a^3*b^7*c^5 + 2560*a^3*b
^8*c^4 + 1664*a^3*b^10*c^2 + 94400*a^4*b^2*c^9 - 144000*a^4*b^3*c^8 - 58880
*a^4*b^4*c^7 + 98560*a^4*b^5*c^6 + 4480*a^4*b^6*c^5 - 2560*a^4*b^8*c^3 - 11
520*a^4*b^9*c^2 + 111168*a^5*b^2*c^8 - 124416*a^5*b^3*c^7 - 28672*a^5*b^4*c
^6 - 4480*a^5*b^6*c^4 + 43520*a^5*b^7*c^3 - 3840*a^5*b^8*c^2 + 51456*a^6*b^
2*c^7 + 28672*a^6*b^4*c^5 - 98560*a^6*b^5*c^4 - 8960*a^6*b^6*c^3 + 38400*a^
6*b^7*c^2 - 51456*a^7*b^2*c^6 + 124416*a^7*b^3*c^5 + 58880*a^7*b^4*c^4 - 10
2400*a^7*b^5*c^3 - 6400*a^7*b^6*c^2 - 111168*a^8*b^2*c^5 + 144000*a^8*b^3*c
^4 + 44800*a^8*b^4*c^3 - 48000*a^8*b^5*c^2 - 94400*a^9*b^2*c^4 + 83200*a^9*
b^3*c^3 + 17920*a^9*b^4*c^2 - 45696*a^10*b^2*c^3 + 26112*a^10*b^3*c^2 - 126
72*a^11*b^2*c^2 + 512*a*b*c^13 - 512*a^13*b*c) - 608*a*b^2*c^11 + 2624*a*b^
3*c^10 + 224*a*b^4*c^9 - 6208*a*b^5*c^8 + 2112*a*b^6*c^7 + 6784*a*b^7*c^6 -
3520*a*b^8*c^5 - 3584*a*b^9*c^4 + 2080*a*b^10*c^3 + 832*a*b^11*c^2 - 3840*
a^2*b*c^11 + 992*a^2*b^11*c - 17280*a^3*b*c^10 + 992*a^3*b^10*c - 46080*a^4
*b*c^9 - 3136*a^4*b^9*c - 80640*a^5*b*c^8 - 320*a^5*b^8*c - 96768*a^6*b*c^7
+ 3776*a^6*b^7*c - 80640*a^7*b*c^6 - 832*a^7*b^6*c - 46080*a^8*b*c^5 - 195

2*a^8*b^5*c - 17280*a^9*b*c^4 + 736*a^9*b^4*c - 3840*a^10*b*c^3 + 352*a^10*
b^3*c - 384*a^11*b*c^2 - 160*a^11*b^2*c - 4192*a^2*b^2*c^10 + 17888*a^2*b^3
*c^9 + 288*a^2*b^4*c^8 - 30080*a^2*b^5*c^7 + 8768*a^2*b^6*c^6 + 22848*a^2*b
^7*c^5 - 8768*a^2*b^8*c^4 - 7808*a^2*b^9*c^3 + 2592*a^2*b^10*c^2 - 15648*a^
3*b^2*c^9 + 60160*a^3*b^3*c^8 + 1152*a^3*b^4*c^7 - 73472*a^3*b^5*c^6 + 1542
4*a^3*b^6*c^5 + 37888*a^3*b^7*c^4 - 8960*a^3*b^8*c^3 - 7552*a^3*b^9*c^2 - 3
6672*a^4*b^2*c^8 + 120512*a^4*b^3*c^7 + 5376*a^4*b^4*c^6 - 104384*a^4*b^5*c
^5 + 12800*a^4*b^6*c^4 + 34112*a^4*b^7*c^3 - 3712*a^4*b^8*c^2 - 57792*a^5*b
^2*c^7 + 155008*a^5*b^3*c^6 + 12096*a^5*b^4*c^5 - 90496*a^5*b^5*c^4 + 3776*
a^5*b^6*c^3 + 16512*a^5*b^7*c^2 - 63168*a^6*b^2*c^6 + 131264*a^6*b^3*c^5 +
14784*a^6*b^4*c^4 - 47488*a^6*b^5*c^3 - 1088*a^6*b^6*c^2 - 48192*a^7*b^2*c^
5 + 72448*a^7*b^3*c^4 + 10368*a^7*b^4*c^3 - 14080*a^7*b^5*c^2 - 25248*a^8*b
^2*c^4 + 24800*a^8*b^3*c^3 + 4032*a^8*b^4*c^2 - 8672*a^9*b^2*c^3 + 4672*a^9
*b^3*c^2 - 1760*a^10*b^2*c^2 - 384*a*b*c^12 - 416*a*b^12*c) + tan(x/2)*(32*
a*b^12 - 512*a*c^12 + 128*b*c^12 + 96*b^12*c - 32*b^13 - 64*c^13 + 96*a^2*b
^11 - 96*a^3*b^10 - 96*a^4*b^9 + 96*a^5*b^8 + 32*a^6*b^7 - 32*a^7*b^6 - 172
8*a^2*c^11 - 3072*a^3*c^10 - 2688*a^4*c^9 + 2688*a^6*c^7 + 3072*a^7*c^6 + 1
728*a^8*c^5 + 512*a^9*c^4 + 64*a^10*c^3 + 160*b^2*c^11 - 544*b^3*c^10 + 64*
b^4*c^9 + 896*b^5*c^8 - 608*b^6*c^7 - 672*b^7*c^6 + 800*b^8*c^5 + 160*b^9*c
^4 - 448*b^10*c^3 + 64*b^11*c^2 + 480*a*b^2*c^10 - 4352*a*b^3*c^9 + 2560*a*
b^4*c^8 + 5248*a*b^5*c^7 - 5664*a*b^6*c^6 - 2240*a*b^7*c^5 + 4320*a*b^8*c^4
 - 256*a*b^9*c^3 - 1216*a*b^10*c^2 + 5632*a^2*b*c^10 - 672*a^2*b^10*c + 143
36*a^3*b*c^9 - 768*a^3*b^9*c + 23296*a^4*b*c^8 + 1248*a^4*b^8*c + 25088*a^5
*b*c^7 + 576*a^5*b^7*c + 17920*a^6*b*c^6 - 864*a^6*b^6*c + 8192*a^7*b*c^5 -
 128*a^7*b^5*c + 2176*a^8*b*c^4 + 192*a^8*b^4*c + 256*a^9*b*c^3 - 1408*a^2*
b^2*c^9 - 14720*a^2*b^3*c^8 + 13440*a^2*b^4*c^7 + 11904*a^2*b^5*c^6 - 16800
*a^2*b^6*c^5 - 1696*a^2*b^7*c^4 + 7168*a^2*b^8*c^3 - 1216*a^2*b^9*c^2 - 985
6*a^3*b^2*c^8 - 27392*a^3*b^3*c^7 + 31232*a^3*b^4*c^6 + 12928*a^3*b^5*c^5 -
 23264*a^3*b^6*c^4 + 1152*a^3*b^7*c^3 + 4800*a^3*b^8*c^2 - 22848*a^4*b^2*c^
7 - 30400*a^4*b^3*c^6 + 39680*a^4*b^4*c^5 + 6272*a^4*b^5*c^4 - 16544*a^4*b^
6*c^3 + 1824*a^4*b^7*c^2 - 29120*a^5*b^2*c^6 - 20224*a^5*b^3*c^5 + 29184*a^
5*b^4*c^4 + 384*a^5*b^5*c^3 - 5856*a^5*b^6*c^2 - 22400*a^6*b^2*c^5 - 7552*a
^6*b^3*c^4 + 12160*a^6*b^4*c^3 - 640*a^6*b^5*c^2 - 10368*a^7*b^2*c^4 - 1280
*a^7*b^3*c^3 + 2560*a^7*b^4*c^2 - 2656*a^8*b^2*c^3 - 32*a^8*b^3*c^2 - 288*a
^9*b^2*c^2 + 1280*a*b*c^11 + 320*a*b^11*c))*(-(8*a*c^7 + b^8 + 24*a^2*c^6 +
 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*
c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^
(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a
*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*
a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(
3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*
c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6
*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c
 + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b
^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*

c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2) + ((-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4 *(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(128*a*c^13 - 64*a*b^13 - 32*b^13*c + 32*b^14 - 96*a^2*b^12 + 256*a^3*b^11 + 64*a^4*b^10 - 384*a^5*b^9 + 64*a^6*b^8 + 256*a^7*b^7 - 96*a^8*b^6 - 64*a^9*b^5 + 32*a^10*b^4 + 1408*a^2*c^12 + 7040*a^3*c^11 + 21120*a^4*c^10 + 42240*a^5*c^9 + 59136*a^6*c^8 + 59136*a^7*c^7 + 42240*a^8*c^6 + 21120*a^9*c^5 + 7040*a^10*c^4 + 1408*a^11*c^3 + 128*a^12*c^2 - 32*b^2*c^12 + 96*b^3*c^11 + 64*b^4*c^10 - 416*b^5*c^9 + 96*b^6*c^8 + 704*b^7*c^7 - 384*b^8*c^6 - 576*b^9*c^5 + 416*b^10*c^4 + 224*b^11*c^3 - 192*b^12*c^2 - tan(x/2)*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8*c)))^(1/2)*(64*a*b^14 - 256*a*c^14 + 256*a^14*c - 64*b^14*c - 128*a^2*b^13 - 256*a^3*b^12 + 640*a^4*b^11 + 320*a^5*b^10 - 1280*a^6*b^9 + 1280*a^8*b^7 - 320*a^9*b^6 - 640*a^10*b^5 + 256*a^11*b^4 + 128*a^12*b^3 - 64*a^13*b^2 - 2816*a^2*c^13 - 13824*a^3*c^12 - 39424*a^4*c^11 - 70400*a^5*c^10 - 76032*a^6*c^9 - 33792*a^7*c^8 + 33792*a^8*c^7 + 76032*a^9*c^6 + 70400*a^10*c^5 + 39424*a^11*c^4 + 13824*a^12*c^3 + 2816*a^13*c^2 + 64*b^2*c^13 - 128*b^3*c^12 - 256*b^4*c^11 + 640*b^5*c^10 + 320*b^6*c^9 - 1280*b^7*c^8 + 1280*b^9*c^6 - 320*b^10*c^5 - 640*b^11*c^4 + 256*b^12*c^3 + 128*b^13*c^2 + 1728*a*b^2*c^12 - 3840*a*b^3*c^11 - 3584*a*b^4*c^10 + 10240*a*b^5*c^9 + 2240*a*b^6*c^8 - 12800*a*b^7*c^7 + 1280*a*b^8*c^6 + 7680*a*b^9*c^5 - 1984*a*b^10*c^4 - 1792*a*b^11*c^3 + 512*a*b^12*c^2 + 5120*a^2*b*c^12 - 512*a^2*b^12*c + 22528*a^3*b*c^11 + 1792*a^3*b^11*c + 56320*a^4*b*c^10 + 1984*a^4*b^10*c + 84480*a^5*b*c^9 - 7680*a^5*b^9*c + 67584*a^6*b*c^8 - 1280*a^6*b^8*c + 12800*a^7*b^7*c - 67584*a^8*b*c^6 - 2240*a^8*b^6*c - 84480*a^9*b*c^5 - 10240*a^9*b^5*c - 56320*a^10*b*c^4 + 3584*a^10*b^4*c - 22528*a^11*b*c^3 + 3840*a^11*b^3*c - 512

0*a^12*b*c^2 - 1728*a^12*b^2*c + 12672*a^2*b^2*c^11 - 26112*a^2*b^3*c^10 -
17920*a^2*b^4*c^9 + 48000*a^2*b^5*c^8 + 6400*a^2*b^6*c^7 - 38400*a^2*b^7*c^
6 + 3840*a^2*b^8*c^5 + 11520*a^2*b^9*c^4 - 1664*a^2*b^10*c^3 + 45696*a^3*b^
2*c^10 - 83200*a^3*b^3*c^9 - 44800*a^3*b^4*c^8 + 102400*a^3*b^5*c^7 + 8960*
a^3*b^6*c^6 - 43520*a^3*b^7*c^5 + 2560*a^3*b^8*c^4 + 1664*a^3*b^10*c^2 + 94
400*a^4*b^2*c^9 - 144000*a^4*b^3*c^8 - 58880*a^4*b^4*c^7 + 98560*a^4*b^5*c^
6 + 4480*a^4*b^6*c^5 - 2560*a^4*b^8*c^3 - 11520*a^4*b^9*c^2 + 111168*a^5*b^
2*c^8 - 124416*a^5*b^3*c^7 - 28672*a^5*b^4*c^6 - 4480*a^5*b^6*c^4 + 43520*a
^5*b^7*c^3 - 3840*a^5*b^8*c^2 + 51456*a^6*b^2*c^7 + 28672*a^6*b^4*c^5 - 985
60*a^6*b^5*c^4 - 8960*a^6*b^6*c^3 + 38400*a^6*b^7*c^2 - 51456*a^7*b^2*c^6 +
124416*a^7*b^3*c^5 + 58880*a^7*b^4*c^4 - 102400*a^7*b^5*c^3 - 6400*a^7*b^6
*c^2 - 111168*a^8*b^2*c^5 + 144000*a^8*b^3*c^4 + 44800*a^8*b^4*c^3 - 48000*
a^8*b^5*c^2 - 94400*a^9*b^2*c^4 + 83200*a^9*b^3*c^3 + 17920*a^9*b^4*c^2 - 4
5696*a^10*b^2*c^3 + 26112*a^10*b^3*c^2 - 12672*a^11*b^2*c^2 + 512*a*b*c^13
- 512*a^13*b*c) - 608*a*b^2*c^11 + 2624*a*b^3*c^10 + 224*a*b^4*c^9 - 6208*a
*b^5*c^8 + 2112*a*b^6*c^7 + 6784*a*b^7*c^6 - 3520*a*b^8*c^5 - 3584*a*b^9*c^
4 + 2080*a*b^10*c^3 + 832*a*b^11*c^2 - 3840*a^2*b*c^11 + 992*a^2*b^11*c - 1
7280*a^3*b*c^10 + 992*a^3*b^10*c - 46080*a^4*b*c^9 - 3136*a^4*b^9*c - 80640
*a^5*b*c^8 - 320*a^5*b^8*c - 96768*a^6*b*c^7 + 3776*a^6*b^7*c - 80640*a^7*b
*c^6 - 832*a^7*b^6*c - 46080*a^8*b*c^5 - 1952*a^8*b^5*c - 17280*a^9*b*c^4 +
736*a^9*b^4*c - 3840*a^10*b*c^3 + 352*a^10*b^3*c - 384*a^11*b*c^2 - 160*a^
11*b^2*c - 4192*a^2*b^2*c^10 + 17888*a^2*b^3*c^9 + 288*a^2*b^4*c^8 - 30080*
a^2*b^5*c^7 + 8768*a^2*b^6*c^6 + 22848*a^2*b^7*c^5 - 8768*a^2*b^8*c^4 - 780
8*a^2*b^9*c^3 + 2592*a^2*b^10*c^2 - 15648*a^3*b^2*c^9 + 60160*a^3*b^3*c^8 +
1152*a^3*b^4*c^7 - 73472*a^3*b^5*c^6 + 15424*a^3*b^6*c^5 + 37888*a^3*b^7*c
^4 - 8960*a^3*b^8*c^3 - 7552*a^3*b^9*c^2 - 36672*a^4*b^2*c^8 + 120512*a^4*b
^3*c^7 + 5376*a^4*b^4*c^6 - 104384*a^4*b^5*c^5 + 12800*a^4*b^6*c^4 + 34112*
a^4*b^7*c^3 - 3712*a^4*b^8*c^2 - 57792*a^5*b^2*c^7 + 155008*a^5*b^3*c^6 + 1
2096*a^5*b^4*c^5 - 90496*a^5*b^5*c^4 + 3776*a^5*b^6*c^3 + 16512*a^5*b^7*c^2
- 63168*a^6*b^2*c^6 + 131264*a^6*b^3*c^5 + 14784*a^6*b^4*c^4 - 47488*a^6*b
^5*c^3 - 1088*a^6*b^6*c^2 - 48192*a^7*b^2*c^5 + 72448*a^7*b^3*c^4 + 10368*a
^7*b^4*c^3 - 14080*a^7*b^5*c^2 - 25248*a^8*b^2*c^4 + 24800*a^8*b^3*c^3 + 40
32*a^8*b^4*c^2 - 8672*a^9*b^2*c^3 + 4672*a^9*b^3*c^2 - 1760*a^10*b^2*c^2 -
384*a*b*c^12 - 416*a*b^12*c) - tan(x/2)*(32*a*b^12 - 512*a*c^12 + 128*b*c^1
2 + 96*b^12*c - 32*b^13 - 64*c^13 + 96*a^2*b^11 - 96*a^3*b^10 - 96*a^4*b^9
+ 96*a^5*b^8 + 32*a^6*b^7 - 32*a^7*b^6 - 1728*a^2*c^11 - 3072*a^3*c^10 - 26
88*a^4*c^9 + 2688*a^6*c^7 + 3072*a^7*c^6 + 1728*a^8*c^5 + 512*a^9*c^4 + 64*
a^10*c^3 + 160*b^2*c^11 - 544*b^3*c^10 + 64*b^4*c^9 + 896*b^5*c^8 - 608*b^6
*c^7 - 672*b^7*c^6 + 800*b^8*c^5 + 160*b^9*c^4 - 448*b^10*c^3 + 64*b^11*c^2
+ 480*a*b^2*c^10 - 4352*a*b^3*c^9 + 2560*a*b^4*c^8 + 5248*a*b^5*c^7 - 5664
*a*b^6*c^6 - 2240*a*b^7*c^5 + 4320*a*b^8*c^4 - 256*a*b^9*c^3 - 1216*a*b^10*
c^2 + 5632*a^2*b*c^10 - 672*a^2*b^10*c + 14336*a^3*b*c^9 - 768*a^3*b^9*c +
23296*a^4*b*c^8 + 1248*a^4*b^8*c + 25088*a^5*b*c^7 + 576*a^5*b^7*c + 17920*
a^6*b*c^6 - 864*a^6*b^6*c + 8192*a^7*b*c^5 - 128*a^7*b^5*c + 2176*a^8*b*c^4
+ 192*a^8*b^4*c + 256*a^9*b*c^3 - 1408*a^2*b^2*c^9 - 14720*a^2*b^3*c^8 + 1

3440*a^2*b^4*c^7 + 11904*a^2*b^5*c^6 - 16800*a^2*b^6*c^5 - 1696*a^2*b^7*c^4
 + 7168*a^2*b^8*c^3 - 1216*a^2*b^9*c^2 - 9856*a^3*b^2*c^8 - 27392*a^3*b^3*c
^7 + 31232*a^3*b^4*c^6 + 12928*a^3*b^5*c^5 - 23264*a^3*b^6*c^4 + 1152*a^3*b
^7*c^3 + 4800*a^3*b^8*c^2 - 22848*a^4*b^2*c^7 - 30400*a^4*b^3*c^6 + 39680*a
^4*b^4*c^5 + 6272*a^4*b^5*c^4 - 16544*a^4*b^6*c^3 + 1824*a^4*b^7*c^2 - 2912
0*a^5*b^2*c^6 - 20224*a^5*b^3*c^5 + 29184*a^5*b^4*c^4 + 384*a^5*b^5*c^3 - 5
856*a^5*b^6*c^2 - 22400*a^6*b^2*c^5 - 7552*a^6*b^3*c^4 + 12160*a^6*b^4*c^3
 - 640*a^6*b^5*c^2 - 10368*a^7*b^2*c^4 - 1280*a^7*b^3*c^3 + 2560*a^7*b^4*c^2
 - 2656*a^8*b^2*c^3 - 32*a^8*b^3*c^2 - 288*a^9*b^2*c^2 + 1280*a*b*c^11 + 32
0*a*b^11*c))*(-(8*a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(
-(4*a*c - b^2)^3)^(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5
 + 24*a*b^4*c^3 - 3*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2
*b^4*c^2 - 38*a^3*b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c
 - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2
) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 +
a^6*b^4 + 16*a^2*c^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4
 + 96*a^7*c^3 + 16*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7
 + 30*a*b^4*c^5 - 36*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c -
 96*a^2*b^2*c^6 + 159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*
a^3*b^4*c^3 - 448*a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*
b^2*c^2 + 14*a*b^8*c)))^(1/2) + 1792*a^2*c^10 + 3584*a^3*c^9 + 4480*a^4*c^8
 + 3584*a^5*c^7 + 1792*a^6*c^6 + 512*a^7*c^5 + 64*a^8*c^4 - 320*b^2*c^10 +
64*b^3*c^9 + 576*b^4*c^8 - 192*b^5*c^7 - 448*b^6*c^6 + 192*b^7*c^5 + 128*b^
8*c^4 - 64*b^9*c^3 - 1984*a*b^2*c^9 + 384*a*b^3*c^8 + 2496*a*b^4*c^7 - 768*
a*b^5*c^6 - 1088*a*b^6*c^5 + 384*a*b^7*c^4 + 64*a*b^8*c^3 - 5184*a^2*b^2*c^
8 + 960*a^2*b^3*c^7 + 4224*a^2*b^4*c^6 - 1152*a^2*b^5*c^5 - 832*a^2*b^6*c^4
 + 192*a^2*b^7*c^3 - 7360*a^3*b^2*c^7 + 1280*a^3*b^3*c^6 + 3456*a^3*b^4*c^5
 - 768*a^3*b^5*c^4 - 192*a^3*b^6*c^3 - 6080*a^4*b^2*c^6 + 960*a^4*b^3*c^5 +
 1344*a^4*b^4*c^4 - 192*a^4*b^5*c^3 - 2880*a^5*b^2*c^5 + 384*a^5*b^3*c^4 +
192*a^5*b^4*c^3 - 704*a^6*b^2*c^4 + 64*a^6*b^3*c^3 - 64*a^7*b^2*c^3))*(-(8*
a*c^7 + b^8 + 24*a^2*c^6 + 24*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^
(1/2) - 2*b^2*c^6 + 3*b^4*c^4 - 3*b^6*c^2 - 18*a*b^2*c^5 + 24*a*b^4*c^3 - 3
*b*c^4*(-(4*a*c - b^2)^3)^(1/2) - 54*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*
b^2*c^3 + 3*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(
4*a*c - b^2)^3)^(1/2) - 6*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4
*a*c - b^2)^3)^(1/2))/(2*(3*a^2*b^8 - b^10 - 3*a^4*b^6 + a^6*b^4 + 16*a^2*c
^8 + 96*a^3*c^7 + 240*a^4*c^6 + 320*a^5*c^5 + 240*a^6*c^4 + 96*a^7*c^3 + 16
*a^8*c^2 + b^4*c^6 - 3*b^6*c^4 + 3*b^8*c^2 - 8*a*b^2*c^7 + 30*a*b^4*c^5 - 3
6*a*b^6*c^3 - 36*a^3*b^6*c + 30*a^5*b^4*c - 8*a^7*b^2*c - 96*a^2*b^2*c^6 +
159*a^2*b^4*c^4 - 82*a^2*b^6*c^2 - 312*a^3*b^2*c^5 + 260*a^3*b^4*c^3 - 448*
a^4*b^2*c^4 + 159*a^4*b^4*c^2 - 312*a^5*b^2*c^3 - 96*a^6*b^2*c^2 + 14*a*b^8
*c)))^(1/2)*2i + tan(x/2)/(2*a - 2*b + 2*c) - (a - b + c)/(tan(x/2)*(a + b
+ c)*(2*a - 2*b + 2*c))

3.9 $\int \frac{\sin(x)}{-2+\cos(x)+\cos^2(x)}\,dx$

Optimal result

Integrand size = 13, antiderivative size = 21

$$\int \frac{\sin(x)}{-2+\cos(x)+\cos^2(x)}\,dx = -\frac{1}{3}\log(1-\cos(x))+\frac{1}{3}\log(2+\cos(x))$$

```
[Out] -1/3*ln(1-cos(x))+1/3*ln(2+cos(x))
```

Rubi [A] (verified)

Time = 0.03 (sec) , antiderivative size = 21, normalized size of antiderivative = 1.00, number of steps used = 4, number of rules used = 3, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.231, Rules used = {3340, 630, 31}

$$\int \frac{\sin(x)}{-2+\cos(x)+\cos^2(x)}\,dx = \frac{1}{3}\log(\cos(x)+2)-\frac{1}{3}\log(1-\cos(x))$$

```
[In] Int[Sin[x]/(-2 + Cos[x] + Cos[x]^2),x]
```

```
[Out] -1/3*Log[1 - Cos[x]] + Log[2 + Cos[x]]/3
```

Rule 31

```
Int[((a_) + (b_.)*(x_))^(-1), x_Symbol] :> Simp[Log[RemoveContent[a + b*x,
x]]/b, x] /; FreeQ[{a, b}, x]
```

Rule 630

```
Int[((a_.) + (b_.)*(x_) + (c_.)*(x_)^2)^(-1), x_Symbol] :> With[{q = Rt[b^2
 - 4*a*c, 2]}, Dist[c/q, Int[1/Simp[b/2 - q/2 + c*x, x], x], x] - Dist[c/q,
 Int[1/Simp[b/2 + q/2 + c*x, x], x], x]] /; FreeQ[{a, b, c}, x] && NeQ[b^2
```

```
- 4*a*c, 0] && PosQ[b^2 - 4*a*c] && PerfectSquareQ[b^2 - 4*a*c]
```

Rule 3340

```
Int[((a_.) + (b_.)*(cos[(d_.) + (e_.)*(x_)]*(f_.))^(n_.) + (c_.)*(cos[(d_.)
 + (e_.)*(x_)]*(f_.))^(n2_.))^(p_.)*sin[(d_.) + (e_.)*(x_)]^(m_.), x_Symbol
] :> Module[{g = FreeFactors[Cos[d + e*x], x]}, Dist[-g/e, Subst[Int[(1 - g
^2*x^2)^((m - 1)/2)*(a + b*(f*g*x)^n + c*(f*g*x)^(2*n))^p, x], x, Cos[d + e
*x]/g], x]] /; FreeQ[{a, b, c, d, e, f, n, p}, x] && EqQ[n2, 2*n] && Intege
rQ[(m - 1)/2]
```

Rubi steps

$$
\begin{aligned}
\text{integral} &= -\text{Subst}\left(\int \frac{1}{-2+x+x^2}\,dx, x, \cos(x)\right)\\
&= -\left(\frac{1}{3}\text{Subst}\left(\int \frac{1}{-1+x}\,dx, x, \cos(x)\right)\right)+\frac{1}{3}\text{Subst}\left(\int \frac{1}{2+x}\,dx, x, \cos(x)\right)\\
&= -\frac{1}{3}\log(1-\cos(x))+\frac{1}{3}\log(2+\cos(x))
\end{aligned}
$$

Mathematica [A] (verified)

Time = 0.02 (sec) , antiderivative size = 19, normalized size of antiderivative = 0.90

$$
\int \frac{\sin(x)}{-2+\cos(x)+\cos^2(x)}\,dx = \frac{1}{3}\left(\log(2+\cos(x))-2\log\left(\sin\left(\frac{x}{2}\right)\right)\right)
$$

```
[In] Integrate[Sin[x]/(-2 + Cos[x] + Cos[x]^2),x]
[Out] (Log[2 + Cos[x]] - 2*Log[Sin[x/2]])/3
```

Maple [A] (verified)

Time = 0.20 (sec) , antiderivative size = 16, normalized size of antiderivative = 0.76

method	result	size
derivativedivides	$-\frac{\ln(\cos(x)-1)}{3}+\frac{\ln(\cos(x)+2)}{3}$	16
default	$-\frac{\ln(\cos(x)-1)}{3}+\frac{\ln(\cos(x)+2)}{3}$	16
norman	$-\frac{2\ln(\tan(\frac{x}{2}))}{3}+\frac{\ln(\tan^2(\frac{x}{2})+3)}{3}$	20
parallelrisch	$\ln\left(\frac{1}{\tan(\frac{x}{2})^{\frac{2}{3}}}\right)+\ln\left(\left(\tan^2\left(\frac{x}{2}\right)+3\right)^{\frac{1}{3}}\right)$	20
risch	$-\frac{2\ln(e^{ix}-1)}{3}+\frac{\ln(e^{2ix}+4e^{ix}+1)}{3}$	29

```
[In] int(sin(x)/(-2+cos(x)+cos(x)^2),x,method=_RETURNVERBOSE)
[Out] -1/3*ln(cos(x)-1)+1/3*ln(cos(x)+2)
```

Fricas [A] (verification not implemented)

none

Time = 0.25 (sec) , antiderivative size = 17, normalized size of antiderivative = 0.81

$$\int \frac{\sin(x)}{-2+\cos(x)+\cos^2(x)}\,dx = \frac{1}{3}\log\left(\cos\left(x\right)+2\right)-\frac{1}{3}\log\left(-\frac{1}{2}\cos\left(x\right)+\frac{1}{2}\right)$$

```
[In] integrate(sin(x)/(-2+cos(x)+cos(x)^2),x, algorithm="fricas")
[Out] 1/3*log(cos(x) + 2) - 1/3*log(-1/2*cos(x) + 1/2)
```

Sympy [A] (verification not implemented)

Time = 0.09 (sec) , antiderivative size = 15, normalized size of antiderivative = 0.71

$$\int \frac{\sin(x)}{-2+\cos(x)+\cos^2(x)}\,dx = -\frac{\log\left(\cos\left(x\right)-1\right)}{3}+\frac{\log\left(\cos\left(x\right)+2\right)}{3}$$

```
[In] integrate(sin(x)/(-2+cos(x)+cos(x)**2),x)
[Out] -log(cos(x) - 1)/3 + log(cos(x) + 2)/3
```

Maxima [A] (verification not implemented)

none

Time = 0.22 (sec) , antiderivative size = 15, normalized size of antiderivative = 0.71

$$\int \frac{\sin(x)}{-2+\cos(x)+\cos^2(x)}\,dx = \frac{1}{3}\,\log\left(\cos\left(x\right)+2\right)-\frac{1}{3}\,\log\left(\cos\left(x\right)-1\right)$$

```
[In] integrate(sin(x)/(-2+cos(x)+cos(x)^2),x, algorithm="maxima")
[Out] 1/3*log(cos(x) + 2) - 1/3*log(cos(x) - 1)
```

Giac [A] (verification not implemented)

none

Time = 0.30 (sec) , antiderivative size = 17, normalized size of antiderivative = 0.81

$$\int \frac{\sin(x)}{-2+\cos(x)+\cos^2(x)}\,dx = \frac{1}{3}\,\log\left(\cos\left(x\right)+2\right)-\frac{1}{3}\,\log\left(-\cos\left(x\right)+1\right)$$

```
[In] integrate(sin(x)/(-2+cos(x)+cos(x)^2),x, algorithm="giac")
[Out] 1/3*log(cos(x) + 2) - 1/3*log(-cos(x) + 1)
```

Mupad [B] (verification not implemented)

Time = 0.20 (sec) , antiderivative size = 9, normalized size of antiderivative = 0.43

$$\int \frac{\sin(x)}{-2+\cos(x)+\cos^2(x)}\,dx = \frac{2\,\mathrm{atanh}\left(\frac{2\,\cos(x)}{3}+\frac{1}{3}\right)}{3}$$

```
[In] int(sin(x)/(cos(x) + cos(x)^2 - 2),x)
[Out] (2*atanh((2*cos(x))/3 + 1/3))/3
```

3.10 $\int \frac{\sin(x)}{4-5\cos(x)+\cos^2(x)}\,dx$

Optimal result

Integrand size = 15, antiderivative size = 23

$$\int \frac{\sin(x)}{4-5\cos(x)+\cos^2(x)}\,dx = \frac{1}{3}\log(1-\cos(x)) - \frac{1}{3}\log(4-\cos(x))$$

```
[Out] 1/3*ln(1-cos(x))-1/3*ln(4-cos(x))
```

Rubi [A] (verified)

Time = 0.03 (sec) , antiderivative size = 23, normalized size of antiderivative = 1.00, number of steps used = 4, number of rules used = 3, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.200, Rules used = {3340, 630, 31}

$$\int \frac{\sin(x)}{4-5\cos(x)+\cos^2(x)}\,dx = \frac{1}{3}\log(1-\cos(x)) - \frac{1}{3}\log(4-\cos(x))$$

```
[In] Int[Sin[x]/(4 - 5*Cos[x] + Cos[x]^2),x]
```

```
[Out] Log[1 - Cos[x]]/3 - Log[4 - Cos[x]]/3
```

Rule 31

```
Int[((a_) + (b_.)*(x_))^(-1), x_Symbol] :> Simp[Log[RemoveContent[a + b*x,
x]]/b, x] /; FreeQ[{a, b}, x]
```

Rule 630

```
Int[((a_.) + (b_.)*(x_) + (c_.)*(x_)^2)^(-1), x_Symbol] :> With[{q = Rt[b^2
 - 4*a*c, 2]}, Dist[c/q, Int[1/Simp[b/2 - q/2 + c*x, x], x], x] - Dist[c/q,
 Int[1/Simp[b/2 + q/2 + c*x, x], x], x]] /; FreeQ[{a, b, c}, x] && NeQ[b^2
```

```
- 4*a*c, 0] && PosQ[b^2 - 4*a*c] && PerfectSquareQ[b^2 - 4*a*c]
```

Rule 3340

```
Int[((a_.) + (b_.)*(cos[(d_.) + (e_.)*(x_)]*(f_.))^(n_.) + (c_.)*(cos[(d_.)
 + (e_.)*(x_)]*(f_.))^(n2_.))^(p_.)*sin[(d_.) + (e_.)*(x_)]^(m_.), x_Symbol
] :> Module[{g = FreeFactors[Cos[d + e*x], x]}, Dist[-g/e, Subst[Int[(1 - g
^2*x^2)^((m - 1)/2)*(a + b*(f*g*x)^n + c*(f*g*x)^(2*n))^p, x], x, Cos[d + e
*x]/g], x]] /; FreeQ[{a, b, c, d, e, f, n, p}, x] && EqQ[n2, 2*n] && Intege
rQ[(m - 1)/2]
```

Rubi steps

$$\begin{aligned}\text{integral} &= -\text{Subst}\left(\int \frac{1}{4-5x+x^2}\,dx, x, \cos(x)\right)\\ &= -\left(\frac{1}{3}\text{Subst}\left(\int \frac{1}{-4+x}\,dx, x, \cos(x)\right)\right)+\frac{1}{3}\text{Subst}\left(\int \frac{1}{-1+x}\,dx, x, \cos(x)\right)\\ &= \frac{1}{3}\log(1-\cos(x))-\frac{1}{3}\log(4-\cos(x))\end{aligned}$$

Mathematica [A] (verified)

Time = 0.01 (sec) , antiderivative size = 29, normalized size of antiderivative = 1.26

$$\int \frac{\sin(x)}{4-5\cos(x)+\cos^2(x)}\,dx = \frac{2}{3}\log\left(\sin\left(\frac{x}{2}\right)\right)-\frac{1}{3}\log\left(3+2\sin^2\left(\frac{x}{2}\right)\right)$$

```
[In] Integrate[Sin[x]/(4 - 5*Cos[x] + Cos[x]^2),x]
[Out] (2*Log[Sin[x/2]])/3 - Log[3 + 2*Sin[x/2]^2]/3
```

Maple [A] (verified)

Time = 0.20 (sec) , antiderivative size = 16, normalized size of antiderivative = 0.70

method	result	size
derivativedivides	$-\frac{\ln(\cos(x)-4)}{3}+\frac{\ln(\cos(x)-1)}{3}$	16
default	$-\frac{\ln(\cos(x)-4)}{3}+\frac{\ln(\cos(x)-1)}{3}$	16
norman	$\frac{2\ln\left(\tan\left(\frac{x}{2}\right)\right)}{3}-\frac{\ln\left(5\left(\tan^2\left(\frac{x}{2}\right)\right)+3\right)}{3}$	22
parallelrisch	$\ln\left(5^{\frac{1}{3}}\left(\tan^2\left(\frac{x}{2}\right)\right)^{\frac{1}{3}}\right)+\ln\left(\frac{1}{\left(5\left(\tan^2\left(\frac{x}{2}\right)\right)+3\right)^{\frac{1}{3}}}\right)$	28
risch	$\frac{2\ln(e^{ix}-1)}{3}-\frac{\ln(e^{2ix}-8\,e^{ix}+1)}{3}$	29

```
[In] int(sin(x)/(4-5*cos(x)+cos(x)^2),x,method=_RETURNVERBOSE)
[Out] -1/3*ln(cos(x)-4)+1/3*ln(cos(x)-1)
```

Fricas [A] (verification not implemented)

none

Time = 0.24 (sec) , antiderivative size = 19, normalized size of antiderivative = 0.83

$$\int \frac{\sin(x)}{4-5\cos(x)+\cos^2(x)}\,dx = \frac{1}{3}\log\left(-\frac{1}{2}\cos(x)+\frac{1}{2}\right)-\frac{1}{3}\log\left(-\cos(x)+4\right)$$

```
[In] integrate(sin(x)/(4-5*cos(x)+cos(x)^2),x, algorithm="fricas")
[Out] 1/3*log(-1/2*cos(x) + 1/2) - 1/3*log(-cos(x) + 4)
```

Sympy [A] (verification not implemented)

Time = 0.10 (sec) , antiderivative size = 15, normalized size of antiderivative = 0.65

$$\int \frac{\sin(x)}{4-5\cos(x)+\cos^2(x)}\,dx = -\frac{\log(\cos(x)-4)}{3}+\frac{\log(\cos(x)-1)}{3}$$

```
[In] integrate(sin(x)/(4-5*cos(x)+cos(x)**2),x)
[Out] -log(cos(x) - 4)/3 + log(cos(x) - 1)/3
```

Maxima [A] (verification not implemented)

none

Time = 0.30 (sec) , antiderivative size = 15, normalized size of antiderivative = 0.65

$$\int \frac{\sin(x)}{4-5\cos(x)+\cos^2(x)}\,dx = \frac{1}{3}\,\log\left(\cos\left(x\right)-1\right)-\frac{1}{3}\,\log\left(\cos\left(x\right)-4\right)$$

```
[In] integrate(sin(x)/(4-5*cos(x)+cos(x)^2),x, algorithm="maxima")
[Out] 1/3*log(cos(x) - 1) - 1/3*log(cos(x) - 4)
```

Giac [A] (verification not implemented)

none

Time = 0.29 (sec) , antiderivative size = 19, normalized size of antiderivative = 0.83

$$\int \frac{\sin(x)}{4-5\cos(x)+\cos^2(x)}\,dx = -\frac{1}{3}\,\log\left(-\cos\left(x\right)+4\right)+\frac{1}{3}\,\log\left(-\cos\left(x\right)+1\right)$$

```
[In] integrate(sin(x)/(4-5*cos(x)+cos(x)^2),x, algorithm="giac")
[Out] -1/3*log(-cos(x) + 4) + 1/3*log(-cos(x) + 1)
```

Mupad [B] (verification not implemented)

Time = 0.14 (sec) , antiderivative size = 9, normalized size of antiderivative = 0.39

$$\int \frac{\sin(x)}{4-5\cos(x)+\cos^2(x)}\,dx = \frac{2\,\mathrm{atanh}\left(\frac{2\,\cos(x)}{3}-\frac{5}{3}\right)}{3}$$

```
[In] int(sin(x)/(cos(x)^2 - 5*cos(x) + 4),x)
[Out] (2*atanh((2*cos(x))/3 - 5/3))/3
```

3.11 $\int \frac{\sin(x)}{3-2\cos(x)+\cos^2(x)}\,dx$

Optimal result

Integrand size = 15, antiderivative size = 19

$$\int \frac{\sin(x)}{3-2\cos(x)+\cos^2(x)}\,dx = \frac{\arctan\left(\frac{1-\cos(x)}{\sqrt{2}}\right)}{\sqrt{2}}$$

```
[Out] 1/2*arctan(1/2*(1-cos(x))*2^(1/2))*2^(1/2)
```

Rubi [A] (verified)

Time = 0.04 (sec) , antiderivative size = 19, normalized size of antiderivative = 1.00, number of steps used = 3, number of rules used = 3, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.200, Rules used = {3340, 632, 210}

$$\int \frac{\sin(x)}{3-2\cos(x)+\cos^2(x)}\,dx = \frac{\arctan\left(\frac{1-\cos(x)}{\sqrt{2}}\right)}{\sqrt{2}}$$

```
[In] Int[Sin[x]/(3 - 2*Cos[x] + Cos[x]^2),x]
```

```
[Out] ArcTan[(1 - Cos[x])/Sqrt[2]]/Sqrt[2]
```

Rule 210

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(-(Rt[-a, 2]*Rt[-b, 2])^(
-1))*ArcTan[Rt[-b, 2]*(x/Rt[-a, 2])], x] /; FreeQ[{a, b}, x] && PosQ[a/b] &
& (LtQ[a, 0] || LtQ[b, 0])
```

Rule 632

```
Int[((a_.) + (b_.)*(x_) + (c_.)*(x_)^2)^(-1), x_Symbol] :> Dist[-2, Subst[I
nt[1/Simp[b^2 - 4*a*c - x^2, x], x], x, b + 2*c*x], x] /; FreeQ[{a, b, c},
```

```
x] && NeQ[b^2 - 4*a*c, 0]
```

Rule 3340

```
Int[((a_.) + (b_.)*(cos[(d_.) + (e_.)*(x_)]*(f_.))^(n_.) + (c_.)*(cos[(d_.)
 + (e_.)*(x_)]*(f_.))^(n2_.))^(p_.)*sin[(d_.) + (e_.)*(x_)]^(m_.), x_Symbol
] :> Module[{g = FreeFactors[Cos[d + e*x], x]}, Dist[-g/e, Subst[Int[(1 - g
^2*x^2)^((m - 1)/2)*(a + b*(f*g*x)^n + c*(f*g*x)^(2*n))^p, x], x, Cos[d + e
*x]/g], x]] /; FreeQ[{a, b, c, d, e, f, n, p}, x] && EqQ[n2, 2*n] && Intege
rQ[(m - 1)/2]
```

Rubi steps

$$\begin{aligned}\text{integral} &= -\text{Subst}\left(\int \frac{1}{3-2x+x^2}\,dx, x, \cos(x)\right)\\ &= 2\text{Subst}\left(\int \frac{1}{-8-x^2}\,dx, x, -2+2\cos(x)\right)\\ &= \frac{\arctan\left(\frac{1-\cos(x)}{\sqrt{2}}\right)}{\sqrt{2}}\end{aligned}$$

Mathematica [A] (verified)

Time = 0.05 (sec) , antiderivative size = 18, normalized size of antiderivative = 0.95

$$\int \frac{\sin(x)}{3-2\cos(x)+\cos^2(x)}\,dx = -\frac{\arctan\left(\frac{-1+\cos(x)}{\sqrt{2}}\right)}{\sqrt{2}}$$

```
[In] Integrate[Sin[x]/(3 - 2*Cos[x] + Cos[x]^2),x]
[Out] -(ArcTan[(-1 + Cos[x])/Sqrt[2]]/Sqrt[2])
```

Maple [A] (verified)

Time = 0.16 (sec) , antiderivative size = 18, normalized size of antiderivative = 0.95

method	result	size
derivativedivides	$-\frac{\sqrt{2}\,\arctan\left(\frac{(2\cos(x)-2)\sqrt{2}}{4}\right)}{2}$	18
default	$-\frac{\sqrt{2}\,\arctan\left(\frac{(2\cos(x)-2)\sqrt{2}}{4}\right)}{2}$	18
risch	$-\frac{i\sqrt{2}\,\ln\left(e^{2ix}+\left(2i\sqrt{2}-2\right)e^{ix}+1\right)}{4}+\frac{i\sqrt{2}\,\ln\left(e^{2ix}+\left(-2i\sqrt{2}-2\right)e^{ix}+1\right)}{4}$	58

```
[In] int(sin(x)/(3-2*cos(x)+cos(x)^2),x,method=_RETURNVERBOSE)
[Out] -1/2*2^(1/2)*arctan(1/4*(2*cos(x)-2)*2^(1/2))
```

Fricas [A] (verification not implemented)

none

Time = 0.24 (sec) , antiderivative size = 19, normalized size of antiderivative = 1.00

$$\int \frac{\sin(x)}{3-2\cos(x)+\cos^2(x)}\,dx = -\frac{1}{2}\,\sqrt{2}\arctan\left(\frac{1}{2}\,\sqrt{2}\cos\left(x\right) - \frac{1}{2}\,\sqrt{2}\right)$$

```
[In] integrate(sin(x)/(3-2*cos(x)+cos(x)^2),x, algorithm="fricas")
[Out] -1/2*sqrt(2)*arctan(1/2*sqrt(2)*cos(x) - 1/2*sqrt(2))
```

Sympy [A] (verification not implemented)

Time = 0.15 (sec) , antiderivative size = 26, normalized size of antiderivative = 1.37

$$\int \frac{\sin(x)}{3-2\cos(x)+\cos^2(x)}\,dx = -\frac{\sqrt{2}\,\mathrm{atan}\left(\frac{\sqrt{2}\cos\left(x\right)}{2} - \frac{\sqrt{2}}{2}\right)}{2}$$

```
[In] integrate(sin(x)/(3-2*cos(x)+cos(x)**2),x)
[Out] -sqrt(2)*atan(sqrt(2)*cos(x)/2 - sqrt(2)/2)/2
```

Maxima [A] (verification not implemented)

none

Time = 0.38 (sec) , antiderivative size = 15, normalized size of antiderivative = 0.79

$$\int \frac{\sin(x)}{3-2\cos(x)+\cos^2(x)}\,dx = -\frac{1}{2}\,\sqrt{2}\arctan\left(\frac{1}{2}\,\sqrt{2}(\cos\left(x\right)-1)\right)$$

```
[In] integrate(sin(x)/(3-2*cos(x)+cos(x)^2),x, algorithm="maxima")
[Out] -1/2*sqrt(2)*arctan(1/2*sqrt(2)*(cos(x) - 1))
```

Giac [A] (verification not implemented)

none

Time = 0.29 (sec) , antiderivative size = 15, normalized size of antiderivative = 0.79

$$\int \frac{\sin(x)}{3-2\cos(x)+\cos^2(x)}\,dx = -\frac{1}{2}\,\sqrt{2}\arctan\left(\frac{1}{2}\,\sqrt{2}(\cos{(x)}-1)\right)$$

```
[In] integrate(sin(x)/(3-2*cos(x)+cos(x)^2),x, algorithm="giac")
[Out] -1/2*sqrt(2)*arctan(1/2*sqrt(2)*(cos(x) - 1))
```

Mupad [B] (verification not implemented)

Time = 0.07 (sec) , antiderivative size = 15, normalized size of antiderivative = 0.79

$$\int \frac{\sin(x)}{3-2\cos(x)+\cos^2(x)}\,dx = -\frac{\sqrt{2}\,\mathrm{atan}\left(\frac{\sqrt{2}\,(\cos(x)-1)}{2}\right)}{2}$$

```
[In] int(sin(x)/(cos(x)^2 - 2*cos(x) + 3),x)
[Out] -(2^(1/2)*atan((2^(1/2)*(cos(x) - 1))/2))/2
```

3.12 $\int \frac{\sin(x)}{(13-4\cos(x)+\cos^2(x))^2}\,dx$

Optimal result

Integrand size = 15, antiderivative size = 36

$$\int \frac{\sin(x)}{(13-4\cos(x)+\cos^2(x))^2}\,dx = -\frac{1}{54}\arctan\left(\frac{1}{3}(-2+\cos(x))\right) + \frac{2-\cos(x)}{18\,(13-4\cos(x)+\cos^2(x))}$$

```
[Out] -1/54*arctan(-2/3+1/3*cos(x))+1/18*(2-cos(x))/(13-4*cos(x)+cos(x)^2)
```

Rubi [A] (verified)

Time = 0.04 (sec) , antiderivative size = 36, normalized size of antiderivative = 1.00, number of steps used = 4, number of rules used = 4, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.267, Rules used = {3340, 628, 632, 210}

$$\int \frac{\sin(x)}{(13-4\cos(x)+\cos^2(x))^2}\,dx = \frac{2-\cos(x)}{18\,(\cos^2(x)-4\cos(x)+13)} - \frac{1}{54}\arctan\left(\frac{1}{3}(\cos(x)-2)\right)$$

```
[In] Int[Sin[x]/(13 - 4*Cos[x] + Cos[x]^2)^2,x]
```

```
[Out] -1/54*ArcTan[(-2 + Cos[x])/3] + (2 - Cos[x])/(18*(13 - 4*Cos[x] + Cos[x]^2)
)
```

Rule 210

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(-(Rt[-a, 2]*Rt[-b, 2])^(
-1))*ArcTan[Rt[-b, 2]*(x/Rt[-a, 2])], x] /; FreeQ[{a, b}, x] && PosQ[a/b] &
& (LtQ[a, 0] || LtQ[b, 0])
```

Rule 628

```
Int[((a_.) + (b_.)*(x_) + (c_.)*(x_)^2)^(p_), x_Symbol] :> Simp[(b + 2*c*x)
*((a + b*x + c*x^2)^(p + 1)/((p + 1)*(b^2 - 4*a*c))), x] - Dist[2*c*((2*p +
 3)/((p + 1)*(b^2 - 4*a*c))), Int[(a + b*x + c*x^2)^(p + 1), x], x] /; Free
Q[{a, b, c}, x] && NeQ[b^2 - 4*a*c, 0] && LtQ[p, -1] && NeQ[p, -3/2] && Int
egerQ[4*p]
```

Rule 632

```
Int[((a_.) + (b_.)*(x_) + (c_.)*(x_)^2)^(-1), x_Symbol] :> Dist[-2, Subst[I
nt[1/Simp[b^2 - 4*a*c - x^2, x], x], x, b + 2*c*x], x] /; FreeQ[{a, b, c},
x] && NeQ[b^2 - 4*a*c, 0]
```

Rule 3340

```
Int[((a_.) + (b_.)*(cos[(d_.) + (e_.)*(x_)]*(f_.))^(n_.) + (c_.)*(cos[(d_.)
 + (e_.)*(x_)]*(f_.))^(n2_.))^(p_.)*sin[(d_.) + (e_.)*(x_)]^(m_.), x_Symbol
] :> Module[{g = FreeFactors[Cos[d + e*x], x]}, Dist[-g/e, Subst[Int[(1 - g
^2*x^2)^((m - 1)/2)*(a + b*(f*g*x)^n + c*(f*g*x)^(2*n))^p, x], x, Cos[d + e
*x]/g], x]] /; FreeQ[{a, b, c, d, e, f, n, p}, x] && EqQ[n2, 2*n] && Intege
rQ[(m - 1)/2]
```

Rubi steps

$$\begin{aligned}
\text{integral} &= -\text{Subst}\left(\int \frac{1}{(13-4x+x^2)^2}\,dx, x, \cos(x)\right) \\
&= \frac{2-\cos(x)}{18\,(13-4\cos(x)+\cos^2(x))} - \frac{1}{18}\text{Subst}\left(\int \frac{1}{13-4x+x^2}\,dx, x, \cos(x)\right) \\
&= \frac{2-\cos(x)}{18\,(13-4\cos(x)+\cos^2(x))} + \frac{1}{9}\text{Subst}\left(\int \frac{1}{-36-x^2}\,dx, x, -4+2\cos(x)\right) \\
&= -\frac{1}{54}\arctan\left(\frac{1}{3}(-2+\cos(x))\right) + \frac{2-\cos(x)}{18\,(13-4\cos(x)+\cos^2(x))}
\end{aligned}$$

Mathematica [A] (verified)

Time = 0.09 (sec) , antiderivative size = 34, normalized size of antiderivative = 0.94

$$\int \frac{\sin(x)}{(13-4\cos(x)+\cos^2(x))^2}\,dx = -\frac{1}{54}\arctan\left(\frac{1}{3}(-2+\cos(x))\right) - \frac{-2+\cos(x)}{18\,(13-4\cos(x)+\cos^2(x))}$$

```
[In] Integrate[Sin[x]/(13 - 4*Cos[x] + Cos[x]^2)^2,x]
[Out] -1/54*ArcTan[(-2 + Cos[x])/3] - (-2 + Cos[x])/(18*(13 - 4*Cos[x] + Cos[x]^2
))
```

Maple [A] (verified)

Time = 0.76 (sec) , antiderivative size = 31, normalized size of antiderivative = 0.86

method	result
derivativedivides	$-\frac{2\cos(x)-4}{36(13-4\cos(x)+\cos^2(x))}-\frac{\arctan\left(-\frac{2}{3}+\frac{\cos(x)}{3}\right)}{54}$
default	$-\frac{2\cos(x)-4}{36(13-4\cos(x)+\cos^2(x))}-\frac{\arctan\left(-\frac{2}{3}+\frac{\cos(x)}{3}\right)}{54}$
risch	$-\frac{e^{3ix}-4\,e^{2ix}+e^{ix}}{9(e^{4ix}-8\,e^{3ix}+54\,e^{2ix}-8\,e^{ix}+1)}+\frac{i\ln(e^{2ix}+(-4-6i)e^{ix}+1)}{108}-\frac{i\ln(e^{2ix}+(-4+6i)e^{ix}+1)}{108}$
parallelrisch	$\frac{-2+i\left(-9\left(\tan^4\left(\frac{x}{2}\right)\right)-12\left(\tan^2\left(\frac{x}{2}\right)\right)-5\right)\ln\left(3\left(\tan^2\left(\frac{x}{2}\right)\right)+2-i\right)+i\left(9\left(\tan^4\left(\frac{x}{2}\right)\right)+12\left(\tan^2\left(\frac{x}{2}\right)\right)+5\right)\ln\left(3\left(\tan^2\left(\frac{x}{2}\right)\right)+2+i\right)}{972\left(\tan^4\left(\frac{x}{2}\right)\right)+1296\left(\tan^2\left(\frac{x}{2}\right)\right)+540}$

```
[In] int(sin(x)/(13-4*cos(x)+cos(x)^2)^2,x,method=_RETURNVERBOSE)
[Out] -1/36*(2*cos(x)-4)/(13-4*cos(x)+cos(x)^2)-1/54*arctan(-2/3+1/3*cos(x))
```

Fricas [A] (verification not implemented)

none

Time = 0.25 (sec) , antiderivative size = 38, normalized size of antiderivative = 1.06

$$\int \frac{\sin(x)}{(13-4\cos(x)+\cos^2(x))^2}\,dx$$
$$=-\frac{\left(\cos\left(x\right)^{2}-4\,\cos\left(x\right)+13\right)\arctan\left(\frac{1}{3}\,\cos\left(x\right)-\frac{2}{3}\right)+3\,\cos\left(x\right)-6}{54\,\left(\cos\left(x\right)^{2}-4\,\cos\left(x\right)+13\right)}$$

```
[In] integrate(sin(x)/(13-4*cos(x)+cos(x)^2)^2,x, algorithm="fricas")
[Out] -1/54*((cos(x)^2 - 4*cos(x) + 13)*arctan(1/3*cos(x) - 2/3) + 3*cos(x) - 6)/
(cos(x)^2 - 4*cos(x) + 13)
```

Sympy [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 116 vs. 2(31) = 62.

Time = 0.47 (sec) , antiderivative size = 116, normalized size of antiderivative = 3.22

$$\int \frac{\sin(x)}{(13-4\cos(x)+\cos^2(x))^2}\,dx = -\frac{\cos^2(x)\operatorname{atan}\left(\frac{\cos(x)}{3}-\frac{2}{3}\right)}{54\cos^2(x)-216\cos(x)+702} + \frac{4\cos(x)\operatorname{atan}\left(\frac{\cos(x)}{3}-\frac{2}{3}\right)}{54\cos^2(x)-216\cos(x)+702} - \frac{3\cos(x)}{54\cos^2(x)-216\cos(x)+702} - \frac{13\operatorname{atan}\left(\frac{\cos(x)}{3}-\frac{2}{3}\right)}{54\cos^2(x)-216\cos(x)+702} + \frac{6}{54\cos^2(x)-216\cos(x)+702}$$

```
[In] integrate(sin(x)/(13-4*cos(x)+cos(x)**2)**2,x)
```

```
[Out] -cos(x)**2*atan(cos(x)/3 - 2/3)/(54*cos(x)**2 - 216*cos(x) + 702) + 4*cos(x
)*atan(cos(x)/3 - 2/3)/(54*cos(x)**2 - 216*cos(x) + 702) - 3*cos(x)/(54*cos
(x)**2 - 216*cos(x) + 702) - 13*atan(cos(x)/3 - 2/3)/(54*cos(x)**2 - 216*co
s(x) + 702) + 6/(54*cos(x)**2 - 216*cos(x) + 702)
```

Maxima [A] (verification not implemented)

none

Time = 0.69 (sec) , antiderivative size = 28, normalized size of antiderivative = 0.78

$$\int \frac{\sin(x)}{(13-4\cos(x)+\cos^2(x))^2}\,dx = -\frac{\cos(x)-2}{18\left(\cos(x)^2-4\cos(x)+13\right)} - \frac{1}{54}\arctan\left(\frac{1}{3}\cos(x)-\frac{2}{3}\right)$$

```
[In] integrate(sin(x)/(13-4*cos(x)+cos(x)^2)^2,x, algorithm="maxima")
```

```
[Out] -1/18*(cos(x) - 2)/(cos(x)^2 - 4*cos(x) + 13) - 1/54*arctan(1/3*cos(x) - 2/
3)
```

Giac [A] (verification not implemented)

none

Time = 0.32 (sec) , antiderivative size = 28, normalized size of antiderivative = 0.78

$$\int \frac{\sin(x)}{(13 - 4\cos(x) + \cos^2(x))^2}\, dx = -\frac{\cos(x) - 2}{18\left(\cos(x)^2 - 4\cos(x) + 13\right)} - \frac{1}{54}\arctan\left(\frac{1}{3}\cos(x) - \frac{2}{3}\right)$$

```
[In] integrate(sin(x)/(13-4*cos(x)+cos(x)^2)^2,x, algorithm="giac")
```

```
[Out] -1/18*(cos(x) - 2)/(cos(x)^2 - 4*cos(x) + 13) - 1/54*arctan(1/3*cos(x) - 2/
3)
```

Mupad [B] (verification not implemented)

Time = 0.08 (sec) , antiderivative size = 30, normalized size of antiderivative = 0.83

$$\int \frac{\sin(x)}{(13 - 4\cos(x) + \cos^2(x))^2}\, dx = -\frac{\mathrm{atan}\left(\frac{\cos(x)}{3} - \frac{2}{3}\right)}{54} - \frac{\frac{\cos(x)}{18} - \frac{1}{9}}{\cos(x)^2 - 4\cos(x) + 13}$$

```
[In] int(sin(x)/(cos(x)^2 - 4*cos(x) + 13)^2,x)
```

```
[Out] - atan(cos(x)/3 - 2/3)/54 - (cos(x)/18 - 1/9)/(cos(x)^2 - 4*cos(x) + 13)
```

3.13 $\int \frac{\cos^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 19, antiderivative size = 326

$$\int \frac{\cos^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{x}{2c}+\frac{(b^2-ac)\,x}{c^3}-\frac{2\left(b^3-2abc-\frac{b^4-4ab^2c+2a^2c^2}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{c^3\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} - \frac{2\left(b^3-2abc+\frac{b^4-4ab^2c+2a^2c^2}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{c^3\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}-\frac{b\sin(x)}{c^2}+\frac{\cos(x)\sin(x)}{2c}$$

```
[Out] 1/2*x/c+(-a*c+b^2)*x/c^3-b*sin(x)/c^2+1/2*cos(x)*sin(x)/c-2*arctan((b-2*c-(
-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2))*(b^3-
2*a*b*c+(-2*a^2*c^2+4*a*b^2*c-b^4)/(-4*a*c+b^2)^(1/2))/c^3/(b-2*c-(-4*a*c+b
^2)^(1/2))^(1/2)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2)-2*arctan((b-2*c+(-4*a*c+b
^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2))*(b^3-2*a*b*c+
(2*a^2*c^2-4*a*b^2*c+b^4)/(-4*a*c+b^2)^(1/2))/c^3/(b-2*c+(-4*a*c+b^2)^(1/2)
)^(1/2)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2)
```

Rubi [A] (verified)

Time = 4.62 (sec) , antiderivative size = 326, normalized size of antiderivative = 1.00, number of steps used = 10, number of rules used = 7, $\frac{\text{number of rules}}{\text{integrand size}} = 0.368$, Rules used = {3338, 2717, 2715, 8, 3374, 2738, 211}

$$\int \frac{\cos^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$$
$$= -\frac{2\left(-\frac{2a^2c^2-4ab^2c+b^4}{\sqrt{b^2-4ac}}-2abc+b^3\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{-\sqrt{b^2-4ac}+b-2c}}{\sqrt{-\sqrt{b^2-4ac}+b+2c}}\right)}{c^3\sqrt{-\sqrt{b^2-4ac}+b-2c}\sqrt{-\sqrt{b^2-4ac}+b+2c}}$$
$$-\frac{2\left(\frac{2a^2c^2-4ab^2c+b^4}{\sqrt{b^2-4ac}}-2abc+b^3\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{\sqrt{b^2-4ac}+b-2c}}{\sqrt{\sqrt{b^2-4ac}+b+2c}}\right)}{c^3\sqrt{\sqrt{b^2-4ac}+b-2c}\sqrt{\sqrt{b^2-4ac}+b+2c}}$$
$$+\frac{x(b^2-ac)}{c^3}-\frac{b\sin(x)}{c^2}+\frac{x}{2c}+\frac{\sin(x)\cos(x)}{2c}$$

```
[In] Int[Cos[x]^4/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] x/(2*c) + ((b^2 - a*c)*x)/c^3 - (2*(b^3 - 2*a*b*c - (b^4 - 4*a*b^2*c + 2*a^
2*c^2)/Sqrt[b^2 - 4*a*c])*ArcTan[(Sqrt[b - 2*c - Sqrt[b^2 - 4*a*c]]*Tan[x/2
])/Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]])/(c^3*Sqrt[b - 2*c - Sqrt[b^2 - 4*a*c
]]*Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]) - (2*(b^3 - 2*a*b*c + (b^4 - 4*a*b^2*
c + 2*a^2*c^2)/Sqrt[b^2 - 4*a*c])*ArcTan[(Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]
*Tan[x/2])/Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]]])/(c^3*Sqrt[b - 2*c + Sqrt[b^2
 - 4*a*c]]*Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]]) - (b*Sin[x])/c^2 + (Cos[x]*Si
n[x])/(2*c)
```

Rule 8

```
Int[a_, x_Symbol] :> Simp[a*x, x] /; FreeQ[a, x]
```

Rule 211

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(Rt[a/b, 2]/a)*ArcTan[x/R
t[a/b, 2]], x] /; FreeQ[{a, b}, x] && PosQ[a/b]
```

Rule 2715

```
Int[((b_.)*sin[(c_.) + (d_.)*(x_)])^(n_), x_Symbol] :> Simp[(-b)*Cos[c + d*
x]*((b*Sin[c + d*x])^(n - 1)/(d*n)), x] + Dist[b^2*((n - 1)/n), Int[(b*Sin[
c + d*x])^(n - 2), x], x] /; FreeQ[{b, c, d}, x] && GtQ[n, 1] && IntegerQ[2
*n]
```

Rule 2717

```
Int[sin[Pi/2 + (c_.) + (d_.)*(x_)], x_Symbol] :> Simp[Sin[c + d*x]/d, x] /;
 FreeQ[{c, d}, x]
```

Rule 2738

```
Int[((a_) + (b_.)*sin[Pi/2 + (c_.) + (d_.)*(x_)])^(-1), x_Symbol] :> With[{
e = FreeFactors[Tan[(c + d*x)/2], x]}, Dist[2*(e/d), Subst[Int[1/(a + b + (
a - b)*e^2*x^2), x], x, Tan[(c + d*x)/2]/e], x]] /; FreeQ[{a, b, c, d}, x]
&& NeQ[a^2 - b^2, 0]
```

Rule 3338

```
Int[cos[(d_.) + (e_.)*(x_)]^(m_.)*((a_.) + cos[(d_.) + (e_.)*(x_)]^(n_.)*(b
_.) + cos[(d_.) + (e_.)*(x_)]^(n2_.)*(c_.))^(p_), x_Symbol] :> Int[ExpandTr
ig[cos[d + e*x]^m*(a + b*cos[d + e*x]^n + c*cos[d + e*x]^(2*n))^p, x], x] /
; FreeQ[{a, b, c, d, e}, x] && EqQ[n2, 2*n] && NeQ[b^2 - 4*a*c, 0] && Integ
ersQ[m, n, p]
```

Rule 3374

```
Int[(cos[(d_.) + (e_.)*(x_)]*(B_.) + (A_))/((a_.) + cos[(d_.) + (e_.)*(x_)]
*(b_.) + cos[(d_.) + (e_.)*(x_)]^2*(c_.)), x_Symbol] :> Module[{q = Rt[b^2
- 4*a*c, 2]}, Dist[B + (b*B - 2*A*c)/q, Int[1/(b + q + 2*c*Cos[d + e*x]), x
], x] + Dist[B - (b*B - 2*A*c)/q, Int[1/(b - q + 2*c*Cos[d + e*x]), x], x]]
 /; FreeQ[{a, b, c, d, e, A, B}, x] && NeQ[b^2 - 4*a*c, 0]
```

Rubi steps

$$\text{integral} = \int \left(\frac{b^2-ac}{c^3}-\frac{b\cos(x)}{c^2}+\frac{\cos^2(x)}{c}+\frac{-ab^2\left(1-\frac{ac}{b^2}\right)-b^3\left(1-\frac{2ac}{b^2}\right)\cos(x)}{c^3\left(a+b\cos(x)+c\cos^2(x)\right)}\right)dx$$

$$=\frac{(b^2-ac)\,x}{c^3}+\frac{\int\frac{-ab^2\left(1-\frac{ac}{b^2}\right)-b^3\left(1-\frac{2ac}{b^2}\right)\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx}{c^3}-\frac{b\int\cos(x)\,dx}{c^2}+\frac{\int\cos^2(x)\,dx}{c}$$

$$=\frac{(b^2-ac)\,x}{c^3}-\frac{b\sin(x)}{c^2}+\frac{\cos(x)\sin(x)}{2c}+\frac{\int 1\,dx}{2c}-\frac{\left(b^3-2abc-\frac{b^4-4ab^2c+2a^2c^2}{\sqrt{b^2-4ac}}\right)\int\frac{1}{b-\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{c^3}-\frac{\left(b^3-2abc+\frac{b^4-4ab^2c+2a^2c^2}{\sqrt{b^2-4ac}}\right)\int\frac{1}{b+\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{c^3}$$

$$= \frac{x}{2c} + \frac{(b^2 - ac)\,x}{c^3} - \frac{b\sin(x)}{c^2} + \frac{\cos(x)\sin(x)}{2c} - \frac{\left(2\left(b^3 - 2abc - \frac{b^4-4ab^2c+2a^2c^2}{\sqrt{b^2-4ac}}\right)\right)\operatorname{Subst}\left(\int \frac{1}{b+2c-\sqrt{b^2-4ac}+\left(b-2c-\sqrt{b^2-4ac}\right)x^2}\,dx, x, \tan\left(\frac{x}{2}\right)\right)}{c^3} - \frac{\left(2\left(b^3 - 2abc + \frac{b^4-4ab^2c+2a^2c^2}{\sqrt{b^2-4ac}}\right)\right)\operatorname{Subst}\left(\int \frac{1}{b+2c+\sqrt{b^2-4ac}+\left(b-2c+\sqrt{b^2-4ac}\right)x^2}\,dx, x, \tan\left(\frac{x}{2}\right)\right)}{c^3}$$

$$= \frac{x}{2c} + \frac{(b^2 - ac)\,x}{c^3} - \frac{2\left(b^3 - 2abc - \frac{b^4-4ab^2c+2a^2c^2}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{c^3\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} - \frac{2\left(b^3 - 2abc + \frac{b^4-4ab^2c+2a^2c^2}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{c^3\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}} - \frac{b\sin(x)}{c^2} + \frac{\cos(x)\sin(x)}{2c}$$

Mathematica [A] (verified)

Time = 0.60 (sec) , antiderivative size = 356, normalized size of antiderivative = 1.09

$$\int \frac{\cos^4(x)}{a + b\cos(x) + c\cos^2(x)}\,dx$$

$$= \frac{4b^2x + 2c(-2a+c)x + \frac{4\sqrt{2}\left(b^4-4ab^2c+2a^2c^2+b^3\sqrt{b^2-4ac}-2abc\sqrt{b^2-4ac}\right)\operatorname{arctanh}\left(\frac{\left(b-2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)-2b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{-b^2+2c(a+c)-b\sqrt{b^2-4ac}}} - \frac{4\sqrt{2}\left(-b^4+\ldots}{\ldots}}{4c^3}$$

```
[In] Integrate[Cos[x]^4/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (4*b^2*x + 2*c*(-2*a + c)*x + (4*Sqrt[2]*(b^4 - 4*a*b^2*c + 2*a^2*c^2 + b^3
*Sqrt[b^2 - 4*a*c] - 2*a*b*c*Sqrt[b^2 - 4*a*c])*ArcTanh[((b - 2*c + Sqrt[b^
2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) - 2*b*Sqrt[b^2 - 4*a*c]]])/
(Sqrt[b^2 - 4*a*c]*Sqrt[-b^2 + 2*c*(a + c) - b*Sqrt[b^2 - 4*a*c]]) - (4*Sqr
t[2]*(-b^4 + 4*a*b^2*c - 2*a^2*c^2 + b^3*Sqrt[b^2 - 4*a*c] - 2*a*b*c*Sqrt[b
^2 - 4*a*c])*ArcTanh[((-b + 2*c + Sqrt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2
+ 4*c*(a + c) + 2*b*Sqrt[b^2 - 4*a*c]]])/(Sqrt[b^2 - 4*a*c]*Sqrt[-b^2 + 2*c
*(a + c) + b*Sqrt[b^2 - 4*a*c]]) - 4*b*c*Sin[x] + c^2*Sin[2*x])/(4*c^3)
```

Maple [A] (verified)

Time = 4.72 (sec) , antiderivative size = 431, normalized size of antiderivative = 1.32

method	result
default	$\frac{2(a-b+c)\left(\frac{\left(a^{2}c\sqrt{-4ac+b^{2}}-a\,b^{2}\sqrt{-4ac+b^{2}}-2cab\sqrt{-4ac+b^{2}}+b^{3}\sqrt{-4ac+b^{2}}+3c\,a^{2}b+2a^{2}c^{2}-a\,b^{3}-4a\,b^{2}c+b^{4}\right)\operatorname{arctanh}\left(\frac{(-a+b-c)\tan\left(\frac{x}{2}\right)}{\sqrt{\left(\sqrt{-4ac+b^{2}}-a+c\right)(a- \cdots}}\right. \cdots}{2\sqrt{-4ac+b^{2}}\,(a-b+c)\sqrt{\left(\sqrt{-4ac+b^{2}}-a+c\right)(a-b+c)}} \cdots\right.}{\cdots}$
risch	Expression too large to display

```
[In] int(cos(x)^4/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] 2/c^3*(a-b+c)*(1/2*(a^2*c*(-4*a*c+b^2)^(1/2)-a*b^2*(-4*a*c+b^2)^(1/2)-2*c*a
*b*(-4*a*c+b^2)^(1/2)+b^3*(-4*a*c+b^2)^(1/2)+3*c*a^2*b+2*a^2*c^2-a*b^3-4*a*
b^2*c+b^4)/(-4*a*c+b^2)^(1/2)/(a-b+c)/(((-4*a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1
/2)*arctanh((-a+b-c)*tan(1/2*x)/(((-4*a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2))+1
/2*(a^2*c*(-4*a*c+b^2)^(1/2)-a*b^2*(-4*a*c+b^2)^(1/2)-2*c*a*b*(-4*a*c+b^2)^
(1/2)+b^3*(-4*a*c+b^2)^(1/2)-3*c*a^2*b-2*a^2*c^2+a*b^3+4*a*b^2*c-b^4)/(-4*a
*c+b^2)^(1/2)/(a-b+c)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)*arctan((a-b+
c)*tan(1/2*x)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)))-2/c^3*(((c*b+1/2*c
^2)*tan(1/2*x)^3+(c*b-1/2*c^2)*tan(1/2*x))/(1+tan(1/2*x)^2)^2+1/2*(2*a*c-2*
b^2-c^2)*arctan(tan(1/2*x)))
```

Fricas [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 8167 vs. $2(282) = 564$.

Time = 5.52 (sec) , antiderivative size = 8167, normalized size of antiderivative = 25.05

$$\int \frac{\cos^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(cos(x)^4/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
```

```
[Out] Too large to include
```

Sympy [F(-1)]

Timed out.

$$\int \frac{\cos^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Timed out}$$

```
[In] integrate(cos(x)**4/(a+b*cos(x)+c*cos(x)**2),x)
```

```
[Out] Timed out
```

Maxima [F]

$$\int \frac{\cos^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \int \frac{\cos(x)^4}{c\cos(x)^2+b\cos(x)+a}\,dx$$

```
[In] integrate(cos(x)^4/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
```

```
[Out] 1/4*(4*c^3*integrate(-2*(2*(b^4 - 2*a*b^2*c)*cos(3*x)^2 + 4*(2*a^2*b^2 - a^
2*c^2 - (2*a^3 - a*b^2)*c)*cos(2*x)^2 + 2*(b^4 - 2*a*b^2*c)*cos(x)^2 + 2*(b
^4 - 2*a*b^2*c)*sin(3*x)^2 + 4*(2*a^2*b^2 - a^2*c^2 - (2*a^3 - a*b^2)*c)*si
n(2*x)^2 + 2*(4*a*b^3 - 2*a*b*c^2 - (6*a^2*b - b^3)*c)*sin(2*x)*sin(x) + 2*
(b^4 - 2*a*b^2*c)*sin(x)^2 + ((b^3*c - 2*a*b*c^2)*cos(3*x) + 2*(a*b^2*c - a
^2*c^2)*cos(2*x) + (b^3*c - 2*a*b*c^2)*cos(x))*cos(4*x) + (b^3*c - 2*a*b*c^
2 + 2*(4*a*b^3 - 2*a*b*c^2 - (6*a^2*b - b^3)*c)*cos(2*x) + 4*(b^4 - 2*a*b^2
*c)*cos(x))*cos(3*x) + 2*(a*b^2*c - a^2*c^2 + (4*a*b^3 - 2*a*b*c^2 - (6*a^2
*b - b^3)*c)*cos(x))*cos(2*x) + (b^3*c - 2*a*b*c^2)*cos(x) + ((b^3*c - 2*a*
b*c^2)*sin(3*x) + 2*(a*b^2*c - a^2*c^2)*sin(2*x) + (b^3*c - 2*a*b*c^2)*sin(
x))*sin(4*x) + 2*((4*a*b^3 - 2*a*b*c^2 - (6*a^2*b - b^3)*c)*sin(2*x) + 2*(b
^4 - 2*a*b^2*c)*sin(x))*sin(3*x))/(c^5*cos(4*x)^2 + 4*b^2*c^3*cos(3*x)^2 +
4*b^2*c^3*cos(x)^2 + c^5*sin(4*x)^2 + 4*b^2*c^3*sin(3*x)^2 + 4*b^2*c^3*sin(
x)^2 + 4*b*c^4*cos(x) + c^5 + 4*(4*a^2*c^3 + 4*a*c^4 + c^5)*cos(2*x)^2 + 4*
(4*a^2*c^3 + 4*a*c^4 + c^5)*sin(2*x)^2 + 8*(2*a*b*c^3 + b*c^4)*sin(2*x)*sin
(x) + 2*(2*b*c^4*cos(3*x) + 2*b*c^4*cos(x) + c^5 + 2*(2*a*c^4 + c^5)*cos(2*
x))*cos(4*x) + 4*(2*b^2*c^3*cos(x) + b*c^4 + 2*(2*a*b*c^3 + b*c^4)*cos(2*x)
)*cos(3*x) + 4*(2*a*c^4 + c^5 + 2*(2*a*b*c^3 + b*c^4)*cos(x))*cos(2*x) + 4*
(b*c^4*sin(3*x) + b*c^4*sin(x) + (2*a*c^4 + c^5)*sin(2*x))*sin(4*x) + 8*(b^
2*c^3*sin(x) + (2*a*b*c^3 + b*c^4)*sin(2*x))*sin(3*x)), x) + c^2*sin(2*x) -
 4*b*c*sin(x) + 2*(2*b^2 - 2*a*c + c^2)*x)/c^3
```

Giac [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 12587 vs. 2(282) = 564.

Time = 2.78 (sec) , antiderivative size = 12587, normalized size of antiderivative = 38.61

$$\int \frac{\cos^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(cos(x)^4/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] -((2*a^3*b^6 - 6*a^2*b^7 + 6*a*b^8 - 2*b^9 - 18*a^4*b^4*c + 56*a^3*b^5*c -
54*a^2*b^6*c + 12*a*b^7*c + 4*b^8*c + 48*a^5*b^2*c^2 - 160*a^4*b^3*c^2 + 14
0*a^3*b^4*c^2 + 12*a^2*b^5*c^2 - 38*a*b^6*c^2 - 2*b^7*c^2 - 32*a^6*c^3 + 12
8*a^5*b*c^3 - 64*a^4*b^2*c^3 - 160*a^3*b^3*c^3 + 110*a^2*b^4*c^3 + 20*a*b^5
*c^3 - 64*a^5*c^4 + 192*a^4*b*c^4 - 80*a^3*b^2*c^4 - 64*a^2*b^3*c^4 - 32*a^
4*c^5 + 64*a^3*b*c^5 + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
- b + c))*sqrt(b^2 - 4*a*c)*a^3*b^4 - 2*(b^2 - 4*a*c)*a^3*b^4 - 5*sqrt(a^2
- a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^
5 + 6*(b^2 - 4*a*c)*a^2*b^5 - 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a
*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^6 - 6*(b^2 - 4*a*c)*a*b^6 + 5*sqrt(a
^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^7
 + 2*(b^2 - 4*a*c)*b^7 - 15*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*
(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^2*c + 10*(b^2 - 4*a*c)*a^4*b^2*c + 28*
sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*
c)*a^3*b^3*c - 32*(b^2 - 4*a*c)*a^3*b^3*c + 27*sqrt(a^2 - a*b + b*c - c^2 +
 sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^4*c + 30*(b^2 - 4*a
*c)*a^2*b^4*c - 38*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*sqrt(b^2 - 4*a*c)*a*b^5*c - 4*(b^2 - 4*a*c)*a*b^5*c - 6*sqrt(a^2 - a*b
+ b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^6*c - 4*(b
^2 - 4*a*c)*b^6*c + 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
b + c))*sqrt(b^2 - 4*a*c)*a^5*c^2 - 8*(b^2 - 4*a*c)*a^5*c^2 - 32*sqrt(a^2 -
 a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b*c
^2 + 32*(b^2 - 4*a*c)*a^4*b*c^2 - 74*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
- 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c^2 - 20*(b^2 - 4*a*c)*a^3*
b^2*c^2 + 94*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sq
rt(b^2 - 4*a*c)*a^2*b^3*c^2 - 28*(b^2 - 4*a*c)*a^2*b^3*c^2 + 31*sqrt(a^2 -
a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4*c^
2 + 22*(b^2 - 4*a*c)*a*b^4*c^2 + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5*c^2 + 2*(b^2 - 4*a*c)*b^5*c^2 + 5
6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*a^4*c^3 - 16*(b^2 - 4*a*c)*a^4*c^3 - 88*sqrt(a^2 - a*b + b*c - c^2 + s
qrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^3 + 48*(b^2 - 4*a*c
)*a^3*b*c^3 - 23*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c)
)*sqrt(b^2 - 4*a*c)*a^2*b^2*c^3 - 22*(b^2 - 4*a*c)*a^2*b^2*c^3 - 30*sqrt(a^
2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^
```

```
3*c^3 - 12*(b^2 - 4*a*c)*a*b^3*c^3 - 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b
^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^4 - 8*(b^2 - 4*a*c)*a^3*c^
4 + 40*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
 - 4*a*c)*a^2*b*c^4 + 16*(b^2 - 4*a*c)*a^2*b*c^4)*c^2*abs(a - b + c) + (4*a
^3*b^6*c - 4*a^2*b^7*c - 4*a*b^8*c + 4*b^9*c - 36*a^4*b^4*c^2 + 32*a^3*b^5*
c^2 + 52*a^2*b^6*c^2 - 40*a*b^7*c^2 - 8*b^8*c^2 + 96*a^5*b^2*c^3 - 64*a^4*b
^3*c^3 - 232*a^3*b^4*c^3 + 120*a^2*b^5*c^3 + 84*a*b^6*c^3 + 4*b^7*c^3 - 64*
a^6*c^4 + 384*a^4*b^2*c^4 - 64*a^3*b^3*c^4 - 292*a^2*b^4*c^4 - 40*a*b^5*c^4
 - 128*a^5*c^5 - 128*a^4*b*c^5 + 352*a^3*b^2*c^5 + 128*a^2*b^3*c^5 - 64*a^4
*c^6 - 128*a^3*b*c^6 + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
- b + c))*a^4*b^4*c - 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
 b + c))*a^3*b^5*c - 8*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
b + c))*a^2*b^6*c + 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
 + c))*a*b^7*c + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*b^8*c - 15*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*
a^5*b^2*c^2 + 7*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))
*a^4*b^3*c^2 + 62*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
))*a^3*b^4*c^2 - 51*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
 c))*a*b^6*c^2 - 11*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
 c))*b^7*c^2 + 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
))*a^6*c^3 + 4*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*
a^5*b*c^3 - 137*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))
*a^4*b^2*c^3 - 56*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
))*a^3*b^3*c^3 + 164*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^2*b^4*c^3 + 86*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
b + c))*a*b^5*c^3 + 11*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
b + c))*b^6*c^3 + 68*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^5*c^4 + 96*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*a^4*b*c^4 - 169*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
 c))*a^3*b^2*c^4 - 197*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
b + c))*a^2*b^3*c^4 - 71*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
- b + c))*a*b^4*c^4 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
 b + c))*b^5*c^4 + 36*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
 + c))*a^4*c^5 + 116*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^3*b*c^5 + 113*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
 + c))*a^2*b^2*c^5 + 30*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
 b + c))*a*b^3*c^5 - 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
 b + c))*a^3*c^6 - 40*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
 + c))*a^2*b*c^6 - 4*(b^2 - 4*a*c)*a^3*b^4*c + 4*(b^2 - 4*a*c)*a^2*b^5*c +
4*(b^2 - 4*a*c)*a*b^6*c - 4*(b^2 - 4*a*c)*b^7*c + 20*(b^2 - 4*a*c)*a^4*b^2*
c^2 - 16*(b^2 - 4*a*c)*a^3*b^3*c^2 - 36*(b^2 - 4*a*c)*a^2*b^4*c^2 + 24*(b^2
 - 4*a*c)*a*b^5*c^2 + 8*(b^2 - 4*a*c)*b^6*c^2 - 16*(b^2 - 4*a*c)*a^5*c^3 +
88*(b^2 - 4*a*c)*a^3*b^2*c^3 - 24*(b^2 - 4*a*c)*a^2*b^3*c^3 - 52*(b^2 - 4*a
*c)*a*b^4*c^3 - 4*(b^2 - 4*a*c)*b^5*c^3 - 32*(b^2 - 4*a*c)*a^4*c^4 - 32*(b^
2 - 4*a*c)*a^3*b*c^4 + 84*(b^2 - 4*a*c)*a^2*b^2*c^4 + 24*(b^2 - 4*a*c)*a*b^
```

```
3*c^4 - 16*(b^2 - 4*a*c)*a^3*c^5 - 32*(b^2 - 4*a*c)*a^2*b*c^5)*abs(a - b +
c)*abs(c) + (2*a^4*b^5*c^2 - 6*a^3*b^6*c^2 + 6*a^2*b^7*c^2 - 2*a*b^8*c^2 -
14*a^5*b^3*c^3 + 44*a^4*b^4*c^3 - 44*a^3*b^5*c^3 + 14*a^2*b^6*c^3 - 2*a*b^7
*c^3 + 2*b^8*c^3 + 24*a^6*b*c^4 - 84*a^5*b^2*c^4 + 82*a^4*b^3*c^4 - 20*a^3*
b^4*c^4 + 12*a^2*b^5*c^4 - 10*a*b^6*c^4 - 4*b^7*c^4 + 16*a^6*c^5 - 8*a^5*b*
c^5 - 20*a^4*b^2*c^5 - 10*a^3*b^3*c^5 - 8*a^2*b^4*c^5 + 30*a*b^5*c^5 + 2*b^
6*c^5 + 16*a^5*c^6 - 24*a^4*b*c^6 + 68*a^3*b^2*c^6 - 58*a^2*b^3*c^6 - 16*a*
b^4*c^6 - 16*a^4*c^7 + 8*a^3*b*c^7 + 36*a^2*b^2*c^7 - 16*a^3*c^8 + 3*sqrt(a
^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4
*b^3*c^2 - 2*(b^2 - 4*a*c)*a^4*b^3*c^2 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqr
t(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^4*c^2 + 6*(b^2 - 4*a*c)
*a^3*b^4*c^2 - 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c)
)*sqrt(b^2 - 4*a*c)*a^2*b^5*c^2 - 6*(b^2 - 4*a*c)*a^2*b^5*c^2 + 5*sqrt(a^2
- a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^6*
c^2 + 2*(b^2 - 4*a*c)*a*b^6*c^2 - 9*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*b*c^3 + 6*(b^2 - 4*a*c)*a^5*b*c^
3 + 18*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
 - 4*a*c)*a^4*b^2*c^3 - 20*(b^2 - 4*a*c)*a^4*b^2*c^3 + 18*sqrt(a^2 - a*b +
b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^3*c^3 +
20*(b^2 - 4*a*c)*a^3*b^3*c^3 - 23*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4
*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^4*c^3 - 6*(b^2 - 4*a*c)*a^2*b^4*
c^3 - 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^
2 - 4*a*c)*a*b^5*c^3 + 2*(b^2 - 4*a*c)*a*b^5*c^3 - 5*sqrt(a^2 - a*b + b*c -
 c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^6*c^3 - 2*(b^2 -
4*a*c)*b^6*c^3 - 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*sqrt(b^2 - 4*a*c)*a^5*c^4 + 4*(b^2 - 4*a*c)*a^5*c^4 - 29*sqrt(a^2 - a*b
 + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b*c^4 -
 2*(b^2 - 4*a*c)*a^4*b*c^4 + 30*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a
*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c^4 - 4*(b^2 - 4*a*c)*a^3*b^2*c^
4 - 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*a^2*b^3*c^4 - 4*(b^2 - 4*a*c)*a^2*b^3*c^4 + 33*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4*c^4 + 2*(b
^2 - 4*a*c)*a*b^4*c^4 + 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
 - b + c))*sqrt(b^2 - 4*a*c)*b^5*c^4 + 4*(b^2 - 4*a*c)*b^5*c^4 - 22*sqrt(a^
2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*
c^5 + 4*(b^2 - 4*a*c)*a^4*c^5 + 41*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^5 - 6*(b^2 - 4*a*c)*a^3*b*c^5
 - 68*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*a^2*b^2*c^5 + 16*(b^2 - 4*a*c)*a^2*b^2*c^5 - 19*sqrt(a^2 - a*b + b
*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c^5 - 14*
(b^2 - 4*a*c)*a*b^3*c^5 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*
(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c^5 - 2*(b^2 - 4*a*c)*b^4*c^5 + 38*sqrt(
a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^
3*c^6 - 4*(b^2 - 4*a*c)*a^3*c^6 - 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^6 + 2*(b^2 - 4*a*c)*a^2*b*c^
```

6 + 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^6 + 8*(b^2 - 4*a*c)*a*b^2*c^6 - 10*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^7 - 4*(b^2 - 4*a*c)*a^2*c^7)*abs(a - b + c))*(pi*floor(1/2*x/pi + 1/2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a*c^3 - 2*c^4 + sqrt(-4*(a*c^3 + b*c^3 + c^4)*(a*c^3 - b*c^3 + c^4) + 4*(a*c^3 - c^4)^2))/(a*c^3 - b*c^3 + c^4))))/((3*a^6*b^2*c^4 - 8*a^5*b^3*c^4 - a^4*b^4*c^4 + 16*a^3*b^5*c^4 - 7*a^2*b^6*c^4 - 8*a*b^7*c^4 + 5*b^8*c^4 - 12*a^7*c^5 + 32*a^6*b*c^5 + 30*a^5*b^2*c^5 - 112*a^4*b^3*c^5 + 8*a^3*b^4*c^5 + 96*a^2*b^5*c^5 - 26*a*b^6*c^5 - 16*b^7*c^5 - 104*a^6*c^6 + 192*a^5*b*c^6 + 149*a^4*b^2*c^6 - 336*a^3*b^3*c^6 - 30*a^2*b^4*c^6 + 112*a*b^5*c^6 + 17*b^6*c^6 - 276*a^5*c^7 + 320*a^4*b*c^7 + 292*a^3*b^2*c^7 - 224*a^2*b^3*c^7 - 120*a*b^4*c^7 - 304*a^4*c^8 + 128*a^3*b*c^8 + 237*a^2*b^2*c^8 + 24*a*b^3*c^8 - 17*b^4*c^8 - 116*a^3*c^9 - 96*a^2*b*c^9 + 62*a*b^2*c^9 + 16*b^3*c^9 + 24*a^2*c^10 - 64*a*b*c^10 - 5*b^2*c^10 + 20*a*c^11)*abs(c)) - ((2*a^3*b^6 - 6*a^2*b^7 + 6*a*b^8 - 2*b^9 - 18*a^4*b^4*c + 56*a^3*b^5*c - 54*a^2*b^6*c + 12*a*b^7*c + 4*b^8*c + 48*a^5*b^2*c^2 - 160*a^4*b^3*c^2 + 140*a^3*b^4*c^2 + 12*a^2*b^5*c^2 - 38*a*b^6*c^2 - 2*b^7*c^2 - 32*a^6*c^3 + 128*a^5*b*c^3 - 64*a^4*b^2*c^3 - 160*a^3*b^3*c^3 + 110*a^2*b^4*c^3 + 20*a*b^5*c^3 - 64*a^5*c^4 + 192*a^4*b*c^4 - 80*a^3*b^2*c^4 - 64*a^2*b^3*c^4 - 32*a^4*c^5 + 64*a^3*b*c^5 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^4 - 2*(b^2 - 4*a*c)*a^3*b^4 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^5 + 6*(b^2 - 4*a*c)*a^2*b^5 - 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^6 - 6*(b^2 - 4*a*c)*a*b^6 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^7 + 2*(b^2 - 4*a*c)*b^7 - 15*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^2*c + 10*(b^2 - 4*a*c)*a^4*b^2*c + 28*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^3*c - 32*(b^2 - 4*a*c)*a^3*b^3*c + 27*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^4*c + 30*(b^2 - 4*a*c)*a^2*b^4*c - 38*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^5*c - 4*(b^2 - 4*a*c)*a*b^5*c - 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^6*c - 4*(b^2 - 4*a*c)*b^6*c + 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*c^2 - 8*(b^2 - 4*a*c)*a^5*c^2 - 32*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b*c^2 + 32*(b^2 - 4*a*c)*a^4*b*c^2 - 74*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c^2 - 20*(b^2 - 4*a*c)*a^3*b^2*c^2 + 94*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3*c^2 - 28*(b^2 - 4*a*c)*a^2*b^3*c^2 + 31*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4*c^2 + 22*(b^2 - 4*a*c)*a*b^4*c^2 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5*c^2 + 2*(b^2 - 4*a*c)*b^5*c^2 + 56*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*c^3 - 16*(b^2 - 4*a*c)*a^4*c^3 - 88*sqrt(a^2 - a*b + b*c -

```
c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^3 + 48*(b^2
- 4*a*c)*a^3*b*c^3 - 23*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
 b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c^3 - 22*(b^2 - 4*a*c)*a^2*b^2*c^3 - 30*
sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*
c)*a*b^3*c^3 - 12*(b^2 - 4*a*c)*a*b^3*c^3 - 20*sqrt(a^2 - a*b + b*c - c^2 -
 sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^4 - 8*(b^2 - 4*a*c)
*a^3*c^4 + 40*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*s
qrt(b^2 - 4*a*c)*a^2*b*c^4 + 16*(b^2 - 4*a*c)*a^2*b*c^4)*c^2*abs(a - b + c)
 - (4*a^3*b^6*c - 4*a^2*b^7*c - 4*a*b^8*c + 4*b^9*c - 36*a^4*b^4*c^2 + 32*a
^3*b^5*c^2 + 52*a^2*b^6*c^2 - 40*a*b^7*c^2 - 8*b^8*c^2 + 96*a^5*b^2*c^3 - 6
4*a^4*b^3*c^3 - 232*a^3*b^4*c^3 + 120*a^2*b^5*c^3 + 84*a*b^6*c^3 + 4*b^7*c^
3 - 64*a^6*c^4 + 384*a^4*b^2*c^4 - 64*a^3*b^3*c^4 - 292*a^2*b^4*c^4 - 40*a*
b^5*c^4 - 128*a^5*c^5 - 128*a^4*b*c^5 + 352*a^3*b^2*c^5 + 128*a^2*b^3*c^5 -
 64*a^4*c^6 - 128*a^3*b*c^6 - 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a
*c)*(a - b + c))*a^4*b^4*c + 2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*a^3*b^5*c + 8*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*a^2*b^6*c - 2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)
*(a - b + c))*a*b^7*c - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*b^8*c + 15*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
 + c))*a^5*b^2*c^2 - 7*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*a^4*b^3*c^2 - 62*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*a^3*b^4*c^2 + 51*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(
a - b + c))*a*b^6*c^2 + 11*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(
a - b + c))*b^7*c^2 - 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*a^6*c^3 - 4*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
 + c))*a^5*b*c^3 + 137*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*a^4*b^2*c^3 + 56*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*a^3*b^3*c^3 - 164*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*
(a - b + c))*a^2*b^4*c^3 - 86*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*a*b^5*c^3 - 11*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*b^6*c^3 - 68*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*
(a - b + c))*a^5*c^4 - 96*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*a^4*b*c^4 + 169*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(
a - b + c))*a^3*b^2*c^4 + 197*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*a^2*b^3*c^4 + 71*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a
*c)*(a - b + c))*a*b^4*c^4 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*b^5*c^4 - 36*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)
*(a - b + c))*a^4*c^5 - 116*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*
(a - b + c))*a^3*b*c^5 - 113*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)
*(a - b + c))*a^2*b^2*c^5 - 30*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*a*b^3*c^5 + 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*a^3*c^6 + 40*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)
*(a - b + c))*a^2*b*c^6 - 4*(b^2 - 4*a*c)*a^3*b^4*c + 4*(b^2 - 4*a*c)*a^2*b
^5*c + 4*(b^2 - 4*a*c)*a*b^6*c - 4*(b^2 - 4*a*c)*b^7*c + 20*(b^2 - 4*a*c)*a
^4*b^2*c^2 - 16*(b^2 - 4*a*c)*a^3*b^3*c^2 - 36*(b^2 - 4*a*c)*a^2*b^4*c^2 +
```

```
24*(b^2 - 4*a*c)*a*b^5*c^2 + 8*(b^2 - 4*a*c)*b^6*c^2 - 16*(b^2 - 4*a*c)*a^5
*c^3 + 88*(b^2 - 4*a*c)*a^3*b^2*c^3 - 24*(b^2 - 4*a*c)*a^2*b^3*c^3 - 52*(b^
2 - 4*a*c)*a*b^4*c^3 - 4*(b^2 - 4*a*c)*b^5*c^3 - 32*(b^2 - 4*a*c)*a^4*c^4 -
 32*(b^2 - 4*a*c)*a^3*b*c^4 + 84*(b^2 - 4*a*c)*a^2*b^2*c^4 + 24*(b^2 - 4*a*
c)*a*b^3*c^4 - 16*(b^2 - 4*a*c)*a^3*c^5 - 32*(b^2 - 4*a*c)*a^2*b*c^5)*abs(a
 - b + c)*abs(c) + (2*a^4*b^5*c^2 - 6*a^3*b^6*c^2 + 6*a^2*b^7*c^2 - 2*a*b^8
*c^2 - 14*a^5*b^3*c^3 + 44*a^4*b^4*c^3 - 44*a^3*b^5*c^3 + 14*a^2*b^6*c^3 -
2*a*b^7*c^3 + 2*b^8*c^3 + 24*a^6*b*c^4 - 84*a^5*b^2*c^4 + 82*a^4*b^3*c^4 -
20*a^3*b^4*c^4 + 12*a^2*b^5*c^4 - 10*a*b^6*c^4 - 4*b^7*c^4 + 16*a^6*c^5 - 8
*a^5*b*c^5 - 20*a^4*b^2*c^5 - 10*a^3*b^3*c^5 - 8*a^2*b^4*c^5 + 30*a*b^5*c^5
 + 2*b^6*c^5 + 16*a^5*c^6 - 24*a^4*b*c^6 + 68*a^3*b^2*c^6 - 58*a^2*b^3*c^6
- 16*a*b^4*c^6 - 16*a^4*c^7 + 8*a^3*b*c^7 + 36*a^2*b^2*c^7 - 16*a^3*c^8 + 3
*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a
*c)*a^4*b^3*c^2 - 2*(b^2 - 4*a*c)*a^4*b^3*c^2 - 5*sqrt(a^2 - a*b + b*c - c^
2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^4*c^2 + 6*(b^2 -
 4*a*c)*a^3*b^4*c^2 - 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
 b + c))*sqrt(b^2 - 4*a*c)*a^2*b^5*c^2 - 6*(b^2 - 4*a*c)*a^2*b^5*c^2 + 5*sq
rt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)
*a*b^6*c^2 + 2*(b^2 - 4*a*c)*a*b^6*c^2 - 9*sqrt(a^2 - a*b + b*c - c^2 - sqr
t(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*b*c^3 + 6*(b^2 - 4*a*c)*a
^5*b*c^3 + 18*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*s
qrt(b^2 - 4*a*c)*a^4*b^2*c^3 - 20*(b^2 - 4*a*c)*a^4*b^2*c^3 + 18*sqrt(a^2 -
 a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^3
*c^3 + 20*(b^2 - 4*a*c)*a^3*b^3*c^3 - 23*sqrt(a^2 - a*b + b*c - c^2 - sqrt(
b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^4*c^3 - 6*(b^2 - 4*a*c)*a
^2*b^4*c^3 - 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*
sqrt(b^2 - 4*a*c)*a*b^5*c^3 + 2*(b^2 - 4*a*c)*a*b^5*c^3 - 5*sqrt(a^2 - a*b
+ b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^6*c^3 - 2*
(b^2 - 4*a*c)*b^6*c^3 - 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*sqrt(b^2 - 4*a*c)*a^5*c^4 + 4*(b^2 - 4*a*c)*a^5*c^4 - 29*sqrt(a^
2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*
b*c^4 - 2*(b^2 - 4*a*c)*a^4*b*c^4 + 30*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^
2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c^4 - 4*(b^2 - 4*a*c)*a^3
*b^2*c^4 - 2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sq
rt(b^2 - 4*a*c)*a^2*b^3*c^4 - 4*(b^2 - 4*a*c)*a^2*b^3*c^4 + 33*sqrt(a^2 - a
*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4*c^4
 + 2*(b^2 - 4*a*c)*a*b^4*c^4 + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5*c^4 + 4*(b^2 - 4*a*c)*b^5*c^4 - 22*
sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*
c)*a^4*c^5 + 4*(b^2 - 4*a*c)*a^4*c^5 + 41*sqrt(a^2 - a*b + b*c - c^2 - sqrt
(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^5 - 6*(b^2 - 4*a*c)*a^
3*b*c^5 - 68*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sq
rt(b^2 - 4*a*c)*a^2*b^2*c^5 + 16*(b^2 - 4*a*c)*a^2*b^2*c^5 - 19*sqrt(a^2 -
a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c^
5 - 14*(b^2 - 4*a*c)*a*b^3*c^5 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
```

```
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c^5 - 2*(b^2 - 4*a*c)*b^4*c^5 + 3
8*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*a^3*c^6 - 4*(b^2 - 4*a*c)*a^3*c^6 - 3*sqrt(a^2 - a*b + b*c - c^2 - sqr
t(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^6 + 2*(b^2 - 4*a*c)*a
^2*b*c^6 + 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*s
qrt(b^2 - 4*a*c)*a*b^2*c^6 + 8*(b^2 - 4*a*c)*a*b^2*c^6 - 10*sqrt(a^2 - a*b
+ b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^7 - 4*
(b^2 - 4*a*c)*a^2*c^7)*abs(a - b + c))*(pi*floor(1/2*x/pi + 1/2) + arctan(2
*sqrt(1/2)*tan(1/2*x)/sqrt((2*a*c^3 - 2*c^4 - sqrt(-4*(a*c^3 + b*c^3 + c^4)
*(a*c^3 - b*c^3 + c^4) + 4*(a*c^3 - c^4)^2))/(a*c^3 - b*c^3 + c^4))))/((3*a
^6*b^2*c^4 - 8*a^5*b^3*c^4 - a^4*b^4*c^4 + 16*a^3*b^5*c^4 - 7*a^2*b^6*c^4 -
 8*a*b^7*c^4 + 5*b^8*c^4 - 12*a^7*c^5 + 32*a^6*b*c^5 + 30*a^5*b^2*c^5 - 112
*a^4*b^3*c^5 + 8*a^3*b^4*c^5 + 96*a^2*b^5*c^5 - 26*a*b^6*c^5 - 16*b^7*c^5 -
 104*a^6*c^6 + 192*a^5*b*c^6 + 149*a^4*b^2*c^6 - 336*a^3*b^3*c^6 - 30*a^2*b
^4*c^6 + 112*a*b^5*c^6 + 17*b^6*c^6 - 276*a^5*c^7 + 320*a^4*b*c^7 + 292*a^3
*b^2*c^7 - 224*a^2*b^3*c^7 - 120*a*b^4*c^7 - 304*a^4*c^8 + 128*a^3*b*c^8 +
237*a^2*b^2*c^8 + 24*a*b^3*c^8 - 17*b^4*c^8 - 116*a^3*c^9 - 96*a^2*b*c^9 +
62*a*b^2*c^9 + 16*b^3*c^9 + 24*a^2*c^10 - 64*a*b*c^10 - 5*b^2*c^10 + 20*a*c
^11)*abs(c)) + 1/2*(2*b^2 - 2*a*c + c^2)*x/c^3 - (2*b*tan(1/2*x)^3 + c*tan(
1/2*x)^3 + 2*b*tan(1/2*x) - c*tan(1/2*x))/((tan(1/2*x)^2 + 1)^2*c^2)
```

Mupad [B] (verification not implemented)

Time = 16.24 (sec) , antiderivative size = 45364, normalized size of antiderivative = 139.15

$$\int \frac{\cos^4(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] int(cos(x)^4/(a + b*cos(x) + c*cos(x)^2),x)
```

```
[Out] atan(((((2048*(12*a^3*c^11 - 28*a^4*c^10 - 44*a^5*c^9 + 72*a^6*c^8 + 88*a^7
*c^7 + 12*a^8*c^6 + b^5*c^9 - 4*b^6*c^8 + 10*b^7*c^7 - 20*b^8*c^6 + 29*b^9*
c^5 - 30*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 6*a*b^3*c^10 + 27*a*b^4*c^9
 - 72*a*b^5*c^8 + 154*a*b^6*c^7 - 238*a*b^7*c^6 + 251*a*b^8*c^5 - 228*a*b^9
*c^4 + 98*a*b^10*c^3 + 20*a*b^11*c^2 + 8*a^2*b*c^11 - 68*a^3*b*c^10 + 112*a
^4*b*c^9 + 100*a^5*b*c^8 - 200*a^6*b*c^7 - 96*a^7*b*c^6 - 47*a^2*b^2*c^10 +
 145*a^2*b^3*c^9 - 354*a^2*b^4*c^8 + 612*a^2*b^5*c^7 - 655*a^2*b^6*c^6 + 63
5*a^2*b^7*c^5 - 202*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 + 239*a^
3*b^2*c^9 - 524*a^3*b^3*c^8 + 536*a^3*b^4*c^7 - 564*a^3*b^5*c^6 - 115*a^3*b
^6*c^5 + 856*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 37*a^4*b^2*c^8
+ 9*a^4*b^3*c^7 + 583*a^4*b^4*c^6 - 1362*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 15
6*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 399*a^5*b^2*c^7 + 904*a^5*b^3*c^6 + 394*a^5
*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 340*a^6*b^2*c^6 + 364*a^6*b^3
*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^2*c^5))/c^8 + (((2048*(16*a^3*c^13 - 32*
a^2*c^14 + 176*a^4*c^12 + 176*a^5*c^11 + 48*a^6*c^10 - 2*b^4*c^12 + 6*b^5*c
```

```
^11 - 18*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 + 16*a*b^2*c^13 - 40*a*b^3*c^12
 + 122*a*b^4*c^11 - 192*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 + 64*a^2*b
*c^13 - 144*a^3*b*c^12 - 352*a^4*b*c^11 - 144*a^5*b*c^10 - 204*a^2*b^2*c^12
 + 388*a^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 2
60*a^3*b^2*c^11 + 496*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 148*
a^4*b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9))/c^8 - (20
48*tan(x/2)*(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)
^3)^(1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^6*c^
2 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*
a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2
)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c
 - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32
*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32
*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^1
5 - 128*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 3
2*b^4*c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*
a*b^3*c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14
 + 352*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*
a^2*b^4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*
b^4*c^10 - 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^8)*(-(a^2*b^8 - b^10 + 8*a^5*c
^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4
*a*c - b^2)^3)^(1/2) - 52*a^2*b^6*c^2 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + 3
3*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3
)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2
)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c -
 b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c
^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*
c^7)))^(1/2) - (2048*tan(x/2)*(8*a*c^14 - 64*a^2*c^13 + 80*a^3*c^12 + 168*a
^4*c^11 - 192*a^5*c^10 - 136*a^6*c^9 + 72*a^7*c^8 - 2*b^2*c^13 + 6*b^3*c^12
 - 17*b^4*c^11 + 33*b^5*c^10 - 49*b^6*c^9 + 61*b^7*c^8 - 52*b^8*c^7 + 36*b^
9*c^6 - 24*b^10*c^5 + 8*b^11*c^4 + 84*a*b^2*c^12 - 178*a*b^3*c^11 + 295*a*b
^4*c^10 - 416*a*b^5*c^9 + 375*a*b^6*c^8 - 308*a*b^7*c^7 + 244*a*b^8*c^6 - 7
2*a*b^9*c^5 - 8*a*b^10*c^4 + 184*a^2*b*c^12 - 328*a^3*b*c^11 - 16*a^4*b*c^1
0 + 496*a^5*b*c^9 - 88*a^6*b*c^8 - 416*a^2*b^2*c^11 + 770*a^2*b^3*c^10 - 72
3*a^2*b^4*c^9 + 779*a^2*b^5*c^8 - 732*a^2*b^6*c^7 + 80*a^2*b^7*c^6 + 112*a^
2*b^8*c^5 - 8*a^2*b^9*c^4 + 180*a^3*b^2*c^10 - 494*a^3*b^3*c^9 + 521*a^3*b^
4*c^8 + 572*a^3*b^5*c^7 - 424*a^3*b^6*c^6 + 56*a^3*b^7*c^5 + 8*a^3*b^8*c^4
+ 234*a^4*b^2*c^9 - 1152*a^4*b^3*c^8 + 416*a^4*b^4*c^7 - 140*a^4*b^5*c^6 -
72*a^4*b^6*c^5 + 64*a^5*b^2*c^8 + 192*a^5*b^3*c^7 + 220*a^5*b^4*c^6 - 256*a
^6*b^2*c^7 - 24*a*b*c^13))/c^8)*(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 -
 b^7*(-(4*a*c - b^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(
1/2) - 52*a^2*b^6*c^2 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 -
38*a^5*b^2*c^3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*
b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*
a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/
```

(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9
 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2))*(-
(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) - 10
*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^6*c^2 + 96*a^3*b^4
*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8*c + 4*a^
3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3
*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/
2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*
a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 +
 a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(20*a*b^12 + 4*b^12*
c - 4*b^13 - 40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 + 2*a^4*c^9
 - 18*a^5*c^8 + 38*a^6*c^7 + 2*a^7*c^6 - 44*a^8*c^5 + 12*a^9*c^4 + b^8*c^5
- b^9*c^4 + 4*b^10*c^3 - 4*b^11*c^2 - 8*a*b^6*c^6 + 4*a*b^7*c^5 - 31*a*b^8*
c^4 + 20*a*b^9*c^3 - 20*a*b^10*c^2 - 160*a^2*b^10*c + 320*a^3*b^9*c + 26*a^
4*b*c^8 - 300*a^4*b^8*c - 84*a^5*b*c^7 + 136*a^5*b^7*c + 2*a^6*b*c^6 - 24*a
^6*b^6*c + 168*a^7*b*c^5 - 92*a^8*b*c^4 + 20*a^2*b^4*c^7 + 8*a^2*b^5*c^6 +
82*a^2*b^6*c^5 + 6*a^2*b^7*c^4 + 8*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 16*a^3*b^
2*c^8 - 40*a^3*b^3*c^7 - 104*a^3*b^4*c^6 - 132*a^3*b^5*c^5 + 34*a^3*b^6*c^4
 + 72*a^3*b^7*c^3 + 460*a^3*b^8*c^2 + 82*a^4*b^2*c^7 + 174*a^4*b^3*c^6 + 41
*a^4*b^4*c^5 - 149*a^4*b^5*c^4 - 660*a^4*b^6*c^3 - 900*a^4*b^7*c^2 - 90*a^5
*b^2*c^6 + 96*a^5*b^3*c^5 + 541*a^5*b^4*c^4 + 1156*a^5*b^5*c^3 + 764*a^5*b^
6*c^2 - 204*a^6*b^2*c^5 - 704*a^6*b^3*c^4 - 840*a^6*b^4*c^3 - 300*a^6*b^5*c
^2 + 384*a^7*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 +
24*a*b^11*c))/c^8)*(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c
 - b^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2
*b^6*c^2 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^
3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*
c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(
-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^
10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c
^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)*1i - (((2048*(12
*a^3*c^11 - 28*a^4*c^10 - 44*a^5*c^9 + 72*a^6*c^8 + 88*a^7*c^7 + 12*a^8*c^6
 + b^5*c^9 - 4*b^6*c^8 + 10*b^7*c^7 - 20*b^8*c^6 + 29*b^9*c^5 - 30*b^10*c^4
 + 26*b^11*c^3 - 12*b^12*c^2 - 6*a*b^3*c^10 + 27*a*b^4*c^9 - 72*a*b^5*c^8 +
 154*a*b^6*c^7 - 238*a*b^7*c^6 + 251*a*b^8*c^5 - 228*a*b^9*c^4 + 98*a*b^10*
c^3 + 20*a*b^11*c^2 + 8*a^2*b*c^11 - 68*a^3*b*c^10 + 112*a^4*b*c^9 + 100*a^
5*b*c^8 - 200*a^6*b*c^7 - 96*a^7*b*c^6 - 47*a^2*b^2*c^10 + 145*a^2*b^3*c^9
- 354*a^2*b^4*c^8 + 612*a^2*b^5*c^7 - 655*a^2*b^6*c^6 + 635*a^2*b^7*c^5 - 2
02*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 + 239*a^3*b^2*c^9 - 524*a
^3*b^3*c^8 + 536*a^3*b^4*c^7 - 564*a^3*b^5*c^6 - 115*a^3*b^6*c^5 + 856*a^3*
b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 37*a^4*b^2*c^8 + 9*a^4*b^3*c^7 +
 583*a^4*b^4*c^6 - 1362*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 156*a^4*b^7*c^3 + 8
*a^4*b^8*c^2 - 399*a^5*b^2*c^7 + 904*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 388*a^
5*b^5*c^4 - 60*a^5*b^6*c^3 - 340*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a^6*b^
4*c^4 - 100*a^7*b^2*c^5))/c^8 + (((2048*(16*a^3*c^13 - 32*a^2*c^14 + 176*a^

4*c^12 + 176*a^5*c^11 + 48*a^6*c^10 - 2*b^4*c^12 + 6*b^5*c^11 - 18*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 + 16*a*b^2*c^13 - 40*a*b^3*c^12 + 122*a*b^4*c^11 - 192*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 + 64*a^2*b*c^13 - 144*a^3*b *c^12 - 352*a^4*b*c^11 - 144*a^5*b*c^10 - 204*a^2*b^2*c^12 + 388*a^2*b^3*c^ 11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 260*a^3*b^2*c^11 + 496*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 148*a^4*b^2*c^10 + 11 6*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9))/c^8 + (2048*tan(x/2)*(-(a^ 2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) - 10*a^ 3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^6*c^2 + 96*a^3*b^4*c^ 3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8*c + 4*a^3*b *c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^ 4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4 *c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^ 2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b ^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184* a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8* a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4 *b^2*c^11 - 96*a*b*c^15))/c^8)*(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1 /2) - 52*a^2*b^6*c^2 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 3 8*a^5*b^2*c^3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b ^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a ^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2) + (2 048*tan(x/2)*(8*a*c^14 - 64*a^2*c^13 + 80*a^3*c^12 + 168*a^4*c^11 - 192*a^5 *c^10 - 136*a^6*c^9 + 72*a^7*c^8 - 2*b^2*c^13 + 6*b^3*c^12 - 17*b^4*c^11 + 33*b^5*c^10 - 49*b^6*c^9 + 61*b^7*c^8 - 52*b^8*c^7 + 36*b^9*c^6 - 24*b^10*c ^5 + 8*b^11*c^4 + 84*a*b^2*c^12 - 178*a*b^3*c^11 + 295*a*b^4*c^10 - 416*a*b ^5*c^9 + 375*a*b^6*c^8 - 308*a*b^7*c^7 + 244*a*b^8*c^6 - 72*a*b^9*c^5 - 8*a *b^10*c^4 + 184*a^2*b*c^12 - 328*a^3*b*c^11 - 16*a^4*b*c^10 + 496*a^5*b*c^9 - 88*a^6*b*c^8 - 416*a^2*b^2*c^11 + 770*a^2*b^3*c^10 - 723*a^2*b^4*c^9 + 7 79*a^2*b^5*c^8 - 732*a^2*b^6*c^7 + 80*a^2*b^7*c^6 + 112*a^2*b^8*c^5 - 8*a^2 *b^9*c^4 + 180*a^3*b^2*c^10 - 494*a^3*b^3*c^9 + 521*a^3*b^4*c^8 + 572*a^3*b ^5*c^7 - 424*a^3*b^6*c^6 + 56*a^3*b^7*c^5 + 8*a^3*b^8*c^4 + 234*a^4*b^2*c^9 - 1152*a^4*b^3*c^8 + 416*a^4*b^4*c^7 - 140*a^4*b^5*c^6 - 72*a^4*b^6*c^5 + 64*a^5*b^2*c^8 + 192*a^5*b^3*c^7 + 220*a^5*b^4*c^6 - 256*a^6*b^2*c^7 - 24*a *b*c^13))/c^8)*(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b ^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^6 *c^2 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*

```
a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 +
 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 -
 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2))*(-(a^2*b^8 - b^10 +
 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*
b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^6*c^2 + 96*a^3*b^4*c^3 - 66*a^4*b^2
*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c
- b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a
*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-
(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8
 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*
a^3*b^2*c^7)))^(1/2) + (2048*tan(x/2)*(20*a*b^12 + 4*b^12*c - 4*b^13 - 40*a
^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 + 2*a^4*c^9 - 18*a^5*c^8 + 3
8*a^6*c^7 + 2*a^7*c^6 - 44*a^8*c^5 + 12*a^9*c^4 + b^8*c^5 - b^9*c^4 + 4*b^1
0*c^3 - 4*b^11*c^2 - 8*a*b^6*c^6 + 4*a*b^7*c^5 - 31*a*b^8*c^4 + 20*a*b^9*c^
3 - 20*a*b^10*c^2 - 160*a^2*b^10*c + 320*a^3*b^9*c + 26*a^4*b*c^8 - 300*a^4
*b^8*c - 84*a^5*b*c^7 + 136*a^5*b^7*c + 2*a^6*b*c^6 - 24*a^6*b^6*c + 168*a^
7*b*c^5 - 92*a^8*b*c^4 + 20*a^2*b^4*c^7 + 8*a^2*b^5*c^6 + 82*a^2*b^6*c^5 +
6*a^2*b^7*c^4 + 8*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 16*a^3*b^2*c^8 - 40*a^3*b^
3*c^7 - 104*a^3*b^4*c^6 - 132*a^3*b^5*c^5 + 34*a^3*b^6*c^4 + 72*a^3*b^7*c^3
 + 460*a^3*b^8*c^2 + 82*a^4*b^2*c^7 + 174*a^4*b^3*c^6 + 41*a^4*b^4*c^5 - 14
9*a^4*b^5*c^4 - 660*a^4*b^6*c^3 - 900*a^4*b^7*c^2 - 90*a^5*b^2*c^6 + 96*a^5
*b^3*c^5 + 541*a^5*b^4*c^4 + 1156*a^5*b^5*c^3 + 764*a^5*b^6*c^2 - 204*a^6*b
^2*c^5 - 704*a^6*b^3*c^4 - 840*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 384*a^7*b^2*
c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 24*a*b^11*c))/c^8
)*(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2)
- 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^6*c^2 + 96*a^3
*b^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8*c +
4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2)
 + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)
^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 +
 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c
^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)*1i)/((4096*(16*a^5*b^7 - 4*a^4*b^
8 - 24*a^6*b^6 + 16*a^7*b^5 - 4*a^8*b^4 + 3*a^6*c^6 - 10*a^7*c^5 + a^8*c^4
+ 14*a^9*c^3 + 4*a^4*b^7*c - 2*a^5*b*c^6 + 4*a^5*b^6*c + 6*a^6*b*c^5 - 40*a
^6*b^5*c + 4*a^7*b*c^4 + 56*a^7*b^4*c - 22*a^8*b*c^3 - 28*a^8*b^3*c + 12*a^
9*b*c^2 + 4*a^9*b^2*c + a^4*b^3*c^5 - a^4*b^4*c^4 + 4*a^4*b^5*c^3 - 4*a^4*b
^6*c^2 - a^5*b^2*c^5 - 8*a^5*b^3*c^4 + 10*a^6*b^2*c^4 - 4*a^6*b^3*c^3 - 8*a
^6*b^4*c^2 + 4*a^7*b^2*c^3 + 48*a^7*b^3*c^2 - 48*a^8*b^2*c^2))/c^8 + (((204
8*(12*a^3*c^11 - 28*a^4*c^10 - 44*a^5*c^9 + 72*a^6*c^8 + 88*a^7*c^7 + 12*a^
8*c^6 + b^5*c^9 - 4*b^6*c^8 + 10*b^7*c^7 - 20*b^8*c^6 + 29*b^9*c^5 - 30*b^1
0*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 6*a*b^3*c^10 + 27*a*b^4*c^9 - 72*a*b^5*
c^8 + 154*a*b^6*c^7 - 238*a*b^7*c^6 + 251*a*b^8*c^5 - 228*a*b^9*c^4 + 98*a*
b^10*c^3 + 20*a*b^11*c^2 + 8*a^2*b*c^11 - 68*a^3*b*c^10 + 112*a^4*b*c^9 + 1
00*a^5*b*c^8 - 200*a^6*b*c^7 - 96*a^7*b*c^6 - 47*a^2*b^2*c^10 + 145*a^2*b^3
*c^9 - 354*a^2*b^4*c^8 + 612*a^2*b^5*c^7 - 655*a^2*b^6*c^6 + 635*a^2*b^7*c^
```

5 - 202*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 + 239*a^3*b^2*c^9 - 524*a^3*b^3*c^8 + 536*a^3*b^4*c^7 - 564*a^3*b^5*c^6 - 115*a^3*b^6*c^5 + 856 *a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 37*a^4*b^2*c^8 + 9*a^4*b^3* c^7 + 583*a^4*b^4*c^6 - 1362*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 156*a^4*b^7*c^ 3 + 8*a^4*b^8*c^2 - 399*a^5*b^2*c^7 + 904*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 3 88*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 340*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a ^6*b^4*c^4 - 100*a^7*b^2*c^5))/c^8 + (((2048*(16*a^3*c^13 - 32*a^2*c^14 + 1 76*a^4*c^12 + 176*a^5*c^11 + 48*a^6*c^10 - 2*b^4*c^12 + 6*b^5*c^11 - 18*b^6 *c^10 + 26*b^7*c^9 - 12*b^8*c^8 + 16*a*b^2*c^13 - 40*a*b^3*c^12 + 122*a*b^4 *c^11 - 192*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 + 64*a^2*b*c^13 - 144* a^3*b*c^12 - 352*a^4*b*c^11 - 144*a^5*b*c^10 - 204*a^2*b^2*c^12 + 388*a^2*b ^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 260*a^3*b^2*c ^11 + 496*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 148*a^4*b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9))/c^8 - (2048*tan(x/2)* (-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^6*c^2 + 96*a^3*b ^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8*c + 4* a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 1 6*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3* c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b *c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 5 6*a^4*b^2*c^11 - 96*a*b*c^15))/c^8)*(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c ^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^ 3)^(1/2) - 52*a^2*b^6*c^2 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^ 2 - 38*a^5*b^2*c^3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4* a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/ 2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2 *c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(8*a*c^14 - 64*a^2*c^13 + 80*a^3*c^12 + 168*a^4*c^11 - 19 2*a^5*c^10 - 136*a^6*c^9 + 72*a^7*c^8 - 2*b^2*c^13 + 6*b^3*c^12 - 17*b^4*c^ 11 + 33*b^5*c^10 - 49*b^6*c^9 + 61*b^7*c^8 - 52*b^8*c^7 + 36*b^9*c^6 - 24*b ^10*c^5 + 8*b^11*c^4 + 84*a*b^2*c^12 - 178*a*b^3*c^11 + 295*a*b^4*c^10 - 41 6*a*b^5*c^9 + 375*a*b^6*c^8 - 308*a*b^7*c^7 + 244*a*b^8*c^6 - 72*a*b^9*c^5 - 8*a*b^10*c^4 + 184*a^2*b*c^12 - 328*a^3*b*c^11 - 16*a^4*b*c^10 + 496*a^5* b*c^9 - 88*a^6*b*c^8 - 416*a^2*b^2*c^11 + 770*a^2*b^3*c^10 - 723*a^2*b^4*c^ 9 + 779*a^2*b^5*c^8 - 732*a^2*b^6*c^7 + 80*a^2*b^7*c^6 + 112*a^2*b^8*c^5 - 8*a^2*b^9*c^4 + 180*a^3*b^2*c^10 - 494*a^3*b^3*c^9 + 521*a^3*b^4*c^8 + 572* a^3*b^5*c^7 - 424*a^3*b^6*c^6 + 56*a^3*b^7*c^5 + 8*a^3*b^8*c^4 + 234*a^4*b^

2*c^9 - 1152*a^4*b^3*c^8 + 416*a^4*b^4*c^7 - 140*a^4*b^5*c^6 - 72*a^4*b^6*c
^5 + 64*a^5*b^2*c^8 + 192*a^5*b^3*c^7 + 220*a^5*b^4*c^6 - 256*a^6*b^2*c^7 -
 24*a*b*c^13))/c^8)*(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*
c - b^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^
2*b^6*c^2 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c
^3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a
*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*
(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c
^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*
c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2))*(-(a^2*b^8 - b
^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) - 10*a^3*b^6*c +
 a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^6*c^2 + 96*a^3*b^4*c^3 - 66*a^
4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(4
*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(
-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5
c(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^
4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6
 - 8*a^3*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(20*a*b^12 + 4*b^12*c - 4*b^13 -
 40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 + 2*a^4*c^9 - 18*a^5*c^
8 + 38*a^6*c^7 + 2*a^7*c^6 - 44*a^8*c^5 + 12*a^9*c^4 + b^8*c^5 - b^9*c^4 +
4*b^10*c^3 - 4*b^11*c^2 - 8*a*b^6*c^6 + 4*a*b^7*c^5 - 31*a*b^8*c^4 + 20*a*b
^9*c^3 - 20*a*b^10*c^2 - 160*a^2*b^10*c + 320*a^3*b^9*c + 26*a^4*b*c^8 - 30
0*a^4*b^8*c - 84*a^5*b*c^7 + 136*a^5*b^7*c + 2*a^6*b*c^6 - 24*a^6*b^6*c + 1
68*a^7*b*c^5 - 92*a^8*b*c^4 + 20*a^2*b^4*c^7 + 8*a^2*b^5*c^6 + 82*a^2*b^6*c
^5 + 6*a^2*b^7*c^4 + 8*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 16*a^3*b^2*c^8 - 40*a
^3*b^3*c^7 - 104*a^3*b^4*c^6 - 132*a^3*b^5*c^5 + 34*a^3*b^6*c^4 + 72*a^3*b^
7*c^3 + 460*a^3*b^8*c^2 + 82*a^4*b^2*c^7 + 174*a^4*b^3*c^6 + 41*a^4*b^4*c^5
 - 149*a^4*b^5*c^4 - 660*a^4*b^6*c^3 - 900*a^4*b^7*c^2 - 90*a^5*b^2*c^6 + 9
6*a^5*b^3*c^5 + 541*a^5*b^4*c^4 + 1156*a^5*b^5*c^3 + 764*a^5*b^6*c^2 - 204*
a^6*b^2*c^5 - 704*a^6*b^3*c^4 - 840*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 384*a^7
*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 24*a*b^11*c)
)/c^8)*(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(
1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^6*c^2 + 9
6*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8
*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^
(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^
2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*
c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*
b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2) + (((2048*(12*a^3*c^11 - 28*
a^4*c^10 - 44*a^5*c^9 + 72*a^6*c^8 + 88*a^7*c^7 + 12*a^8*c^6 + b^5*c^9 - 4*
b^6*c^8 + 10*b^7*c^7 - 20*b^8*c^6 + 29*b^9*c^5 - 30*b^10*c^4 + 26*b^11*c^3
- 12*b^12*c^2 - 6*a*b^3*c^10 + 27*a*b^4*c^9 - 72*a*b^5*c^8 + 154*a*b^6*c^7
- 238*a*b^7*c^6 + 251*a*b^8*c^5 - 228*a*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^11
*c^2 + 8*a^2*b*c^11 - 68*a^3*b*c^10 + 112*a^4*b*c^9 + 100*a^5*b*c^8 - 200*a
^6*b*c^7 - 96*a^7*b*c^6 - 47*a^2*b^2*c^10 + 145*a^2*b^3*c^9 - 354*a^2*b^4*c

^8 + 612*a^2*b^5*c^7 - 655*a^2*b^6*c^6 + 635*a^2*b^7*c^5 - 202*a^2*b^8*c^4
- 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 + 239*a^3*b^2*c^9 - 524*a^3*b^3*c^8 + 53
6*a^3*b^4*c^7 - 564*a^3*b^5*c^6 - 115*a^3*b^6*c^5 + 856*a^3*b^7*c^4 + 2*a^3
*b^8*c^3 - 20*a^3*b^9*c^2 - 37*a^4*b^2*c^8 + 9*a^4*b^3*c^7 + 583*a^4*b^4*c^
6 - 1362*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^8*c^2 -
399*a^5*b^2*c^7 + 904*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*
a^5*b^6*c^3 - 340*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7
*b^2*c^5))/c^8 + (((2048*(16*a^3*c^13 - 32*a^2*c^14 + 176*a^4*c^12 + 176*a^
5*c^11 + 48*a^6*c^10 - 2*b^4*c^12 + 6*b^5*c^11 - 18*b^6*c^10 + 26*b^7*c^9 -
12*b^8*c^8 + 16*a*b^2*c^13 - 40*a*b^3*c^12 + 122*a*b^4*c^11 - 192*a*b^5*c^
10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 + 64*a^2*b*c^13 - 144*a^3*b*c^12 - 352*a^4
*b*c^11 - 144*a^5*b*c^10 - 204*a^2*b^2*c^12 + 388*a^2*b^3*c^11 - 50*a^2*b^4
*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 260*a^3*b^2*c^11 + 496*a^3*b^3*c^
10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 148*a^4*b^2*c^10 + 116*a^4*b^3*c^9 +
8*a^4*b^4*c^8 - 44*a^5*b^2*c^9))/c^8 + (2048*tan(x/2)*(-(a^2*b^8 - b^10 +
8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*b
^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^6*c^2 + 96*a^3*b^4*c^3 - 66*a^4*b^2*
c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c -
b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*
c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(
4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8
- b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a
^3*b^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13 +
96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b^
6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 56
*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^1
2 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 -
272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 96*
a*b*c^15))/c^8)*(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c -
b^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^
6*c^2 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 +
12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c -
b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4
*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10
+ 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7
- 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2) + (2048*tan(x/2)*(8
*a*c^14 - 64*a^2*c^13 + 80*a^3*c^12 + 168*a^4*c^11 - 192*a^5*c^10 - 136*a^6
*c^9 + 72*a^7*c^8 - 2*b^2*c^13 + 6*b^3*c^12 - 17*b^4*c^11 + 33*b^5*c^10 - 4
9*b^6*c^9 + 61*b^7*c^8 - 52*b^8*c^7 + 36*b^9*c^6 - 24*b^10*c^5 + 8*b^11*c^4
+ 84*a*b^2*c^12 - 178*a*b^3*c^11 + 295*a*b^4*c^10 - 416*a*b^5*c^9 + 375*a*
b^6*c^8 - 308*a*b^7*c^7 + 244*a*b^8*c^6 - 72*a*b^9*c^5 - 8*a*b^10*c^4 + 184
*a^2*b*c^12 - 328*a^3*b*c^11 - 16*a^4*b*c^10 + 496*a^5*b*c^9 - 88*a^6*b*c^8
- 416*a^2*b^2*c^11 + 770*a^2*b^3*c^10 - 723*a^2*b^4*c^9 + 779*a^2*b^5*c^8
- 732*a^2*b^6*c^7 + 80*a^2*b^7*c^6 + 112*a^2*b^8*c^5 - 8*a^2*b^9*c^4 + 180*
a^3*b^2*c^10 - 494*a^3*b^3*c^9 + 521*a^3*b^4*c^8 + 572*a^3*b^5*c^7 - 424*a^

3*b^6*c^6 + 56*a^3*b^7*c^5 + 8*a^3*b^8*c^4 + 234*a^4*b^2*c^9 - 1152*a^4*b^3
*c^8 + 416*a^4*b^4*c^7 - 140*a^4*b^5*c^6 - 72*a^4*b^6*c^5 + 64*a^5*b^2*c^8
+ 192*a^5*b^3*c^7 + 220*a^5*b^4*c^6 - 256*a^6*b^2*c^7 - 24*a*b*c^13))/c^8)*
(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) -
10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^6*c^2 + 96*a^3*b
^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8*c + 4*
a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) +
 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(
1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 1
6*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8
 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2))*(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*
a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4*a*c -
b^2)^3)^(1/2) - 52*a^2*b^6*c^2 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b
^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2)
 - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1
/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3
)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*
a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^
(1/2) + (2048*tan(x/2)*(20*a*b^12 + 4*b^12*c - 4*b^13 - 40*a^2*b^11 + 40*a^
3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 + 2*a^4*c^9 - 18*a^5*c^8 + 38*a^6*c^7 + 2*a
^7*c^6 - 44*a^8*c^5 + 12*a^9*c^4 + b^8*c^5 - b^9*c^4 + 4*b^10*c^3 - 4*b^11*
c^2 - 8*a*b^6*c^6 + 4*a*b^7*c^5 - 31*a*b^8*c^4 + 20*a*b^9*c^3 - 20*a*b^10*c
^2 - 160*a^2*b^10*c + 320*a^3*b^9*c + 26*a^4*b*c^8 - 300*a^4*b^8*c - 84*a^5
*b*c^7 + 136*a^5*b^7*c + 2*a^6*b*c^6 - 24*a^6*b^6*c + 168*a^7*b*c^5 - 92*a^
8*b*c^4 + 20*a^2*b^4*c^7 + 8*a^2*b^5*c^6 + 82*a^2*b^6*c^5 + 6*a^2*b^7*c^4 +
 8*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 16*a^3*b^2*c^8 - 40*a^3*b^3*c^7 - 104*a^3
*b^4*c^6 - 132*a^3*b^5*c^5 + 34*a^3*b^6*c^4 + 72*a^3*b^7*c^3 + 460*a^3*b^8*
c^2 + 82*a^4*b^2*c^7 + 174*a^4*b^3*c^6 + 41*a^4*b^4*c^5 - 149*a^4*b^5*c^4 -
 660*a^4*b^6*c^3 - 900*a^4*b^7*c^2 - 90*a^5*b^2*c^6 + 96*a^5*b^3*c^5 + 541*
a^5*b^4*c^4 + 1156*a^5*b^5*c^3 + 764*a^5*b^6*c^2 - 204*a^6*b^2*c^5 - 704*a^
6*b^3*c^4 - 840*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 384*a^7*b^2*c^4 + 272*a^7*b
^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 24*a*b^11*c))/c^8)*(-(a^2*b^8 -
b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) - 10*a^3*b^6*c
+ a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) - 52*a^2*b^6*c^2 + 96*a^3*b^4*c^3 - 66*a
^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*a^5*b^2*c^3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(
4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*
(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^
5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b
^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^
6 - 8*a^3*b^2*c^7)))^(1/2)))*(-(a^2*b^8 - b^10 + 8*a^5*c^5 + 8*a^6*c^4 - b^
7*(-(4*a*c - b^2)^3)^(1/2) - 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2
) - 52*a^2*b^6*c^2 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + 33*a^4*b^4*c^2 - 38*
a^5*b^2*c^3 + 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3
c(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2
*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*

(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 +
10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)*2i - (
(tan(x/2)*(2*b - c))/c^2 + (tan(x/2)^3*(2*b + c))/c^2)/(2*tan(x/2)^2 + tan(
x/2)^4 + 1) + atan(((((2048*(12*a^3*c^11 - 28*a^4*c^10 - 44*a^5*c^9 + 72*a^
6*c^8 + 88*a^7*c^7 + 12*a^8*c^6 + b^5*c^9 - 4*b^6*c^8 + 10*b^7*c^7 - 20*b^8
*c^6 + 29*b^9*c^5 - 30*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 6*a*b^3*c^10
+ 27*a*b^4*c^9 - 72*a*b^5*c^8 + 154*a*b^6*c^7 - 238*a*b^7*c^6 + 251*a*b^8*c
^5 - 228*a*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^11*c^2 + 8*a^2*b*c^11 - 68*a^3*
b*c^10 + 112*a^4*b*c^9 + 100*a^5*b*c^8 - 200*a^6*b*c^7 - 96*a^7*b*c^6 - 47*
a^2*b^2*c^10 + 145*a^2*b^3*c^9 - 354*a^2*b^4*c^8 + 612*a^2*b^5*c^7 - 655*a^
2*b^6*c^6 + 635*a^2*b^7*c^5 - 202*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^1
0*c^2 + 239*a^3*b^2*c^9 - 524*a^3*b^3*c^8 + 536*a^3*b^4*c^7 - 564*a^3*b^5*c
^6 - 115*a^3*b^6*c^5 + 856*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 3
7*a^4*b^2*c^8 + 9*a^4*b^3*c^7 + 583*a^4*b^4*c^6 - 1362*a^4*b^5*c^5 - 152*a^
4*b^6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 399*a^5*b^2*c^7 + 904*a^5*b^3
*c^6 + 394*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 340*a^6*b^2*c^6
 + 364*a^6*b^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^2*c^5))/c^8 + (((2048*(16*
a^3*c^13 - 32*a^2*c^14 + 176*a^4*c^12 + 176*a^5*c^11 + 48*a^6*c^10 - 2*b^4*
c^12 + 6*b^5*c^11 - 18*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 + 16*a*b^2*c^13 -
 40*a*b^3*c^12 + 122*a*b^4*c^11 - 192*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*
c^8 + 64*a^2*b*c^13 - 144*a^3*b*c^12 - 352*a^4*b*c^11 - 144*a^5*b*c^10 - 20
4*a^2*b^2*c^12 + 388*a^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a
^2*b^6*c^8 - 260*a^3*b^2*c^11 + 496*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*
b^5*c^8 - 148*a^4*b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c
^9))/c^8 - (2048*tan(x/2)*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-
(4*a*c - b^2)^3)^(1/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) +
52*a^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*
b^2*c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(
-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3
c^2(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*
a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a
*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)*(32*a*c^16
 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24
*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^
2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 2
88*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b
^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*
c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^8)*((b^10 - a^2*b
^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) + 10*a^3*b^6*c +
a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^4
*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4*
a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-
(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*
c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4
*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6

```
- 8*a^3*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(8*a*c^14 - 64*a^2*c^13 + 80*a^3*
c^12 + 168*a^4*c^11 - 192*a^5*c^10 - 136*a^6*c^9 + 72*a^7*c^8 - 2*b^2*c^13
+ 6*b^3*c^12 - 17*b^4*c^11 + 33*b^5*c^10 - 49*b^6*c^9 + 61*b^7*c^8 - 52*b^8
*c^7 + 36*b^9*c^6 - 24*b^10*c^5 + 8*b^11*c^4 + 84*a*b^2*c^12 - 178*a*b^3*c^
11 + 295*a*b^4*c^10 - 416*a*b^5*c^9 + 375*a*b^6*c^8 - 308*a*b^7*c^7 + 244*a
*b^8*c^6 - 72*a*b^9*c^5 - 8*a*b^10*c^4 + 184*a^2*b*c^12 - 328*a^3*b*c^11 -
16*a^4*b*c^10 + 496*a^5*b*c^9 - 88*a^6*b*c^8 - 416*a^2*b^2*c^11 + 770*a^2*b
^3*c^10 - 723*a^2*b^4*c^9 + 779*a^2*b^5*c^8 - 732*a^2*b^6*c^7 + 80*a^2*b^7*
c^6 + 112*a^2*b^8*c^5 - 8*a^2*b^9*c^4 + 180*a^3*b^2*c^10 - 494*a^3*b^3*c^9
+ 521*a^3*b^4*c^8 + 572*a^3*b^5*c^7 - 424*a^3*b^6*c^6 + 56*a^3*b^7*c^5 + 8*
a^3*b^8*c^4 + 234*a^4*b^2*c^9 - 1152*a^4*b^3*c^8 + 416*a^4*b^4*c^7 - 140*a^
4*b^5*c^6 - 72*a^4*b^6*c^5 + 64*a^5*b^2*c^8 + 192*a^5*b^3*c^7 + 220*a^5*b^4
*c^6 - 256*a^6*b^2*c^7 - 24*a*b*c^13))/c^8)*((b^10 - a^2*b^8 - 8*a^5*c^5 -
8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c
- b^2)^3)^(1/2) + 52*a^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4
*b^4*c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/
2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^
(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)
^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 -
8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7))
)^(1/2))*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^
(1/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a^2*b^6*c^2 -
96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*c^3 - 12*a*b^
8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)
^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b
^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3
*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2
*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(20*a*b^12
 + 4*b^12*c - 4*b^13 - 40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 +
 2*a^4*c^9 - 18*a^5*c^8 + 38*a^6*c^7 + 2*a^7*c^6 - 44*a^8*c^5 + 12*a^9*c^4
+ b^8*c^5 - b^9*c^4 + 4*b^10*c^3 - 4*b^11*c^2 - 8*a*b^6*c^6 + 4*a*b^7*c^5 -
 31*a*b^8*c^4 + 20*a*b^9*c^3 - 20*a*b^10*c^2 - 160*a^2*b^10*c + 320*a^3*b^9
*c + 26*a^4*b*c^8 - 300*a^4*b^8*c - 84*a^5*b*c^7 + 136*a^5*b^7*c + 2*a^6*b*
c^6 - 24*a^6*b^6*c + 168*a^7*b*c^5 - 92*a^8*b*c^4 + 20*a^2*b^4*c^7 + 8*a^2*
b^5*c^6 + 82*a^2*b^6*c^5 + 6*a^2*b^7*c^4 + 8*a^2*b^8*c^3 - 44*a^2*b^9*c^2 -
 16*a^3*b^2*c^8 - 40*a^3*b^3*c^7 - 104*a^3*b^4*c^6 - 132*a^3*b^5*c^5 + 34*a
^3*b^6*c^4 + 72*a^3*b^7*c^3 + 460*a^3*b^8*c^2 + 82*a^4*b^2*c^7 + 174*a^4*b^
3*c^6 + 41*a^4*b^4*c^5 - 149*a^4*b^5*c^4 - 660*a^4*b^6*c^3 - 900*a^4*b^7*c^
2 - 90*a^5*b^2*c^6 + 96*a^5*b^3*c^5 + 541*a^5*b^4*c^4 + 1156*a^5*b^5*c^3 +
764*a^5*b^6*c^2 - 204*a^6*b^2*c^5 - 704*a^6*b^3*c^4 - 840*a^6*b^4*c^3 - 300
*a^6*b^5*c^2 + 384*a^7*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*
b^2*c^3 + 24*a*b^11*c))/c^8)*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7
*(-(4*a*c - b^2)^3)^(1/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2)
 + 52*a^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a
^5*b^2*c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*
```

```
c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*
b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(
16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 1
0*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)*1i - ((
(2048*(12*a^3*c^11 - 28*a^4*c^10 - 44*a^5*c^9 + 72*a^6*c^8 + 88*a^7*c^7 + 1
2*a^8*c^6 + b^5*c^9 - 4*b^6*c^8 + 10*b^7*c^7 - 20*b^8*c^6 + 29*b^9*c^5 - 30
*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 6*a*b^3*c^10 + 27*a*b^4*c^9 - 72*a*
b^5*c^8 + 154*a*b^6*c^7 - 238*a*b^7*c^6 + 251*a*b^8*c^5 - 228*a*b^9*c^4 + 9
8*a*b^10*c^3 + 20*a*b^11*c^2 + 8*a^2*b*c^11 - 68*a^3*b*c^10 + 112*a^4*b*c^9
 + 100*a^5*b*c^8 - 200*a^6*b*c^7 - 96*a^7*b*c^6 - 47*a^2*b^2*c^10 + 145*a^2
*b^3*c^9 - 354*a^2*b^4*c^8 + 612*a^2*b^5*c^7 - 655*a^2*b^6*c^6 + 635*a^2*b^
7*c^5 - 202*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 + 239*a^3*b^2*c^
9 - 524*a^3*b^3*c^8 + 536*a^3*b^4*c^7 - 564*a^3*b^5*c^6 - 115*a^3*b^6*c^5 +
 856*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 37*a^4*b^2*c^8 + 9*a^4*
b^3*c^7 + 583*a^4*b^4*c^6 - 1362*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 156*a^4*b^
7*c^3 + 8*a^4*b^8*c^2 - 399*a^5*b^2*c^7 + 904*a^5*b^3*c^6 + 394*a^5*b^4*c^5
 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 340*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 1
36*a^6*b^4*c^4 - 100*a^7*b^2*c^5))/c^8 + (((2048*(16*a^3*c^13 - 32*a^2*c^14
 + 176*a^4*c^12 + 176*a^5*c^11 + 48*a^6*c^10 - 2*b^4*c^12 + 6*b^5*c^11 - 18
*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 + 16*a*b^2*c^13 - 40*a*b^3*c^12 + 122*a
*b^4*c^11 - 192*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 + 64*a^2*b*c^13 -
144*a^3*b*c^12 - 352*a^4*b*c^11 - 144*a^5*b*c^10 - 204*a^2*b^2*c^12 + 388*a
^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 260*a^3*b
^2*c^11 + 496*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 148*a^4*b^2*
c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9))/c^8 + (2048*tan(x
/2)*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2)
 + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a^2*b^6*c^2 - 96*a^
3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*c +
 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2
) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3
)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9
+ 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*
c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a
^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^1
3 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^1
3 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^
3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c
^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10
- 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^8)*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6
*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2
)^3)^(1/2) + 52*a^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*
c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) -
4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2)
 - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(
1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b
```

^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/
2) + (2048*tan(x/2)*(8*a*c^14 - 64*a^2*c^13 + 80*a^3*c^12 + 168*a^4*c^11 -
192*a^5*c^10 - 136*a^6*c^9 + 72*a^7*c^8 - 2*b^2*c^13 + 6*b^3*c^12 - 17*b^4*
c^11 + 33*b^5*c^10 - 49*b^6*c^9 + 61*b^7*c^8 - 52*b^8*c^7 + 36*b^9*c^6 - 24
*b^10*c^5 + 8*b^11*c^4 + 84*a*b^2*c^12 - 178*a*b^3*c^11 + 295*a*b^4*c^10 -
416*a*b^5*c^9 + 375*a*b^6*c^8 - 308*a*b^7*c^7 + 244*a*b^8*c^6 - 72*a*b^9*c^
5 - 8*a*b^10*c^4 + 184*a^2*b*c^12 - 328*a^3*b*c^11 - 16*a^4*b*c^10 + 496*a^
5*b*c^9 - 88*a^6*b*c^8 - 416*a^2*b^2*c^11 + 770*a^2*b^3*c^10 - 723*a^2*b^4*
c^9 + 779*a^2*b^5*c^8 - 732*a^2*b^6*c^7 + 80*a^2*b^7*c^6 + 112*a^2*b^8*c^5
- 8*a^2*b^9*c^4 + 180*a^3*b^2*c^10 - 494*a^3*b^3*c^9 + 521*a^3*b^4*c^8 + 57
2*a^3*b^5*c^7 - 424*a^3*b^6*c^6 + 56*a^3*b^7*c^5 + 8*a^3*b^8*c^4 + 234*a^4*
b^2*c^9 - 1152*a^4*b^3*c^8 + 416*a^4*b^4*c^7 - 140*a^4*b^5*c^6 - 72*a^4*b^6
*c^5 + 64*a^5*b^2*c^8 + 192*a^5*b^3*c^7 + 220*a^5*b^4*c^6 - 256*a^6*b^2*c^7
- 24*a*b*c^13))/c^8)*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a
*c - b^2)^3)^(1/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a
^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*
c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*
a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2
*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*
c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4
*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2))*((b^10 - a^2*
b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) + 10*a^3*b^6*c +
a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^
4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4
*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(
-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5
c(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^
4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6
- 8*a^3*b^2*c^7)))^(1/2) + (2048*tan(x/2)*(20*a*b^12 + 4*b^12*c - 4*b^13 -
40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 + 2*a^4*c^9 - 18*a^5*c^
8 + 38*a^6*c^7 + 2*a^7*c^6 - 44*a^8*c^5 + 12*a^9*c^4 + b^8*c^5 - b^9*c^4 +
4*b^10*c^3 - 4*b^11*c^2 - 8*a*b^6*c^6 + 4*a*b^7*c^5 - 31*a*b^8*c^4 + 20*a*b
^9*c^3 - 20*a*b^10*c^2 - 160*a^2*b^10*c + 320*a^3*b^9*c + 26*a^4*b*c^8 - 30
0*a^4*b^8*c - 84*a^5*b*c^7 + 136*a^5*b^7*c + 2*a^6*b*c^6 - 24*a^6*b^6*c + 1
68*a^7*b*c^5 - 92*a^8*b*c^4 + 20*a^2*b^4*c^7 + 8*a^2*b^5*c^6 + 82*a^2*b^6*c
^5 + 6*a^2*b^7*c^4 + 8*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 16*a^3*b^2*c^8 - 40*a
^3*b^3*c^7 - 104*a^3*b^4*c^6 - 132*a^3*b^5*c^5 + 34*a^3*b^6*c^4 + 72*a^3*b^
7*c^3 + 460*a^3*b^8*c^2 + 82*a^4*b^2*c^7 + 174*a^4*b^3*c^6 + 41*a^4*b^4*c^5
- 149*a^4*b^5*c^4 - 660*a^4*b^6*c^3 - 900*a^4*b^7*c^2 - 90*a^5*b^2*c^6 + 9
6*a^5*b^3*c^5 + 541*a^5*b^4*c^4 + 1156*a^5*b^5*c^3 + 764*a^5*b^6*c^2 - 204*
a^6*b^2*c^5 - 704*a^6*b^3*c^4 - 840*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 384*a^7
*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 24*a*b^11*c)
)/c^8)*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1
/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a^2*b^6*c^2 - 96
*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*

c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(
1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2
)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c
^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b
^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)*1i)/((4096*(16*a^5*b^7 - 4*a^
4*b^8 - 24*a^6*b^6 + 16*a^7*b^5 - 4*a^8*b^4 + 3*a^6*c^6 - 10*a^7*c^5 + a^8*
c^4 + 14*a^9*c^3 + 4*a^4*b^7*c - 2*a^5*b*c^6 + 4*a^5*b^6*c + 6*a^6*b*c^5 -
40*a^6*b^5*c + 4*a^7*b*c^4 + 56*a^7*b^4*c - 22*a^8*b*c^3 - 28*a^8*b^3*c + 1
2*a^9*b*c^2 + 4*a^9*b^2*c + a^4*b^3*c^5 - a^4*b^4*c^4 + 4*a^4*b^5*c^3 - 4*a
^4*b^6*c^2 - a^5*b^2*c^5 - 8*a^5*b^3*c^4 + 10*a^6*b^2*c^4 - 4*a^6*b^3*c^3 -
8*a^6*b^4*c^2 + 4*a^7*b^2*c^3 + 48*a^7*b^3*c^2 - 48*a^8*b^2*c^2))/c^8 + ((
(2048*(12*a^3*c^11 - 28*a^4*c^10 - 44*a^5*c^9 + 72*a^6*c^8 + 88*a^7*c^7 + 1
2*a^8*c^6 + b^5*c^9 - 4*b^6*c^8 + 10*b^7*c^7 - 20*b^8*c^6 + 29*b^9*c^5 - 30
*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 6*a*b^3*c^10 + 27*a*b^4*c^9 - 72*a*
b^5*c^8 + 154*a*b^6*c^7 - 238*a*b^7*c^6 + 251*a*b^8*c^5 - 228*a*b^9*c^4 + 9
8*a*b^10*c^3 + 20*a*b^11*c^2 + 8*a^2*b*c^11 - 68*a^3*b*c^10 + 112*a^4*b*c^9
+ 100*a^5*b*c^8 - 200*a^6*b*c^7 - 96*a^7*b*c^6 - 47*a^2*b^2*c^10 + 145*a^2
*b^3*c^9 - 354*a^2*b^4*c^8 + 612*a^2*b^5*c^7 - 655*a^2*b^6*c^6 + 635*a^2*b^
7*c^5 - 202*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 + 239*a^3*b^2*c^
9 - 524*a^3*b^3*c^8 + 536*a^3*b^4*c^7 - 564*a^3*b^5*c^6 - 115*a^3*b^6*c^5 +
856*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 37*a^4*b^2*c^8 + 9*a^4*
b^3*c^7 + 583*a^4*b^4*c^6 - 1362*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 156*a^4*b^
7*c^3 + 8*a^4*b^8*c^2 - 399*a^5*b^2*c^7 + 904*a^5*b^3*c^6 + 394*a^5*b^4*c^5
- 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 340*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 1
36*a^6*b^4*c^4 - 100*a^7*b^2*c^5))/c^8 + (((2048*(16*a^3*c^13 - 32*a^2*c^14
+ 176*a^4*c^12 + 176*a^5*c^11 + 48*a^6*c^10 - 2*b^4*c^12 + 6*b^5*c^11 - 18
*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 + 16*a*b^2*c^13 - 40*a*b^3*c^12 + 122*a
*b^4*c^11 - 192*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 + 64*a^2*b*c^13 -
144*a^3*b*c^12 - 352*a^4*b*c^11 - 144*a^5*b*c^10 - 204*a^2*b^2*c^12 + 388*a
^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 260*a^3*b
^2*c^11 + 496*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 148*a^4*b^2*
c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9))/c^8 - (2048*tan(x
/2)*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2)
+ 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a^2*b^6*c^2 - 96*a^
3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*c +
4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2
) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3
)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9
+ 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*
c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a
^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^1
3 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^1
3 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^
3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c
^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10

- 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^8)*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6
c^4 - b^7(-(4*a*c - b^2)^3)^(1/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2
)^3)^(1/2) + 52*a^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*
c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) -
4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2)
 - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(
1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b
^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/
2) - (2048*tan(x/2)*(8*a*c^14 - 64*a^2*c^13 + 80*a^3*c^12 + 168*a^4*c^11 -
192*a^5*c^10 - 136*a^6*c^9 + 72*a^7*c^8 - 2*b^2*c^13 + 6*b^3*c^12 - 17*b^4*
c^11 + 33*b^5*c^10 - 49*b^6*c^9 + 61*b^7*c^8 - 52*b^8*c^7 + 36*b^9*c^6 - 24
*b^10*c^5 + 8*b^11*c^4 + 84*a*b^2*c^12 - 178*a*b^3*c^11 + 295*a*b^4*c^10 -
416*a*b^5*c^9 + 375*a*b^6*c^8 - 308*a*b^7*c^7 + 244*a*b^8*c^6 - 72*a*b^9*c^
5 - 8*a*b^10*c^4 + 184*a^2*b*c^12 - 328*a^3*b*c^11 - 16*a^4*b*c^10 + 496*a^
5*b*c^9 - 88*a^6*b*c^8 - 416*a^2*b^2*c^11 + 770*a^2*b^3*c^10 - 723*a^2*b^4*
c^9 + 779*a^2*b^5*c^8 - 732*a^2*b^6*c^7 + 80*a^2*b^7*c^6 + 112*a^2*b^8*c^5
- 8*a^2*b^9*c^4 + 180*a^3*b^2*c^10 - 494*a^3*b^3*c^9 + 521*a^3*b^4*c^8 + 57
2*a^3*b^5*c^7 - 424*a^3*b^6*c^6 + 56*a^3*b^7*c^5 + 8*a^3*b^8*c^4 + 234*a^4*
b^2*c^9 - 1152*a^4*b^3*c^8 + 416*a^4*b^4*c^7 - 140*a^4*b^5*c^6 - 72*a^4*b^6
*c^5 + 64*a^5*b^2*c^8 + 192*a^5*b^3*c^7 + 220*a^5*b^4*c^6 - 256*a^6*b^2*c^7
 - 24*a*b*c^13))/c^8)*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a
*c - b^2)^3)^(1/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a
^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*
c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*
a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2
*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*
c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4
*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2))*((b^10 - a^2*
b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) + 10*a^3*b^6*c +
 a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^
4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4
*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(
-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5
c(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^
4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6
 - 8*a^3*b^2*c^7)))^(1/2) - (2048*tan(x/2)*(20*a*b^12 + 4*b^12*c - 4*b^13 -
 40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 + 2*a^4*c^9 - 18*a^5*c^
8 + 38*a^6*c^7 + 2*a^7*c^6 - 44*a^8*c^5 + 12*a^9*c^4 + b^8*c^5 - b^9*c^4 +
4*b^10*c^3 - 4*b^11*c^2 - 8*a*b^6*c^6 + 4*a*b^7*c^5 - 31*a*b^8*c^4 + 20*a*b
^9*c^3 - 20*a*b^10*c^2 - 160*a^2*b^10*c + 320*a^3*b^9*c + 26*a^4*b*c^8 - 30
0*a^4*b^8*c - 84*a^5*b*c^7 + 136*a^5*b^7*c + 2*a^6*b*c^6 - 24*a^6*b^6*c + 1
68*a^7*b*c^5 - 92*a^8*b*c^4 + 20*a^2*b^4*c^7 + 8*a^2*b^5*c^6 + 82*a^2*b^6*c
^5 + 6*a^2*b^7*c^4 + 8*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 16*a^3*b^2*c^8 - 40*a
^3*b^3*c^7 - 104*a^3*b^4*c^6 - 132*a^3*b^5*c^5 + 34*a^3*b^6*c^4 + 72*a^3*b^
7*c^3 + 460*a^3*b^8*c^2 + 82*a^4*b^2*c^7 + 174*a^4*b^3*c^6 + 41*a^4*b^4*c^5

- 149*a^4*b^5*c^4 - 660*a^4*b^6*c^3 - 900*a^4*b^7*c^2 - 90*a^5*b^2*c^6 + 9
6*a^5*b^3*c^5 + 541*a^5*b^4*c^4 + 1156*a^5*b^5*c^3 + 764*a^5*b^6*c^2 - 204*
a^6*b^2*c^5 - 704*a^6*b^3*c^4 - 840*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 384*a^7
*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 24*a*b^11*c)
)/c^8)*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1
/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a^2*b^6*c^2 - 96
*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*
c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(
1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2
)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c
^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b
^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2) + (((2048*(12*a^3*c^11 - 28*a
^4*c^10 - 44*a^5*c^9 + 72*a^6*c^8 + 88*a^7*c^7 + 12*a^8*c^6 + b^5*c^9 - 4*b
^6*c^8 + 10*b^7*c^7 - 20*b^8*c^6 + 29*b^9*c^5 - 30*b^10*c^4 + 26*b^11*c^3 -
12*b^12*c^2 - 6*a*b^3*c^10 + 27*a*b^4*c^9 - 72*a*b^5*c^8 + 154*a*b^6*c^7 -
238*a*b^7*c^6 + 251*a*b^8*c^5 - 228*a*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^11*
c^2 + 8*a^2*b*c^11 - 68*a^3*b*c^10 + 112*a^4*b*c^9 + 100*a^5*b*c^8 - 200*a^
6*b*c^7 - 96*a^7*b*c^6 - 47*a^2*b^2*c^10 + 145*a^2*b^3*c^9 - 354*a^2*b^4*c^
8 + 612*a^2*b^5*c^7 - 655*a^2*b^6*c^6 + 635*a^2*b^7*c^5 - 202*a^2*b^8*c^4 -
222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 + 239*a^3*b^2*c^9 - 524*a^3*b^3*c^8 + 536
*a^3*b^4*c^7 - 564*a^3*b^5*c^6 - 115*a^3*b^6*c^5 + 856*a^3*b^7*c^4 + 2*a^3*
b^8*c^3 - 20*a^3*b^9*c^2 - 37*a^4*b^2*c^8 + 9*a^4*b^3*c^7 + 583*a^4*b^4*c^6
- 1362*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 3
99*a^5*b^2*c^7 + 904*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a
^5*b^6*c^3 - 340*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*
b^2*c^5))/c^8 + (((2048*(16*a^3*c^13 - 32*a^2*c^14 + 176*a^4*c^12 + 176*a^5
*c^11 + 48*a^6*c^10 - 2*b^4*c^12 + 6*b^5*c^11 - 18*b^6*c^10 + 26*b^7*c^9 -
12*b^8*c^8 + 16*a*b^2*c^13 - 40*a*b^3*c^12 + 122*a*b^4*c^11 - 192*a*b^5*c^1
0 + 74*a*b^6*c^9 + 20*a*b^7*c^8 + 64*a^2*b*c^13 - 144*a^3*b*c^12 - 352*a^4*
b*c^11 - 144*a^5*b*c^10 - 204*a^2*b^2*c^12 + 388*a^2*b^3*c^11 - 50*a^2*b^4*
c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 260*a^3*b^2*c^11 + 496*a^3*b^3*c^1
0 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 148*a^4*b^2*c^10 + 116*a^4*b^3*c^9 +
8*a^4*b^4*c^8 - 44*a^5*b^2*c^9))/c^8 + (2048*tan(x/2)*((b^10 - a^2*b^8 - 8*
a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) + 10*a^3*b^6*c + a^2*b^5
*(-(4*a*c - b^2)^3)^(1/2) + 52*a^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^
4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b
^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c
- b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*
a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 -
b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3
*b^2*c^7)))^(1/2)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13 + 9
6*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b^6*
c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 56*a
*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^12
- 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 - 27

```
2*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 96*a*
b*c^15))/c^8)*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a*c - b^2
)^3)^(1/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a^2*b^6*c
^2 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*c^3 - 12
*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^
2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*
c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 3
2*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 3
2*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2) + (2048*tan(x/2)*(8*a*
c^14 - 64*a^2*c^13 + 80*a^3*c^12 + 168*a^4*c^11 - 192*a^5*c^10 - 136*a^6*c^
9 + 72*a^7*c^8 - 2*b^2*c^13 + 6*b^3*c^12 - 17*b^4*c^11 + 33*b^5*c^10 - 49*b
^6*c^9 + 61*b^7*c^8 - 52*b^8*c^7 + 36*b^9*c^6 - 24*b^10*c^5 + 8*b^11*c^4 +
84*a*b^2*c^12 - 178*a*b^3*c^11 + 295*a*b^4*c^10 - 416*a*b^5*c^9 + 375*a*b^6
*c^8 - 308*a*b^7*c^7 + 244*a*b^8*c^6 - 72*a*b^9*c^5 - 8*a*b^10*c^4 + 184*a^
2*b*c^12 - 328*a^3*b*c^11 - 16*a^4*b*c^10 + 496*a^5*b*c^9 - 88*a^6*b*c^8 -
416*a^2*b^2*c^11 + 770*a^2*b^3*c^10 - 723*a^2*b^4*c^9 + 779*a^2*b^5*c^8 - 7
32*a^2*b^6*c^7 + 80*a^2*b^7*c^6 + 112*a^2*b^8*c^5 - 8*a^2*b^9*c^4 + 180*a^3
*b^2*c^10 - 494*a^3*b^3*c^9 + 521*a^3*b^4*c^8 + 572*a^3*b^5*c^7 - 424*a^3*b
^6*c^6 + 56*a^3*b^7*c^5 + 8*a^3*b^8*c^4 + 234*a^4*b^2*c^9 - 1152*a^4*b^3*c^
8 + 416*a^4*b^4*c^7 - 140*a^4*b^5*c^6 - 72*a^4*b^6*c^5 + 64*a^5*b^2*c^8 + 1
92*a^5*b^3*c^7 + 220*a^5*b^4*c^6 - 256*a^6*b^2*c^7 - 24*a*b*c^13))/c^8)*((b
^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) + 10*a
^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a^2*b^6*c^2 - 96*a^3*b^4*c
^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*c + 4*a^3*
b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a
^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2)
 + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^
4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a
^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2))*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c
^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^
3)^(1/2) + 52*a^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^
2 + 38*a^5*b^2*c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*
a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) -
 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/
2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2
*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)
 + (2048*tan(x/2)*(20*a*b^12 + 4*b^12*c - 4*b^13 - 40*a^2*b^11 + 40*a^3*b^1
0 - 20*a^4*b^9 + 4*a^5*b^8 + 2*a^4*c^9 - 18*a^5*c^8 + 38*a^6*c^7 + 2*a^7*c^
6 - 44*a^8*c^5 + 12*a^9*c^4 + b^8*c^5 - b^9*c^4 + 4*b^10*c^3 - 4*b^11*c^2 -
 8*a*b^6*c^6 + 4*a*b^7*c^5 - 31*a*b^8*c^4 + 20*a*b^9*c^3 - 20*a*b^10*c^2 -
160*a^2*b^10*c + 320*a^3*b^9*c + 26*a^4*b*c^8 - 300*a^4*b^8*c - 84*a^5*b*c^
7 + 136*a^5*b^7*c + 2*a^6*b*c^6 - 24*a^6*b^6*c + 168*a^7*b*c^5 - 92*a^8*b*c
^4 + 20*a^2*b^4*c^7 + 8*a^2*b^5*c^6 + 82*a^2*b^6*c^5 + 6*a^2*b^7*c^4 + 8*a^
2*b^8*c^3 - 44*a^2*b^9*c^2 - 16*a^3*b^2*c^8 - 40*a^3*b^3*c^7 - 104*a^3*b^4*
c^6 - 132*a^3*b^5*c^5 + 34*a^3*b^6*c^4 + 72*a^3*b^7*c^3 + 460*a^3*b^8*c^2 +
```

82*a^4*b^2*c^7 + 174*a^4*b^3*c^6 + 41*a^4*b^4*c^5 - 149*a^4*b^5*c^4 - 660*
a^4*b^6*c^3 - 900*a^4*b^7*c^2 - 90*a^5*b^2*c^6 + 96*a^5*b^3*c^5 + 541*a^5*b
^4*c^4 + 1156*a^5*b^5*c^3 + 764*a^5*b^6*c^2 - 204*a^6*b^2*c^5 - 704*a^6*b^3
*c^4 - 840*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 384*a^7*b^2*c^4 + 272*a^7*b^3*c^
3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 24*a*b^11*c))/c^8)*((b^10 - a^2*b^8 -
8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*a*c - b^2)^3)^(1/2) + 10*a^3*b^6*c + a^2*
b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*a^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^4*b^2
*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2*c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c
- b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a
*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-
(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8
- b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*
a^3*b^2*c^7)))^(1/2)))*((b^10 - a^2*b^8 - 8*a^5*c^5 - 8*a^6*c^4 - b^7*(-(4*
a*c - b^2)^3)^(1/2) + 10*a^3*b^6*c + a^2*b^5*(-(4*a*c - b^2)^3)^(1/2) + 52*
a^2*b^6*c^2 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 - 33*a^4*b^4*c^2 + 38*a^5*b^2
*c^3 - 12*a*b^8*c + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a^3*b^3*c*(-(4
*a*c - b^2)^3)^(1/2) + 3*a^4*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^
2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2
*c^10 + 32*a^3*c^9 + 16*a^4*c^8 + b^4*c^8 - b^6*c^6 - 8*a*b^2*c^9 + 10*a*b^
4*c^7 - 32*a^2*b^2*c^8 + a^2*b^4*c^6 - 8*a^3*b^2*c^7)))^(1/2)*2i + (atan(-(
(((2048*tan(x/2)*(20*a*b^12 + 4*b^12*c - 4*b^13 - 40*a^2*b^11 + 40*a^3*b^10
- 20*a^4*b^9 + 4*a^5*b^8 + 2*a^4*c^9 - 18*a^5*c^8 + 38*a^6*c^7 + 2*a^7*c^6
- 44*a^8*c^5 + 12*a^9*c^4 + b^8*c^5 - b^9*c^4 + 4*b^10*c^3 - 4*b^11*c^2 -
8*a*b^6*c^6 + 4*a*b^7*c^5 - 31*a*b^8*c^4 + 20*a*b^9*c^3 - 20*a*b^10*c^2 - 1
60*a^2*b^10*c + 320*a^3*b^9*c + 26*a^4*b*c^8 - 300*a^4*b^8*c - 84*a^5*b*c^7
+ 136*a^5*b^7*c + 2*a^6*b*c^6 - 24*a^6*b^6*c + 168*a^7*b*c^5 - 92*a^8*b*c^
4 + 20*a^2*b^4*c^7 + 8*a^2*b^5*c^6 + 82*a^2*b^6*c^5 + 6*a^2*b^7*c^4 + 8*a^2
*b^8*c^3 - 44*a^2*b^9*c^2 - 16*a^3*b^2*c^8 - 40*a^3*b^3*c^7 - 104*a^3*b^4*c
^6 - 132*a^3*b^5*c^5 + 34*a^3*b^6*c^4 + 72*a^3*b^7*c^3 + 460*a^3*b^8*c^2 +
82*a^4*b^2*c^7 + 174*a^4*b^3*c^6 + 41*a^4*b^4*c^5 - 149*a^4*b^5*c^4 - 660*a
^4*b^6*c^3 - 900*a^4*b^7*c^2 - 90*a^5*b^2*c^6 + 96*a^5*b^3*c^5 + 541*a^5*b^
4*c^4 + 1156*a^5*b^5*c^3 + 764*a^5*b^6*c^2 - 204*a^6*b^2*c^5 - 704*a^6*b^3*
c^4 - 840*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 384*a^7*b^2*c^4 + 272*a^7*b^3*c^3
+ 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 24*a*b^11*c))/c^8 - (((2048*(12*a^3*c^
11 - 28*a^4*c^10 - 44*a^5*c^9 + 72*a^6*c^8 + 88*a^7*c^7 + 12*a^8*c^6 + b^5*
c^9 - 4*b^6*c^8 + 10*b^7*c^7 - 20*b^8*c^6 + 29*b^9*c^5 - 30*b^10*c^4 + 26*b
^11*c^3 - 12*b^12*c^2 - 6*a*b^3*c^10 + 27*a*b^4*c^9 - 72*a*b^5*c^8 + 154*a*
b^6*c^7 - 238*a*b^7*c^6 + 251*a*b^8*c^5 - 228*a*b^9*c^4 + 98*a*b^10*c^3 + 2
0*a*b^11*c^2 + 8*a^2*b*c^11 - 68*a^3*b*c^10 + 112*a^4*b*c^9 + 100*a^5*b*c^8
- 200*a^6*b*c^7 - 96*a^7*b*c^6 - 47*a^2*b^2*c^10 + 145*a^2*b^3*c^9 - 354*a
^2*b^4*c^8 + 612*a^2*b^5*c^7 - 655*a^2*b^6*c^6 + 635*a^2*b^7*c^5 - 202*a^2*
b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 + 239*a^3*b^2*c^9 - 524*a^3*b^3*
c^8 + 536*a^3*b^4*c^7 - 564*a^3*b^5*c^6 - 115*a^3*b^6*c^5 + 856*a^3*b^7*c^4
+ 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 37*a^4*b^2*c^8 + 9*a^4*b^3*c^7 + 583*a^
4*b^4*c^6 - 1362*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^

8*c^2 - 399*a^5*b^2*c^7 + 904*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 388*a^5*b^5*c
^4 - 60*a^5*b^6*c^3 - 340*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a^6*b^4*c^4 -
 100*a^7*b^2*c^5))/c^8 - (((2048*tan(x/2)*(8*a*c^14 - 64*a^2*c^13 + 80*a^3*
c^12 + 168*a^4*c^11 - 192*a^5*c^10 - 136*a^6*c^9 + 72*a^7*c^8 - 2*b^2*c^13
+ 6*b^3*c^12 - 17*b^4*c^11 + 33*b^5*c^10 - 49*b^6*c^9 + 61*b^7*c^8 - 52*b^8
*c^7 + 36*b^9*c^6 - 24*b^10*c^5 + 8*b^11*c^4 + 84*a*b^2*c^12 - 178*a*b^3*c^
11 + 295*a*b^4*c^10 - 416*a*b^5*c^9 + 375*a*b^6*c^8 - 308*a*b^7*c^7 + 244*a
*b^8*c^6 - 72*a*b^9*c^5 - 8*a*b^10*c^4 + 184*a^2*b*c^12 - 328*a^3*b*c^11 -
16*a^4*b*c^10 + 496*a^5*b*c^9 - 88*a^6*b*c^8 - 416*a^2*b^2*c^11 + 770*a^2*b
^3*c^10 - 723*a^2*b^4*c^9 + 779*a^2*b^5*c^8 - 732*a^2*b^6*c^7 + 80*a^2*b^7*
c^6 + 112*a^2*b^8*c^5 - 8*a^2*b^9*c^4 + 180*a^3*b^2*c^10 - 494*a^3*b^3*c^9
+ 521*a^3*b^4*c^8 + 572*a^3*b^5*c^7 - 424*a^3*b^6*c^6 + 56*a^3*b^7*c^5 + 8*
a^3*b^8*c^4 + 234*a^4*b^2*c^9 - 1152*a^4*b^3*c^8 + 416*a^4*b^4*c^7 - 140*a^
4*b^5*c^6 - 72*a^4*b^6*c^5 + 64*a^5*b^2*c^8 + 192*a^5*b^3*c^7 + 220*a^5*b^4
*c^6 - 256*a^6*b^2*c^7 - 24*a*b*c^13))/c^8 - (((2048*(16*a^3*c^13 - 32*a^2*
c^14 + 176*a^4*c^12 + 176*a^5*c^11 + 48*a^6*c^10 - 2*b^4*c^12 + 6*b^5*c^11
- 18*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 + 16*a*b^2*c^13 - 40*a*b^3*c^12 + 1
22*a*b^4*c^11 - 192*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8 + 64*a^2*b*c^1
3 - 144*a^3*b*c^12 - 352*a^4*b*c^11 - 144*a^5*b*c^10 - 204*a^2*b^2*c^12 + 3
88*a^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 260*a
^3*b^2*c^11 + 496*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 148*a^4*
b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9))/c^8 - (1024*t
an(x/2)*(b^2*2i - a*c*2i + c^2*1i)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14
+ 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b
^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184*
a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13
 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*
a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4
*b^2*c^11 - 96*a*b*c^15))/c^11)*(b^2*2i - a*c*2i + c^2*1i))/(2*c^3))*(b^2*2
i - a*c*2i + c^2*1i))/(2*c^3))*(b^2*2i - a*c*2i + c^2*1i))/(2*c^3))*(b^2*2i
 - a*c*2i + c^2*1i)*1i)/(2*c^3) + (((2048*tan(x/2)*(20*a*b^12 + 4*b^12*c -
4*b^13 - 40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 + 2*a^4*c^9 - 1
8*a^5*c^8 + 38*a^6*c^7 + 2*a^7*c^6 - 44*a^8*c^5 + 12*a^9*c^4 + b^8*c^5 - b^
9*c^4 + 4*b^10*c^3 - 4*b^11*c^2 - 8*a*b^6*c^6 + 4*a*b^7*c^5 - 31*a*b^8*c^4
+ 20*a*b^9*c^3 - 20*a*b^10*c^2 - 160*a^2*b^10*c + 320*a^3*b^9*c + 26*a^4*b*
c^8 - 300*a^4*b^8*c - 84*a^5*b*c^7 + 136*a^5*b^7*c + 2*a^6*b*c^6 - 24*a^6*b
^6*c + 168*a^7*b*c^5 - 92*a^8*b*c^4 + 20*a^2*b^4*c^7 + 8*a^2*b^5*c^6 + 82*a
^2*b^6*c^5 + 6*a^2*b^7*c^4 + 8*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 16*a^3*b^2*c^
8 - 40*a^3*b^3*c^7 - 104*a^3*b^4*c^6 - 132*a^3*b^5*c^5 + 34*a^3*b^6*c^4 + 7
2*a^3*b^7*c^3 + 460*a^3*b^8*c^2 + 82*a^4*b^2*c^7 + 174*a^4*b^3*c^6 + 41*a^4
*b^4*c^5 - 149*a^4*b^5*c^4 - 660*a^4*b^6*c^3 - 900*a^4*b^7*c^2 - 90*a^5*b^2
*c^6 + 96*a^5*b^3*c^5 + 541*a^5*b^4*c^4 + 1156*a^5*b^5*c^3 + 764*a^5*b^6*c^
2 - 204*a^6*b^2*c^5 - 704*a^6*b^3*c^4 - 840*a^6*b^4*c^3 - 300*a^6*b^5*c^2 +
 384*a^7*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 24*a
*b^11*c))/c^8 + (((2048*(12*a^3*c^11 - 28*a^4*c^10 - 44*a^5*c^9 + 72*a^6*c^

8 + 88*a^7*c^7 + 12*a^8*c^6 + b^5*c^9 - 4*b^6*c^8 + 10*b^7*c^7 - 20*b^8*c^6 + 29*b^9*c^5 - 30*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 6*a*b^3*c^10 + 27 *a*b^4*c^9 - 72*a*b^5*c^8 + 154*a*b^6*c^7 - 238*a*b^7*c^6 + 251*a*b^8*c^5 - 228*a*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^11*c^2 + 8*a^2*b*c^11 - 68*a^3*b*c^ 10 + 112*a^4*b*c^9 + 100*a^5*b*c^8 - 200*a^6*b*c^7 - 96*a^7*b*c^6 - 47*a^2* b^2*c^10 + 145*a^2*b^3*c^9 - 354*a^2*b^4*c^8 + 612*a^2*b^5*c^7 - 655*a^2*b^ 6*c^6 + 635*a^2*b^7*c^5 - 202*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^ 2 + 239*a^3*b^2*c^9 - 524*a^3*b^3*c^8 + 536*a^3*b^4*c^7 - 564*a^3*b^5*c^6 - 115*a^3*b^6*c^5 + 856*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 37*a^ 4*b^2*c^8 + 9*a^4*b^3*c^7 + 583*a^4*b^4*c^6 - 1362*a^4*b^5*c^5 - 152*a^4*b^ 6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 399*a^5*b^2*c^7 + 904*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 340*a^6*b^2*c^6 + 3 64*a^6*b^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^2*c^5))/c^8 + (((2048*tan(x/2) *(8*a*c^14 - 64*a^2*c^13 + 80*a^3*c^12 + 168*a^4*c^11 - 192*a^5*c^10 - 136* a^6*c^9 + 72*a^7*c^8 - 2*b^2*c^13 + 6*b^3*c^12 - 17*b^4*c^11 + 33*b^5*c^10 - 49*b^6*c^9 + 61*b^7*c^8 - 52*b^8*c^7 + 36*b^9*c^6 - 24*b^10*c^5 + 8*b^11* c^4 + 84*a*b^2*c^12 - 178*a*b^3*c^11 + 295*a*b^4*c^10 - 416*a*b^5*c^9 + 375 *a*b^6*c^8 - 308*a*b^7*c^7 + 244*a*b^8*c^6 - 72*a*b^9*c^5 - 8*a*b^10*c^4 + 184*a^2*b*c^12 - 328*a^3*b*c^11 - 16*a^4*b*c^10 + 496*a^5*b*c^9 - 88*a^6*b* c^8 - 416*a^2*b^2*c^11 + 770*a^2*b^3*c^10 - 723*a^2*b^4*c^9 + 779*a^2*b^5*c ^8 - 732*a^2*b^6*c^7 + 80*a^2*b^7*c^6 + 112*a^2*b^8*c^5 - 8*a^2*b^9*c^4 + 1 80*a^3*b^2*c^10 - 494*a^3*b^3*c^9 + 521*a^3*b^4*c^8 + 572*a^3*b^5*c^7 - 424 *a^3*b^6*c^6 + 56*a^3*b^7*c^5 + 8*a^3*b^8*c^4 + 234*a^4*b^2*c^9 - 1152*a^4* b^3*c^8 + 416*a^4*b^4*c^7 - 140*a^4*b^5*c^6 - 72*a^4*b^6*c^5 + 64*a^5*b^2*c ^8 + 192*a^5*b^3*c^7 + 220*a^5*b^4*c^6 - 256*a^6*b^2*c^7 - 24*a*b*c^13))/c^ 8 + (((2048*(16*a^3*c^13 - 32*a^2*c^14 + 176*a^4*c^12 + 176*a^5*c^11 + 48*a ^6*c^10 - 2*b^4*c^12 + 6*b^5*c^11 - 18*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 + 16*a*b^2*c^13 - 40*a*b^3*c^12 + 122*a*b^4*c^11 - 192*a*b^5*c^10 + 74*a*b^6 *c^9 + 20*a*b^7*c^8 + 64*a^2*b*c^13 - 144*a^3*b*c^12 - 352*a^4*b*c^11 - 144 *a^5*b*c^10 - 204*a^2*b^2*c^12 + 388*a^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a ^2*b^5*c^9 + 4*a^2*b^6*c^8 - 260*a^3*b^2*c^11 + 496*a^3*b^3*c^10 + 10*a^3*b ^4*c^9 - 20*a^3*b^5*c^8 - 148*a^4*b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^ 8 - 44*a^5*b^2*c^9))/c^8 + (1024*tan(x/2)*(b^2*2i - a*c*2i + c^2*1i)*(32*a* c^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144* a*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a ^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3* b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^11)*(b^2*2i - a*c*2i + c^2*1i))/(2*c^3))*(b^2*2i - a*c*2i + c^2*1i))/(2*c^3))*(b^2*2i - a*c*2i + c^2*1i))/(2*c^3))*(b^2*2i - a*c*2i + c^2*1i)*1i)/(2*c^3))/((4096*(16*a^5*b^7 - 4*a^4*b^8 - 24*a^6*b^6 + 16*a^7*b^5 - 4*a^8*b^4 + 3*a^6*c^6 - 10*a^7*c^5 + a^8*c^4 + 14*a^9*c^3 + 4*a^4*b^7*c - 2*a^5*b*c^6 + 4*a^5*b^6*c + 6*a^6*b*c^5 - 40*a^6*b^5*c + 4*a^7*b*c^4 + 56*a^7*b^4*c - 22*a^8*b*c^3 - 28*a^8*b^3*c + 12*a^9*b*c^2 + 4*a^9*b^2*c + a^4*b^3*c^5 - a^4*b^4*c^4 + 4*

a^4*b^5*c^3 - 4*a^4*b^6*c^2 - a^5*b^2*c^5 - 8*a^5*b^3*c^4 + 10*a^6*b^2*c^4 - 4*a^6*b^3*c^3 - 8*a^6*b^4*c^2 + 4*a^7*b^2*c^3 + 48*a^7*b^3*c^2 - 48*a^8*b ^2*c^2))/c^8 - (((2048*tan(x/2)*(20*a*b^12 + 4*b^12*c - 4*b^13 - 40*a^2*b^1 1 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 + 2*a^4*c^9 - 18*a^5*c^8 + 38*a^6* c^7 + 2*a^7*c^6 - 44*a^8*c^5 + 12*a^9*c^4 + b^8*c^5 - b^9*c^4 + 4*b^10*c^3 - 4*b^11*c^2 - 8*a*b^6*c^6 + 4*a*b^7*c^5 - 31*a*b^8*c^4 + 20*a*b^9*c^3 - 20 *a*b^10*c^2 - 160*a^2*b^10*c + 320*a^3*b^9*c + 26*a^4*b*c^8 - 300*a^4*b^8*c - 84*a^5*b*c^7 + 136*a^5*b^7*c + 2*a^6*b*c^6 - 24*a^6*b^6*c + 168*a^7*b*c^ 5 - 92*a^8*b*c^4 + 20*a^2*b^4*c^7 + 8*a^2*b^5*c^6 + 82*a^2*b^6*c^5 + 6*a^2* b^7*c^4 + 8*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 16*a^3*b^2*c^8 - 40*a^3*b^3*c^7 - 104*a^3*b^4*c^6 - 132*a^3*b^5*c^5 + 34*a^3*b^6*c^4 + 72*a^3*b^7*c^3 + 460 *a^3*b^8*c^2 + 82*a^4*b^2*c^7 + 174*a^4*b^3*c^6 + 41*a^4*b^4*c^5 - 149*a^4* b^5*c^4 - 660*a^4*b^6*c^3 - 900*a^4*b^7*c^2 - 90*a^5*b^2*c^6 + 96*a^5*b^3*c ^5 + 541*a^5*b^4*c^4 + 1156*a^5*b^5*c^3 + 764*a^5*b^6*c^2 - 204*a^6*b^2*c^5 - 704*a^6*b^3*c^4 - 840*a^6*b^4*c^3 - 300*a^6*b^5*c^2 + 384*a^7*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^2*c^3 + 24*a*b^11*c))/c^8 - (((2048*(12*a^3*c^11 - 28*a^4*c^10 - 44*a^5*c^9 + 72*a^6*c^8 + 88*a^7*c^7 + 12 *a^8*c^6 + b^5*c^9 - 4*b^6*c^8 + 10*b^7*c^7 - 20*b^8*c^6 + 29*b^9*c^5 - 30* b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 6*a*b^3*c^10 + 27*a*b^4*c^9 - 72*a*b ^5*c^8 + 154*a*b^6*c^7 - 238*a*b^7*c^6 + 251*a*b^8*c^5 - 228*a*b^9*c^4 + 98 *a*b^10*c^3 + 20*a*b^11*c^2 + 8*a^2*b*c^11 - 68*a^3*b*c^10 + 112*a^4*b*c^9 + 100*a^5*b*c^8 - 200*a^6*b*c^7 - 96*a^7*b*c^6 - 47*a^2*b^2*c^10 + 145*a^2* b^3*c^9 - 354*a^2*b^4*c^8 + 612*a^2*b^5*c^7 - 655*a^2*b^6*c^6 + 635*a^2*b^7 *c^5 - 202*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4*a^2*b^10*c^2 + 239*a^3*b^2*c^9 - 524*a^3*b^3*c^8 + 536*a^3*b^4*c^7 - 564*a^3*b^5*c^6 - 115*a^3*b^6*c^5 + 856*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9*c^2 - 37*a^4*b^2*c^8 + 9*a^4*b ^3*c^7 + 583*a^4*b^4*c^6 - 1362*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 156*a^4*b^7 *c^3 + 8*a^4*b^8*c^2 - 399*a^5*b^2*c^7 + 904*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 340*a^6*b^2*c^6 + 364*a^6*b^3*c^5 + 13 6*a^6*b^4*c^4 - 100*a^7*b^2*c^5))/c^8 - (((2048*tan(x/2)*(8*a*c^14 - 64*a^2 *c^13 + 80*a^3*c^12 + 168*a^4*c^11 - 192*a^5*c^10 - 136*a^6*c^9 + 72*a^7*c^ 8 - 2*b^2*c^13 + 6*b^3*c^12 - 17*b^4*c^11 + 33*b^5*c^10 - 49*b^6*c^9 + 61*b ^7*c^8 - 52*b^8*c^7 + 36*b^9*c^6 - 24*b^10*c^5 + 8*b^11*c^4 + 84*a*b^2*c^12 - 178*a*b^3*c^11 + 295*a*b^4*c^10 - 416*a*b^5*c^9 + 375*a*b^6*c^8 - 308*a* b^7*c^7 + 244*a*b^8*c^6 - 72*a*b^9*c^5 - 8*a*b^10*c^4 + 184*a^2*b*c^12 - 32 8*a^3*b*c^11 - 16*a^4*b*c^10 + 496*a^5*b*c^9 - 88*a^6*b*c^8 - 416*a^2*b^2*c ^11 + 770*a^2*b^3*c^10 - 723*a^2*b^4*c^9 + 779*a^2*b^5*c^8 - 732*a^2*b^6*c^ 7 + 80*a^2*b^7*c^6 + 112*a^2*b^8*c^5 - 8*a^2*b^9*c^4 + 180*a^3*b^2*c^10 - 4 94*a^3*b^3*c^9 + 521*a^3*b^4*c^8 + 572*a^3*b^5*c^7 - 424*a^3*b^6*c^6 + 56*a ^3*b^7*c^5 + 8*a^3*b^8*c^4 + 234*a^4*b^2*c^9 - 1152*a^4*b^3*c^8 + 416*a^4*b ^4*c^7 - 140*a^4*b^5*c^6 - 72*a^4*b^6*c^5 + 64*a^5*b^2*c^8 + 192*a^5*b^3*c^ 7 + 220*a^5*b^4*c^6 - 256*a^6*b^2*c^7 - 24*a*b*c^13))/c^8 - (((2048*(16*a^3 *c^13 - 32*a^2*c^14 + 176*a^4*c^12 + 176*a^5*c^11 + 48*a^6*c^10 - 2*b^4*c^1 2 + 6*b^5*c^11 - 18*b^6*c^10 + 26*b^7*c^9 - 12*b^8*c^8 + 16*a*b^2*c^13 - 40 *a*b^3*c^12 + 122*a*b^4*c^11 - 192*a*b^5*c^10 + 74*a*b^6*c^9 + 20*a*b^7*c^8

```
 + 64*a^2*b*c^13 - 144*a^3*b*c^12 - 352*a^4*b*c^11 - 144*a^5*b*c^10 - 204*a
^2*b^2*c^12 + 388*a^2*b^3*c^11 - 50*a^2*b^4*c^10 - 182*a^2*b^5*c^9 + 4*a^2*
b^6*c^8 - 260*a^3*b^2*c^11 + 496*a^3*b^3*c^10 + 10*a^3*b^4*c^9 - 20*a^3*b^5
*c^8 - 148*a^4*b^2*c^10 + 116*a^4*b^3*c^9 + 8*a^4*b^4*c^8 - 44*a^5*b^2*c^9)
)/c^8 - (1024*tan(x/2)*(b^2*2i - a*c*2i + c^2*1i)*(32*a*c^16 - 64*a^2*c^15
- 128*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 - 8*b^2*c^15 + 24*b^3*c^14 - 32*
b^4*c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7*c^10 + 144*a*b^2*c^14 - 200*a*
b^3*c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 - 8*a*b^6*c^10 + 288*a^2*b*c^14 +
 352*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^2*c^13 + 8*a^2*b^3*c^12 + 96*a^
2*b^4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^12 + 40*a^3*b^3*c^11 + 8*a^3*b^
4*c^10 - 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^11)*(b^2*2i - a*c*2i + c^2*1i))/
(2*c^3))*(b^2*2i - a*c*2i + c^2*1i))/(2*c^3))*(b^2*2i - a*c*2i + c^2*1i))/(
2*c^3))*(b^2*2i - a*c*2i + c^2*1i))/(2*c^3) + (((2048*tan(x/2)*(20*a*b^12 +
 4*b^12*c - 4*b^13 - 40*a^2*b^11 + 40*a^3*b^10 - 20*a^4*b^9 + 4*a^5*b^8 + 2
*a^4*c^9 - 18*a^5*c^8 + 38*a^6*c^7 + 2*a^7*c^6 - 44*a^8*c^5 + 12*a^9*c^4 +
b^8*c^5 - b^9*c^4 + 4*b^10*c^3 - 4*b^11*c^2 - 8*a*b^6*c^6 + 4*a*b^7*c^5 - 3
1*a*b^8*c^4 + 20*a*b^9*c^3 - 20*a*b^10*c^2 - 160*a^2*b^10*c + 320*a^3*b^9*c
 + 26*a^4*b*c^8 - 300*a^4*b^8*c - 84*a^5*b*c^7 + 136*a^5*b^7*c + 2*a^6*b*c^
6 - 24*a^6*b^6*c + 168*a^7*b*c^5 - 92*a^8*b*c^4 + 20*a^2*b^4*c^7 + 8*a^2*b^
5*c^6 + 82*a^2*b^6*c^5 + 6*a^2*b^7*c^4 + 8*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 1
6*a^3*b^2*c^8 - 40*a^3*b^3*c^7 - 104*a^3*b^4*c^6 - 132*a^3*b^5*c^5 + 34*a^3
*b^6*c^4 + 72*a^3*b^7*c^3 + 460*a^3*b^8*c^2 + 82*a^4*b^2*c^7 + 174*a^4*b^3*
c^6 + 41*a^4*b^4*c^5 - 149*a^4*b^5*c^4 - 660*a^4*b^6*c^3 - 900*a^4*b^7*c^2
- 90*a^5*b^2*c^6 + 96*a^5*b^3*c^5 + 541*a^5*b^4*c^4 + 1156*a^5*b^5*c^3 + 76
4*a^5*b^6*c^2 - 204*a^6*b^2*c^5 - 704*a^6*b^3*c^4 - 840*a^6*b^4*c^3 - 300*a
^6*b^5*c^2 + 384*a^7*b^2*c^4 + 272*a^7*b^3*c^3 + 44*a^7*b^4*c^2 - 32*a^8*b^
2*c^3 + 24*a*b^11*c))/c^8 + (((2048*(12*a^3*c^11 - 28*a^4*c^10 - 44*a^5*c^9
 + 72*a^6*c^8 + 88*a^7*c^7 + 12*a^8*c^6 + b^5*c^9 - 4*b^6*c^8 + 10*b^7*c^7
- 20*b^8*c^6 + 29*b^9*c^5 - 30*b^10*c^4 + 26*b^11*c^3 - 12*b^12*c^2 - 6*a*b
^3*c^10 + 27*a*b^4*c^9 - 72*a*b^5*c^8 + 154*a*b^6*c^7 - 238*a*b^7*c^6 + 251
*a*b^8*c^5 - 228*a*b^9*c^4 + 98*a*b^10*c^3 + 20*a*b^11*c^2 + 8*a^2*b*c^11 -
 68*a^3*b*c^10 + 112*a^4*b*c^9 + 100*a^5*b*c^8 - 200*a^6*b*c^7 - 96*a^7*b*c
^6 - 47*a^2*b^2*c^10 + 145*a^2*b^3*c^9 - 354*a^2*b^4*c^8 + 612*a^2*b^5*c^7
- 655*a^2*b^6*c^6 + 635*a^2*b^7*c^5 - 202*a^2*b^8*c^4 - 222*a^2*b^9*c^3 + 4
*a^2*b^10*c^2 + 239*a^3*b^2*c^9 - 524*a^3*b^3*c^8 + 536*a^3*b^4*c^7 - 564*a
^3*b^5*c^6 - 115*a^3*b^6*c^5 + 856*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 20*a^3*b^9
*c^2 - 37*a^4*b^2*c^8 + 9*a^4*b^3*c^7 + 583*a^4*b^4*c^6 - 1362*a^4*b^5*c^5
- 152*a^4*b^6*c^4 + 156*a^4*b^7*c^3 + 8*a^4*b^8*c^2 - 399*a^5*b^2*c^7 + 904
*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 388*a^5*b^5*c^4 - 60*a^5*b^6*c^3 - 340*a^6
*b^2*c^6 + 364*a^6*b^3*c^5 + 136*a^6*b^4*c^4 - 100*a^7*b^2*c^5))/c^8 + (((2
048*tan(x/2)*(8*a*c^14 - 64*a^2*c^13 + 80*a^3*c^12 + 168*a^4*c^11 - 192*a^5
*c^10 - 136*a^6*c^9 + 72*a^7*c^8 - 2*b^2*c^13 + 6*b^3*c^12 - 17*b^4*c^11 +
33*b^5*c^10 - 49*b^6*c^9 + 61*b^7*c^8 - 52*b^8*c^7 + 36*b^9*c^6 - 24*b^10*c
^5 + 8*b^11*c^4 + 84*a*b^2*c^12 - 178*a*b^3*c^11 + 295*a*b^4*c^10 - 416*a*b
^5*c^9 + 375*a*b^6*c^8 - 308*a*b^7*c^7 + 244*a*b^8*c^6 - 72*a*b^9*c^5 - 8*a
```

```
*b^10*c^4 + 184*a^2*b*c^12 - 328*a^3*b*c^11 - 16*a^4*b*c^10 + 496*a^5*b*c^9
 - 88*a^6*b*c^8 - 416*a^2*b^2*c^11 + 770*a^2*b^3*c^10 - 723*a^2*b^4*c^9 + 7
79*a^2*b^5*c^8 - 732*a^2*b^6*c^7 + 80*a^2*b^7*c^6 + 112*a^2*b^8*c^5 - 8*a^2
*b^9*c^4 + 180*a^3*b^2*c^10 - 494*a^3*b^3*c^9 + 521*a^3*b^4*c^8 + 572*a^3*b
^5*c^7 - 424*a^3*b^6*c^6 + 56*a^3*b^7*c^5 + 8*a^3*b^8*c^4 + 234*a^4*b^2*c^9
 - 1152*a^4*b^3*c^8 + 416*a^4*b^4*c^7 - 140*a^4*b^5*c^6 - 72*a^4*b^6*c^5 +
64*a^5*b^2*c^8 + 192*a^5*b^3*c^7 + 220*a^5*b^4*c^6 - 256*a^6*b^2*c^7 - 24*a
*b*c^13))/c^8 + (((2048*(16*a^3*c^13 - 32*a^2*c^14 + 176*a^4*c^12 + 176*a^5
*c^11 + 48*a^6*c^10 - 2*b^4*c^12 + 6*b^5*c^11 - 18*b^6*c^10 + 26*b^7*c^9 -
12*b^8*c^8 + 16*a*b^2*c^13 - 40*a*b^3*c^12 + 122*a*b^4*c^11 - 192*a*b^5*c^1
0 + 74*a*b^6*c^9 + 20*a*b^7*c^8 + 64*a^2*b*c^13 - 144*a^3*b*c^12 - 352*a^4*
b*c^11 - 144*a^5*b*c^10 - 204*a^2*b^2*c^12 + 388*a^2*b^3*c^11 - 50*a^2*b^4*
c^10 - 182*a^2*b^5*c^9 + 4*a^2*b^6*c^8 - 260*a^3*b^2*c^11 + 496*a^3*b^3*c^1
0 + 10*a^3*b^4*c^9 - 20*a^3*b^5*c^8 - 148*a^4*b^2*c^10 + 116*a^4*b^3*c^9 +
8*a^4*b^4*c^8 - 44*a^5*b^2*c^9))/c^8 + (1024*tan(x/2)*(b^2*2i - a*c*2i + c^
2*1i)*(32*a*c^16 - 64*a^2*c^15 - 128*a^3*c^14 + 64*a^4*c^13 + 96*a^5*c^12 -
 8*b^2*c^15 + 24*b^3*c^14 - 32*b^4*c^13 + 32*b^5*c^12 - 24*b^6*c^11 + 8*b^7
*c^10 + 144*a*b^2*c^14 - 200*a*b^3*c^13 + 184*a*b^4*c^12 - 56*a*b^5*c^11 -
8*a*b^6*c^10 + 288*a^2*b*c^14 + 352*a^3*b*c^13 - 32*a^4*b*c^12 - 320*a^2*b^
2*c^13 + 8*a^2*b^3*c^12 + 96*a^2*b^4*c^11 - 8*a^2*b^5*c^10 - 272*a^3*b^2*c^
12 + 40*a^3*b^3*c^11 + 8*a^3*b^4*c^10 - 56*a^4*b^2*c^11 - 96*a*b*c^15))/c^1
1)*(b^2*2i - a*c*2i + c^2*1i))/(2*c^3))*(b^2*2i - a*c*2i + c^2*1i))/(2*c^3)
)*(b^2*2i - a*c*2i + c^2*1i))/(2*c^3))*(b^2*2i - a*c*2i + c^2*1i))/(2*c^3))
)*(b^2*2i - a*c*2i + c^2*1i)*1i)/c^3
```

3.14 $\int \frac{\cos^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 19, antiderivative size = 299

$$\int \frac{\cos^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{bx}{c^2}+\frac{2\left(b^2-ac-\frac{b^3}{\sqrt{b^2-4ac}}+\frac{3abc}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{c^2\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} + \frac{2\left(b^2-ac+\frac{b^3}{\sqrt{b^2-4ac}}-\frac{3abc}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{c^2\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}+\frac{\sin(x)}{c}$$

```
[Out] -b*x/c^2+sin(x)/c+2*arctan((b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2
*c-(-4*a*c+b^2)^(1/2))^(1/2))*(b^2-a*c-b^3/(-4*a*c+b^2)^(1/2)+3*a*b*c/(-4*a
*c+b^2)^(1/2))/c^2/(b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)/(b+2*c-(-4*a*c+b^2)^(1/
2))^(1/2)+2*arctan((b-2*c+(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c+(-4*a
*c+b^2)^(1/2))^(1/2))*(b^2-a*c+b^3/(-4*a*c+b^2)^(1/2)-3*a*b*c/(-4*a*c+b^2)^
(1/2))/c^2/(b-2*c+(-4*a*c+b^2)^(1/2))^(1/2)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2
)
```

Rubi [A] (verified)

Time = 7.40 (sec) , antiderivative size = 299, normalized size of antiderivative = 1.00, number of steps used = 8, number of rules used = 5, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.263, Rules used = {3338, 2717, 3374, 2738, 211}

$$\int \frac{\cos^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$$
$$=\frac{2\left(\frac{3abc}{\sqrt{b^2-4ac}}-\frac{b^3}{\sqrt{b^2-4ac}}-ac+b^2\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{-\sqrt{b^2-4ac}+b-2c}}{\sqrt{-\sqrt{b^2-4ac}+b+2c}}\right)}{c^2\sqrt{-\sqrt{b^2-4ac}+b-2c}\sqrt{-\sqrt{b^2-4ac}+b+2c}}$$
$$+\frac{2\left(-\frac{3abc}{\sqrt{b^2-4ac}}+\frac{b^3}{\sqrt{b^2-4ac}}-ac+b^2\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{\sqrt{b^2-4ac}+b-2c}}{\sqrt{\sqrt{b^2-4ac}+b+2c}}\right)}{c^2\sqrt{\sqrt{b^2-4ac}+b-2c}\sqrt{\sqrt{b^2-4ac}+b+2c}}-\frac{bx}{c^2}+\frac{\sin(x)}{c}$$

```
[In] Int[Cos[x]^3/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] -((b*x)/c^2) + (2*(b^2 - a*c - b^3/Sqrt[b^2 - 4*a*c] + (3*a*b*c)/Sqrt[b^2 -
 4*a*c])*ArcTan[(Sqrt[b - 2*c - Sqrt[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c -
 Sqrt[b^2 - 4*a*c]]])/(c^2*Sqrt[b - 2*c - Sqrt[b^2 - 4*a*c]]*Sqrt[b + 2*c -
 Sqrt[b^2 - 4*a*c]]) + (2*(b^2 - a*c + b^3/Sqrt[b^2 - 4*a*c] - (3*a*b*c)/Sq
rt[b^2 - 4*a*c])*ArcTan[(Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b
 + 2*c + Sqrt[b^2 - 4*a*c]]])/(c^2*Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]*Sqrt[b
 + 2*c + Sqrt[b^2 - 4*a*c]]) + Sin[x]/c
```

Rule 211

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(Rt[a/b, 2]/a)*ArcTan[x/R
t[a/b, 2]], x] /; FreeQ[{a, b}, x] && PosQ[a/b]
```

Rule 2717

```
Int[sin[Pi/2 + (c_.) + (d_.)*(x_)], x_Symbol] :> Simp[Sin[c + d*x]/d, x] /;
 FreeQ[{c, d}, x]
```

Rule 2738

```
Int[((a_) + (b_.)*sin[Pi/2 + (c_.) + (d_.)*(x_)])^(-1), x_Symbol] :> With[{
e = FreeFactors[Tan[(c + d*x)/2], x]}, Dist[2*(e/d), Subst[Int[1/(a + b + (
a - b)*e^2*x^2), x], x, Tan[(c + d*x)/2]/e], x]] /; FreeQ[{a, b, c, d}, x]
&& NeQ[a^2 - b^2, 0]
```

Rule 3338

```
Int[cos[(d_.) + (e_.)*(x_)]^(m_.)*((a_.) + cos[(d_.) + (e_.)*(x_)]^(n_.)*(b
_.) + cos[(d_.) + (e_.)*(x_)]^(n2_.)*(c_.))^(p_), x_Symbol] :> Int[ExpandTr
```

```
ig[cos[d + e*x]^m*(a + b*cos[d + e*x]^n + c*cos[d + e*x]^(2*n))^p, x], x] /
; FreeQ[{a, b, c, d, e}, x] && EqQ[n2, 2*n] && NeQ[b^2 - 4*a*c, 0] && Integ
ersQ[m, n, p]
```

Rule 3374

```
Int[(cos[(d_.) + (e_.)*(x_)]*(B_.) + (A_))/((a_.) + cos[(d_.) + (e_.)*(x_)]
*(b_.) + cos[(d_.) + (e_.)*(x_)]^2*(c_.)), x_Symbol] :> Module[{q = Rt[b^2
- 4*a*c, 2]}, Dist[B + (b*B - 2*A*c)/q, Int[1/(b + q + 2*c*Cos[d + e*x]), x
], x] + Dist[B - (b*B - 2*A*c)/q, Int[1/(b - q + 2*c*Cos[d + e*x]), x], x]]
 /; FreeQ[{a, b, c, d, e, A, B}, x] && NeQ[b^2 - 4*a*c, 0]
```

Rubi steps

$$\begin{aligned}
\text{integral} &= \int \left(-\frac{b}{c^2}+\frac{\cos(x)}{c}+\frac{ab+b^2\left(1-\frac{ac}{b^2}\right)\cos(x)}{c^2\left(a+b\cos(x)+c\cos^2(x)\right)}\right)dx\\
&= -\frac{bx}{c^2}+\frac{\int \frac{ab+b^2\left(1-\frac{ac}{b^2}\right)\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx}{c^2}+\frac{\int \cos(x)\,dx}{c}\\
&= -\frac{bx}{c^2}+\frac{\sin(x)}{c}+\frac{\left(b^2-ac+\frac{b^3}{\sqrt{b^2-4ac}}-\frac{3abc}{\sqrt{b^2-4ac}}\right)\int \frac{1}{b+\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{c^2}\\
&\quad+\frac{\left(b^2-ac-\frac{b^3}{\sqrt{b^2-4ac}}+\frac{3abc}{\sqrt{b^2-4ac}}\right)\int \frac{1}{b-\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{c^2}\\
&= -\frac{bx}{c^2}+\frac{\sin(x)}{c}\\
&\quad+\frac{\left(2\left(b^2-ac+\frac{b^3}{\sqrt{b^2-4ac}}-\frac{3abc}{\sqrt{b^2-4ac}}\right)\right)\text{Subst}\left(\int \frac{1}{b+2c+\sqrt{b^2-4ac}+\left(b-2c+\sqrt{b^2-4ac}\right)x^2}\,dx,x,\tan\left(\frac{x}{2}\right)\right)}{c^2}\\
&\quad+\frac{\left(2\left(b^2-ac-\frac{b^3}{\sqrt{b^2-4ac}}+\frac{3abc}{\sqrt{b^2-4ac}}\right)\right)\text{Subst}\left(\int \frac{1}{b+2c-\sqrt{b^2-4ac}+\left(b-2c-\sqrt{b^2-4ac}\right)x^2}\,dx,x,\tan\left(\frac{x}{2}\right)\right)}{c^2}\\
&= -\frac{bx}{c^2}+\frac{2\left(b^2-ac-\frac{b^3}{\sqrt{b^2-4ac}}+\frac{3abc}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{c^2\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}}\\
&\quad+\frac{2\left(b^2-ac+\frac{b^3}{\sqrt{b^2-4ac}}-\frac{3abc}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{c^2\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}+\frac{\sin(x)}{c}
\end{aligned}$$

Mathematica [A] (verified)

Time = 0.47 (sec) , antiderivative size = 309, normalized size of antiderivative = 1.03

$$\int \frac{\cos^3(x)}{a + b\cos(x) + c\cos^2(x)}\,dx$$
$$= \frac{-bx - \frac{\sqrt{2}\left(b^3-3abc+b^2\sqrt{b^2-4ac}-ac\sqrt{b^2-4ac}\right)\operatorname{arctanh}\left(\frac{\left(b-2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)-2b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{-b^2+2c(a+c)-b\sqrt{b^2-4ac}}} + \frac{\sqrt{2}\left(-b^3+3abc+b^2\sqrt{b^2-4ac}-ac\sqrt{b^2-4ac}\right)\operatorname{arct}}{\sqrt{b^2-4ac}\sqrt{-b^2+2c(a+}}}{c^2}$$

```
[In] Integrate[Cos[x]^3/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (-(b*x) - (Sqrt[2]*(b^3 - 3*a*b*c + b^2*Sqrt[b^2 - 4*a*c] - a*c*Sqrt[b^2 -
4*a*c])*ArcTanh[((b - 2*c + Sqrt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*
(a + c) - 2*b*Sqrt[b^2 - 4*a*c]]])/(Sqrt[b^2 - 4*a*c]*Sqrt[-b^2 + 2*c*(a +
c) - b*Sqrt[b^2 - 4*a*c]]) + (Sqrt[2]*(-b^3 + 3*a*b*c + b^2*Sqrt[b^2 - 4*a*
c] - a*c*Sqrt[b^2 - 4*a*c])*ArcTanh[((-b + 2*c + Sqrt[b^2 - 4*a*c])*Tan[x/2
])/Sqrt[-2*b^2 + 4*c*(a + c) + 2*b*Sqrt[b^2 - 4*a*c]]])/(Sqrt[b^2 - 4*a*c]*
Sqrt[-b^2 + 2*c*(a + c) + b*Sqrt[b^2 - 4*a*c]]) + c*Sin[x])/c^2
```

Maple [A] (verified)

Time = 3.28 (sec) , antiderivative size = 331, normalized size of antiderivative = 1.11

method	result
default	$\frac{2(a-b+c)\left(\frac{\left(ab\sqrt{-4ac+b^2}+\sqrt{-4ac+b^2}\,ac-\sqrt{-4ac+b^2}\,b^2-2a^2c+a\,b^2+3cab-b^3\right)\operatorname{arctanh}\left(\frac{(-a+b-c)\tan\left(\frac{x}{2}\right)}{\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}\right)}{2\sqrt{-4ac+b^2}\,(a-b+c)\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}+\frac{\left(ab\sqrt{-4ac+b^2}+\sqrt{-}\right.}{}\right.}{c^2}$
risch	Expression too large to display

```
[In] int(cos(x)^3/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] 2/c^2*(a-b+c)*(1/2*(a*b*(-4*a*c+b^2)^(1/2)+(-4*a*c+b^2)^(1/2)*a*c-(-4*a*c+b
^2)^(1/2)*b^2-2*a^2*c+a*b^2+3*c*a*b-b^3)/(-4*a*c+b^2)^(1/2)/(a-b+c)/(((-4*a
*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2)*arctanh((-a+b-c)*tan(1/2*x)/(((-4*a*c+b^2
)^(1/2)-a+c)*(a-b+c))^(1/2))+1/2*(a*b*(-4*a*c+b^2)^(1/2)+(-4*a*c+b^2)^(1/2)
*a*c-(-4*a*c+b^2)^(1/2)*b^2+2*a^2*c-a*b^2-3*c*a*b+b^3)/(-4*a*c+b^2)^(1/2)/(
a-b+c)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)*arctan((a-b+c)*tan(1/2*x)/(
((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)))-2/c^2*(-c*tan(1/2*x)/(1+tan(1/2*x
)^2)+b*arctan(tan(1/2*x)))
```

Fricas [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 6529 vs. 2(255) = 510.

Time = 2.35 (sec) , antiderivative size = 6529, normalized size of antiderivative = 21.84

$$\int \frac{\cos^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(cos(x)^3/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
[Out] Too large to include
```

Sympy [F(-1)]

Timed out.

$$\int \frac{\cos^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Timed out}$$

```
[In] integrate(cos(x)**3/(a+b*cos(x)+c*cos(x)**2),x)
[Out] Timed out
```

Maxima [F]

$$\int \frac{\cos^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \int \frac{\cos(x)^3}{c\cos(x)^2+b\cos(x)+a}\,dx$$

```
[In] integrate(cos(x)^3/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
[Out] -(c^2*integrate(-2*(2*(b^3 - a*b*c)*cos(3*x)^2 + 4*(2*a^2*b + a*b*c)*cos(2*
x)^2 + 2*(b^3 - a*b*c)*cos(x)^2 + 2*(b^3 - a*b*c)*sin(3*x)^2 + 4*(2*a^2*b +
 a*b*c)*sin(2*x)^2 + 2*(4*a*b^2 - a*c^2 - (2*a^2 - b^2)*c)*sin(2*x)*sin(x)
+ 2*(b^3 - a*b*c)*sin(x)^2 + (2*a*b*c*cos(2*x) + (b^2*c - a*c^2)*cos(3*x) +
 (b^2*c - a*c^2)*cos(x))*cos(4*x) + (b^2*c - a*c^2 + 2*(4*a*b^2 - a*c^2 - (
2*a^2 - b^2)*c)*cos(2*x) + 4*(b^3 - a*b*c)*cos(x))*cos(3*x) + 2*(a*b*c + (4
*a*b^2 - a*c^2 - (2*a^2 - b^2)*c)*cos(x))*cos(2*x) + (b^2*c - a*c^2)*cos(x)
 + (2*a*b*c*sin(2*x) + (b^2*c - a*c^2)*sin(3*x) + (b^2*c - a*c^2)*sin(x))*s
in(4*x) + 2*((4*a*b^2 - a*c^2 - (2*a^2 - b^2)*c)*sin(2*x) + 2*(b^3 - a*b*c)
*sin(x))*sin(3*x))/(c^4*cos(4*x)^2 + 4*b^2*c^2*cos(3*x)^2 + 4*b^2*c^2*cos(x
)^2 + c^4*sin(4*x)^2 + 4*b^2*c^2*sin(3*x)^2 + 4*b^2*c^2*sin(x)^2 + 4*b*c^3*
cos(x) + c^4 + 4*(4*a^2*c^2 + 4*a*c^3 + c^4)*cos(2*x)^2 + 4*(4*a^2*c^2 + 4*
a*c^3 + c^4)*sin(2*x)^2 + 8*(2*a*b*c^2 + b*c^3)*sin(2*x)*sin(x) + 2*(2*b*c^
3*cos(3*x) + 2*b*c^3*cos(x) + c^4 + 2*(2*a*c^3 + c^4)*cos(2*x))*cos(4*x) +
4*(2*b^2*c^2*cos(x) + b*c^3 + 2*(2*a*b*c^2 + b*c^3)*cos(2*x))*cos(3*x) + 4*
(2*a*c^3 + c^4 + 2*(2*a*b*c^2 + b*c^3)*cos(x))*cos(2*x) + 4*(b*c^3*sin(3*x)
 + b*c^3*sin(x) + (2*a*c^3 + c^4)*sin(2*x))*sin(4*x) + 8*(b^2*c^2*sin(x) +
(2*a*b*c^2 + b*c^3)*sin(2*x))*sin(3*x)), x) + b*x - c*sin(x))/c^2
```

Giac [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 4810 vs. $2(255) = 510$.

Time = 3.24 (sec) , antiderivative size = 4810, normalized size of antiderivative = 16.09

$$\int \frac{\cos^3(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(cos(x)^3/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] -(2*a^3*b^4 - 6*a^2*b^5 - 2*a*b^6 + 6*b^7 - 12*a^4*b^2*c + 42*a^3*b^3*c + 2
4*a^2*b^4*c - 54*a*b^5*c - 8*b^6*c + 16*a^5*c^2 - 72*a^4*b*c^2 - 88*a^3*b^2
*c^2 + 148*a^2*b^3*c^2 + 66*a*b^4*c^2 + 2*b^5*c^2 + 96*a^4*c^3 - 112*a^3*b*
c^3 - 156*a^2*b^2*c^3 - 14*a*b^3*c^3 + 80*a^3*c^4 + 24*a^2*b*c^4 - 3*sqrt(a
^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^3 - sqrt(a^2 -
a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^4 + 7*sqrt(a^2 - a*b
 + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^5 + 5*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^6 + 12*sqrt(a^2 - a*b + b*c - c^
2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*b*c + 7*sqrt(a^2 - a*b + b*c - c^2 +
 sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^2*c - 44*sqrt(a^2 - a*b + b*c - c^2 +
 sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^3*c - 45*sqrt(a^2 - a*b + b*c - c^2 +
 sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^4*c - 6*sqrt(a^2 - a*b + b*c - c^2 + sq
rt(b^2 - 4*a*c)*(a - b + c))*b^5*c - 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b
^2 - 4*a*c)*(a - b + c))*a^4*c^2 + 64*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
 - 4*a*c)*(a - b + c))*a^3*b*c^2 + 114*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^
2 - 4*a*c)*(a - b + c))*a^2*b^2*c^2 + 35*sqrt(a^2 - a*b + b*c - c^2 + sqrt(
b^2 - 4*a*c)*(a - b + c))*a*b^3*c^2 + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b
^2 - 4*a*c)*(a - b + c))*b^4*c^2 - 56*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
 - 4*a*c)*(a - b + c))*a^3*c^3 - 44*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
 4*a*c)*(a - b + c))*a^2*b*c^3 - 25*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
 4*a*c)*(a - b + c))*a*b^2*c^3 + 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
 4*a*c)*(a - b + c))*a^2*c^4 + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2 - 2*(b^2 - 4*a*c)*a^3*b^2 + sqr
t(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*
a^2*b^3 + 6*(b^2 - 4*a*c)*a^2*b^3 - 7*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4 + 2*(b^2 - 4*a*c)*a*b^4 - 5*
sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*
c)*b^5 - 6*(b^2 - 4*a*c)*b^5 - 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*c + 4*(b^2 - 4*a*c)*a^4*c - 5*sqrt(
a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^
3*b*c - 18*(b^2 - 4*a*c)*a^3*b*c + 30*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c - 16*(b^2 - 4*a*c)*a^2*b
^2*c + 35*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(
b^2 - 4*a*c)*a*b^3*c + 30*(b^2 - 4*a*c)*a*b^3*c + 6*sqrt(a^2 - a*b + b*c -
c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c + 8*(b^2 - 4*a
```

```
*c)*b^4*c - 28*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*
sqrt(b^2 - 4*a*c)*a^3*c^2 + 24*(b^2 - 4*a*c)*a^3*c^2 - 54*sqrt(a^2 - a*b +
b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^2 - 28
*(b^2 - 4*a*c)*a^2*b*c^2 - 23*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^2 - 34*(b^2 - 4*a*c)*a*b^2*c^2 - 5
*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a
*c)*b^3*c^2 - 2*(b^2 - 4*a*c)*b^3*c^2 + 10*sqrt(a^2 - a*b + b*c - c^2 + sqr
t(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^3 + 20*(b^2 - 4*a*c)*a^
2*c^3 + 15*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt
(b^2 - 4*a*c)*a*b*c^3 + 6*(b^2 - 4*a*c)*a*b*c^3)*(pi*floor(1/2*x/pi + 1/2)
+ arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a*c^2 - 2*c^3 + sqrt(-4*(a*c^2 + b*
c^2 + c^3)*(a*c^2 - b*c^2 + c^3) + 4*(a*c^2 - c^3)^2))/(a*c^2 - b*c^2 + c^3
))))*abs(a - b + c)/(3*a^5*b^2*c^2 - 5*a^4*b^3*c^2 - 6*a^3*b^4*c^2 + 10*a^2
*b^5*c^2 + 3*a*b^6*c^2 - 5*b^7*c^2 - 12*a^6*c^3 + 20*a^5*b*c^3 + 47*a^4*b^2
*c^3 - 60*a^3*b^3*c^3 - 46*a^2*b^4*c^3 + 40*a*b^5*c^3 + 11*b^6*c^3 - 92*a^5
*c^4 + 80*a^4*b*c^4 + 182*a^3*b^2*c^4 - 94*a^2*b^3*c^4 - 78*a*b^4*c^4 - 6*b
^5*c^4 - 184*a^4*c^5 + 56*a^3*b*c^5 + 166*a^2*b^2*c^5 + 36*a*b^3*c^5 - 6*b^
4*c^5 - 120*a^3*c^6 - 48*a^2*b*c^6 + 23*a*b^2*c^6 + 11*b^3*c^6 + 4*a^2*c^7
- 44*a*b*c^7 - 5*b^2*c^7 + 20*a*c^8) + (2*a^3*b^4 - 6*a^2*b^5 - 2*a*b^6 + 6
*b^7 - 12*a^4*b^2*c + 42*a^3*b^3*c + 24*a^2*b^4*c - 54*a*b^5*c - 8*b^6*c +
16*a^5*c^2 - 72*a^4*b*c^2 - 88*a^3*b^2*c^2 + 148*a^2*b^3*c^2 + 66*a*b^4*c^2
 + 2*b^5*c^2 + 96*a^4*c^3 - 112*a^3*b*c^3 - 156*a^2*b^2*c^3 - 14*a*b^3*c^3
+ 80*a^3*c^4 + 24*a^2*b*c^4 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a
*c)*(a - b + c))*a^3*b^3 + sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(
a - b + c))*a^2*b^4 - 7*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
 b + c))*a*b^5 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
c))*b^6 - 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^
4*b*c - 7*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b
^2*c + 44*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b
^3*c + 45*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^4
*c + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*b^5*c +
12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*c^2 - 64
*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b*c^2 - 11
4*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^2*c^2 -
 35*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^3*c^2 -
 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*b^4*c^2 + 56
*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*c^3 + 44*s
qrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b*c^3 + 25*s
qrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^2*c^3 - 20*s
qrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*c^4 + 3*sqrt
(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a
^3*b^2 - 2*(b^2 - 4*a*c)*a^3*b^2 + sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3 + 6*(b^2 - 4*a*c)*a^2*b^3 - 7
*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a
*c)*a*b^4 + 2*(b^2 - 4*a*c)*a*b^4 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
```

```
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5 - 6*(b^2 - 4*a*c)*b^5 - 6*sqrt
(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a
^4*c + 4*(b^2 - 4*a*c)*a^4*c - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c - 18*(b^2 - 4*a*c)*a^3*b*c + 30
*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a
*c)*a^2*b^2*c - 16*(b^2 - 4*a*c)*a^2*b^2*c + 35*sqrt(a^2 - a*b + b*c - c^2
- sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c + 30*(b^2 - 4*a*
c)*a*b^3*c + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*
sqrt(b^2 - 4*a*c)*b^4*c + 8*(b^2 - 4*a*c)*b^4*c - 28*sqrt(a^2 - a*b + b*c -
 c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^2 + 24*(b^2 -
 4*a*c)*a^3*c^2 - 54*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*sqrt(b^2 - 4*a*c)*a^2*b*c^2 - 28*(b^2 - 4*a*c)*a^2*b*c^2 - 23*sqrt(a^
2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^
2*c^2 - 34*(b^2 - 4*a*c)*a*b^2*c^2 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^
2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^3*c^2 - 2*(b^2 - 4*a*c)*b^3*c^2
 + 10*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*a^2*c^3 + 20*(b^2 - 4*a*c)*a^2*c^3 + 15*sqrt(a^2 - a*b + b*c - c^2
 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*c^3 + 6*(b^2 - 4*a*
c)*a*b*c^3)*(pi*floor(1/2*x/pi + 1/2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt(
(2*a*c^2 - 2*c^3 - sqrt(-4*(a*c^2 + b*c^2 + c^3)*(a*c^2 - b*c^2 + c^3) + 4*
(a*c^2 - c^3)^2))/(a*c^2 - b*c^2 + c^3))))*abs(a - b + c)/(3*a^5*b^2*c^2 -
5*a^4*b^3*c^2 - 6*a^3*b^4*c^2 + 10*a^2*b^5*c^2 + 3*a*b^6*c^2 - 5*b^7*c^2 -
12*a^6*c^3 + 20*a^5*b*c^3 + 47*a^4*b^2*c^3 - 60*a^3*b^3*c^3 - 46*a^2*b^4*c^
3 + 40*a*b^5*c^3 + 11*b^6*c^3 - 92*a^5*c^4 + 80*a^4*b*c^4 + 182*a^3*b^2*c^4
 - 94*a^2*b^3*c^4 - 78*a*b^4*c^4 - 6*b^5*c^4 - 184*a^4*c^5 + 56*a^3*b*c^5 +
 166*a^2*b^2*c^5 + 36*a*b^3*c^5 - 6*b^4*c^5 - 120*a^3*c^6 - 48*a^2*b*c^6 +
23*a*b^2*c^6 + 11*b^3*c^6 + 4*a^2*c^7 - 44*a*b*c^7 - 5*b^2*c^7 + 20*a*c^8)
- b*x/c^2 + 2*tan(1/2*x)/((tan(1/2*x)^2 + 1)*c)
```

Mupad [B] (verification not implemented)

Time = 14.65 (sec) , antiderivative size = 29362, normalized size of antiderivative = 98.20

$$\int \frac{\cos^3(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] int(cos(x)^3/(a + b*cos(x) + c*cos(x)^2),x)
```

```
[Out] sin(x)/c - atan((((((8192*(4*a^2*c^10 - 4*a^3*c^9 - 20*a^4*c^8 - 12*a^5*c^
7 + b^4*c^8 - 5*b^5*c^7 + 7*b^6*c^6 - 3*b^7*c^5 - 5*a*b^2*c^9 + 31*a*b^3*c^
8 - 46*a*b^4*c^7 + 15*a*b^5*c^6 + 5*a*b^6*c^5 - 44*a^2*b*c^9 - 64*a^3*b*c^8
 - 28*a^4*b*c^7 - 8*a^5*b*c^6 + 73*a^2*b^2*c^8 + 4*a^2*b^3*c^7 - 40*a^2*b^4
*c^6 + a^2*b^5*c^5 + 85*a^3*b^2*c^7 + 3*a^3*b^3*c^6 - 5*a^3*b^4*c^5 + 23*a^
4*b^2*c^6 + 2*a^4*b^3*c^5))/c^4 - (8192*tan(x/2)*((b^8 - a^2*b^6 + 8*a^4*c^
4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a
```

*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*
a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^
3)^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7
+ 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*
c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2)*(8*a*c^12 - 16*a^2*c^11 - 32*a^3
*c^10 + 16*a^4*c^9 + 24*a^5*c^8 - 2*b^2*c^11 + 6*b^3*c^10 - 8*b^4*c^9 + 8*b
^5*c^8 - 6*b^6*c^7 + 2*b^7*c^6 + 36*a*b^2*c^10 - 50*a*b^3*c^9 + 46*a*b^4*c^
8 - 14*a*b^5*c^7 - 2*a*b^6*c^6 + 72*a^2*b*c^10 + 88*a^3*b*c^9 - 8*a^4*b*c^8
 - 80*a^2*b^2*c^9 + 2*a^2*b^3*c^8 + 24*a^2*b^4*c^7 - 2*a^2*b^5*c^6 - 68*a^3
*b^2*c^8 + 10*a^3*b^3*c^7 + 2*a^3*b^4*c^6 - 14*a^4*b^2*c^7 - 24*a*b*c^11))/
c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(1/2)
 + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3
*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/
2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2
))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c
^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2) +
 (8192*tan(x/2)*(2*a^3*c^8 - 2*a^4*c^7 + 6*a^5*c^6 + 10*a^6*c^5 + 2*b^4*c^7
 - 6*b^5*c^6 + 8*b^6*c^5 - 8*b^7*c^4 + 6*b^8*c^3 - 2*b^9*c^2 - 8*a*b^2*c^8
+ 24*a*b^3*c^7 - 38*a*b^4*c^6 + 56*a*b^5*c^5 - 50*a*b^6*c^4 + 14*a*b^7*c^3
+ 2*a*b^8*c^2 + 18*a^3*b*c^7 + 12*a^4*b*c^6 - 22*a^5*b*c^5 + 23*a^2*b^2*c^7
 - 99*a^2*b^3*c^6 + 93*a^2*b^4*c^5 + 7*a^2*b^5*c^4 - 24*a^2*b^6*c^3 + 2*a^2
*b^7*c^2 + 37*a^3*b^2*c^6 - 122*a^3*b^3*c^5 + 59*a^3*b^4*c^4 - 10*a^3*b^5*c
^3 - 2*a^3*b^6*c^2 + 11*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 14*a^4*b^4*c^3 - 27*
a^5*b^2*c^4))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*c -
 b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^
4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c
 - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*c*(-(4*a*c
- b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c
^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*
c^5)))^(1/2) + (8192*(2*a^5*c^5 - a^4*c^6 - 3*b^9*c + 3*a^6*c^4 + b^6*c^4 -
 4*b^7*c^3 + 6*b^8*c^2 - 5*a*b^4*c^5 + 23*a*b^5*c^4 - 38*a*b^6*c^3 + 16*a*b
^7*c^2 + a^2*b^7*c - 5*a^3*b^6*c + 6*a^4*b*c^5 + 2*a^4*b^5*c + 10*a^5*b*c^4
 + 8*a^6*b*c^3 + 4*a^2*b^2*c^6 - 28*a^2*b^3*c^5 + 57*a^2*b^4*c^4 - 3*a^2*b^
5*c^3 - 41*a^2*b^6*c^2 - 3*a^3*b^2*c^5 - 55*a^3*b^3*c^4 + 91*a^3*b^4*c^3 +
4*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 36*a^4*b^3*c^3 + 25*a^4*b^4*c^2 - 20*a^5*b
^2*c^3 - 10*a^5*b^3*c^2 + 5*a*b^8*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*
a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b
^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*
c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/
2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a
^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 +
a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2) - (8192*tan(x/2)*(5*a*b^8 + b^8*c - b^
9 - 10*a^2*b^7 + 10*a^3*b^6 - 5*a^4*b^5 + a^5*b^4 + a^6*c^3 + a^7*c^2 - 6*a
*b^6*c^2 - 20*a^2*b^6*c + 40*a^3*b^5*c - 35*a^4*b^4*c + 14*a^5*b^3*c - a^6*
b*c^2 - 2*a^6*b^2*c + 9*a^2*b^4*c^3 + 11*a^2*b^5*c^2 - 2*a^3*b^2*c^4 - 18*a

^3*b^3*c^3 + 5*a^3*b^4*c^2 + 10*a^4*b^2*c^3 - 20*a^4*b^3*c^2 + 10*a^5*b^2*c
^2 + 2*a*b^7*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*
c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2
*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*
a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*c*(-(4*a
c - b^2)^3)^(1/2))/(2(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^
6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b
^2*c^5)))^(1/2)*1i - (((((8192*(4*a^2*c^10 - 4*a^3*c^9 - 20*a^4*c^8 - 12*a^
5*c^7 + b^4*c^8 - 5*b^5*c^7 + 7*b^6*c^6 - 3*b^7*c^5 - 5*a*b^2*c^9 + 31*a*b^
3*c^8 - 46*a*b^4*c^7 + 15*a*b^5*c^6 + 5*a*b^6*c^5 - 44*a^2*b*c^9 - 64*a^3*b
*c^8 - 28*a^4*b*c^7 - 8*a^5*b*c^6 + 73*a^2*b^2*c^8 + 4*a^2*b^3*c^7 - 40*a^2
*b^4*c^6 + a^2*b^5*c^5 + 85*a^3*b^2*c^7 + 3*a^3*b^3*c^6 - 5*a^3*b^4*c^5 + 2
3*a^4*b^2*c^6 + 2*a^4*b^3*c^5))/c^4 + (8192*tan(x/2)*((b^8 - a^2*b^6 + 8*a^
4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-
(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 -
10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b
^2)^3)^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*
c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*
b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2)*(8*a*c^12 - 16*a^2*c^11 - 32
*a^3*c^10 + 16*a^4*c^9 + 24*a^5*c^8 - 2*b^2*c^11 + 6*b^3*c^10 - 8*b^4*c^9 +
8*b^5*c^8 - 6*b^6*c^7 + 2*b^7*c^6 + 36*a*b^2*c^10 - 50*a*b^3*c^9 + 46*a*b^
4*c^8 - 14*a*b^5*c^7 - 2*a*b^6*c^6 + 72*a^2*b*c^10 + 88*a^3*b*c^9 - 8*a^4*b
*c^8 - 80*a^2*b^2*c^9 + 2*a^2*b^3*c^8 + 24*a^2*b^4*c^7 - 2*a^2*b^5*c^6 - 68
*a^3*b^2*c^8 + 10*a^3*b^3*c^7 + 2*a^3*b^4*c^6 - 14*a^4*b^2*c^7 - 24*a*b*c^1
1))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(
1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38
*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)
^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^
(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b
^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/
2) - (8192*tan(x/2)*(2*a^3*c^8 - 2*a^4*c^7 + 6*a^5*c^6 + 10*a^6*c^5 + 2*b^4
*c^7 - 6*b^5*c^6 + 8*b^6*c^5 - 8*b^7*c^4 + 6*b^8*c^3 - 2*b^9*c^2 - 8*a*b^2*
c^8 + 24*a*b^3*c^7 - 38*a*b^4*c^6 + 56*a*b^5*c^5 - 50*a*b^6*c^4 + 14*a*b^7*
c^3 + 2*a*b^8*c^2 + 18*a^3*b*c^7 + 12*a^4*b*c^6 - 22*a^5*b*c^5 + 23*a^2*b^2
*c^7 - 99*a^2*b^3*c^6 + 93*a^2*b^4*c^5 + 7*a^2*b^5*c^4 - 24*a^2*b^6*c^3 + 2
*a^2*b^7*c^2 + 37*a^3*b^2*c^6 - 122*a^3*b^3*c^5 + 59*a^3*b^4*c^4 - 10*a^3*b
^5*c^3 - 2*a^3*b^6*c^2 + 11*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 14*a^4*b^4*c^3 -
27*a^5*b^2*c^4))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a
*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^
2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4
*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*c*(-(4*
a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b
^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*
b^2*c^5)))^(1/2) + (8192*(2*a^5*c^5 - a^4*c^6 - 3*b^9*c + 3*a^6*c^4 + b^6*c
^4 - 4*b^7*c^3 + 6*b^8*c^2 - 5*a*b^4*c^5 + 23*a*b^5*c^4 - 38*a*b^6*c^3 + 16

*a*b^7*c^2 + a^2*b^7*c - 5*a^3*b^6*c + 6*a^4*b*c^5 + 2*a^4*b^5*c + 10*a^5*b
*c^4 + 8*a^6*b*c^3 + 4*a^2*b^2*c^6 - 28*a^2*b^3*c^5 + 57*a^2*b^4*c^4 - 3*a^
2*b^5*c^3 - 41*a^2*b^6*c^2 - 3*a^3*b^2*c^5 - 55*a^3*b^3*c^4 + 91*a^3*b^4*c^
3 + 4*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 36*a^4*b^3*c^3 + 25*a^4*b^4*c^2 - 20*a
^5*b^2*c^3 - 10*a^5*b^3*c^2 + 5*a*b^8*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4
+ 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c
 - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*
b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)
^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 +
16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^
6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2) + (8192*tan(x/2)*(5*a*b^8 + b^8*c
- b^9 - 10*a^2*b^7 + 10*a^3*b^6 - 5*a^4*b^5 + a^5*b^4 + a^6*c^3 + a^7*c^2 -
 6*a*b^6*c^2 - 20*a^2*b^6*c + 40*a^3*b^5*c - 35*a^4*b^4*c + 14*a^5*b^3*c -
a^6*b*c^2 - 2*a^6*b^2*c + 9*a^2*b^4*c^3 + 11*a^2*b^5*c^2 - 2*a^3*b^2*c^4 -
18*a^3*b^3*c^3 + 5*a^3*b^4*c^2 + 10*a^4*b^2*c^3 - 20*a^4*b^3*c^2 + 10*a^5*b
^2*c^2 + 2*a*b^7*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(
4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33
*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c^2*(
-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*c*(-
(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6
- b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a
^3*b^2*c^5)))^(1/2)*1i)/((((((8192*(4*a^2*c^10 - 4*a^3*c^9 - 20*a^4*c^8 - 1
2*a^5*c^7 + b^4*c^8 - 5*b^5*c^7 + 7*b^6*c^6 - 3*b^7*c^5 - 5*a*b^2*c^9 + 31*
a*b^3*c^8 - 46*a*b^4*c^7 + 15*a*b^5*c^6 + 5*a*b^6*c^5 - 44*a^2*b*c^9 - 64*a
^3*b*c^8 - 28*a^4*b*c^7 - 8*a^5*b*c^6 + 73*a^2*b^2*c^8 + 4*a^2*b^3*c^7 - 40
*a^2*b^4*c^6 + a^2*b^5*c^5 + 85*a^3*b^2*c^7 + 3*a^3*b^3*c^6 - 5*a^3*b^4*c^5
 + 23*a^4*b^2*c^6 + 2*a^4*b^3*c^5))/c^4 - (8192*tan(x/2)*((b^8 - a^2*b^6 +
8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^
3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c
^2 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c
 - b^2)^3)^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*
a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*
a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2)*(8*a*c^12 - 16*a^2*c^11
- 32*a^3*c^10 + 16*a^4*c^9 + 24*a^5*c^8 - 2*b^2*c^11 + 6*b^3*c^10 - 8*b^4*c
^9 + 8*b^5*c^8 - 6*b^6*c^7 + 2*b^7*c^6 + 36*a*b^2*c^10 - 50*a*b^3*c^9 + 46*
a*b^4*c^8 - 14*a*b^5*c^7 - 2*a*b^6*c^6 + 72*a^2*b*c^10 + 88*a^3*b*c^9 - 8*a
^4*b*c^8 - 80*a^2*b^2*c^9 + 2*a^2*b^3*c^8 + 24*a^2*b^4*c^7 - 2*a^2*b^5*c^6
- 68*a^3*b^2*c^8 + 10*a^3*b^3*c^7 + 2*a^3*b^4*c^6 - 14*a^4*b^2*c^7 - 24*a*b
c^11))/c^4)((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^
3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2
- 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2
)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)
^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8
*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))
^(1/2) + (8192*tan(x/2)*(2*a^3*c^8 - 2*a^4*c^7 + 6*a^5*c^6 + 10*a^6*c^5 + 2

*b^4*c^7 - 6*b^5*c^6 + 8*b^6*c^5 - 8*b^7*c^4 + 6*b^8*c^3 - 2*b^9*c^2 - 8*a*
b^2*c^8 + 24*a*b^3*c^7 - 38*a*b^4*c^6 + 56*a*b^5*c^5 - 50*a*b^6*c^4 + 14*a*
b^7*c^3 + 2*a*b^8*c^2 + 18*a^3*b*c^7 + 12*a^4*b*c^6 - 22*a^5*b*c^5 + 23*a^2
*b^2*c^7 - 99*a^2*b^3*c^6 + 93*a^2*b^4*c^5 + 7*a^2*b^5*c^4 - 24*a^2*b^6*c^3
 + 2*a^2*b^7*c^2 + 37*a^3*b^2*c^6 - 122*a^3*b^3*c^5 + 59*a^3*b^4*c^4 - 10*a
^3*b^5*c^3 - 2*a^3*b^6*c^2 + 11*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 14*a^4*b^4*c
^3 - 27*a^5*b^2*c^4))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-
(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 3
3*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c^2*
(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*c*(
-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6
 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*
a^3*b^2*c^5)))^(1/2) + (8192*(2*a^5*c^5 - a^4*c^6 - 3*b^9*c + 3*a^6*c^4 + b
^6*c^4 - 4*b^7*c^3 + 6*b^8*c^2 - 5*a*b^4*c^5 + 23*a*b^5*c^4 - 38*a*b^6*c^3
+ 16*a*b^7*c^2 + a^2*b^7*c - 5*a^3*b^6*c + 6*a^4*b*c^5 + 2*a^4*b^5*c + 10*a
^5*b*c^4 + 8*a^6*b*c^3 + 4*a^2*b^2*c^6 - 28*a^2*b^3*c^5 + 57*a^2*b^4*c^4 -
3*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 3*a^3*b^2*c^5 - 55*a^3*b^3*c^4 + 91*a^3*b^
4*c^3 + 4*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 36*a^4*b^3*c^3 + 25*a^4*b^4*c^2 -
20*a^5*b^2*c^3 - 10*a^5*b^3*c^2 + 5*a*b^8*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*
c^4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4
*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 1
0*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2
)^3)^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^
7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^
2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2) - (8192*tan(x/2)*(5*a*b^8 + b^
8*c - b^9 - 10*a^2*b^7 + 10*a^3*b^6 - 5*a^4*b^5 + a^5*b^4 + a^6*c^3 + a^7*c
^2 - 6*a*b^6*c^2 - 20*a^2*b^6*c + 40*a^3*b^5*c - 35*a^4*b^4*c + 14*a^5*b^3*
c - a^6*b*c^2 - 2*a^6*b^2*c + 9*a^2*b^4*c^3 + 11*a^2*b^5*c^2 - 2*a^3*b^2*c^
4 - 18*a^3*b^3*c^3 + 5*a^3*b^4*c^2 + 10*a^4*b^2*c^3 - 20*a^4*b^3*c^2 + 10*a
^5*b^2*c^2 + 2*a*b^7*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5
*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2)
+ 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c
^2*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*
c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*
c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 -
 8*a^3*b^2*c^5)))^(1/2) + (((((8192*(4*a^2*c^10 - 4*a^3*c^9 - 20*a^4*c^8 -
12*a^5*c^7 + b^4*c^8 - 5*b^5*c^7 + 7*b^6*c^6 - 3*b^7*c^5 - 5*a*b^2*c^9 + 31
*a*b^3*c^8 - 46*a*b^4*c^7 + 15*a*b^5*c^6 + 5*a*b^6*c^5 - 44*a^2*b*c^9 - 64*
a^3*b*c^8 - 28*a^4*b*c^7 - 8*a^5*b*c^6 + 73*a^2*b^2*c^8 + 4*a^2*b^3*c^7 - 4
0*a^2*b^4*c^6 + a^2*b^5*c^5 + 85*a^3*b^2*c^7 + 3*a^3*b^3*c^6 - 5*a^3*b^4*c^
5 + 23*a^4*b^2*c^6 + 2*a^4*b^3*c^5))/c^4 + (8192*tan(x/2)*((b^8 - a^2*b^6 +
 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b
^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*
c^2 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*
c - b^2)^3)^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32

*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2)*(8*a*c^12 - 16*a^2*c^11 - 32*a^3*c^10 + 16*a^4*c^9 + 24*a^5*c^8 - 2*b^2*c^11 + 6*b^3*c^10 - 8*b^4*c^9 + 8*b^5*c^8 - 6*b^6*c^7 + 2*b^7*c^6 + 36*a*b^2*c^10 - 50*a*b^3*c^9 + 46*a*b^4*c^8 - 14*a*b^5*c^7 - 2*a*b^6*c^6 + 72*a^2*b*c^10 + 88*a^3*b*c^9 - 8*a^4*b*c^8 - 80*a^2*b^2*c^9 + 2*a^2*b^3*c^8 + 24*a^2*b^4*c^7 - 2*a^2*b^5*c^6 - 68*a^3*b^2*c^8 + 10*a^3*b^3*c^7 + 2*a^3*b^4*c^6 - 14*a^4*b^2*c^7 - 24*a*b*c^11))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2) - (8192*tan(x/2)*(2*a^3*c^8 - 2*a^4*c^7 + 6*a^5*c^6 + 10*a^6*c^5 + 2*b^4*c^7 - 6*b^5*c^6 + 8*b^6*c^5 - 8*b^7*c^4 + 6*b^8*c^3 - 2*b^9*c^2 - 8*a*b^2*c^8 + 24*a*b^3*c^7 - 38*a*b^4*c^6 + 56*a*b^5*c^5 - 50*a*b^6*c^4 + 14*a*b^7*c^3 + 2*a*b^8*c^2 + 18*a^3*b*c^7 + 12*a^4*b*c^6 - 22*a^5*b*c^5 + 23*a^2*b^2*c^7 - 99*a^2*b^3*c^6 + 93*a^2*b^4*c^5 + 7*a^2*b^5*c^4 - 24*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 37*a^3*b^2*c^6 - 122*a^3*b^3*c^5 + 59*a^3*b^4*c^4 - 10*a^3*b^5*c^3 - 2*a^3*b^6*c^2 + 11*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 14*a^4*b^4*c^3 - 27*a^5*b^2*c^4))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2) + (8192*(2*a^5*c^5 - a^4*c^6 - 3*b^9*c + 3*a^6*c^4 + b^6*c^4 - 4*b^7*c^3 + 6*b^8*c^2 - 5*a*b^4*c^5 + 23*a*b^5*c^4 - 38*a*b^6*c^3 + 16*a*b^7*c^2 + a^2*b^7*c - 5*a^3*b^6*c + 6*a^4*b*c^5 + 2*a^4*b^5*c + 10*a^5*b*c^4 + 8*a^6*b*c^3 + 4*a^2*b^2*c^6 - 28*a^2*b^3*c^5 + 57*a^2*b^4*c^4 - 3*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 3*a^3*b^2*c^5 - 55*a^3*b^3*c^4 + 91*a^3*b^4*c^3 + 4*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 36*a^4*b^3*c^3 + 25*a^4*b^4*c^2 - 20*a^5*b^2*c^3 - 10*a^5*b^3*c^2 + 5*a*b^8*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2) + (8192*tan(x/2)*(5*a*b^8 + b^8*c - b^9 - 10*a^2*b^7 + 10*a^3*b^6 - 5*a^4*b^5 + a^5*b^4 + a^6*c^3 + a^7*c^2 - 6*a*b^6*c^2 - 20*a^2*b^6*c + 40*a^3*b^5*c - 35*a^4*b^4*c + 14*a^5*b^3*c - a^6*b*c^2 - 2*a^6*b^2*c + 9*a^2*b^4*c^3 + 11*a^2*b^5*c^2 - 2*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 5*a^3*b^4*c^2 + 10*a^4*b^2*c^3 - 20*a^4*b^3*c^2 + 10*a^5*b^2*c^2 + 2*a*b^7*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2)

+ 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*
c^2*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b
c(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4
*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4
- 8*a^3*b^2*c^5)))^(1/2) - (16384*(a^7*b + a^3*b^5 - 4*a^4*b^4 + 6*a^5*b^3
- 4*a^6*b^2 - a^3*b^4*c + 2*a^4*b^3*c - 2*a^5*b^2*c + a^4*b^2*c^2 + a^6*b*c
))/c^4))*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 + b^5*(-(4*a*c - b^2)^3)^(
1/2) + 8*a^3*b^4*c - a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38
*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)
^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) + 2*a^3*b*c*(-(4*a*c - b^2)^3)^
(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b
^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/
2)*2i - atan(((((((8192*(4*a^2*c^10 - 4*a^3*c^9 - 20*a^4*c^8 - 12*a^5*c^7 +
b^4*c^8 - 5*b^5*c^7 + 7*b^6*c^6 - 3*b^7*c^5 - 5*a*b^2*c^9 + 31*a*b^3*c^8 -
46*a*b^4*c^7 + 15*a*b^5*c^6 + 5*a*b^6*c^5 - 44*a^2*b*c^9 - 64*a^3*b*c^8 -
28*a^4*b*c^7 - 8*a^5*b*c^6 + 73*a^2*b^2*c^8 + 4*a^2*b^3*c^7 - 40*a^2*b^4*c^
6 + a^2*b^5*c^5 + 85*a^3*b^2*c^7 + 3*a^3*b^3*c^6 - 5*a^3*b^4*c^5 + 23*a^4*b
^2*c^6 + 2*a^4*b^3*c^5))/c^4 - (8192*tan(x/2)*((b^8 - a^2*b^6 + 8*a^4*c^4 +
8*a^5*c^3 - b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c
- b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b
^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^
(1/2) - 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 1
6*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6
+ a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2)*(8*a*c^12 - 16*a^2*c^11 - 32*a^3*c^
10 + 16*a^4*c^9 + 24*a^5*c^8 - 2*b^2*c^11 + 6*b^3*c^10 - 8*b^4*c^9 + 8*b^5*
c^8 - 6*b^6*c^7 + 2*b^7*c^6 + 36*a*b^2*c^10 - 50*a*b^3*c^9 + 46*a*b^4*c^8 -
14*a*b^5*c^7 - 2*a*b^6*c^6 + 72*a^2*b*c^10 + 88*a^3*b*c^9 - 8*a^4*b*c^8 -
80*a^2*b^2*c^9 + 2*a^2*b^3*c^8 + 24*a^2*b^4*c^7 - 2*a^2*b^5*c^6 - 68*a^3*b^
2*c^8 + 10*a^3*b^3*c^7 + 2*a^3*b^4*c^6 - 14*a^4*b^2*c^7 - 24*a*b*c^11))/c^4
)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(-(4*a*c - b^2)^3)^(1/2) +
8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^
2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2)
+ 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/
(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7
+ 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2) + (8
192*tan(x/2)*(2*a^3*c^8 - 2*a^4*c^7 + 6*a^5*c^6 + 10*a^6*c^5 + 2*b^4*c^7 -
6*b^5*c^6 + 8*b^6*c^5 - 8*b^7*c^4 + 6*b^8*c^3 - 2*b^9*c^2 - 8*a*b^2*c^8 + 2
4*a*b^3*c^7 - 38*a*b^4*c^6 + 56*a*b^5*c^5 - 50*a*b^6*c^4 + 14*a*b^7*c^3 + 2
*a*b^8*c^2 + 18*a^3*b*c^7 + 12*a^4*b*c^6 - 22*a^5*b*c^5 + 23*a^2*b^2*c^7 -
99*a^2*b^3*c^6 + 93*a^2*b^4*c^5 + 7*a^2*b^5*c^4 - 24*a^2*b^6*c^3 + 2*a^2*b^
7*c^2 + 37*a^3*b^2*c^6 - 122*a^3*b^3*c^5 + 59*a^3*b^4*c^4 - 10*a^3*b^5*c^3
- 2*a^3*b^6*c^2 + 11*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 14*a^4*b^4*c^3 - 27*a^5
*b^2*c^4))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(-(4*a*c - b^
2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c
^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c -

b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*(-(4*a*c - b
^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4
- 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5
)))^(1/2) + (8192*(2*a^5*c^5 - a^4*c^6 - 3*b^9*c + 3*a^6*c^4 + b^6*c^4 - 4*
b^7*c^3 + 6*b^8*c^2 - 5*a*b^4*c^5 + 23*a*b^5*c^4 - 38*a*b^6*c^3 + 16*a*b^7*
c^2 + a^2*b^7*c - 5*a^3*b^6*c + 6*a^4*b*c^5 + 2*a^4*b^5*c + 10*a^5*b*c^4 +
8*a^6*b*c^3 + 4*a^2*b^2*c^6 - 28*a^2*b^3*c^5 + 57*a^2*b^4*c^4 - 3*a^2*b^5*c
^3 - 41*a^2*b^6*c^2 - 3*a^3*b^2*c^5 - 55*a^3*b^3*c^4 + 91*a^3*b^4*c^3 + 4*a
^3*b^5*c^2 - 24*a^4*b^2*c^4 - 36*a^4*b^3*c^3 + 25*a^4*b^4*c^2 - 20*a^5*b^2*
c^3 - 10*a^5*b^3*c^2 + 5*a*b^8*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5
c^3 - b^5(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)
^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c -
3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2)
- 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*
c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2
*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2) - (8192*tan(x/2)*(5*a*b^8 + b^8*c - b^9 -
10*a^2*b^7 + 10*a^3*b^6 - 5*a^4*b^5 + a^5*b^4 + a^6*c^3 + a^7*c^2 - 6*a*b^
6*c^2 - 20*a^2*b^6*c + 40*a^3*b^5*c - 35*a^4*b^4*c + 14*a^5*b^3*c - a^6*b*c
^2 - 2*a^6*b^2*c + 9*a^2*b^4*c^3 + 11*a^2*b^5*c^2 - 2*a^3*b^2*c^4 - 18*a^3*
b^3*c^3 + 5*a^3*b^4*c^2 + 10*a^4*b^2*c^3 - 20*a^4*b^3*c^2 + 10*a^5*b^2*c^2
+ 2*a*b^7*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(-(4*a*c -
b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^
4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c
- b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*(-(4*a*c
- b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c
^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*
c^5)))^(1/2)*1i - (((((8192*(4*a^2*c^10 - 4*a^3*c^9 - 20*a^4*c^8 - 12*a^5*c
^7 + b^4*c^8 - 5*b^5*c^7 + 7*b^6*c^6 - 3*b^7*c^5 - 5*a*b^2*c^9 + 31*a*b^3*c
^8 - 46*a*b^4*c^7 + 15*a*b^5*c^6 + 5*a*b^6*c^5 - 44*a^2*b*c^9 - 64*a^3*b*c^
8 - 28*a^4*b*c^7 - 8*a^5*b*c^6 + 73*a^2*b^2*c^8 + 4*a^2*b^3*c^7 - 40*a^2*b^
4*c^6 + a^2*b^5*c^5 + 85*a^3*b^2*c^7 + 3*a^3*b^3*c^6 - 5*a^3*b^4*c^5 + 23*a
^4*b^2*c^6 + 2*a^4*b^3*c^5))/c^4 + (8192*tan(x/2)*((b^8 - a^2*b^6 + 8*a^4*c
^4 + 8*a^5*c^3 - b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*
a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10
*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)
^3)^(1/2) - 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7
+ 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2
*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2)*(8*a*c^12 - 16*a^2*c^11 - 32*a^
3*c^10 + 16*a^4*c^9 + 24*a^5*c^8 - 2*b^2*c^11 + 6*b^3*c^10 - 8*b^4*c^9 + 8*
b^5*c^8 - 6*b^6*c^7 + 2*b^7*c^6 + 36*a*b^2*c^10 - 50*a*b^3*c^9 + 46*a*b^4*c
^8 - 14*a*b^5*c^7 - 2*a*b^6*c^6 + 72*a^2*b*c^10 + 88*a^3*b*c^9 - 8*a^4*b*c^
8 - 80*a^2*b^2*c^9 + 2*a^2*b^3*c^8 + 24*a^2*b^4*c^7 - 2*a^2*b^5*c^6 - 68*a^
3*b^2*c^8 + 10*a^3*b^3*c^7 + 2*a^3*b^4*c^6 - 14*a^4*b^2*c^7 - 24*a*b*c^11))
/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(-(4*a*c - b^2)^3)^(1/2
) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^

3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1
/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/
2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*
c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2)
- (8192*tan(x/2)*(2*a^3*c^8 - 2*a^4*c^7 + 6*a^5*c^6 + 10*a^6*c^5 + 2*b^4*c^
7 - 6*b^5*c^6 + 8*b^6*c^5 - 8*b^7*c^4 + 6*b^8*c^3 - 2*b^9*c^2 - 8*a*b^2*c^8
 + 24*a*b^3*c^7 - 38*a*b^4*c^6 + 56*a*b^5*c^5 - 50*a*b^6*c^4 + 14*a*b^7*c^3
 + 2*a*b^8*c^2 + 18*a^3*b*c^7 + 12*a^4*b*c^6 - 22*a^5*b*c^5 + 23*a^2*b^2*c^
7 - 99*a^2*b^3*c^6 + 93*a^2*b^4*c^5 + 7*a^2*b^5*c^4 - 24*a^2*b^6*c^3 + 2*a^
2*b^7*c^2 + 37*a^3*b^2*c^6 - 122*a^3*b^3*c^5 + 59*a^3*b^4*c^4 - 10*a^3*b^5*
c^3 - 2*a^3*b^6*c^2 + 11*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 14*a^4*b^4*c^3 - 27
*a^5*b^2*c^4))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(-(4*a*c
- b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b
^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*
c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*(-(4*a*c
 - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*
c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2
c^5)))^(1/2) + (8192(2*a^5*c^5 - a^4*c^6 - 3*b^9*c + 3*a^6*c^4 + b^6*c^4
- 4*b^7*c^3 + 6*b^8*c^2 - 5*a*b^4*c^5 + 23*a*b^5*c^4 - 38*a*b^6*c^3 + 16*a*
b^7*c^2 + a^2*b^7*c - 5*a^3*b^6*c + 6*a^4*b*c^5 + 2*a^4*b^5*c + 10*a^5*b*c^
4 + 8*a^6*b*c^3 + 4*a^2*b^2*c^6 - 28*a^2*b^3*c^5 + 57*a^2*b^4*c^4 - 3*a^2*b
^5*c^3 - 41*a^2*b^6*c^2 - 3*a^3*b^2*c^5 - 55*a^3*b^3*c^4 + 91*a^3*b^4*c^3 +
 4*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 36*a^4*b^3*c^3 + 25*a^4*b^4*c^2 - 20*a^5*
b^2*c^3 - 10*a^5*b^3*c^2 + 5*a*b^8*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8
*a^5*c^3 - b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c -
b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6
*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1
/2) - 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*
a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 +
 a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2) + (8192*tan(x/2)*(5*a*b^8 + b^8*c - b
^9 - 10*a^2*b^7 + 10*a^3*b^6 - 5*a^4*b^5 + a^5*b^4 + a^6*c^3 + a^7*c^2 - 6*
a*b^6*c^2 - 20*a^2*b^6*c + 40*a^3*b^5*c - 35*a^4*b^4*c + 14*a^5*b^3*c - a^6
*b*c^2 - 2*a^6*b^2*c + 9*a^2*b^4*c^3 + 11*a^2*b^5*c^2 - 2*a^3*b^2*c^4 - 18*
a^3*b^3*c^3 + 5*a^3*b^4*c^2 + 10*a^4*b^2*c^3 - 20*a^4*b^3*c^2 + 10*a^5*b^2*
c^2 + 2*a*b^7*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(-(4*a
*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^
2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2*(-(4
*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*(-(4*
a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b
^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*
b^2*c^5)))^(1/2)*1i)/((((((8192*(4*a^2*c^10 - 4*a^3*c^9 - 20*a^4*c^8 - 12*a
^5*c^7 + b^4*c^8 - 5*b^5*c^7 + 7*b^6*c^6 - 3*b^7*c^5 - 5*a*b^2*c^9 + 31*a*b
^3*c^8 - 46*a*b^4*c^7 + 15*a*b^5*c^6 + 5*a*b^6*c^5 - 44*a^2*b*c^9 - 64*a^3*
b*c^8 - 28*a^4*b*c^7 - 8*a^5*b*c^6 + 73*a^2*b^2*c^8 + 4*a^2*b^3*c^7 - 40*a^
2*b^4*c^6 + a^2*b^5*c^5 + 85*a^3*b^2*c^7 + 3*a^3*b^3*c^6 - 5*a^3*b^4*c^5 +

23*a^4*b^2*c^6 + 2*a^4*b^3*c^5))/c^4 - (8192*tan(x/2)*((b^8 - a^2*b^6 + 8*a
^4*c^4 + 8*a^5*c^3 - b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(
-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2
- 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c -
b^2)^3)^(1/2) - 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3
*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2
*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2)*(8*a*c^12 - 16*a^2*c^11 - 3
2*a^3*c^10 + 16*a^4*c^9 + 24*a^5*c^8 - 2*b^2*c^11 + 6*b^3*c^10 - 8*b^4*c^9
+ 8*b^5*c^8 - 6*b^6*c^7 + 2*b^7*c^6 + 36*a*b^2*c^10 - 50*a*b^3*c^9 + 46*a*b
^4*c^8 - 14*a*b^5*c^7 - 2*a*b^6*c^6 + 72*a^2*b*c^10 + 88*a^3*b*c^9 - 8*a^4*
b*c^8 - 80*a^2*b^2*c^9 + 2*a^2*b^3*c^8 + 24*a^2*b^4*c^7 - 2*a^2*b^5*c^6 - 6
8*a^3*b^2*c^8 + 10*a^3*b^3*c^7 + 2*a^3*b^4*c^6 - 14*a^4*b^2*c^7 - 24*a*b*c^
11))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(-(4*a*c - b^2)^3)^
(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 3
8*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3
)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*(-(4*a*c - b^2)^3)
^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*
b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1
/2) + (8192*tan(x/2)*(2*a^3*c^8 - 2*a^4*c^7 + 6*a^5*c^6 + 10*a^6*c^5 + 2*b^
4*c^7 - 6*b^5*c^6 + 8*b^6*c^5 - 8*b^7*c^4 + 6*b^8*c^3 - 2*b^9*c^2 - 8*a*b^2
*c^8 + 24*a*b^3*c^7 - 38*a*b^4*c^6 + 56*a*b^5*c^5 - 50*a*b^6*c^4 + 14*a*b^7
*c^3 + 2*a*b^8*c^2 + 18*a^3*b*c^7 + 12*a^4*b*c^6 - 22*a^5*b*c^5 + 23*a^2*b^
2*c^7 - 99*a^2*b^3*c^6 + 93*a^2*b^4*c^5 + 7*a^2*b^5*c^4 - 24*a^2*b^6*c^3 +
2*a^2*b^7*c^2 + 37*a^3*b^2*c^6 - 122*a^3*b^3*c^5 + 59*a^3*b^4*c^4 - 10*a^3*
b^5*c^3 - 2*a^3*b^6*c^2 + 11*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 14*a^4*b^4*c^3
- 27*a^5*b^2*c^4))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(-(4*
a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a
^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2*(-(
4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*(-(4
*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 -
b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3
*b^2*c^5)))^(1/2) + (8192*(2*a^5*c^5 - a^4*c^6 - 3*b^9*c + 3*a^6*c^4 + b^6*
c^4 - 4*b^7*c^3 + 6*b^8*c^2 - 5*a*b^4*c^5 + 23*a*b^5*c^4 - 38*a*b^6*c^3 + 1
6*a*b^7*c^2 + a^2*b^7*c - 5*a^3*b^6*c + 6*a^4*b*c^5 + 2*a^4*b^5*c + 10*a^5*
b*c^4 + 8*a^6*b*c^3 + 4*a^2*b^2*c^6 - 28*a^2*b^3*c^5 + 57*a^2*b^4*c^4 - 3*a
^2*b^5*c^3 - 41*a^2*b^6*c^2 - 3*a^3*b^2*c^5 - 55*a^3*b^3*c^4 + 91*a^3*b^4*c
^3 + 4*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 36*a^4*b^3*c^3 + 25*a^4*b^4*c^2 - 20*
a^5*b^2*c^3 - 10*a^5*b^3*c^2 + 5*a*b^8*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4
 + 8*a^5*c^3 - b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*
c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a
*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3
)^(1/2) - 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 +
 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c
^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2) - (8192*tan(x/2)*(5*a*b^8 + b^8*c
 - b^9 - 10*a^2*b^7 + 10*a^3*b^6 - 5*a^4*b^5 + a^5*b^4 + a^6*c^3 + a^7*c^2

```
- 6*a*b^6*c^2 - 20*a^2*b^6*c + 40*a^3*b^5*c - 35*a^4*b^4*c + 14*a^5*b^3*c -
 a^6*b*c^2 - 2*a^6*b^2*c + 9*a^2*b^4*c^3 + 11*a^2*b^5*c^2 - 2*a^3*b^2*c^4 -
 18*a^3*b^3*c^3 + 5*a^3*b^4*c^2 + 10*a^4*b^2*c^3 - 20*a^4*b^3*c^2 + 10*a^5*
b^2*c^2 + 2*a*b^7*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(-
(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 3
3*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2*
(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*(
-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6
 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*
a^3*b^2*c^5)))^(1/2) + (((((8192*(4*a^2*c^10 - 4*a^3*c^9 - 20*a^4*c^8 - 12*
a^5*c^7 + b^4*c^8 - 5*b^5*c^7 + 7*b^6*c^6 - 3*b^7*c^5 - 5*a*b^2*c^9 + 31*a*
b^3*c^8 - 46*a*b^4*c^7 + 15*a*b^5*c^6 + 5*a*b^6*c^5 - 44*a^2*b*c^9 - 64*a^3
*b*c^8 - 28*a^4*b*c^7 - 8*a^5*b*c^6 + 73*a^2*b^2*c^8 + 4*a^2*b^3*c^7 - 40*a
^2*b^4*c^6 + a^2*b^5*c^5 + 85*a^3*b^2*c^7 + 3*a^3*b^3*c^6 - 5*a^3*b^4*c^5 +
 23*a^4*b^2*c^6 + 2*a^4*b^3*c^5))/c^4 + (8192*tan(x/2)*((b^8 - a^2*b^6 + 8*
a^4*c^4 + 8*a^5*c^3 - b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*
(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2
 - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c -
 b^2)^3)^(1/2) - 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^
3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^
2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2)*(8*a*c^12 - 16*a^2*c^11 -
32*a^3*c^10 + 16*a^4*c^9 + 24*a^5*c^8 - 2*b^2*c^11 + 6*b^3*c^10 - 8*b^4*c^9
 + 8*b^5*c^8 - 6*b^6*c^7 + 2*b^7*c^6 + 36*a*b^2*c^10 - 50*a*b^3*c^9 + 46*a*
b^4*c^8 - 14*a*b^5*c^7 - 2*a*b^6*c^6 + 72*a^2*b*c^10 + 88*a^3*b*c^9 - 8*a^4
*b*c^8 - 80*a^2*b^2*c^9 + 2*a^2*b^3*c^8 + 24*a^2*b^4*c^7 - 2*a^2*b^5*c^6 -
68*a^3*b^2*c^8 + 10*a^3*b^3*c^7 + 2*a^3*b^4*c^6 - 14*a^4*b^2*c^7 - 24*a*b*c
^11))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(-(4*a*c - b^2)^3)
^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 -
38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^
3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*(-(4*a*c - b^2)^3
)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a
*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(
1/2) - (8192*tan(x/2)*(2*a^3*c^8 - 2*a^4*c^7 + 6*a^5*c^6 + 10*a^6*c^5 + 2*b
^4*c^7 - 6*b^5*c^6 + 8*b^6*c^5 - 8*b^7*c^4 + 6*b^8*c^3 - 2*b^9*c^2 - 8*a*b^
2*c^8 + 24*a*b^3*c^7 - 38*a*b^4*c^6 + 56*a*b^5*c^5 - 50*a*b^6*c^4 + 14*a*b^
7*c^3 + 2*a*b^8*c^2 + 18*a^3*b*c^7 + 12*a^4*b*c^6 - 22*a^5*b*c^5 + 23*a^2*b
^2*c^7 - 99*a^2*b^3*c^6 + 93*a^2*b^4*c^5 + 7*a^2*b^5*c^4 - 24*a^2*b^6*c^3 +
 2*a^2*b^7*c^2 + 37*a^3*b^2*c^6 - 122*a^3*b^3*c^5 + 59*a^3*b^4*c^4 - 10*a^3
*b^5*c^3 - 2*a^3*b^6*c^2 + 11*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 14*a^4*b^4*c^3
 - 27*a^5*b^2*c^4))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(-(4
*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*
a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2*(-
(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*(-(
4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 -
 b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^
```

```
3*b^2*c^5)))^(1/2) + (8192*(2*a^5*c^5 - a^4*c^6 - 3*b^9*c + 3*a^6*c^4 + b^6
*c^4 - 4*b^7*c^3 + 6*b^8*c^2 - 5*a*b^4*c^5 + 23*a*b^5*c^4 - 38*a*b^6*c^3 +
16*a*b^7*c^2 + a^2*b^7*c - 5*a^3*b^6*c + 6*a^4*b*c^5 + 2*a^4*b^5*c + 10*a^5
*b*c^4 + 8*a^6*b*c^3 + 4*a^2*b^2*c^6 - 28*a^2*b^3*c^5 + 57*a^2*b^4*c^4 - 3*
a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 3*a^3*b^2*c^5 - 55*a^3*b^3*c^4 + 91*a^3*b^4*
c^3 + 4*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 36*a^4*b^3*c^3 + 25*a^4*b^4*c^2 - 20
*a^5*b^2*c^3 - 10*a^5*b^3*c^2 + 5*a*b^8*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^
4 + 8*a^5*c^3 - b^5*(-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a
*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*
a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^
3)^(1/2) - 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7
+ 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*
c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2) + (8192*tan(x/2)*(5*a*b^8 + b^8*
c - b^9 - 10*a^2*b^7 + 10*a^3*b^6 - 5*a^4*b^5 + a^5*b^4 + a^6*c^3 + a^7*c^2
 - 6*a*b^6*c^2 - 20*a^2*b^6*c + 40*a^3*b^5*c - 35*a^4*b^4*c + 14*a^5*b^3*c
- a^6*b*c^2 - 2*a^6*b^2*c + 9*a^2*b^4*c^3 + 11*a^2*b^5*c^2 - 2*a^3*b^2*c^4
- 18*a^3*b^3*c^3 + 5*a^3*b^4*c^2 + 10*a^4*b^2*c^3 - 20*a^4*b^3*c^2 + 10*a^5
*b^2*c^2 + 2*a*b^7*c))/c^4)*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(
-(4*a*c - b^2)^3)^(1/2) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) +
33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2
*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*
(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^
6 - b^6*c^4 - 8*a*b^2*c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8
*a^3*b^2*c^5)))^(1/2) - (16384*(a^7*b + a^3*b^5 - 4*a^4*b^4 + 6*a^5*b^3 - 4
*a^6*b^2 - a^3*b^4*c + 2*a^4*b^3*c - 2*a^5*b^2*c + a^4*b^2*c^2 + a^6*b*c))/
c^4))*((b^8 - a^2*b^6 + 8*a^4*c^4 + 8*a^5*c^3 - b^5*(-(4*a*c - b^2)^3)^(1/2
) + 8*a^3*b^4*c + a^2*b^3*(-(4*a*c - b^2)^3)^(1/2) + 33*a^2*b^4*c^2 - 38*a^
3*b^2*c^3 - 18*a^4*b^2*c^2 - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1
/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2) - 2*a^3*b*c*(-(4*a*c - b^2)^3)^(1/
2))/(2*(16*a^2*c^8 + 32*a^3*c^7 + 16*a^4*c^6 + b^4*c^6 - b^6*c^4 - 8*a*b^2*
c^7 + 10*a*b^4*c^5 - 32*a^2*b^2*c^6 + a^2*b^4*c^4 - 8*a^3*b^2*c^5)))^(1/2)*
2i - (2*b*atan(((b*((8192*tan(x/2)*(5*a*b^8 + b^8*c - b^9 - 10*a^2*b^7 + 10
*a^3*b^6 - 5*a^4*b^5 + a^5*b^4 + a^6*c^3 + a^7*c^2 - 6*a*b^6*c^2 - 20*a^2*b
^6*c + 40*a^3*b^5*c - 35*a^4*b^4*c + 14*a^5*b^3*c - a^6*b*c^2 - 2*a^6*b^2*c
 + 9*a^2*b^4*c^3 + 11*a^2*b^5*c^2 - 2*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 5*a^3*
b^4*c^2 + 10*a^4*b^2*c^3 - 20*a^4*b^3*c^2 + 10*a^5*b^2*c^2 + 2*a*b^7*c))/c^
4 - (b*((8192*(2*a^5*c^5 - a^4*c^6 - 3*b^9*c + 3*a^6*c^4 + b^6*c^4 - 4*b^7*
c^3 + 6*b^8*c^2 - 5*a*b^4*c^5 + 23*a*b^5*c^4 - 38*a*b^6*c^3 + 16*a*b^7*c^2
+ a^2*b^7*c - 5*a^3*b^6*c + 6*a^4*b*c^5 + 2*a^4*b^5*c + 10*a^5*b*c^4 + 8*a^
6*b*c^3 + 4*a^2*b^2*c^6 - 28*a^2*b^3*c^5 + 57*a^2*b^4*c^4 - 3*a^2*b^5*c^3 -
 41*a^2*b^6*c^2 - 3*a^3*b^2*c^5 - 55*a^3*b^3*c^4 + 91*a^3*b^4*c^3 + 4*a^3*b
^5*c^2 - 24*a^4*b^2*c^4 - 36*a^4*b^3*c^3 + 25*a^4*b^4*c^2 - 20*a^5*b^2*c^3
- 10*a^5*b^3*c^2 + 5*a*b^8*c))/c^4 + (b*((b*((8192*(4*a^2*c^10 - 4*a^3*c^9
- 20*a^4*c^8 - 12*a^5*c^7 + b^4*c^8 - 5*b^5*c^7 + 7*b^6*c^6 - 3*b^7*c^5 - 5
*a*b^2*c^9 + 31*a*b^3*c^8 - 46*a*b^4*c^7 + 15*a*b^5*c^6 + 5*a*b^6*c^5 - 44*
```

a^2*b*c^9 - 64*a^3*b*c^8 - 28*a^4*b*c^7 - 8*a^5*b*c^6 + 73*a^2*b^2*c^8 + 4*
a^2*b^3*c^7 - 40*a^2*b^4*c^6 + a^2*b^5*c^5 + 85*a^3*b^2*c^7 + 3*a^3*b^3*c^6
 - 5*a^3*b^4*c^5 + 23*a^4*b^2*c^6 + 2*a^4*b^3*c^5))/c^4 - (b*tan(x/2)*(8*a*
c^12 - 16*a^2*c^11 - 32*a^3*c^10 + 16*a^4*c^9 + 24*a^5*c^8 - 2*b^2*c^11 + 6
*b^3*c^10 - 8*b^4*c^9 + 8*b^5*c^8 - 6*b^6*c^7 + 2*b^7*c^6 + 36*a*b^2*c^10 -
 50*a*b^3*c^9 + 46*a*b^4*c^8 - 14*a*b^5*c^7 - 2*a*b^6*c^6 + 72*a^2*b*c^10 +
 88*a^3*b*c^9 - 8*a^4*b*c^8 - 80*a^2*b^2*c^9 + 2*a^2*b^3*c^8 + 24*a^2*b^4*c
^7 - 2*a^2*b^5*c^6 - 68*a^3*b^2*c^8 + 10*a^3*b^3*c^7 + 2*a^3*b^4*c^6 - 14*a
^4*b^2*c^7 - 24*a*b*c^11)*8192i)/c^6)*1i)/c^2 + (8192*tan(x/2)*(2*a^3*c^8 -
 2*a^4*c^7 + 6*a^5*c^6 + 10*a^6*c^5 + 2*b^4*c^7 - 6*b^5*c^6 + 8*b^6*c^5 - 8
*b^7*c^4 + 6*b^8*c^3 - 2*b^9*c^2 - 8*a*b^2*c^8 + 24*a*b^3*c^7 - 38*a*b^4*c^
6 + 56*a*b^5*c^5 - 50*a*b^6*c^4 + 14*a*b^7*c^3 + 2*a*b^8*c^2 + 18*a^3*b*c^7
 + 12*a^4*b*c^6 - 22*a^5*b*c^5 + 23*a^2*b^2*c^7 - 99*a^2*b^3*c^6 + 93*a^2*b
^4*c^5 + 7*a^2*b^5*c^4 - 24*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 37*a^3*b^2*c^6 -
122*a^3*b^3*c^5 + 59*a^3*b^4*c^4 - 10*a^3*b^5*c^3 - 2*a^3*b^6*c^2 + 11*a^4*
b^2*c^5 + 15*a^4*b^3*c^4 + 14*a^4*b^4*c^3 - 27*a^5*b^2*c^4))/c^4)*1i)/c^2)*
1i)/c^2))/c^2 + (b*((8192*tan(x/2)*(5*a*b^8 + b^8*c - b^9 - 10*a^2*b^7 + 10
*a^3*b^6 - 5*a^4*b^5 + a^5*b^4 + a^6*c^3 + a^7*c^2 - 6*a*b^6*c^2 - 20*a^2*b
^6*c + 40*a^3*b^5*c - 35*a^4*b^4*c + 14*a^5*b^3*c - a^6*b*c^2 - 2*a^6*b^2*c
 + 9*a^2*b^4*c^3 + 11*a^2*b^5*c^2 - 2*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 5*a^3*
b^4*c^2 + 10*a^4*b^2*c^3 - 20*a^4*b^3*c^2 + 10*a^5*b^2*c^2 + 2*a*b^7*c))/c^
4 + (b*((8192*(2*a^5*c^5 - a^4*c^6 - 3*b^9*c + 3*a^6*c^4 + b^6*c^4 - 4*b^7*
c^3 + 6*b^8*c^2 - 5*a*b^4*c^5 + 23*a*b^5*c^4 - 38*a*b^6*c^3 + 16*a*b^7*c^2
+ a^2*b^7*c - 5*a^3*b^6*c + 6*a^4*b*c^5 + 2*a^4*b^5*c + 10*a^5*b*c^4 + 8*a^
6*b*c^3 + 4*a^2*b^2*c^6 - 28*a^2*b^3*c^5 + 57*a^2*b^4*c^4 - 3*a^2*b^5*c^3 -
 41*a^2*b^6*c^2 - 3*a^3*b^2*c^5 - 55*a^3*b^3*c^4 + 91*a^3*b^4*c^3 + 4*a^3*b
^5*c^2 - 24*a^4*b^2*c^4 - 36*a^4*b^3*c^3 + 25*a^4*b^4*c^2 - 20*a^5*b^2*c^3
- 10*a^5*b^3*c^2 + 5*a*b^8*c))/c^4 + (b*((b*((8192*(4*a^2*c^10 - 4*a^3*c^9
- 20*a^4*c^8 - 12*a^5*c^7 + b^4*c^8 - 5*b^5*c^7 + 7*b^6*c^6 - 3*b^7*c^5 - 5
*a*b^2*c^9 + 31*a*b^3*c^8 - 46*a*b^4*c^7 + 15*a*b^5*c^6 + 5*a*b^6*c^5 - 44*
a^2*b*c^9 - 64*a^3*b*c^8 - 28*a^4*b*c^7 - 8*a^5*b*c^6 + 73*a^2*b^2*c^8 + 4*
a^2*b^3*c^7 - 40*a^2*b^4*c^6 + a^2*b^5*c^5 + 85*a^3*b^2*c^7 + 3*a^3*b^3*c^6
 - 5*a^3*b^4*c^5 + 23*a^4*b^2*c^6 + 2*a^4*b^3*c^5))/c^4 + (b*tan(x/2)*(8*a*
c^12 - 16*a^2*c^11 - 32*a^3*c^10 + 16*a^4*c^9 + 24*a^5*c^8 - 2*b^2*c^11 + 6
*b^3*c^10 - 8*b^4*c^9 + 8*b^5*c^8 - 6*b^6*c^7 + 2*b^7*c^6 + 36*a*b^2*c^10 -
 50*a*b^3*c^9 + 46*a*b^4*c^8 - 14*a*b^5*c^7 - 2*a*b^6*c^6 + 72*a^2*b*c^10 +
 88*a^3*b*c^9 - 8*a^4*b*c^8 - 80*a^2*b^2*c^9 + 2*a^2*b^3*c^8 + 24*a^2*b^4*c
^7 - 2*a^2*b^5*c^6 - 68*a^3*b^2*c^8 + 10*a^3*b^3*c^7 + 2*a^3*b^4*c^6 - 14*a
^4*b^2*c^7 - 24*a*b*c^11)*8192i)/c^6)*1i)/c^2 - (8192*tan(x/2)*(2*a^3*c^8 -
 2*a^4*c^7 + 6*a^5*c^6 + 10*a^6*c^5 + 2*b^4*c^7 - 6*b^5*c^6 + 8*b^6*c^5 - 8
*b^7*c^4 + 6*b^8*c^3 - 2*b^9*c^2 - 8*a*b^2*c^8 + 24*a*b^3*c^7 - 38*a*b^4*c^
6 + 56*a*b^5*c^5 - 50*a*b^6*c^4 + 14*a*b^7*c^3 + 2*a*b^8*c^2 + 18*a^3*b*c^7
 + 12*a^4*b*c^6 - 22*a^5*b*c^5 + 23*a^2*b^2*c^7 - 99*a^2*b^3*c^6 + 93*a^2*b
^4*c^5 + 7*a^2*b^5*c^4 - 24*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 37*a^3*b^2*c^6 -
122*a^3*b^3*c^5 + 59*a^3*b^4*c^4 - 10*a^3*b^5*c^3 - 2*a^3*b^6*c^2 + 11*a^4*

b^2*c^5 + 15*a^4*b^3*c^4 + 14*a^4*b^4*c^3 - 27*a^5*b^2*c^4))/c^4)*1i)/c^2)*
1i)/c^2))/c^2)/((16384*(a^7*b + a^3*b^5 - 4*a^4*b^4 + 6*a^5*b^3 - 4*a^6*b^2
 - a^3*b^4*c + 2*a^4*b^3*c - 2*a^5*b^2*c + a^4*b^2*c^2 + a^6*b*c))/c^4 + (b
*((8192*tan(x/2)*(5*a*b^8 + b^8*c - b^9 - 10*a^2*b^7 + 10*a^3*b^6 - 5*a^4*b
^5 + a^5*b^4 + a^6*c^3 + a^7*c^2 - 6*a*b^6*c^2 - 20*a^2*b^6*c + 40*a^3*b^5*
c - 35*a^4*b^4*c + 14*a^5*b^3*c - a^6*b*c^2 - 2*a^6*b^2*c + 9*a^2*b^4*c^3 +
 11*a^2*b^5*c^2 - 2*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 5*a^3*b^4*c^2 + 10*a^4*b
^2*c^3 - 20*a^4*b^3*c^2 + 10*a^5*b^2*c^2 + 2*a*b^7*c))/c^4 - (b*((8192*(2*a
^5*c^5 - a^4*c^6 - 3*b^9*c + 3*a^6*c^4 + b^6*c^4 - 4*b^7*c^3 + 6*b^8*c^2 -
5*a*b^4*c^5 + 23*a*b^5*c^4 - 38*a*b^6*c^3 + 16*a*b^7*c^2 + a^2*b^7*c - 5*a^
3*b^6*c + 6*a^4*b*c^5 + 2*a^4*b^5*c + 10*a^5*b*c^4 + 8*a^6*b*c^3 + 4*a^2*b^
2*c^6 - 28*a^2*b^3*c^5 + 57*a^2*b^4*c^4 - 3*a^2*b^5*c^3 - 41*a^2*b^6*c^2 -
3*a^3*b^2*c^5 - 55*a^3*b^3*c^4 + 91*a^3*b^4*c^3 + 4*a^3*b^5*c^2 - 24*a^4*b^
2*c^4 - 36*a^4*b^3*c^3 + 25*a^4*b^4*c^2 - 20*a^5*b^2*c^3 - 10*a^5*b^3*c^2 +
 5*a*b^8*c))/c^4 + (b*((b*((8192*(4*a^2*c^10 - 4*a^3*c^9 - 20*a^4*c^8 - 12*
a^5*c^7 + b^4*c^8 - 5*b^5*c^7 + 7*b^6*c^6 - 3*b^7*c^5 - 5*a*b^2*c^9 + 31*a*
b^3*c^8 - 46*a*b^4*c^7 + 15*a*b^5*c^6 + 5*a*b^6*c^5 - 44*a^2*b*c^9 - 64*a^3
*b*c^8 - 28*a^4*b*c^7 - 8*a^5*b*c^6 + 73*a^2*b^2*c^8 + 4*a^2*b^3*c^7 - 40*a
^2*b^4*c^6 + a^2*b^5*c^5 + 85*a^3*b^2*c^7 + 3*a^3*b^3*c^6 - 5*a^3*b^4*c^5 +
 23*a^4*b^2*c^6 + 2*a^4*b^3*c^5))/c^4 - (b*tan(x/2)*(8*a*c^12 - 16*a^2*c^11
 - 32*a^3*c^10 + 16*a^4*c^9 + 24*a^5*c^8 - 2*b^2*c^11 + 6*b^3*c^10 - 8*b^4*
c^9 + 8*b^5*c^8 - 6*b^6*c^7 + 2*b^7*c^6 + 36*a*b^2*c^10 - 50*a*b^3*c^9 + 46
*a*b^4*c^8 - 14*a*b^5*c^7 - 2*a*b^6*c^6 + 72*a^2*b*c^10 + 88*a^3*b*c^9 - 8*
a^4*b*c^8 - 80*a^2*b^2*c^9 + 2*a^2*b^3*c^8 + 24*a^2*b^4*c^7 - 2*a^2*b^5*c^6
 - 68*a^3*b^2*c^8 + 10*a^3*b^3*c^7 + 2*a^3*b^4*c^6 - 14*a^4*b^2*c^7 - 24*a*
b*c^11)*8192i)/c^6)*1i)/c^2 + (8192*tan(x/2)*(2*a^3*c^8 - 2*a^4*c^7 + 6*a^5
*c^6 + 10*a^6*c^5 + 2*b^4*c^7 - 6*b^5*c^6 + 8*b^6*c^5 - 8*b^7*c^4 + 6*b^8*c
^3 - 2*b^9*c^2 - 8*a*b^2*c^8 + 24*a*b^3*c^7 - 38*a*b^4*c^6 + 56*a*b^5*c^5 -
 50*a*b^6*c^4 + 14*a*b^7*c^3 + 2*a*b^8*c^2 + 18*a^3*b*c^7 + 12*a^4*b*c^6 -
22*a^5*b*c^5 + 23*a^2*b^2*c^7 - 99*a^2*b^3*c^6 + 93*a^2*b^4*c^5 + 7*a^2*b^5
*c^4 - 24*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 37*a^3*b^2*c^6 - 122*a^3*b^3*c^5 +
59*a^3*b^4*c^4 - 10*a^3*b^5*c^3 - 2*a^3*b^6*c^2 + 11*a^4*b^2*c^5 + 15*a^4*b
^3*c^4 + 14*a^4*b^4*c^3 - 27*a^5*b^2*c^4))/c^4)*1i)/c^2)*1i)/c^2)*1i)/c^2 -
 (b*((8192*tan(x/2)*(5*a*b^8 + b^8*c - b^9 - 10*a^2*b^7 + 10*a^3*b^6 - 5*a^
4*b^5 + a^5*b^4 + a^6*c^3 + a^7*c^2 - 6*a*b^6*c^2 - 20*a^2*b^6*c + 40*a^3*b
^5*c - 35*a^4*b^4*c + 14*a^5*b^3*c - a^6*b*c^2 - 2*a^6*b^2*c + 9*a^2*b^4*c^
3 + 11*a^2*b^5*c^2 - 2*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 5*a^3*b^4*c^2 + 10*a^
4*b^2*c^3 - 20*a^4*b^3*c^2 + 10*a^5*b^2*c^2 + 2*a*b^7*c))/c^4 + (b*((8192*(
2*a^5*c^5 - a^4*c^6 - 3*b^9*c + 3*a^6*c^4 + b^6*c^4 - 4*b^7*c^3 + 6*b^8*c^2
 - 5*a*b^4*c^5 + 23*a*b^5*c^4 - 38*a*b^6*c^3 + 16*a*b^7*c^2 + a^2*b^7*c - 5
*a^3*b^6*c + 6*a^4*b*c^5 + 2*a^4*b^5*c + 10*a^5*b*c^4 + 8*a^6*b*c^3 + 4*a^2
*b^2*c^6 - 28*a^2*b^3*c^5 + 57*a^2*b^4*c^4 - 3*a^2*b^5*c^3 - 41*a^2*b^6*c^2
 - 3*a^3*b^2*c^5 - 55*a^3*b^3*c^4 + 91*a^3*b^4*c^3 + 4*a^3*b^5*c^2 - 24*a^4
*b^2*c^4 - 36*a^4*b^3*c^3 + 25*a^4*b^4*c^2 - 20*a^5*b^2*c^3 - 10*a^5*b^3*c^
2 + 5*a*b^8*c))/c^4 + (b*((b*((8192*(4*a^2*c^10 - 4*a^3*c^9 - 20*a^4*c^8 -

```
12*a^5*c^7 + b^4*c^8 - 5*b^5*c^7 + 7*b^6*c^6 - 3*b^7*c^5 - 5*a*b^2*c^9 + 31
*a*b^3*c^8 - 46*a*b^4*c^7 + 15*a*b^5*c^6 + 5*a*b^6*c^5 - 44*a^2*b*c^9 - 64*
a^3*b*c^8 - 28*a^4*b*c^7 - 8*a^5*b*c^6 + 73*a^2*b^2*c^8 + 4*a^2*b^3*c^7 - 4
0*a^2*b^4*c^6 + a^2*b^5*c^5 + 85*a^3*b^2*c^7 + 3*a^3*b^3*c^6 - 5*a^3*b^4*c^
5 + 23*a^4*b^2*c^6 + 2*a^4*b^3*c^5))/c^4 + (b*tan(x/2)*(8*a*c^12 - 16*a^2*c
^11 - 32*a^3*c^10 + 16*a^4*c^9 + 24*a^5*c^8 - 2*b^2*c^11 + 6*b^3*c^10 - 8*b
^4*c^9 + 8*b^5*c^8 - 6*b^6*c^7 + 2*b^7*c^6 + 36*a*b^2*c^10 - 50*a*b^3*c^9 +
 46*a*b^4*c^8 - 14*a*b^5*c^7 - 2*a*b^6*c^6 + 72*a^2*b*c^10 + 88*a^3*b*c^9 -
 8*a^4*b*c^8 - 80*a^2*b^2*c^9 + 2*a^2*b^3*c^8 + 24*a^2*b^4*c^7 - 2*a^2*b^5*
c^6 - 68*a^3*b^2*c^8 + 10*a^3*b^3*c^7 + 2*a^3*b^4*c^6 - 14*a^4*b^2*c^7 - 24
*a*b*c^11)*8192i)/c^6)*1i)/c^2 - (8192*tan(x/2)*(2*a^3*c^8 - 2*a^4*c^7 + 6*
a^5*c^6 + 10*a^6*c^5 + 2*b^4*c^7 - 6*b^5*c^6 + 8*b^6*c^5 - 8*b^7*c^4 + 6*b^
8*c^3 - 2*b^9*c^2 - 8*a*b^2*c^8 + 24*a*b^3*c^7 - 38*a*b^4*c^6 + 56*a*b^5*c^
5 - 50*a*b^6*c^4 + 14*a*b^7*c^3 + 2*a*b^8*c^2 + 18*a^3*b*c^7 + 12*a^4*b*c^6
 - 22*a^5*b*c^5 + 23*a^2*b^2*c^7 - 99*a^2*b^3*c^6 + 93*a^2*b^4*c^5 + 7*a^2*
b^5*c^4 - 24*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 37*a^3*b^2*c^6 - 122*a^3*b^3*c^5
 + 59*a^3*b^4*c^4 - 10*a^3*b^5*c^3 - 2*a^3*b^6*c^2 + 11*a^4*b^2*c^5 + 15*a^
4*b^3*c^4 + 14*a^4*b^4*c^3 - 27*a^5*b^2*c^4))/c^4)*1i)/c^2)*1i)/c^2)*1i)/c^
2)))/c^2
```

3.15 $\int \frac{\cos^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 19, antiderivative size = 255

$$\int \frac{\cos^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{x}{c}-\frac{2\left(b-\frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{c\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}}-\frac{2\left(b+\frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{c\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}$$

```
[Out] x/c-2*arctan((b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c-(-4*a*c+b^2
)^(1/2))^(1/2))*(b+(2*a*c-b^2)/(-4*a*c+b^2)^(1/2))/c/(b-2*c-(-4*a*c+b^2)^(1
/2))^(1/2)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2)-2*arctan((b-2*c+(-4*a*c+b^2)^(1
/2))^(1/2)*tan(1/2*x)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2))*(b+(-2*a*c+b^2)/(-4
*a*c+b^2)^(1/2))/c/(b-2*c+(-4*a*c+b^2)^(1/2))^(1/2)/(b+2*c+(-4*a*c+b^2)^(1/
2))^(1/2)
```

Rubi [A] (verified)

Time = 1.30 (sec) , antiderivative size = 255, normalized size of antiderivative = 1.00, number of steps used = 7, number of rules used = 4, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.211, Rules used = {3338, 3374, 2738, 211}

$$\int \frac{\cos^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{2\left(b-\frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{-\sqrt{b^2-4ac}+b-2c}}{\sqrt{-\sqrt{b^2-4ac}+b+2c}}\right)}{c\sqrt{-\sqrt{b^2-4ac}+b-2c}\sqrt{-\sqrt{b^2-4ac}+b+2c}}-\frac{2\left(\frac{b^2-2ac}{\sqrt{b^2-4ac}}+b\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{\sqrt{b^2-4ac}+b-2c}}{\sqrt{\sqrt{b^2-4ac}+b+2c}}\right)}{c\sqrt{\sqrt{b^2-4ac}+b-2c}\sqrt{\sqrt{b^2-4ac}+b+2c}}+\frac{x}{c}$$

```
[In] Int[Cos[x]^2/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] x/c - (2*(b - (b^2 - 2*a*c)/Sqrt[b^2 - 4*a*c])*ArcTan[(Sqrt[b - 2*c - Sqrt[
b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]])/(c*Sqrt[b - 2*c
 - Sqrt[b^2 - 4*a*c]]*Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]) - (2*(b + (b^2 - 2
*a*c)/Sqrt[b^2 - 4*a*c])*ArcTan[(Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]*Tan[x/2]
)/Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]]])/(c*Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]*
Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]])
```

Rule 211

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(Rt[a/b, 2]/a)*ArcTan[x/R
t[a/b, 2]], x] /; FreeQ[{a, b}, x] && PosQ[a/b]
```

Rule 2738

```
Int[((a_) + (b_.)*sin[Pi/2 + (c_.) + (d_.)*(x_)])^(-1), x_Symbol] :> With[{
e = FreeFactors[Tan[(c + d*x)/2], x]}, Dist[2*(e/d), Subst[Int[1/(a + b + (
a - b)*e^2*x^2), x], x, Tan[(c + d*x)/2]/e], x]] /; FreeQ[{a, b, c, d}, x]
&& NeQ[a^2 - b^2, 0]
```

Rule 3338

```
Int[cos[(d_.) + (e_.)*(x_)]^(m_.)*((a_.) + cos[(d_.) + (e_.)*(x_)]^(n_.)*(b
_.) + cos[(d_.) + (e_.)*(x_)]^(n2_.)*(c_.))^(p_), x_Symbol] :> Int[ExpandTr
ig[cos[d + e*x]^m*(a + b*cos[d + e*x]^n + c*cos[d + e*x]^(2*n))^p, x], x] /
; FreeQ[{a, b, c, d, e}, x] && EqQ[n2, 2*n] && NeQ[b^2 - 4*a*c, 0] && Integ
ersQ[m, n, p]
```

Rule 3374

```
Int[(cos[(d_.) + (e_.)*(x_)]*(B_.) + (A_))/((a_.) + cos[(d_.) + (e_.)*(x_)]
*(b_.) + cos[(d_.) + (e_.)*(x_)]^2*(c_.)), x_Symbol] :> Module[{q = Rt[b^2
- 4*a*c, 2]}, Dist[B + (b*B - 2*A*c)/q, Int[1/(b + q + 2*c*Cos[d + e*x]), x
], x] + Dist[B - (b*B - 2*A*c)/q, Int[1/(b - q + 2*c*Cos[d + e*x]), x], x]]
 /; FreeQ[{a, b, c, d, e, A, B}, x] && NeQ[b^2 - 4*a*c, 0]
```

Rubi steps

$$\text{integral} = \int \left(\frac{1}{c} + \frac{-a - b\cos(x)}{c\,(a + b\cos(x) + c\cos^2(x))}\right) dx$$

$$= \frac{x}{c} + \frac{\int \frac{-a-b\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx}{c}$$

$$= \frac{x}{c} - \frac{\left(b - \frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\int \frac{1}{b-\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{c} - \frac{\left(b + \frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\int \frac{1}{b+\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{c}$$

$$= \frac{x}{c} - \frac{\left(2\left(b - \frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\right) \text{Subst}\left(\int \frac{1}{b+2c-\sqrt{b^2-4ac}+\left(b-2c-\sqrt{b^2-4ac}\right)x^2}\,dx, x, \tan\left(\frac{x}{2}\right)\right)}{c} - \frac{\left(2\left(b + \frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\right) \text{Subst}\left(\int \frac{1}{b+2c+\sqrt{b^2-4ac}+\left(b-2c+\sqrt{b^2-4ac}\right)x^2}\,dx, x, \tan\left(\frac{x}{2}\right)\right)}{c}$$

$$= \frac{x}{c} - \frac{2\left(b - \frac{b^2-2ac}{\sqrt{b^2-4ac}}\right) \arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{c\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} - \frac{2\left(b + \frac{b^2-2ac}{\sqrt{b^2-4ac}}\right) \arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{c\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}$$

Mathematica [A] (verified)

Time = 1.36 (sec) , antiderivative size = 264, normalized size of antiderivative = 1.04

$$\int \frac{\cos^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{x + \frac{\sqrt{2}\left(b^2-2ac+b\sqrt{b^2-4ac}\right)\text{arctanh}\left(\frac{\left(b-2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)-2b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{-b^2+2c(a+c)-b\sqrt{b^2-4ac}}} - \frac{\sqrt{2}\left(-b^2+2ac+b\sqrt{b^2-4ac}\right)\text{arctanh}\left(\frac{\left(-b+2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)+2b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{-b^2+2c(a+c)+b\sqrt{b^2-4ac}}}}{c}$$

```
[In] Integrate[Cos[x]^2/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (x + (Sqrt[2]*(b^2 - 2*a*c + b*Sqrt[b^2 - 4*a*c])*ArcTanh[((b - 2*c + Sqrt[
b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) - 2*b*Sqrt[b^2 - 4*a*c]]]
)/(Sqrt[b^2 - 4*a*c]*Sqrt[-b^2 + 2*c*(a + c) - b*Sqrt[b^2 - 4*a*c]]) - (Sqr
t[2]*(-b^2 + 2*a*c + b*Sqrt[b^2 - 4*a*c])*ArcTanh[((-b + 2*c + Sqrt[b^2 - 4
*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) + 2*b*Sqrt[b^2 - 4*a*c]]])/(Sqrt
[b^2 - 4*a*c]*Sqrt[-b^2 + 2*c*(a + c) + b*Sqrt[b^2 - 4*a*c]]))/c
```

Maple [A] (verified)

Time = 2.20 (sec) , antiderivative size = 261, normalized size of antiderivative = 1.02

method	result
default	$\frac{2(a-b+c)\left(\frac{\left(-a\sqrt{-4ac+b^2}+b\sqrt{-4ac+b^2}-ab-2ac+b^2\right)\operatorname{arctanh}\left(\frac{(-a+b-c)\tan\left(\frac{x}{2}\right)}{\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}\right)}{2\sqrt{-4ac+b^2}\,(a-b+c)\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}+\frac{\left(-a\sqrt{-4ac+b^2}+b\sqrt{-4ac+b^2}+ab+2ac-b^2\right) a}{2\sqrt{-4ac+b^2}\,(a-b+c)\sqrt{\left(\sqrt{-\right.}}}}{c}$
risch	Expression too large to display

```
[In] int(cos(x)^2/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] 2/c*(a-b+c)*(1/2*(-a*(-4*a*c+b^2)^(1/2)+b*(-4*a*c+b^2)^(1/2)-a*b-2*a*c+b^2)
/(-4*a*c+b^2)^(1/2)/(a-b+c)/(((-4*a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2)*arctan
h((-a+b-c)*tan(1/2*x)/(((-4*a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2))+1/2*(-a*(-4
*a*c+b^2)^(1/2)+b*(-4*a*c+b^2)^(1/2)+a*b+2*a*c-b^2)/(-4*a*c+b^2)^(1/2)/(a-b
+c)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)*arctan((a-b+c)*tan(1/2*x)/(((-
4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)))+2/c*arctan(tan(1/2*x))
```

Fricas [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 4983 vs. $2(215) = 430$.

Time = 1.06 (sec) , antiderivative size = 4983, normalized size of antiderivative = 19.54

$$\int \frac{\cos^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(cos(x)^2/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
```

```
[Out] -1/4*(sqrt(2)*c*sqrt((a^2*b^2 - b^4 - 2*a^2*c^2 - 2*(a^3 - 2*a*b^2)*c - (4*
a*c^5 + (8*a^2 - b^2)*c^4 + 2*(2*a^3 - 3*a*b^2)*c^3 - (a^2*b^2 - b^4)*c^2)*
sqrt(-(a^4*b^2 - 2*a^2*b^4 + b^6 + 4*a^2*b^2*c^2 + 4*(a^3*b^2 - a*b^4)*c)/(
4*a*c^9 + (16*a^2 - b^2)*c^8 + 12*(2*a^3 - a*b^2)*c^7 + 2*(8*a^4 - 11*a^2*b
^2 + b^4)*c^6 + 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c^5 - (a^4*b^2 - 2*a^2*b^4 +
b^6)*c^4)))/(4*a*c^5 + (8*a^2 - b^2)*c^4 + 2*(2*a^3 - 3*a*b^2)*c^3 - (a^2*b
^2 - b^4)*c^2))*log(4*a^3*b*c^2 - (4*a^3*c^5 + (8*a^4 - a^2*b^2)*c^4 + 2*(2
*a^5 - 3*a^3*b^2)*c^3 - (a^4*b^2 - a^2*b^4)*c^2)*sqrt(-(a^4*b^2 - 2*a^2*b^4
 + b^6 + 4*a^2*b^2*c^2 + 4*(a^3*b^2 - a*b^4)*c)/(4*a*c^9 + (16*a^2 - b^2)*c
^8 + 12*(2*a^3 - a*b^2)*c^7 + 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^6 + 4*(a^5 - 3
*a^3*b^2 + 2*a*b^4)*c^5 - (a^4*b^2 - 2*a^2*b^4 + b^6)*c^4))*cos(x) + 2*(a^4
*b - a^2*b^3)*c + 1/2*sqrt(2)*((8*a^2*c^7 + 6*(4*a^3 - a*b^2)*c^6 + (24*a^4
 - 22*a^2*b^2 + b^4)*c^5 + 2*(4*a^5 - 9*a^3*b^2 + 4*a*b^4)*c^4 - (2*a^4*b^2
 - 3*a^2*b^4 + b^6)*c^3)*sqrt(-(a^4*b^2 - 2*a^2*b^4 + b^6 + 4*a^2*b^2*c^2 +
 4*(a^3*b^2 - a*b^4)*c)/(4*a*c^9 + (16*a^2 - b^2)*c^8 + 12*(2*a^3 - a*b^2)*
c^7 + 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^6 + 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c^5
- (a^4*b^2 - 2*a^2*b^4 + b^6)*c^4))*sin(x) + (8*a^2*b^2*c^3 + 2*(2*a^3*b^2
```

```
- 3*a*b^4)*c^2 - (a^2*b^4 - b^6)*c)*sin(x))*sqrt((a^2*b^2 - b^4 - 2*a^2*c^2
 - 2*(a^3 - 2*a*b^2)*c - (4*a*c^5 + (8*a^2 - b^2)*c^4 + 2*(2*a^3 - 3*a*b^2)
*c^3 - (a^2*b^2 - b^4)*c^2)*sqrt(-(a^4*b^2 - 2*a^2*b^4 + b^6 + 4*a^2*b^2*c^
2 + 4*(a^3*b^2 - a*b^4)*c)/(4*a*c^9 + (16*a^2 - b^2)*c^8 + 12*(2*a^3 - a*b^
2)*c^7 + 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^6 + 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c
^5 - (a^4*b^2 - 2*a^2*b^4 + b^6)*c^4)))/(4*a*c^5 + (8*a^2 - b^2)*c^4 + 2*(2
*a^3 - 3*a*b^2)*c^3 - (a^2*b^2 - b^4)*c^2)) + (a^4*b^2 - a^2*b^4 + 2*a^3*b^
2*c)*cos(x)) - sqrt(2)*c*sqrt((a^2*b^2 - b^4 - 2*a^2*c^2 - 2*(a^3 - 2*a*b^2
)*c - (4*a*c^5 + (8*a^2 - b^2)*c^4 + 2*(2*a^3 - 3*a*b^2)*c^3 - (a^2*b^2 - b
^4)*c^2)*sqrt(-(a^4*b^2 - 2*a^2*b^4 + b^6 + 4*a^2*b^2*c^2 + 4*(a^3*b^2 - a*
b^4)*c)/(4*a*c^9 + (16*a^2 - b^2)*c^8 + 12*(2*a^3 - a*b^2)*c^7 + 2*(8*a^4 -
 11*a^2*b^2 + b^4)*c^6 + 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c^5 - (a^4*b^2 - 2*a
^2*b^4 + b^6)*c^4)))/(4*a*c^5 + (8*a^2 - b^2)*c^4 + 2*(2*a^3 - 3*a*b^2)*c^3
 - (a^2*b^2 - b^4)*c^2))*log(4*a^3*b*c^2 - (4*a^3*c^5 + (8*a^4 - a^2*b^2)*c
^4 + 2*(2*a^5 - 3*a^3*b^2)*c^3 - (a^4*b^2 - a^2*b^4)*c^2)*sqrt(-(a^4*b^2 -
2*a^2*b^4 + b^6 + 4*a^2*b^2*c^2 + 4*(a^3*b^2 - a*b^4)*c)/(4*a*c^9 + (16*a^2
 - b^2)*c^8 + 12*(2*a^3 - a*b^2)*c^7 + 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^6 + 4
*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c^5 - (a^4*b^2 - 2*a^2*b^4 + b^6)*c^4))*cos(x)
 + 2*(a^4*b - a^2*b^3)*c - 1/2*sqrt(2)*((8*a^2*c^7 + 6*(4*a^3 - a*b^2)*c^6
+ (24*a^4 - 22*a^2*b^2 + b^4)*c^5 + 2*(4*a^5 - 9*a^3*b^2 + 4*a*b^4)*c^4 - (
2*a^4*b^2 - 3*a^2*b^4 + b^6)*c^3)*sqrt(-(a^4*b^2 - 2*a^2*b^4 + b^6 + 4*a^2*
b^2*c^2 + 4*(a^3*b^2 - a*b^4)*c)/(4*a*c^9 + (16*a^2 - b^2)*c^8 + 12*(2*a^3
- a*b^2)*c^7 + 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^6 + 4*(a^5 - 3*a^3*b^2 + 2*a*
b^4)*c^5 - (a^4*b^2 - 2*a^2*b^4 + b^6)*c^4))*sin(x) + (8*a^2*b^2*c^3 + 2*(2
*a^3*b^2 - 3*a*b^4)*c^2 - (a^2*b^4 - b^6)*c)*sin(x))*sqrt((a^2*b^2 - b^4 -
2*a^2*c^2 - 2*(a^3 - 2*a*b^2)*c - (4*a*c^5 + (8*a^2 - b^2)*c^4 + 2*(2*a^3 -
 3*a*b^2)*c^3 - (a^2*b^2 - b^4)*c^2)*sqrt(-(a^4*b^2 - 2*a^2*b^4 + b^6 + 4*a
^2*b^2*c^2 + 4*(a^3*b^2 - a*b^4)*c)/(4*a*c^9 + (16*a^2 - b^2)*c^8 + 12*(2*a
^3 - a*b^2)*c^7 + 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^6 + 4*(a^5 - 3*a^3*b^2 + 2
*a*b^4)*c^5 - (a^4*b^2 - 2*a^2*b^4 + b^6)*c^4)))/(4*a*c^5 + (8*a^2 - b^2)*c
^4 + 2*(2*a^3 - 3*a*b^2)*c^3 - (a^2*b^2 - b^4)*c^2)) + (a^4*b^2 - a^2*b^4 +
 2*a^3*b^2*c)*cos(x)) + sqrt(2)*c*sqrt((a^2*b^2 - b^4 - 2*a^2*c^2 - 2*(a^3
- 2*a*b^2)*c + (4*a*c^5 + (8*a^2 - b^2)*c^4 + 2*(2*a^3 - 3*a*b^2)*c^3 - (a^
2*b^2 - b^4)*c^2)*sqrt(-(a^4*b^2 - 2*a^2*b^4 + b^6 + 4*a^2*b^2*c^2 + 4*(a^3
*b^2 - a*b^4)*c)/(4*a*c^9 + (16*a^2 - b^2)*c^8 + 12*(2*a^3 - a*b^2)*c^7 + 2
*(8*a^4 - 11*a^2*b^2 + b^4)*c^6 + 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c^5 - (a^4*
b^2 - 2*a^2*b^4 + b^6)*c^4)))/(4*a*c^5 + (8*a^2 - b^2)*c^4 + 2*(2*a^3 - 3*a
*b^2)*c^3 - (a^2*b^2 - b^4)*c^2))*log(-4*a^3*b*c^2 - (4*a^3*c^5 + (8*a^4 -
a^2*b^2)*c^4 + 2*(2*a^5 - 3*a^3*b^2)*c^3 - (a^4*b^2 - a^2*b^4)*c^2)*sqrt(-(
a^4*b^2 - 2*a^2*b^4 + b^6 + 4*a^2*b^2*c^2 + 4*(a^3*b^2 - a*b^4)*c)/(4*a*c^9
 + (16*a^2 - b^2)*c^8 + 12*(2*a^3 - a*b^2)*c^7 + 2*(8*a^4 - 11*a^2*b^2 + b^
4)*c^6 + 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c^5 - (a^4*b^2 - 2*a^2*b^4 + b^6)*c^
4))*cos(x) - 2*(a^4*b - a^2*b^3)*c + 1/2*sqrt(2)*((8*a^2*c^7 + 6*(4*a^3 - a
*b^2)*c^6 + (24*a^4 - 22*a^2*b^2 + b^4)*c^5 + 2*(4*a^5 - 9*a^3*b^2 + 4*a*b^
4)*c^4 - (2*a^4*b^2 - 3*a^2*b^4 + b^6)*c^3)*sqrt(-(a^4*b^2 - 2*a^2*b^4 + b^
```

```
6 + 4*a^2*b^2*c^2 + 4*(a^3*b^2 - a*b^4)*c)/(4*a*c^9 + (16*a^2 - b^2)*c^8 +
12*(2*a^3 - a*b^2)*c^7 + 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^6 + 4*(a^5 - 3*a^3*
b^2 + 2*a*b^4)*c^5 - (a^4*b^2 - 2*a^2*b^4 + b^6)*c^4))*sin(x) - (8*a^2*b^2*
c^3 + 2*(2*a^3*b^2 - 3*a*b^4)*c^2 - (a^2*b^4 - b^6)*c)*sin(x))*sqrt((a^2*b^
2 - b^4 - 2*a^2*c^2 - 2*(a^3 - 2*a*b^2)*c + (4*a*c^5 + (8*a^2 - b^2)*c^4 +
2*(2*a^3 - 3*a*b^2)*c^3 - (a^2*b^2 - b^4)*c^2)*sqrt(-(a^4*b^2 - 2*a^2*b^4 +
 b^6 + 4*a^2*b^2*c^2 + 4*(a^3*b^2 - a*b^4)*c)/(4*a*c^9 + (16*a^2 - b^2)*c^8
 + 12*(2*a^3 - a*b^2)*c^7 + 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^6 + 4*(a^5 - 3*a
^3*b^2 + 2*a*b^4)*c^5 - (a^4*b^2 - 2*a^2*b^4 + b^6)*c^4)))/(4*a*c^5 + (8*a^
2 - b^2)*c^4 + 2*(2*a^3 - 3*a*b^2)*c^3 - (a^2*b^2 - b^4)*c^2)) - (a^4*b^2 -
 a^2*b^4 + 2*a^3*b^2*c)*cos(x)) - sqrt(2)*c*sqrt((a^2*b^2 - b^4 - 2*a^2*c^2
 - 2*(a^3 - 2*a*b^2)*c + (4*a*c^5 + (8*a^2 - b^2)*c^4 + 2*(2*a^3 - 3*a*b^2)
*c^3 - (a^2*b^2 - b^4)*c^2)*sqrt(-(a^4*b^2 - 2*a^2*b^4 + b^6 + 4*a^2*b^2*c^
2 + 4*(a^3*b^2 - a*b^4)*c)/(4*a*c^9 + (16*a^2 - b^2)*c^8 + 12*(2*a^3 - a*b^
2)*c^7 + 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^6 + 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c
^5 - (a^4*b^2 - 2*a^2*b^4 + b^6)*c^4)))/(4*a*c^5 + (8*a^2 - b^2)*c^4 + 2*(2
*a^3 - 3*a*b^2)*c^3 - (a^2*b^2 - b^4)*c^2))*log(-4*a^3*b*c^2 - (4*a^3*c^5 +
 (8*a^4 - a^2*b^2)*c^4 + 2*(2*a^5 - 3*a^3*b^2)*c^3 - (a^4*b^2 - a^2*b^4)*c^
2)*sqrt(-(a^4*b^2 - 2*a^2*b^4 + b^6 + 4*a^2*b^2*c^2 + 4*(a^3*b^2 - a*b^4)*c
)/(4*a*c^9 + (16*a^2 - b^2)*c^8 + 12*(2*a^3 - a*b^2)*c^7 + 2*(8*a^4 - 11*a^
2*b^2 + b^4)*c^6 + 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c^5 - (a^4*b^2 - 2*a^2*b^4
 + b^6)*c^4))*cos(x) - 2*(a^4*b - a^2*b^3)*c - 1/2*sqrt(2)*((8*a^2*c^7 + 6*
(4*a^3 - a*b^2)*c^6 + (24*a^4 - 22*a^2*b^2 + b^4)*c^5 + 2*(4*a^5 - 9*a^3*b^
2 + 4*a*b^4)*c^4 - (2*a^4*b^2 - 3*a^2*b^4 + b^6)*c^3)*sqrt(-(a^4*b^2 - 2*a^
2*b^4 + b^6 + 4*a^2*b^2*c^2 + 4*(a^3*b^2 - a*b^4)*c)/(4*a*c^9 + (16*a^2 - b
^2)*c^8 + 12*(2*a^3 - a*b^2)*c^7 + 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^6 + 4*(a^
5 - 3*a^3*b^2 + 2*a*b^4)*c^5 - (a^4*b^2 - 2*a^2*b^4 + b^6)*c^4))*sin(x) - (
8*a^2*b^2*c^3 + 2*(2*a^3*b^2 - 3*a*b^4)*c^2 - (a^2*b^4 - b^6)*c)*sin(x))*sq
rt((a^2*b^2 - b^4 - 2*a^2*c^2 - 2*(a^3 - 2*a*b^2)*c + (4*a*c^5 + (8*a^2 - b
^2)*c^4 + 2*(2*a^3 - 3*a*b^2)*c^3 - (a^2*b^2 - b^4)*c^2)*sqrt(-(a^4*b^2 - 2
*a^2*b^4 + b^6 + 4*a^2*b^2*c^2 + 4*(a^3*b^2 - a*b^4)*c)/(4*a*c^9 + (16*a^2
- b^2)*c^8 + 12*(2*a^3 - a*b^2)*c^7 + 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^6 + 4*
(a^5 - 3*a^3*b^2 + 2*a*b^4)*c^5 - (a^4*b^2 - 2*a^2*b^4 + b^6)*c^4)))/(4*a*c
^5 + (8*a^2 - b^2)*c^4 + 2*(2*a^3 - 3*a*b^2)*c^3 - (a^2*b^2 - b^4)*c^2)) -
(a^4*b^2 - a^2*b^4 + 2*a^3*b^2*c)*cos(x)) - 4*x)/c
```

Sympy [F(-1)]

Timed out.

$$\int \frac{\cos^2(x)}{a + b\cos(x) + c\cos^2(x)}\, dx = \text{Timed out}$$

```
[In] integrate(cos(x)**2/(a+b*cos(x)+c*cos(x)**2),x)
[Out] Timed out
```

Maxima [F]

$$\int \frac{\cos^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx=\int \frac{\cos(x)^2}{c\cos(x)^2+b\cos(x)+a}\,dx$$

```
[In] integrate(cos(x)^2/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
```

```
[Out] -(c*integrate(2*(2*b^2*cos(3*x)^2 + 2*b^2*cos(x)^2 + 2*b^2*sin(3*x)^2 + 2*b
^2*sin(x)^2 + 4*(2*a^2 + a*c)*cos(2*x)^2 + b*c*cos(x) + 4*(2*a^2 + a*c)*sin
(2*x)^2 + 2*(4*a*b + b*c)*sin(2*x)*sin(x) + (b*c*cos(3*x) + 2*a*c*cos(2*x)
+ b*c*cos(x))*cos(4*x) + (4*b^2*cos(x) + b*c + 2*(4*a*b + b*c)*cos(2*x))*co
s(3*x) + 2*(a*c + (4*a*b + b*c)*cos(x))*cos(2*x) + (b*c*sin(3*x) + 2*a*c*si
n(2*x) + b*c*sin(x))*sin(4*x) + 2*(2*b^2*sin(x) + (4*a*b + b*c)*sin(2*x))*s
in(3*x))/(c^3*cos(4*x)^2 + 4*b^2*c*cos(3*x)^2 + 4*b^2*c*cos(x)^2 + c^3*sin(
4*x)^2 + 4*b^2*c*sin(3*x)^2 + 4*b^2*c*sin(x)^2 + 4*b*c^2*cos(x) + c^3 + 4*(
4*a^2*c + 4*a*c^2 + c^3)*cos(2*x)^2 + 4*(4*a^2*c + 4*a*c^2 + c^3)*sin(2*x)^
2 + 8*(2*a*b*c + b*c^2)*sin(2*x)*sin(x) + 2*(2*b*c^2*cos(3*x) + 2*b*c^2*cos
(x) + c^3 + 2*(2*a*c^2 + c^3)*cos(2*x))*cos(4*x) + 4*(2*b^2*c*cos(x) + b*c^
2 + 2*(2*a*b*c + b*c^2)*cos(2*x))*cos(3*x) + 4*(2*a*c^2 + c^3 + 2*(2*a*b*c
+ b*c^2)*cos(x))*cos(2*x) + 4*(b*c^2*sin(3*x) + b*c^2*sin(x) + (2*a*c^2 + c
^3)*sin(2*x))*sin(4*x) + 8*(b^2*c*sin(x) + (2*a*b*c + b*c^2)*sin(2*x))*sin(
3*x)), x) - x)/c
```

Giac [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 9030 vs. $2(215) = 430$.

Time = 2.16 (sec) , antiderivative size = 9030, normalized size of antiderivative = 35.41

$$\int \frac{\cos^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx=\text{Too large to display}$$

```
[In] integrate(cos(x)^2/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] x/c + ((2*a^3*b^4 - 6*a^2*b^5 + 6*a*b^6 - 2*b^7 - 16*a^4*b^2*c + 48*a^3*b^3
*c - 44*a^2*b^4*c + 8*a*b^5*c + 4*b^6*c + 32*a^5*c^2 - 96*a^4*b*c^2 + 64*a^
3*b^2*c^2 + 32*a^2*b^3*c^2 - 30*a*b^4*c^2 - 2*b^5*c^2 + 64*a^4*c^3 - 128*a^
3*b*c^3 + 48*a^2*b^2*c^3 + 16*a*b^3*c^3 + 32*a^3*c^4 - 32*a^2*b*c^4 + 3*sqr
t(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*
a^3*b^2 - 2*(b^2 - 4*a*c)*a^3*b^2 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3 + 6*(b^2 - 4*a*c)*a^2*b^3
- 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
4*a*c)*a*b^4 - 6*(b^2 - 4*a*c)*a*b^4 + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(
b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5 + 2*(b^2 - 4*a*c)*b^5 - 12*
```

sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*
c)*a^4*c + 8*(b^2 - 4*a*c)*a^4*c + 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c - 24*(b^2 - 4*a*c)*a^3*b*c
 + 26*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*a^2*b^2*c + 20*(b^2 - 4*a*c)*a^2*b^2*c - 28*sqrt(a^2 - a*b + b*c -
 c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c - 6*sqrt(a^
2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*
c - 4*(b^2 - 4*a*c)*b^4*c - 56*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*
c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^2 + 16*(b^2 - 4*a*c)*a^3*c^2 + 32*s
qrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c
)*a^2*b*c^2 - 32*(b^2 - 4*a*c)*a^2*b*c^2 + 19*sqrt(a^2 - a*b + b*c - c^2 +
sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^2 + 14*(b^2 - 4*a*
c)*a*b^2*c^2 + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c)
)*sqrt(b^2 - 4*a*c)*b^3*c^2 + 2*(b^2 - 4*a*c)*b^3*c^2 + 20*sqrt(a^2 - a*b +
 b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^3 + 8*(
b^2 - 4*a*c)*a^2*c^3 - 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
 - b + c))*sqrt(b^2 - 4*a*c)*a*b*c^3 - 8*(b^2 - 4*a*c)*a*b*c^3)*c^2*abs(a -
 b + c) - (4*a^3*b^4*c - 4*a^2*b^5*c - 4*a*b^6*c + 4*b^7*c - 32*a^4*b^2*c^2
 + 32*a^3*b^3*c^2 + 40*a^2*b^4*c^2 - 32*a*b^5*c^2 - 8*b^6*c^2 + 64*a^5*c^3
- 64*a^4*b*c^3 - 128*a^3*b^2*c^3 + 64*a^2*b^3*c^3 + 68*a*b^4*c^3 + 4*b^5*c^
3 + 128*a^4*c^4 - 160*a^2*b^2*c^4 - 32*a*b^3*c^4 + 64*a^3*c^5 + 64*a^2*b*c^
5 + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*b^2*c
 - 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^3*c
- 8*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^4*c +
 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^5*c + 5*
sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^6*c - 12*sqrt
(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^5*c^2 + 8*sqrt(a^
2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*b*c^2 + 49*sqrt(a^
2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^2*c^2 - sqrt(a^2
 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^3*c^2 - 41*sqrt(a
^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^4*c^2 - 11*sqrt(a
^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^5*c^2 - 68*sqrt(a^2
 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*c^3 - 28*sqrt(a^2 -
 a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b*c^3 + 93*sqrt(a^2 -
 a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^2*c^3 + 64*sqrt(a^2
 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^3*c^3 + 11*sqrt(a^2
 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^4*c^3 - 36*sqrt(a^2 -
 a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*c^4 - 80*sqrt(a^2 - a
*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b*c^4 - 49*sqrt(a^2 - a
*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^2*c^4 - 5*sqrt(a^2 - a*
b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^3*c^4 + 20*sqrt(a^2 - a*b
+ b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*c^5 + 20*sqrt(a^2 - a*b +
b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b*c^5 - 4*(b^2 - 4*a*c)*a^3*b^
2*c + 4*(b^2 - 4*a*c)*a^2*b^3*c + 4*(b^2 - 4*a*c)*a*b^4*c - 4*(b^2 - 4*a*c)
*b^5*c + 16*(b^2 - 4*a*c)*a^4*c^2 - 16*(b^2 - 4*a*c)*a^3*b*c^2 - 24*(b^2 -

```
4*a*c)*a^2*b^2*c^2 + 16*(b^2 - 4*a*c)*a*b^3*c^2 + 8*(b^2 - 4*a*c)*b^4*c^2 +
 32*(b^2 - 4*a*c)*a^3*c^3 - 36*(b^2 - 4*a*c)*a*b^2*c^3 - 4*(b^2 - 4*a*c)*b^
3*c^3 + 16*(b^2 - 4*a*c)*a^2*c^4 + 16*(b^2 - 4*a*c)*a*b*c^4)*abs(a - b + c)
*abs(c) + (2*a^4*b^3*c^2 - 6*a^3*b^4*c^2 + 6*a^2*b^5*c^2 - 2*a*b^6*c^2 - 8*
a^5*b*c^3 + 28*a^4*b^2*c^3 - 30*a^3*b^3*c^3 + 10*a^2*b^4*c^3 - 2*a*b^5*c^3
+ 2*b^6*c^3 - 16*a^5*c^4 + 24*a^4*b*c^4 - 4*a^3*b^2*c^4 + 6*a^2*b^3*c^4 - 6
*a*b^4*c^4 - 4*b^5*c^4 - 16*a^4*c^5 + 8*a^3*b*c^5 - 12*a^2*b^2*c^5 + 22*a*b
^3*c^5 + 2*b^4*c^5 + 16*a^3*c^6 - 24*a^2*b*c^6 - 12*a*b^2*c^6 + 16*a^2*c^7
+ 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
4*a*c)*a^4*b*c^2 - 2*(b^2 - 4*a*c)*a^4*b*c^2 - 5*sqrt(a^2 - a*b + b*c - c^2
 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c^2 + 6*(b^2 -
4*a*c)*a^3*b^2*c^2 - 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3*c^2 - 6*(b^2 - 4*a*c)*a^2*b^3*c^2 + 5*sqr
t(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*
a*b^4*c^2 + 2*(b^2 - 4*a*c)*a*b^4*c^2 + 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt
(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*c^3 - 4*(b^2 - 4*a*c)*a^4*
c^3 + 7*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^
2 - 4*a*c)*a^3*b*c^3 + 6*(b^2 - 4*a*c)*a^3*b*c^3 - 13*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c^3 - 2*(b
^2 - 4*a*c)*a^2*b^2*c^3 - 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*
(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c^3 + 2*(b^2 - 4*a*c)*a*b^3*c^3 - 5*sq
rt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)
*b^4*c^3 - 2*(b^2 - 4*a*c)*b^4*c^3 + 22*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b
^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^4 - 4*(b^2 - 4*a*c)*a^3*c^
4 - 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*a^2*b*c^4 + 2*(b^2 - 4*a*c)*a^2*b*c^4 + 23*sqrt(a^2 - a*b + b*c -
c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^4 - 2*(b^2 -
 4*a*c)*a*b^2*c^4 + 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
 + c))*sqrt(b^2 - 4*a*c)*b^3*c^4 + 4*(b^2 - 4*a*c)*b^3*c^4 - 38*sqrt(a^2 -
a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^5
+ 4*(b^2 - 4*a*c)*a^2*c^5 - 7*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*c^5 - 6*(b^2 - 4*a*c)*a*b*c^5 - 5*sqrt
(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b
^2*c^5 - 2*(b^2 - 4*a*c)*b^2*c^5 + 10*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*c^6 + 4*(b^2 - 4*a*c)*a*c^6)*abs
(a - b + c))*(pi*floor(1/2*x/pi + 1/2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt
((2*a*c - 2*c^2 + sqrt(-4*(a*c + b*c + c^2)*(a*c - b*c + c^2) + 4*(a*c - c^
2)^2))/(a*c - b*c + c^2)))))/((3*a^6*b^2*c^2 - 8*a^5*b^3*c^2 - a^4*b^4*c^2 +
 16*a^3*b^5*c^2 - 7*a^2*b^6*c^2 - 8*a*b^7*c^2 + 5*b^8*c^2 - 12*a^7*c^3 + 32
*a^6*b*c^3 + 30*a^5*b^2*c^3 - 112*a^4*b^3*c^3 + 8*a^3*b^4*c^3 + 96*a^2*b^5*
c^3 - 26*a*b^6*c^3 - 16*b^7*c^3 - 104*a^6*c^4 + 192*a^5*b*c^4 + 149*a^4*b^2
*c^4 - 336*a^3*b^3*c^4 - 30*a^2*b^4*c^4 + 112*a*b^5*c^4 + 17*b^6*c^4 - 276*
a^5*c^5 + 320*a^4*b*c^5 + 292*a^3*b^2*c^5 - 224*a^2*b^3*c^5 - 120*a*b^4*c^5
 - 304*a^4*c^6 + 128*a^3*b*c^6 + 237*a^2*b^2*c^6 + 24*a*b^3*c^6 - 17*b^4*c^
6 - 116*a^3*c^7 - 96*a^2*b*c^7 + 62*a*b^2*c^7 + 16*b^3*c^7 + 24*a^2*c^8 - 6
```

4*a*b*c^8 - 5*b^2*c^8 + 20*a*c^9)*abs(c)) - ((2*a^3*b^4 - 6*a^2*b^5 + 6*a*b
^6 - 2*b^7 - 16*a^4*b^2*c + 48*a^3*b^3*c - 44*a^2*b^4*c + 8*a*b^5*c + 4*b^6
*c + 32*a^5*c^2 - 96*a^4*b*c^2 + 64*a^3*b^2*c^2 + 32*a^2*b^3*c^2 - 30*a*b^4
*c^2 - 2*b^5*c^2 + 64*a^4*c^3 - 128*a^3*b*c^3 + 48*a^2*b^2*c^3 + 16*a*b^3*c
^3 + 32*a^3*c^4 - 32*a^2*b*c^4 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2 - 2*(b^2 - 4*a*c)*a^3*b^2 - 5
*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a
*c)*a^2*b^3 + 6*(b^2 - 4*a*c)*a^2*b^3 - 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt
(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4 - 6*(b^2 - 4*a*c)*a*b^4
+ 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
4*a*c)*b^5 + 2*(b^2 - 4*a*c)*b^5 - 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*c + 8*(b^2 - 4*a*c)*a^4*c + 20
*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a
*c)*a^3*b*c - 24*(b^2 - 4*a*c)*a^3*b*c + 26*sqrt(a^2 - a*b + b*c - c^2 - sq
rt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c + 20*(b^2 - 4*a*c)
*a^2*b^2*c - 28*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))
*sqrt(b^2 - 4*a*c)*a*b^3*c - 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c - 4*(b^2 - 4*a*c)*b^4*c - 56*sqrt(a
^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3
c^2 + 16(b^2 - 4*a*c)*a^3*c^2 + 32*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^2 - 32*(b^2 - 4*a*c)*a^2*b*
c^2 + 19*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b
^2 - 4*a*c)*a*b^2*c^2 + 14*(b^2 - 4*a*c)*a*b^2*c^2 + 5*sqrt(a^2 - a*b + b*c
 - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^3*c^2 + 2*(b^2
 - 4*a*c)*b^3*c^2 + 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
 + c))*sqrt(b^2 - 4*a*c)*a^2*c^3 + 8*(b^2 - 4*a*c)*a^2*c^3 - 20*sqrt(a^2 -
a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*c^3
 - 8*(b^2 - 4*a*c)*a*b*c^3)*c^2*abs(a - b + c) - (4*a^3*b^4*c - 4*a^2*b^5*c
 - 4*a*b^6*c + 4*b^7*c - 32*a^4*b^2*c^2 + 32*a^3*b^3*c^2 + 40*a^2*b^4*c^2 -
32*a*b^5*c^2 - 8*b^6*c^2 + 64*a^5*c^3 - 64*a^4*b*c^3 - 128*a^3*b^2*c^3 + 64
*a^2*b^3*c^3 + 68*a*b^4*c^3 + 4*b^5*c^3 + 128*a^4*c^4 - 160*a^2*b^2*c^4 - 3
2*a*b^3*c^4 + 64*a^3*c^5 + 64*a^2*b*c^5 - 3*sqrt(a^2 - a*b + b*c - c^2 - sq
rt(b^2 - 4*a*c)*(a - b + c))*a^4*b^2*c + 2*sqrt(a^2 - a*b + b*c - c^2 - sqr
t(b^2 - 4*a*c)*(a - b + c))*a^3*b^3*c + 8*sqrt(a^2 - a*b + b*c - c^2 - sqrt
(b^2 - 4*a*c)*(a - b + c))*a^2*b^4*c - 2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(
b^2 - 4*a*c)*(a - b + c))*a*b^5*c - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
 - 4*a*c)*(a - b + c))*b^6*c + 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4
*a*c)*(a - b + c))*a^5*c^2 - 8*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*a^4*b*c^2 - 49*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*a^3*b^2*c^2 + sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*a^2*b^3*c^2 + 41*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a
c)(a - b + c))*a*b^4*c^2 + 11*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a
c)(a - b + c))*b^5*c^2 + 68*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*a^4*c^3 + 28*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*
(a - b + c))*a^3*b*c^3 - 93*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*

```
(a - b + c))*a^2*b^2*c^3 - 64*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*a*b^3*c^3 - 11*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*b^4*c^3 + 36*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*
(a - b + c))*a^3*c^4 + 80*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*a^2*b*c^4 + 49*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*a*b^2*c^4 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*b^3*c^4 - 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*a^2*c^5 - 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*a*b*c^5 - 4*(b^2 - 4*a*c)*a^3*b^2*c + 4*(b^2 - 4*a*c)*a^2*b^3*c + 4*(
b^2 - 4*a*c)*a*b^4*c - 4*(b^2 - 4*a*c)*b^5*c + 16*(b^2 - 4*a*c)*a^4*c^2 - 1
6*(b^2 - 4*a*c)*a^3*b*c^2 - 24*(b^2 - 4*a*c)*a^2*b^2*c^2 + 16*(b^2 - 4*a*c)
*a*b^3*c^2 + 8*(b^2 - 4*a*c)*b^4*c^2 + 32*(b^2 - 4*a*c)*a^3*c^3 - 36*(b^2 -
 4*a*c)*a*b^2*c^3 - 4*(b^2 - 4*a*c)*b^3*c^3 + 16*(b^2 - 4*a*c)*a^2*c^4 + 16
*(b^2 - 4*a*c)*a*b*c^4)*abs(a - b + c)*abs(c) + (2*a^4*b^3*c^2 - 6*a^3*b^4*
c^2 + 6*a^2*b^5*c^2 - 2*a*b^6*c^2 - 8*a^5*b*c^3 + 28*a^4*b^2*c^3 - 30*a^3*b
^3*c^3 + 10*a^2*b^4*c^3 - 2*a*b^5*c^3 + 2*b^6*c^3 - 16*a^5*c^4 + 24*a^4*b*c
^4 - 4*a^3*b^2*c^4 + 6*a^2*b^3*c^4 - 6*a*b^4*c^4 - 4*b^5*c^4 - 16*a^4*c^5 +
 8*a^3*b*c^5 - 12*a^2*b^2*c^5 + 22*a*b^3*c^5 + 2*b^4*c^5 + 16*a^3*c^6 - 24*
a^2*b*c^6 - 12*a*b^2*c^6 + 16*a^2*c^7 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt
(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b*c^2 - 2*(b^2 - 4*a*c)*a^
4*b*c^2 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*a^3*b^2*c^2 + 6*(b^2 - 4*a*c)*a^3*b^2*c^2 - 3*sqrt(a^2 - a*b
 + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3*c^2
 - 6*(b^2 - 4*a*c)*a^2*b^3*c^2 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4*c^2 + 2*(b^2 - 4*a*c)*a*b^4*c^2
 + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
 4*a*c)*a^4*c^3 - 4*(b^2 - 4*a*c)*a^4*c^3 + 7*sqrt(a^2 - a*b + b*c - c^2 -
sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^3 + 6*(b^2 - 4*a*c
)*a^3*b*c^3 - 13*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c)
)*sqrt(b^2 - 4*a*c)*a^2*b^2*c^3 - 2*(b^2 - 4*a*c)*a^2*b^2*c^3 - 3*sqrt(a^2
- a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*
c^3 + 2*(b^2 - 4*a*c)*a*b^3*c^3 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c^3 - 2*(b^2 - 4*a*c)*b^4*c^3 +
22*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4
*a*c)*a^3*c^4 - 4*(b^2 - 4*a*c)*a^3*c^4 - 3*sqrt(a^2 - a*b + b*c - c^2 - sq
rt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^4 + 2*(b^2 - 4*a*c)*
a^2*b*c^4 + 23*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*
sqrt(b^2 - 4*a*c)*a*b^2*c^4 - 2*(b^2 - 4*a*c)*a*b^2*c^4 + 6*sqrt(a^2 - a*b
+ b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^3*c^4 + 4*
(b^2 - 4*a*c)*b^3*c^4 - 38*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(
a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^5 + 4*(b^2 - 4*a*c)*a^2*c^5 - 7*sqrt(a^
2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*
c^5 - 6*(b^2 - 4*a*c)*a*b*c^5 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4
*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c^5 - 2*(b^2 - 4*a*c)*b^2*c^5 + 10
*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a
```

```
*c)*a*c^6 + 4*(b^2 - 4*a*c)*a*c^6)*abs(a - b + c))*(pi*floor(1/2*x/pi + 1/2
) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a*c - 2*c^2 - sqrt(-4*(a*c + b*c
+ c^2)*(a*c - b*c + c^2) + 4*(a*c - c^2)^2))/(a*c - b*c + c^2))))/((3*a^6*b
^2*c^2 - 8*a^5*b^3*c^2 - a^4*b^4*c^2 + 16*a^3*b^5*c^2 - 7*a^2*b^6*c^2 - 8*a
*b^7*c^2 + 5*b^8*c^2 - 12*a^7*c^3 + 32*a^6*b*c^3 + 30*a^5*b^2*c^3 - 112*a^4
*b^3*c^3 + 8*a^3*b^4*c^3 + 96*a^2*b^5*c^3 - 26*a*b^6*c^3 - 16*b^7*c^3 - 104
*a^6*c^4 + 192*a^5*b*c^4 + 149*a^4*b^2*c^4 - 336*a^3*b^3*c^4 - 30*a^2*b^4*c
^4 + 112*a*b^5*c^4 + 17*b^6*c^4 - 276*a^5*c^5 + 320*a^4*b*c^5 + 292*a^3*b^2
*c^5 - 224*a^2*b^3*c^5 - 120*a*b^4*c^5 - 304*a^4*c^6 + 128*a^3*b*c^6 + 237*
a^2*b^2*c^6 + 24*a*b^3*c^6 - 17*b^4*c^6 - 116*a^3*c^7 - 96*a^2*b*c^7 + 62*a
*b^2*c^7 + 16*b^3*c^7 + 24*a^2*c^8 - 64*a*b*c^8 - 5*b^2*c^8 + 20*a*c^9)*abs
(c))
```

Mupad [B] (verification not implemented)

Time = 16.96 (sec) , antiderivative size = 20133, normalized size of antiderivative = 78.95

$$\int \frac{\cos^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] int(cos(x)^2/(a + b*cos(x) + c*cos(x)^2),x)
```

```
[Out] (2*atan((540672*a^4*tan(x/2))/(131072*a*b^3 - 393216*a^3*b + 655360*a^3*c +
 540672*a^4 + 16384*b^4 - 360448*a^2*b^2 + 262144*a^2*c^2 + (147456*a^5)/c
- (16384*b^5)/c + (49152*a*b^4)/c - (147456*a^4*b)/c - (32768*a*b^5)/c^2 +
(229376*a^2*b^3)/c - (262144*a^3*b^2)/c + (32768*a^2*b^4)/c^2 + (32768*a^3*
b^3)/c^2 - (32768*a^4*b^2)/c^2 - 131072*a*b^2*c - 262144*a^2*b*c) + (16384*
b^4*tan(x/2))/(131072*a*b^3 - 393216*a^3*b + 655360*a^3*c + 540672*a^4 + 16
384*b^4 - 360448*a^2*b^2 + 262144*a^2*c^2 + (147456*a^5)/c - (16384*b^5)/c
+ (49152*a*b^4)/c - (147456*a^4*b)/c - (32768*a*b^5)/c^2 + (229376*a^2*b^3)
/c - (262144*a^3*b^2)/c + (32768*a^2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (3276
8*a^4*b^2)/c^2 - 131072*a*b^2*c - 262144*a^2*b*c) + (147456*a^5*tan(x/2))/(
49152*a*b^4 - 147456*a^4*b + 540672*a^4*c + 16384*b^4*c + 147456*a^5 - 1638
4*b^5 + 229376*a^2*b^3 - 262144*a^3*b^2 + 262144*a^2*c^3 + 655360*a^3*c^2 -
 131072*a*b^2*c^2 - 262144*a^2*b*c^2 - 360448*a^2*b^2*c - (32768*a*b^5)/c +
 (32768*a^2*b^4)/c + (32768*a^3*b^3)/c - (32768*a^4*b^2)/c + 131072*a*b^3*c
 - 393216*a^3*b*c) - (16384*b^5*tan(x/2))/(49152*a*b^4 - 147456*a^4*b + 540
672*a^4*c + 16384*b^4*c + 147456*a^5 - 16384*b^5 + 229376*a^2*b^3 - 262144*
a^3*b^2 + 262144*a^2*c^3 + 655360*a^3*c^2 - 131072*a*b^2*c^2 - 262144*a^2*b
*c^2 - 360448*a^2*b^2*c - (32768*a*b^5)/c + (32768*a^2*b^4)/c + (32768*a^3*
b^3)/c - (32768*a^4*b^2)/c + 131072*a*b^3*c - 393216*a^3*b*c) - (360448*a^2
*b^2*tan(x/2))/(131072*a*b^3 - 393216*a^3*b + 655360*a^3*c + 540672*a^4 + 1
6384*b^4 - 360448*a^2*b^2 + 262144*a^2*c^2 + (147456*a^5)/c - (16384*b^5)/c
 + (49152*a*b^4)/c - (147456*a^4*b)/c - (32768*a*b^5)/c^2 + (229376*a^2*b^3
)/c - (262144*a^3*b^2)/c + (32768*a^2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (327
```

68*a^4*b^2)/c^2 - 131072*a*b^2*c - 262144*a^2*b*c) + (262144*a^2*c^2*tan(x/
2))/(131072*a*b^3 - 393216*a^3*b + 655360*a^3*c + 540672*a^4 + 16384*b^4 -
360448*a^2*b^2 + 262144*a^2*c^2 + (147456*a^5)/c - (16384*b^5)/c + (49152*a
*b^4)/c - (147456*a^4*b)/c - (32768*a*b^5)/c^2 + (229376*a^2*b^3)/c - (2621
44*a^3*b^2)/c + (32768*a^2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32768*a^4*b^2)
/c^2 - 131072*a*b^2*c - 262144*a^2*b*c) + (49152*a*b^4*tan(x/2))/(49152*a*b
^4 - 147456*a^4*b + 540672*a^4*c + 16384*b^4*c + 147456*a^5 - 16384*b^5 + 2
29376*a^2*b^3 - 262144*a^3*b^2 + 262144*a^2*c^3 + 655360*a^3*c^2 - 131072*a
*b^2*c^2 - 262144*a^2*b*c^2 - 360448*a^2*b^2*c - (32768*a*b^5)/c + (32768*a
^2*b^4)/c + (32768*a^3*b^3)/c - (32768*a^4*b^2)/c + 131072*a*b^3*c - 393216
*a^3*b*c) - (147456*a^4*b*tan(x/2))/(49152*a*b^4 - 147456*a^4*b + 540672*a^
4*c + 16384*b^4*c + 147456*a^5 - 16384*b^5 + 229376*a^2*b^3 - 262144*a^3*b^
2 + 262144*a^2*c^3 + 655360*a^3*c^2 - 131072*a*b^2*c^2 - 262144*a^2*b*c^2 -
360448*a^2*b^2*c - (32768*a*b^5)/c + (32768*a^2*b^4)/c + (32768*a^3*b^3)/c
- (32768*a^4*b^2)/c + 131072*a*b^3*c - 393216*a^3*b*c) - (32768*a*b^5*tan(
x/2))/(147456*a^5*c - 32768*a*b^5 - 16384*b^5*c + 32768*a^2*b^4 + 32768*a^3
*b^3 - 32768*a^4*b^2 + 262144*a^2*c^4 + 655360*a^3*c^3 + 540672*a^4*c^2 + 1
6384*b^4*c^2 - 131072*a*b^2*c^3 + 131072*a*b^3*c^2 - 262144*a^2*b*c^3 + 229
376*a^2*b^3*c - 393216*a^3*b*c^2 - 262144*a^3*b^2*c - 360448*a^2*b^2*c^2 +
49152*a*b^4*c - 147456*a^4*b*c) + (229376*a^2*b^3*tan(x/2))/(49152*a*b^4 -
147456*a^4*b + 540672*a^4*c + 16384*b^4*c + 147456*a^5 - 16384*b^5 + 229376
*a^2*b^3 - 262144*a^3*b^2 + 262144*a^2*c^3 + 655360*a^3*c^2 - 131072*a*b^2*
c^2 - 262144*a^2*b*c^2 - 360448*a^2*b^2*c - (32768*a*b^5)/c + (32768*a^2*b^
4)/c + (32768*a^3*b^3)/c - (32768*a^4*b^2)/c + 131072*a*b^3*c - 393216*a^3*
b*c) - (262144*a^3*b^2*tan(x/2))/(49152*a*b^4 - 147456*a^4*b + 540672*a^4*c
+ 16384*b^4*c + 147456*a^5 - 16384*b^5 + 229376*a^2*b^3 - 262144*a^3*b^2 +
262144*a^2*c^3 + 655360*a^3*c^2 - 131072*a*b^2*c^2 - 262144*a^2*b*c^2 - 36
0448*a^2*b^2*c - (32768*a*b^5)/c + (32768*a^2*b^4)/c + (32768*a^3*b^3)/c -
(32768*a^4*b^2)/c + 131072*a*b^3*c - 393216*a^3*b*c) + (131072*a*b^3*tan(x/
2))/(131072*a*b^3 - 393216*a^3*b + 655360*a^3*c + 540672*a^4 + 16384*b^4 -
360448*a^2*b^2 + 262144*a^2*c^2 + (147456*a^5)/c - (16384*b^5)/c + (49152*a
*b^4)/c - (147456*a^4*b)/c - (32768*a*b^5)/c^2 + (229376*a^2*b^3)/c - (2621
44*a^3*b^2)/c + (32768*a^2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32768*a^4*b^2)
/c^2 - 131072*a*b^2*c - 262144*a^2*b*c) - (393216*a^3*b*tan(x/2))/(131072*a
*b^3 - 393216*a^3*b + 655360*a^3*c + 540672*a^4 + 16384*b^4 - 360448*a^2*b^
2 + 262144*a^2*c^2 + (147456*a^5)/c - (16384*b^5)/c + (49152*a*b^4)/c - (14
7456*a^4*b)/c - (32768*a*b^5)/c^2 + (229376*a^2*b^3)/c - (262144*a^3*b^2)/c
+ (32768*a^2*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32768*a^4*b^2)/c^2 - 131072
*a*b^2*c - 262144*a^2*b*c) + (655360*a^3*c*tan(x/2))/(131072*a*b^3 - 393216
*a^3*b + 655360*a^3*c + 540672*a^4 + 16384*b^4 - 360448*a^2*b^2 + 262144*a^
2*c^2 + (147456*a^5)/c - (16384*b^5)/c + (49152*a*b^4)/c - (147456*a^4*b)/c
- (32768*a*b^5)/c^2 + (229376*a^2*b^3)/c - (262144*a^3*b^2)/c + (32768*a^2
*b^4)/c^2 + (32768*a^3*b^3)/c^2 - (32768*a^4*b^2)/c^2 - 131072*a*b^2*c - 26
2144*a^2*b*c) + (32768*a^2*b^4*tan(x/2))/(147456*a^5*c - 32768*a*b^5 - 1638
4*b^5*c + 32768*a^2*b^4 + 32768*a^3*b^3 - 32768*a^4*b^2 + 262144*a^2*c^4 +

655360*a^3*c^3 + 540672*a^4*c^2 + 16384*b^4*c^2 - 131072*a*b^2*c^3 + 131072
*a*b^3*c^2 - 262144*a^2*b*c^3 + 229376*a^2*b^3*c - 393216*a^3*b*c^2 - 26214
4*a^3*b^2*c - 360448*a^2*b^2*c^2 + 49152*a*b^4*c - 147456*a^4*b*c) + (32768
*a^3*b^3*tan(x/2))/(147456*a^5*c - 32768*a*b^5 - 16384*b^5*c + 32768*a^2*b^
4 + 32768*a^3*b^3 - 32768*a^4*b^2 + 262144*a^2*c^4 + 655360*a^3*c^3 + 54067
2*a^4*c^2 + 16384*b^4*c^2 - 131072*a*b^2*c^3 + 131072*a*b^3*c^2 - 262144*a^
2*b*c^3 + 229376*a^2*b^3*c - 393216*a^3*b*c^2 - 262144*a^3*b^2*c - 360448*a
^2*b^2*c^2 + 49152*a*b^4*c - 147456*a^4*b*c) - (32768*a^4*b^2*tan(x/2))/(14
7456*a^5*c - 32768*a*b^5 - 16384*b^5*c + 32768*a^2*b^4 + 32768*a^3*b^3 - 32
768*a^4*b^2 + 262144*a^2*c^4 + 655360*a^3*c^3 + 540672*a^4*c^2 + 16384*b^4*
c^2 - 131072*a*b^2*c^3 + 131072*a*b^3*c^2 - 262144*a^2*b*c^3 + 229376*a^2*b
^3*c - 393216*a^3*b*c^2 - 262144*a^3*b^2*c - 360448*a^2*b^2*c^2 + 49152*a*b
^4*c - 147456*a^4*b*c) - (131072*a*b^2*c*tan(x/2))/(131072*a*b^3 - 393216*a
^3*b + 655360*a^3*c + 540672*a^4 + 16384*b^4 - 360448*a^2*b^2 + 262144*a^2*
c^2 + (147456*a^5)/c - (16384*b^5)/c + (49152*a*b^4)/c - (147456*a^4*b)/c -
(32768*a*b^5)/c^2 + (229376*a^2*b^3)/c - (262144*a^3*b^2)/c + (32768*a^2*b
^4)/c^2 + (32768*a^3*b^3)/c^2 - (32768*a^4*b^2)/c^2 - 131072*a*b^2*c - 2621
44*a^2*b*c) - (262144*a^2*b*c*tan(x/2))/(131072*a*b^3 - 393216*a^3*b + 6553
60*a^3*c + 540672*a^4 + 16384*b^4 - 360448*a^2*b^2 + 262144*a^2*c^2 + (1474
56*a^5)/c - (16384*b^5)/c + (49152*a*b^4)/c - (147456*a^4*b)/c - (32768*a*b
^5)/c^2 + (229376*a^2*b^3)/c - (262144*a^3*b^2)/c + (32768*a^2*b^4)/c^2 + (
32768*a^3*b^3)/c^2 - (32768*a^4*b^2)/c^2 - 131072*a*b^2*c - 262144*a^2*b*c)
))/c + atan(-(((-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b^
2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^2
+ 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^
5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^
2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*((tan(x/2)*(16384*a*b^6 - 6553
6*a*c^6 + 49152*b^6*c - 16384*b^7 + 16384*a^2*b^5 - 16384*a^3*b^4 + 245760*
a^2*c^5 + 671744*a^3*c^4 + 212992*a^4*c^3 - 147456*a^5*c^2 + 16384*b^2*c^5
- 49152*b^3*c^4 + 65536*b^4*c^3 - 65536*b^5*c^2 - 327680*a*b^2*c^4 + 475136
*a*b^3*c^3 - 393216*a*b^4*c^2 - 802816*a^2*b*c^4 - 180224*a^2*b^4*c - 10813
44*a^3*b*c^3 - 65536*a^3*b^3*c + 49152*a^4*b*c^2 + 98304*a^4*b^2*c + 557056
*a^2*b^2*c^3 + 180224*a^2*b^3*c^2 + 344064*a^3*b^2*c^2 + 196608*a*b*c^5 + 9
8304*a*b^5*c) - (-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b
^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^
2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c
^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b
^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(8192*b^3*c^5 - 557056*a^3*c^
5 - 425984*a^4*c^4 - 98304*a^5*c^3 - 229376*a^2*c^6 - 40960*b^4*c^4 + 57344
*b^5*c^3 - 24576*b^6*c^2 + 221184*a*b^2*c^5 - 327680*a*b^3*c^4 + 90112*a*b^
4*c^3 + 49152*a*b^5*c^2 + 393216*a^2*b*c^5 + 622592*a^3*b*c^4 + 196608*a^4*
b*c^3 + tan(x/2)*(-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c -
b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c
^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*
c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*

b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(65536*a*c^8 - 131072*a^2*c^7 - 262144*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c^4 - 16384*b^2*c^7 + 49152*b^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c^3 + 16384*b^7*c^2 + 294912*a*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^4*c^4 - 114688*a*b^5*c^3 - 16384*a*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*b*c^5 - 65536*a^4*b*c^4 - 655360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^4*c^2 - 114688*a^4*b^2*c^3 - 196608*a*b*c^7) + 172032*a^2*b^2*c^4 - 352256*a^2*b^3*c^3 + 106496*a^3*b^2*c^3 - 49152*a^3*b^3*c^2 + 24576*a^4*b^2*c^2 - 32768*a*b*c^6))*(-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2) - 32768*a*b^5 + 24576*a^5*c - 49152*b^5*c + 24576*b^6 - 16384*a^2*b^4 + 32768*a^3*b^3 - 8192*a^4*b^2 + 98304*a^2*c^4 + 253952*a^3*c^3 + 180224*a^4*c^2 - 8192*b^3*c^3 + 32768*b^4*c^2 - 155648*a*b^2*c^3 + 262144*a*b^3*c^2 - 270336*a^2*b*c^3 + 237568*a^2*b^3*c - 458752*a^3*b*c^2 + 24576*a^3*b^2*c + 16384*a^2*b^2*c^2 + 32768*a*b*c^4 - 114688*a*b^4*c - 122880*a^4*b*c) - tan(x/2)*(40960*a*b^4 - 57344*a^4*b - 73728*a^4*c + 8192*b^4*c + 24576*a^5 - 8192*b^5 - 81920*a^2*b^3 + 81920*a^3*b^2 + 16384*a^2*c^3 - 81920*a^3*c^2 - 32768*a*b^2*c^2 + 81920*a^2*b*c^2 - 81920*a^2*b^2*c + 163840*a^3*b*c))*(-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*1i - ((-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(24576*a^5*c - 32768*a*b^5 - (tan(x/2)*(16384*a*b^6 - 65536*a*c^6 + 49152*b^6*c - 16384*b^7 + 16384*a^2*b^5 - 16384*a^3*b^4 + 245760*a^2*c^5 + 671744*a^3*c^4 + 212992*a^4*c^3 - 147456*a^5*c^2 + 16384*b^2*c^5 - 49152*b^3*c^4 + 65536*b^4*c^3 - 65536*b^5*c^2 - 327680*a*b^2*c^4 + 475136*a*b^3*c^3 - 393216*a*b^4*c^2 - 802816*a^2*b*c^4 - 180224*a^2*b^4*c - 1081344*a^3*b*c^3 - 65536*a^3*b^3*c + 49152*a^4*b*c^2 + 98304*a^4*b^2*c + 557056*a^2*b^2*c^3 + 180224*a^2*b^3*c^2 + 344064*a^3*b^2*c^2 + 196608*a*b*c^5 + 98304*a*b^5*c) - (-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(229376*a^2*c^6 + 557056*a^3*c^5 + 425984*a^4*c^4 + 98304*a^5*c^3 - 8192*b^3*c^5 + 40960*b^4*c^4 - 57344*b^5*c^3 + 24576*b^6*c^2 - 221184*a*b^2*c^5 + 327680*a*b^3*c^4 - 90112*a*b^4*c^3 - 49152*a*b^5*c^2 - 393216*a^2*b*c^5 - 622592*a^3*b*c^4 - 196608*a^4*b*c

^3 + tan(x/2)*(-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b^2
)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^2
+ 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5
 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2
*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(65536*a*c^8 - 131072*a^2*c^7 -
 262144*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c^4 - 16384*b^2*c^7 + 49152*b
^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c^3 + 16384*b^7*c^2 + 29
4912*a*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^4*c^4 - 114688*a*b^5*c^3 - 1
6384*a*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*b*c^5 - 65536*a^4*b*c^4 - 65
5360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^4*c^3 - 16384*a^2*b^5*c
^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^4*c^2 - 114688*a^
4*b^2*c^3 - 196608*a*b*c^7) - 172032*a^2*b^2*c^4 + 352256*a^2*b^3*c^3 - 106
496*a^3*b^2*c^3 + 49152*a^3*b^3*c^2 - 24576*a^4*b^2*c^2 + 32768*a*b*c^6))*(
-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^
2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2
*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 +
 b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*
c^2 - 8*a^3*b^2*c^3)))^(1/2) - 49152*b^5*c + 24576*b^6 - 16384*a^2*b^4 + 32
768*a^3*b^3 - 8192*a^4*b^2 + 98304*a^2*c^4 + 253952*a^3*c^3 + 180224*a^4*c^
2 - 8192*b^3*c^3 + 32768*b^4*c^2 - 155648*a*b^2*c^3 + 262144*a*b^3*c^2 - 27
0336*a^2*b*c^3 + 237568*a^2*b^3*c - 458752*a^3*b*c^2 + 24576*a^3*b^2*c + 16
384*a^2*b^2*c^2 + 32768*a*b*c^4 - 114688*a*b^4*c - 122880*a^4*b*c) + tan(x/
2)*(40960*a*b^4 - 57344*a^4*b - 73728*a^4*c + 8192*b^4*c + 24576*a^5 - 8192
*b^5 - 81920*a^2*b^3 + 81920*a^3*b^2 + 16384*a^2*c^3 - 81920*a^3*c^2 - 3276
8*a*b^2*c^2 + 81920*a^2*b*c^2 - 81920*a^2*b^2*c + 163840*a^3*b*c))*(-(a^2*b
^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(
4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*
(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^
4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8
*a^3*b^2*c^3)))^(1/2)*1i)/(((-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*
(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 1
8*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^
6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^
3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*((tan(x/2)*(16384
*a*b^6 - 65536*a*c^6 + 49152*b^6*c - 16384*b^7 + 16384*a^2*b^5 - 16384*a^3*
b^4 + 245760*a^2*c^5 + 671744*a^3*c^4 + 212992*a^4*c^3 - 147456*a^5*c^2 + 1
6384*b^2*c^5 - 49152*b^3*c^4 + 65536*b^4*c^3 - 65536*b^5*c^2 - 327680*a*b^2
*c^4 + 475136*a*b^3*c^3 - 393216*a*b^4*c^2 - 802816*a^2*b*c^4 - 180224*a^2*
b^4*c - 1081344*a^3*b*c^3 - 65536*a^3*b^3*c + 49152*a^4*b*c^2 + 98304*a^4*b
^2*c + 557056*a^2*b^2*c^3 + 180224*a^2*b^3*c^2 + 344064*a^3*b^2*c^2 + 19660
8*a*b*c^5 + 98304*a*b^5*c) - (-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3
*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c -
18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c
^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c
^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(8192*b^3*c^5 -

557056*a^3*c^5 - 425984*a^4*c^4 - 98304*a^5*c^3 - 229376*a^2*c^6 - 40960*b^
4*c^4 + 57344*b^5*c^3 - 24576*b^6*c^2 + 221184*a*b^2*c^5 - 327680*a*b^3*c^4
 + 90112*a*b^4*c^3 + 49152*a*b^5*c^2 + 393216*a^2*b*c^5 + 622592*a^3*b*c^4
+ 196608*a^4*b*c^3 + tan(x/2)*(-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^
3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c -
 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*
c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*
c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(65536*a*c^8 -
131072*a^2*c^7 - 262144*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c^4 - 16384*b
^2*c^7 + 49152*b^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c^3 + 16
384*b^7*c^2 + 294912*a*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^4*c^4 - 1146
88*a*b^5*c^3 - 16384*a*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*b*c^5 - 6553
6*a^4*b*c^4 - 655360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^4*c^3 -
 16384*a^2*b^5*c^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^4
*c^2 - 114688*a^4*b^2*c^3 - 196608*a*b*c^7) + 172032*a^2*b^2*c^4 - 352256*a
^2*b^3*c^3 + 106496*a^3*b^2*c^3 - 49152*a^3*b^3*c^2 + 24576*a^4*b^2*c^2 - 3
2768*a*b*c^6))*(-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b^
2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^2
 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^
5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^
2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2) - 32768*a*b^5 + 24576*a^5*c -
49152*b^5*c + 24576*b^6 - 16384*a^2*b^4 + 32768*a^3*b^3 - 8192*a^4*b^2 + 98
304*a^2*c^4 + 253952*a^3*c^3 + 180224*a^4*c^2 - 8192*b^3*c^3 + 32768*b^4*c^
2 - 155648*a*b^2*c^3 + 262144*a*b^3*c^2 - 270336*a^2*b*c^3 + 237568*a^2*b^3
*c - 458752*a^3*b*c^2 + 24576*a^3*b^2*c + 16384*a^2*b^2*c^2 + 32768*a*b*c^4
 - 114688*a*b^4*c - 122880*a^4*b*c) - tan(x/2)*(40960*a*b^4 - 57344*a^4*b -
 73728*a^4*c + 8192*b^4*c + 24576*a^5 - 8192*b^5 - 81920*a^2*b^3 + 81920*a^
3*b^2 + 16384*a^2*c^3 - 81920*a^3*c^2 - 32768*a*b^2*c^2 + 81920*a^2*b*c^2 -
 81920*a^2*b^2*c + 163840*a^3*b*c))*(-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^
2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b
^2*c - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(1
6*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*
a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2) + ((-(a^2
*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(
-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*
c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*
c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 -
 8*a^3*b^2*c^3)))^(1/2)*(24576*a^5*c - 32768*a*b^5 - (tan(x/2)*(16384*a*b^6
 - 65536*a*c^6 + 49152*b^6*c - 16384*b^7 + 16384*a^2*b^5 - 16384*a^3*b^4 +
245760*a^2*c^5 + 671744*a^3*c^4 + 212992*a^4*c^3 - 147456*a^5*c^2 + 16384*b
^2*c^5 - 49152*b^3*c^4 + 65536*b^4*c^3 - 65536*b^5*c^2 - 327680*a*b^2*c^4 +
 475136*a*b^3*c^3 - 393216*a*b^4*c^2 - 802816*a^2*b*c^4 - 180224*a^2*b^4*c
- 1081344*a^3*b*c^3 - 65536*a^3*b^3*c + 49152*a^4*b*c^2 + 98304*a^4*b^2*c +
 557056*a^2*b^2*c^3 + 180224*a^2*b^3*c^2 + 344064*a^3*b^2*c^2 + 196608*a*b*
c^5 + 98304*a*b^5*c) - (-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*

a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2
*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 3
2*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 3
2*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(229376*a^2*c^6 + 5570
56*a^3*c^5 + 425984*a^4*c^4 + 98304*a^5*c^3 - 8192*b^3*c^5 + 40960*b^4*c^4
- 57344*b^5*c^3 + 24576*b^6*c^2 - 221184*a*b^2*c^5 + 327680*a*b^3*c^4 - 901
12*a*b^4*c^3 - 49152*a*b^5*c^2 - 393216*a^2*b*c^5 - 622592*a^3*b*c^4 - 1966
08*a^4*b*c^3 + tan(x/2)*(-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4
*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^
2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 +
32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 -
32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(65536*a*c^8 - 131072
*a^2*c^7 - 262144*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c^4 - 16384*b^2*c^7
+ 49152*b^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c^3 + 16384*b^
7*c^2 + 294912*a*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^4*c^4 - 114688*a*b
^5*c^3 - 16384*a*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*b*c^5 - 65536*a^4*
b*c^4 - 655360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^4*c^3 - 16384
*a^2*b^5*c^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^4*c^2 -
114688*a^4*b^2*c^3 - 196608*a*b*c^7) - 172032*a^2*b^2*c^4 + 352256*a^2*b^3
*c^3 - 106496*a^3*b^2*c^3 + 49152*a^3*b^3*c^2 - 24576*a^4*b^2*c^2 + 32768*a
*b*c^6))*(-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^
(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^2 + 8*a
*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16
*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4
+ a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2) - 49152*b^5*c + 24576*b^6 - 16384*a^
2*b^4 + 32768*a^3*b^3 - 8192*a^4*b^2 + 98304*a^2*c^4 + 253952*a^3*c^3 + 180
224*a^4*c^2 - 8192*b^3*c^3 + 32768*b^4*c^2 - 155648*a*b^2*c^3 + 262144*a*b^
3*c^2 - 270336*a^2*b*c^3 + 237568*a^2*b^3*c - 458752*a^3*b*c^2 + 24576*a^3*
b^2*c + 16384*a^2*b^2*c^2 + 32768*a*b*c^4 - 114688*a*b^4*c - 122880*a^4*b*c
) + tan(x/2)*(40960*a*b^4 - 57344*a^4*b - 73728*a^4*c + 8192*b^4*c + 24576*
a^5 - 8192*b^5 - 81920*a^2*b^3 + 81920*a^3*b^2 + 16384*a^2*c^3 - 81920*a^3*
c^2 - 32768*a*b^2*c^2 + 81920*a^2*b*c^2 - 81920*a^2*b^2*c + 163840*a^3*b*c)
)*(-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) +
a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c - 18*a^2*b^2*c^2 + 8*a*b^4*c
+ 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^
4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b
^4*c^2 - 8*a^3*b^2*c^3)))^(1/2) - 65536*a^3*b + 49152*a^3*c + 49152*a^4 + 1
6384*a^2*b^2 - 16384*a^2*b*c))*(-(a^2*b^4 - b^6 + 8*a^3*c^3 + 8*a^4*c^2 - b
^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) - 6*a^3*b^2*c
- 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2
*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4
*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*2i + atan(((ta
n(x/2)*(40960*a*b^4 - 57344*a^4*b - 73728*a^4*c + 8192*b^4*c + 24576*a^5 -
8192*b^5 - 81920*a^2*b^3 + 81920*a^3*b^2 + 16384*a^2*c^3 - 81920*a^3*c^2 -
32768*a*b^2*c^2 + 81920*a^2*b*c^2 - 81920*a^2*b^2*c + 163840*a^3*b*c) + ((b

^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b
*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*
b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^
4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2
 - 8*a^3*b^2*c^3)))^(1/2)*(24576*a^5*c - 32768*a*b^5 - 49152*b^5*c + 24576*
b^6 - 16384*a^2*b^4 + 32768*a^3*b^3 - 8192*a^4*b^2 + 98304*a^2*c^4 + 253952
*a^3*c^3 + 180224*a^4*c^2 - 8192*b^3*c^3 + 32768*b^4*c^2 + (((b^6 - a^2*b^4
 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c -
b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c
 - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*
c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2
c^3)))^(1/2)(229376*a^2*c^6 + 557056*a^3*c^5 + 425984*a^4*c^4 + 98304*a^5
*c^3 - 8192*b^3*c^5 + 40960*b^4*c^4 - 57344*b^5*c^3 + 24576*b^6*c^2 - 22118
4*a*b^2*c^5 + 327680*a*b^3*c^4 - 90112*a*b^4*c^3 - 49152*a*b^5*c^2 - 393216
*a^2*b*c^5 - 622592*a^3*b*c^4 - 196608*a^4*b*c^3 - 172032*a^2*b^2*c^4 + 352
256*a^2*b^3*c^3 - 106496*a^3*b^2*c^3 + 49152*a^3*b^3*c^2 - 24576*a^4*b^2*c^
2 + 32768*a*b*c^6 + tan(x/2)*((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*
(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 1
8*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^
6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^
3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(65536*a*c^8 - 13
1072*a^2*c^7 - 262144*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c^4 - 16384*b^2
*c^7 + 49152*b^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c^3 + 1638
4*b^7*c^2 + 294912*a*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^4*c^4 - 114688
*a*b^5*c^3 - 16384*a*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*b*c^5 - 65536*
a^4*b*c^4 - 655360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^4*c^3 - 1
6384*a^2*b^5*c^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^4*c
^2 - 114688*a^4*b^2*c^3 - 196608*a*b*c^7)) - tan(x/2)*(16384*a*b^6 - 65536*
a*c^6 + 49152*b^6*c - 16384*b^7 + 16384*a^2*b^5 - 16384*a^3*b^4 + 245760*a^
2*c^5 + 671744*a^3*c^4 + 212992*a^4*c^3 - 147456*a^5*c^2 + 16384*b^2*c^5 -
49152*b^3*c^4 + 65536*b^4*c^3 - 65536*b^5*c^2 - 327680*a*b^2*c^4 + 475136*a
*b^3*c^3 - 393216*a*b^4*c^2 - 802816*a^2*b*c^4 - 180224*a^2*b^4*c - 1081344
*a^3*b*c^3 - 65536*a^3*b^3*c + 49152*a^4*b*c^2 + 98304*a^4*b^2*c + 557056*a
^2*b^2*c^3 + 180224*a^2*b^3*c^2 + 344064*a^3*b^2*c^2 + 196608*a*b*c^5 + 983
04*a*b^5*c))*((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^
3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 -
8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 +
 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c
^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2) - 155648*a*b^2*c^3 + 262144*a*b^3
*c^2 - 270336*a^2*b*c^3 + 237568*a^2*b^3*c - 458752*a^3*b*c^2 + 24576*a^3*b
^2*c + 16384*a^2*b^2*c^2 + 32768*a*b*c^4 - 114688*a*b^4*c - 122880*a^4*b*c)
)*((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) +
a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c +
 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4
 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^

4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*1i + (tan(x/2)*(40960*a*b^4 - 57344*a^4*b -
73728*a^4*c + 8192*b^4*c + 24576*a^5 - 8192*b^5 - 81920*a^2*b^3 + 81920*a^3
*b^2 + 16384*a^2*c^3 - 81920*a^3*c^2 - 32768*a*b^2*c^2 + 81920*a^2*b*c^2 -
81920*a^2*b^2*c + 163840*a^3*b*c) - ((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2
 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^
2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16
*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a
*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(24576*a^5
*c - 32768*a*b^5 - 49152*b^5*c + 24576*b^6 - 16384*a^2*b^4 + 32768*a^3*b^3
- 8192*a^4*b^2 + 98304*a^2*c^4 + 253952*a^3*c^3 + 180224*a^4*c^2 - 8192*b^3
*c^3 + 32768*b^4*c^2 - (((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*
a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2
*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 3
2*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 3
2*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(8192*b^3*c^5 - 557056
*a^3*c^5 - 425984*a^4*c^4 - 98304*a^5*c^3 - 229376*a^2*c^6 - 40960*b^4*c^4
+ 57344*b^5*c^3 - 24576*b^6*c^2 + 221184*a*b^2*c^5 - 327680*a*b^3*c^4 + 901
12*a*b^4*c^3 + 49152*a*b^5*c^2 + 393216*a^2*b*c^5 + 622592*a^3*b*c^4 + 1966
08*a^4*b*c^3 + 172032*a^2*b^2*c^4 - 352256*a^2*b^3*c^3 + 106496*a^3*b^2*c^3
 - 49152*a^3*b^3*c^2 + 24576*a^4*b^2*c^2 - 32768*a*b*c^6 + tan(x/2)*((b^6 -
 a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(
4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*
(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^
4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8
*a^3*b^2*c^3)))^(1/2)*(65536*a*c^8 - 131072*a^2*c^7 - 262144*a^3*c^6 + 1310
72*a^4*c^5 + 196608*a^5*c^4 - 16384*b^2*c^7 + 49152*b^3*c^6 - 65536*b^4*c^5
 + 65536*b^5*c^4 - 49152*b^6*c^3 + 16384*b^7*c^2 + 294912*a*b^2*c^6 - 40960
0*a*b^3*c^5 + 376832*a*b^4*c^4 - 114688*a*b^5*c^3 - 16384*a*b^6*c^2 + 58982
4*a^2*b*c^6 + 720896*a^3*b*c^5 - 65536*a^4*b*c^4 - 655360*a^2*b^2*c^5 + 163
84*a^2*b^3*c^4 + 196608*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 557056*a^3*b^2*c^
4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^4*c^2 - 114688*a^4*b^2*c^3 - 196608*a*b
c^7)) - tan(x/2)(16384*a*b^6 - 65536*a*c^6 + 49152*b^6*c - 16384*b^7 + 16
384*a^2*b^5 - 16384*a^3*b^4 + 245760*a^2*c^5 + 671744*a^3*c^4 + 212992*a^4*
c^3 - 147456*a^5*c^2 + 16384*b^2*c^5 - 49152*b^3*c^4 + 65536*b^4*c^3 - 6553
6*b^5*c^2 - 327680*a*b^2*c^4 + 475136*a*b^3*c^3 - 393216*a*b^4*c^2 - 802816
*a^2*b*c^4 - 180224*a^2*b^4*c - 1081344*a^3*b*c^3 - 65536*a^3*b^3*c + 49152
*a^4*b*c^2 + 98304*a^4*b^2*c + 557056*a^2*b^2*c^3 + 180224*a^2*b^3*c^2 + 34
4064*a^3*b^2*c^2 + 196608*a*b*c^5 + 98304*a*b^5*c))*((b^6 - a^2*b^4 - 8*a^3
*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^
(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^
3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*
a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^
(1/2) - 155648*a*b^2*c^3 + 262144*a*b^3*c^2 - 270336*a^2*b*c^3 + 237568*a^2
*b^3*c - 458752*a^3*b*c^2 + 24576*a^3*b^2*c + 16384*a^2*b^2*c^2 + 32768*a*b
*c^4 - 114688*a*b^4*c - 122880*a^4*b*c))*((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^

4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a
^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(
2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 +
 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*1i)/(
(tan(x/2)*(40960*a*b^4 - 57344*a^4*b - 73728*a^4*c + 8192*b^4*c + 24576*a^5
 - 8192*b^5 - 81920*a^2*b^3 + 81920*a^3*b^2 + 16384*a^2*c^3 - 81920*a^3*c^2
 - 32768*a*b^2*c^2 + 81920*a^2*b*c^2 - 81920*a^2*b^2*c + 163840*a^3*b*c) +
((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^
2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2
*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 +
 b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*
c^2 - 8*a^3*b^2*c^3)))^(1/2)*(24576*a^5*c - 32768*a*b^5 - 49152*b^5*c + 245
76*b^6 - 16384*a^2*b^4 + 32768*a^3*b^3 - 8192*a^4*b^2 + 98304*a^2*c^4 + 253
952*a^3*c^3 + 180224*a^4*c^2 - 8192*b^3*c^3 + 32768*b^4*c^2 + (((b^6 - a^2*
b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c
 - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*
a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b
^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*
b^2*c^3)))^(1/2)*(229376*a^2*c^6 + 557056*a^3*c^5 + 425984*a^4*c^4 + 98304*
a^5*c^3 - 8192*b^3*c^5 + 40960*b^4*c^4 - 57344*b^5*c^3 + 24576*b^6*c^2 - 22
1184*a*b^2*c^5 + 327680*a*b^3*c^4 - 90112*a*b^4*c^3 - 49152*a*b^5*c^2 - 393
216*a^2*b*c^5 - 622592*a^3*b*c^4 - 196608*a^4*b*c^3 - 172032*a^2*b^2*c^4 +
352256*a^2*b^3*c^3 - 106496*a^3*b^2*c^3 + 49152*a^3*b^3*c^2 - 24576*a^4*b^2
*c^2 + 32768*a*b*c^6 + tan(x/2)*((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b
^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c
 + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2
*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4
*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(65536*a*c^8 -
 131072*a^2*c^7 - 262144*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c^4 - 16384*
b^2*c^7 + 49152*b^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c^3 + 1
6384*b^7*c^2 + 294912*a*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^4*c^4 - 114
688*a*b^5*c^3 - 16384*a*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*b*c^5 - 655
36*a^4*b*c^4 - 655360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^4*c^3
 - 16384*a^2*b^5*c^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^
4*c^2 - 114688*a^4*b^2*c^3 - 196608*a*b*c^7)) - tan(x/2)*(16384*a*b^6 - 655
36*a*c^6 + 49152*b^6*c - 16384*b^7 + 16384*a^2*b^5 - 16384*a^3*b^4 + 245760
*a^2*c^5 + 671744*a^3*c^4 + 212992*a^4*c^3 - 147456*a^5*c^2 + 16384*b^2*c^5
 - 49152*b^3*c^4 + 65536*b^4*c^3 - 65536*b^5*c^2 - 327680*a*b^2*c^4 + 47513
6*a*b^3*c^3 - 393216*a*b^4*c^2 - 802816*a^2*b*c^4 - 180224*a^2*b^4*c - 1081
344*a^3*b*c^3 - 65536*a^3*b^3*c + 49152*a^4*b*c^2 + 98304*a^4*b^2*c + 55705
6*a^2*b^2*c^3 + 180224*a^2*b^3*c^2 + 344064*a^3*b^2*c^2 + 196608*a*b*c^5 +
98304*a*b^5*c))*((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^
2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2
 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^
5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^

2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2) - 155648*a*b^2*c^3 + 262144*a*b^3*c^2 - 270336*a^2*b*c^3 + 237568*a^2*b^3*c - 458752*a^3*b*c^2 + 24576*a^3*b^2*c + 16384*a^2*b^2*c^2 + 32768*a*b*c^4 - 114688*a*b^4*c - 122880*a^4*b*c))*((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2) - (tan(x/2)*(40960*a*b^4 - 57344*a^4*b - 73728*a^4*c + 8192*b^4*c + 24576*a^5 - 8192*b^5 - 81920*a^2*b^3 + 81920*a^3*b^2 + 16384*a^2*c^3 - 81920*a^3*c^2 - 32768*a*b^2*c^2 + 81920*a^2*b*c^2 - 81920*a^2*b^2*c + 163840*a^3*b*c) - ((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(24576*a^5*c - 32768*a*b^5 - 49152*b^5*c + 24576*b^6 - 16384*a^2*b^4 + 32768*a^3*b^3 - 8192*a^4*b^2 + 98304*a^2*c^4 + 253952*a^3*c^3 + 180224*a^4*c^2 - 8192*b^3*c^3 + 32768*b^4*c^2 - (((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(8192*b^3*c^5 - 557056*a^3*c^5 - 425984*a^4*c^4 - 98304*a^5*c^3 - 229376*a^2*c^6 - 40960*b^4*c^4 + 57344*b^5*c^3 - 24576*b^6*c^2 + 221184*a*b^2*c^5 - 327680*a*b^3*c^4 + 90112*a*b^4*c^3 + 49152*a*b^5*c^2 + 393216*a^2*b*c^5 + 622592*a^3*b*c^4 + 196608*a^4*b*c^3 + 172032*a^2*b^2*c^4 - 352256*a^2*b^3*c^3 + 106496*a^3*b^2*c^3 - 49152*a^3*b^3*c^2 + 24576*a^4*b^2*c^2 - 32768*a*b*c^6 + tan(x/2)*((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2)*(65536*a*c^8 - 131072*a^2*c^7 - 262144*a^3*c^6 + 131072*a^4*c^5 + 196608*a^5*c^4 - 16384*b^2*c^7 + 49152*b^3*c^6 - 65536*b^4*c^5 + 65536*b^5*c^4 - 49152*b^6*c^3 + 16384*b^7*c^2 + 294912*a*b^2*c^6 - 409600*a*b^3*c^5 + 376832*a*b^4*c^4 - 114688*a*b^5*c^3 - 16384*a*b^6*c^2 + 589824*a^2*b*c^6 + 720896*a^3*b*c^5 - 65536*a^4*b*c^4 - 655360*a^2*b^2*c^5 + 16384*a^2*b^3*c^4 + 196608*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 557056*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 16384*a^3*b^4*c^2 - 114688*a^4*b^2*c^3 - 196608*a*b*c^7)) - tan(x/2)*(16384*a*b^6 - 65536*a*c^6 + 49152*b^6*c - 16384*b^7 + 16384*a^2*b^5 - 16384*a^3*b^4 + 245760*a^2*c^5 + 671744*a^3*c^4 + 212992*a^4*c^3 - 147456*a^5*c^2 + 16384*b^2*c^5 - 49152*b^3*c^4 + 65536*b^4*c^3 - 65536*b^5*c^2 - 327680*a*b^2*c^4 + 475136*a*b^3*c^3 - 393216*a*b^4*c^2 - 802816*a^2*b*c^4 - 180224*a^2*b^4*c - 1081344*a^3*b*c^3 - 65536*a^3*b^3*c + 49152*a^4*b*c^2 + 98304*a^4*b^2*c + 557056*a^2*b^2*c^3 + 180224*a^2*b^3*c^2 + 344064*a^3*b^2*c^2 + 196608*a*b*c^5 + 98304*a*b^5*c))*((b^6 - a^2*b^4 - 8*a^3

```
*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^
(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^
3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*
a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^
(1/2) - 155648*a*b^2*c^3 + 262144*a*b^3*c^2 - 270336*a^2*b*c^3 + 237568*a^2
*b^3*c - 458752*a^3*b*c^2 + 24576*a^3*b^2*c + 16384*a^2*b^2*c^2 + 32768*a*b
*c^4 - 114688*a*b^4*c - 122880*a^4*b*c))*((b^6 - a^2*b^4 - 8*a^3*c^3 - 8*a^
4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-(4*a*c - b^2)^3)^(1/2) + 6*a
^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(
2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c^4 - b^6*c^2 - 8*a*b^2*c^5 +
 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 - 8*a^3*b^2*c^3)))^(1/2) - 655
36*a^3*b + 49152*a^3*c + 49152*a^4 + 16384*a^2*b^2 - 16384*a^2*b*c))*((b^6
- a^2*b^4 - 8*a^3*c^3 - 8*a^4*c^2 - b^3*(-(4*a*c - b^2)^3)^(1/2) + a^2*b*(-
(4*a*c - b^2)^3)^(1/2) + 6*a^3*b^2*c + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c
*(-(4*a*c - b^2)^3)^(1/2))/(2*(16*a^2*c^6 + 32*a^3*c^5 + 16*a^4*c^4 + b^4*c
^4 - b^6*c^2 - 8*a*b^2*c^5 + 10*a*b^4*c^3 - 32*a^2*b^2*c^4 + a^2*b^4*c^2 -
8*a^3*b^2*c^3)))^(1/2)*2i
```

3.16 $\int \frac{\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 17, antiderivative size = 230

$$\int \frac{\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{2\left(1-\frac{b}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} + \frac{2\left(1+\frac{b}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}$$

```
[Out] 2*arctan((b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c-(-4*a*c+b^2)^(1
/2))^(1/2))*(1-b/(-4*a*c+b^2)^(1/2))/(b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)/(b+2*
c-(-4*a*c+b^2)^(1/2))^(1/2)+2*arctan((b-2*c+(-4*a*c+b^2)^(1/2))^(1/2)*tan(1
/2*x)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2))*(1+b/(-4*a*c+b^2)^(1/2))/(b-2*c+(-4
*a*c+b^2)^(1/2))^(1/2)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2)
```

Rubi [A] (verified)

Time = 0.59 (sec) , antiderivative size = 230, normalized size of antiderivative = 1.00, number of steps used = 6, number of rules used = 3, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.176, Rules used = {3338, 2738, 211}

$$\int \frac{\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{2\left(1-\frac{b}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{-\sqrt{b^2-4ac}+b-2c}}{\sqrt{-\sqrt{b^2-4ac}+b+2c}}\right)}{\sqrt{-\sqrt{b^2-4ac}+b-2c}\sqrt{-\sqrt{b^2-4ac}+b+2c}} + \frac{2\left(\frac{b}{\sqrt{b^2-4ac}}+1\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{\sqrt{b^2-4ac}+b-2c}}{\sqrt{\sqrt{b^2-4ac}+b+2c}}\right)}{\sqrt{\sqrt{b^2-4ac}+b-2c}\sqrt{\sqrt{b^2-4ac}+b+2c}}$$

```
[In] Int[Cos[x]/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (2*(1 - b/Sqrt[b^2 - 4*a*c])*ArcTan[(Sqrt[b - 2*c - Sqrt[b^2 - 4*a*c]]*Tan[
x/2])/Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]])/(Sqrt[b - 2*c - Sqrt[b^2 - 4*a*c]
]*Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]) + (2*(1 + b/Sqrt[b^2 - 4*a*c])*ArcTan[
(Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c + Sqrt[b^2 - 4*a*
c]]])/(Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]*Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]])
```

Rule 211

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(Rt[a/b, 2]/a)*ArcTan[x/R
t[a/b, 2]], x] /; FreeQ[{a, b}, x] && PosQ[a/b]
```

Rule 2738

```
Int[((a_) + (b_.)*sin[Pi/2 + (c_.) + (d_.)*(x_)])^(-1), x_Symbol] :> With[{
e = FreeFactors[Tan[(c + d*x)/2], x]}, Dist[2*(e/d), Subst[Int[1/(a + b + (
a - b)*e^2*x^2), x], x, Tan[(c + d*x)/2]/e], x]] /; FreeQ[{a, b, c, d}, x]
&& NeQ[a^2 - b^2, 0]
```

Rule 3338

```
Int[cos[(d_.) + (e_.)*(x_)]^(m_.)*((a_.) + cos[(d_.) + (e_.)*(x_)]^(n_.)*(b
_.) + cos[(d_.) + (e_.)*(x_)]^(n2_.)*(c_.))^(p_), x_Symbol] :> Int[ExpandTr
ig[cos[d + e*x]^m*(a + b*cos[d + e*x]^n + c*cos[d + e*x]^(2*n))^p, x], x] /
; FreeQ[{a, b, c, d, e}, x] && EqQ[n2, 2*n] && NeQ[b^2 - 4*a*c, 0] && Integ
ersQ[m, n, p]
```

Rubi steps

$$\text{integral} = \int \left(\frac{1 - \frac{b}{\sqrt{b^2-4ac}}}{b - \sqrt{b^2-4ac} + 2c\cos(x)} + \frac{1 + \frac{b}{\sqrt{b^2-4ac}}}{b + \sqrt{b^2-4ac} + 2c\cos(x)} \right) dx$$

$$= \left(1 - \frac{b}{\sqrt{b^2-4ac}}\right) \int \frac{1}{b - \sqrt{b^2-4ac} + 2c\cos(x)}\, dx + \left(1 + \frac{b}{\sqrt{b^2-4ac}}\right) \int \frac{1}{b + \sqrt{b^2-4ac} + 2c\cos(x)}\, dx$$

$$= \left(2\left(1 - \frac{b}{\sqrt{b^2-4ac}}\right)\right) \text{Subst}\left(\int \frac{1}{b + 2c - \sqrt{b^2-4ac} + \left(b - 2c - \sqrt{b^2-4ac}\right)x^2}\, dx, x, \tan\left(\frac{x}{2}\right)\right) + \left(2\left(1 + \frac{b}{\sqrt{b^2-4ac}}\right)\right) \text{Subst}\left(\int \frac{1}{b + 2c + \sqrt{b^2-4ac} + \left(b - 2c + \sqrt{b^2-4ac}\right)x^2}\, dx, x, \tan\left(\frac{x}{2}\right)\right)$$

$$= \frac{2\left(1-\frac{b}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} + \frac{2\left(1+\frac{b}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}$$

Mathematica [A] (verified)

Time = 0.84 (sec) , antiderivative size = 227, normalized size of antiderivative = 0.99

$$\int \frac{\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{\sqrt{2}\left(-\frac{\left(b+\sqrt{b^2-4ac}\right)\operatorname{arctanh}\left(\frac{\left(b-2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)-2b\sqrt{b^2-4ac}}}\right)}{\sqrt{-b^2+2c(a+c)-b\sqrt{b^2-4ac}}}+\frac{\left(-b+\sqrt{b^2-4ac}\right)\operatorname{arctanh}\left(\frac{\left(-b+2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)+2b\sqrt{b^2-4ac}}}\right)}{\sqrt{-b^2+2c(a+c)+b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}}$$

```
[In] Integrate[Cos[x]/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (Sqrt[2]*(-(((b + Sqrt[b^2 - 4*a*c])*ArcTanh[((b - 2*c + Sqrt[b^2 - 4*a*c])
*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) - 2*b*Sqrt[b^2 - 4*a*c]]])/Sqrt[-b^2 +
 2*c*(a + c) - b*Sqrt[b^2 - 4*a*c]]) + ((-b + Sqrt[b^2 - 4*a*c])*ArcTanh[((
-b + 2*c + Sqrt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) + 2*b*Sqr
t[b^2 - 4*a*c]]])/Sqrt[-b^2 + 2*c*(a + c) + b*Sqrt[b^2 - 4*a*c]]))/Sqrt[b^2
 - 4*a*c]
```

Maple [A] (verified)

Time = 1.17 (sec) , antiderivative size = 208, normalized size of antiderivative = 0.90

method	result
default	$2(a-b+c)\left(\frac{\left(2a-b-\sqrt{-4ac+b^2}\right)\operatorname{arctanh}\left(\frac{(-a+b-c)\tan\left(\frac{x}{2}\right)}{\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}\right)}{2\sqrt{-4ac+b^2}\,(a-b+c)\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}+\frac{\left(-2a+b-\sqrt{-4ac+b^2}\right)\arctan\left(\frac{(a-b+c)\,\mathrm{ta}}{\sqrt{\left(\sqrt{-4ac+b^2}+c\right.}}\right.}{2\sqrt{-4ac+b^2}\,(a-b+c)\sqrt{\left(\sqrt{-4ac+b^2}+a-c\right.}}\right.$
risch	$\sum_{\_R=\operatorname{RootOf}\left(\left(16a^4c^2-8a^3b^2c+32c^3a^3+a^2b^4-32a^2b^2c^2+16a^2c^4+10a\,b^4c-8a\,b^2c^3-b^6+b^4c^2\right)\_Z^4+\left(-8a^3c+2a^2b^2-8a^2c^2+6a\,b^2c-b\right.\right.}$

```
[In] int(cos(x)/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] 2*(a-b+c)*(1/2*(2*a-b-(-4*a*c+b^2)^(1/2))/(-4*a*c+b^2)^(1/2)/(a-b+c)/(((-4*
a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2)*arctanh((-a+b-c)*tan(1/2*x)/(((-4*a*c+b^
```

```
2)^(1/2)-a+c)*(a-b+c))^(1/2))+1/2*(-2*a+b-(-4*a*c+b^2)^(1/2))/(-4*a*c+b^2)^
(1/2)/(a-b+c)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)*arctan((a-b+c)*tan(1
/2*x)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)))
```

Fricas [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 3513 vs. $2(189) = 378$.

Time = 0.57 (sec) , antiderivative size = 3513, normalized size of antiderivative = 15.27

$$\int \frac{\cos(x)}{a + b\cos(x) + c\cos^2(x)} \, dx = \text{Too large to display}$$

```
[In] integrate(cos(x)/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
```

```
[Out] -1/4*sqrt(2)*sqrt(-(2*a^2 - b^2 + 2*a*c - (a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2
 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 -
4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b
^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c)))/(a^2*b^2 - b^4 - 4*a*c^3
 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c))*log(a*b^2*cos(x) + 2*a*b*c +
 (a^3*b^2 - a*b^4 - 4*a^2*c^3 - (8*a^3 - a*b^2)*c^2 - 2*(2*a^4 - 3*a^2*b^2)
*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12
*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^
2 + 2*a*b^4)*c))*cos(x) + 1/2*sqrt(2)*((a^3*b^3 - a*b^5 + 4*a*b*c^4 + (4*a^
2*b - b^3)*c^3 - (4*a^3*b + 5*a*b^3)*c^2 - (4*a^4*b - 5*a^2*b^3 - b^5)*c)*s
qrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a
^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2
*a*b^4)*c))*sin(x) + (a*b^3 - 4*a*b*c^2 - (4*a^2*b - b^3)*c)*sin(x))*sqrt(-
(2*a^2 - b^2 + 2*a*c - (a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*
a^3 - 3*a*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 -
 b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(
a^5 - 3*a^3*b^2 + 2*a*b^4)*c)))/(a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^
2 - 2*(2*a^3 - 3*a*b^2)*c))) + 1/4*sqrt(2)*sqrt(-(2*a^2 - b^2 + 2*a*c - (a^
2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)*sqrt(b^2
/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*
b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)
*c)))/(a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)
)*log(a*b^2*cos(x) + 2*a*b*c + (a^3*b^2 - a*b^4 - 4*a^2*c^3 - (8*a^3 - a*b^
2)*c^2 - 2*(2*a^4 - 3*a^2*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a
*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2
+ b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c))*cos(x) - 1/2*sqrt(2)*((a^3*b
^3 - a*b^5 + 4*a*b*c^4 + (4*a^2*b - b^3)*c^3 - (4*a^3*b + 5*a*b^3)*c^2 - (4
*a^4*b - 5*a^2*b^3 - b^5)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5
- (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4
)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c))*sin(x) + (a*b^3 - 4*a*b*c^2 - (4*
a^2*b - b^3)*c)*sin(x))*sqrt(-(2*a^2 - b^2 + 2*a*c - (a^2*b^2 - b^4 - 4*a*c
```

```
^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b
^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4
 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c)))/(a^2*b^2 - b^
4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c))) - 1/4*sqrt(2)*sq
rt(-(2*a^2 - b^2 + 2*a*c + (a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2
*(2*a^3 - 3*a*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a
^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 -
 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c)))/(a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2
)*c^2 - 2*(2*a^3 - 3*a*b^2)*c))*log(-a*b^2*cos(x) - 2*a*b*c + (a^3*b^2 - a*
b^4 - 4*a^2*c^3 - (8*a^3 - a*b^2)*c^2 - 2*(2*a^4 - 3*a^2*b^2)*c)*sqrt(b^2/(
a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^
2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c
))*cos(x) + 1/2*sqrt(2)*((a^3*b^3 - a*b^5 + 4*a*b*c^4 + (4*a^2*b - b^3)*c^3
 - (4*a^3*b + 5*a*b^3)*c^2 - (4*a^4*b - 5*a^2*b^3 - b^5)*c)*sqrt(b^2/(a^4*b
^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^
3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c))*si
n(x) - (a*b^3 - 4*a*b*c^2 - (4*a^2*b - b^3)*c)*sin(x))*sqrt(-(2*a^2 - b^2 +
 2*a*c + (a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)
*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12
*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^
2 + 2*a*b^4)*c)))/(a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 -
 3*a*b^2)*c))) + 1/4*sqrt(2)*sqrt(-(2*a^2 - b^2 + 2*a*c + (a^2*b^2 - b^4 -
4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*
a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(
8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c)))/(a^2*b^2
 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c))*log(-a*b^2*c
os(x) - 2*a*b*c + (a^3*b^2 - a*b^4 - 4*a^2*c^3 - (8*a^3 - a*b^2)*c^2 - 2*(2
*a^4 - 3*a^2*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^
2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 -
4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c))*cos(x) - 1/2*sqrt(2)*((a^3*b^3 - a*b^5 +
4*a*b*c^4 + (4*a^2*b - b^3)*c^3 - (4*a^3*b + 5*a*b^3)*c^2 - (4*a^4*b - 5*a^
2*b^3 - b^5)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b
^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^
5 - 3*a^3*b^2 + 2*a*b^4)*c))*sin(x) - (a*b^3 - 4*a*b*c^2 - (4*a^2*b - b^3)*
c)*sin(x))*sqrt(-(2*a^2 - b^2 + 2*a*c + (a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 -
 b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*
a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2
 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c)))/(a^2*b^2 - b^4 - 4*a*c^3 -
 (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)))
```

Sympy [F(-1)]

Timed out.

$$\int \frac{\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Timed out}$$

```
[In] integrate(cos(x)/(a+b*cos(x)+c*cos(x)**2),x)
[Out] Timed out
```

Maxima [F]

$$\int \frac{\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \int \frac{\cos(x)}{c\cos(x)^2+b\cos(x)+a}\,dx$$

```
[In] integrate(cos(x)/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
[Out] integrate(cos(x)/(c*cos(x)^2 + b*cos(x) + a), x)
```

Giac [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 2973 vs. $2(189) = 378$.

Time = 2.34 (sec) , antiderivative size = 2973, normalized size of antiderivative = 12.93

$$\int \frac{\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(cos(x)/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
[Out] (4*a^3*b^2 - 6*a^2*b^3 + 4*a*b^4 - 2*b^5 - 16*a^4*c + 24*a^3*b*c - 24*a^2*b
^2*c + 20*a*b^3*c + 32*a^3*c^2 - 48*a^2*b*c^2 - 12*a*b^2*c^2 + 2*b^3*c^2 +
48*a^2*c^3 - 8*a*b*c^3 + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(
a - b + c))*a^2*b^2 - 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
 b + c))*a*b^3 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*b^4 - 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^
3*c + 8*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b*c
 + 34*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^2*c +
 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^3*c - 56*s
qrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*c^2 - 24*sqr
t(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b*c^2 - 5*sqrt(a
^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^2*c^2 + 20*sqrt(a^2
 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*c^3 + 6*sqrt(a^2 - a*
b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3 - 4*(b
^2 - 4*a*c)*a^3 - sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
```

```
))*sqrt(b^2 - 4*a*c)*a^2*b + 6*(b^2 - 4*a*c)*a^2*b - 12*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2 - 4*(b^2 -
 4*a*c)*a*b^2 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
))*sqrt(b^2 - 4*a*c)*b^3 + 2*(b^2 - 4*a*c)*b^3 + 28*sqrt(a^2 - a*b + b*c -
c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c + 8*(b^2 - 4*a
*c)*a^2*c + 26*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*
sqrt(b^2 - 4*a*c)*a*b*c - 12*(b^2 - 4*a*c)*a*b*c + 6*sqrt(a^2 - a*b + b*c -
 c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c - 10*sqrt(a^2
 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*c^2
 + 12*(b^2 - 4*a*c)*a*c^2 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*b*c^2 - 2*(b^2 - 4*a*c)*b*c^2)*(pi*floor(1
/2*x/pi + 1/2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a - 2*c + sqrt(-4*(a
 + b + c)*(a - b + c) + 4*(a - c)^2))/(a - b + c))))*abs(a - b + c)/(3*a^5*
b^2 - 5*a^4*b^3 - 6*a^3*b^4 + 10*a^2*b^5 + 3*a*b^6 - 5*b^7 - 12*a^6*c + 20*
a^5*b*c + 47*a^4*b^2*c - 60*a^3*b^3*c - 46*a^2*b^4*c + 40*a*b^5*c + 11*b^6*
c - 92*a^5*c^2 + 80*a^4*b*c^2 + 182*a^3*b^2*c^2 - 94*a^2*b^3*c^2 - 78*a*b^4
*c^2 - 6*b^5*c^2 - 184*a^4*c^3 + 56*a^3*b*c^3 + 166*a^2*b^2*c^3 + 36*a*b^3*
c^3 - 6*b^4*c^3 - 120*a^3*c^4 - 48*a^2*b*c^4 + 23*a*b^2*c^4 + 11*b^3*c^4 +
4*a^2*c^5 - 44*a*b*c^5 - 5*b^2*c^5 + 20*a*c^6) + (4*a^3*b^2 - 6*a^2*b^3 - 4
*a*b^4 + 6*b^5 - 16*a^4*c + 24*a^3*b*c + 40*a^2*b^2*c - 44*a*b^3*c - 8*b^4*
c - 96*a^3*c^2 + 80*a^2*b*c^2 + 52*a*b^2*c^2 + 2*b^3*c^2 - 80*a^2*c^3 - 8*a
*b*c^3 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*
b^2 - 2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^3 -
 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*b^4 - 12*sqr
t(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*c + 8*sqrt(a^2
 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b*c + 34*sqrt(a^2 -
 a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^2*c + 6*sqrt(a^2 - a*
b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*b^3*c - 56*sqrt(a^2 - a*b +
b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*c^2 - 24*sqrt(a^2 - a*b + b*
c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b*c^2 - 5*sqrt(a^2 - a*b + b*c -
 c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*b^2*c^2 + 20*sqrt(a^2 - a*b + b*c - c
^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*c^3 + 6*sqrt(a^2 - a*b + b*c - c^2 -
sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3 - 4*(b^2 - 4*a*c)*a^3
- sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*a^2*b + 6*(b^2 - 4*a*c)*a^2*b - 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b
^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2 + 4*(b^2 - 4*a*c)*a*b^2 -
5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*b^3 - 6*(b^2 - 4*a*c)*b^3 + 28*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c - 24*(b^2 - 4*a*c)*a^2*c + 26*
sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*
c)*a*b*c + 20*(b^2 - 4*a*c)*a*b*c + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c + 8*(b^2 - 4*a*c)*b^2*c - 10
*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a
*c)*a*c^2 - 20*(b^2 - 4*a*c)*a*c^2 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^
2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b*c^2 - 2*(b^2 - 4*a*c)*b*c^2)*(p
```

```
i*floor(1/2*x/pi + 1/2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a - 2*c - s
qrt(-4*(a + b + c)*(a - b + c) + 4*(a - c)^2))/(a - b + c))))*abs(a - b + c
)/(3*a^5*b^2 - 5*a^4*b^3 - 6*a^3*b^4 + 10*a^2*b^5 + 3*a*b^6 - 5*b^7 - 12*a^
6*c + 20*a^5*b*c + 47*a^4*b^2*c - 60*a^3*b^3*c - 46*a^2*b^4*c + 40*a*b^5*c
+ 11*b^6*c - 92*a^5*c^2 + 80*a^4*b*c^2 + 182*a^3*b^2*c^2 - 94*a^2*b^3*c^2 -
 78*a*b^4*c^2 - 6*b^5*c^2 - 184*a^4*c^3 + 56*a^3*b*c^3 + 166*a^2*b^2*c^3 +
36*a*b^3*c^3 - 6*b^4*c^3 - 120*a^3*c^4 - 48*a^2*b*c^4 + 23*a*b^2*c^4 + 11*b
^3*c^4 + 4*a^2*c^5 - 44*a*b*c^5 - 5*b^2*c^5 + 20*a*c^6)
```

Mupad [B] (verification not implemented)

Time = 13.30 (sec) , antiderivative size = 5488, normalized size of antiderivative = 23.86

$$\int \frac{\cos(x)}{a + b\cos(x) + c\cos^2(x)}\, dx = \text{Too large to display}$$

```
[In] int(cos(x)/(a + b*cos(x) + c*cos(x)^2),x)
```

```
[Out] atan(((tan(x/2)*(96*a*b^2 - 128*a^2*b - 64*a*c^2 + 32*b^2*c + 64*a^3 - 32*b
^3) + ((8*a^3*c + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^2*c^2
- 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4
*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64
*a*b^3 + 128*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32*a^2*b^2 + 256*a^2*c
^2 - 32*b^2*c^2 + tan(x/2)*((8*a^3*c + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2
*a^2*b^2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c
^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10
*a*b^4*c)))^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c - 64*b^4*c - 128*a^2*b^
3 + 64*a^3*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^3 + 128*b^3*c^2 + 192
*a*b^2*c^2 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c) - 256*a*b*c^2 + 64*
a*b^2*c - 256*a^2*b*c))*((8*a^3*c + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^
2*b^2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3
+ 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*
b^4*c)))^(1/2)*1i + (tan(x/2)*(96*a*b^2 - 128*a^2*b - 64*a*c^2 + 32*b^2*c +
 64*a^3 - 32*b^3) - ((8*a^3*c + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^
2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16
*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*
c)))^(1/2)*(64*a*b^3 + 128*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32*a^2*b
^2 + 256*a^2*c^2 - 32*b^2*c^2 - tan(x/2)*((8*a^3*c + b*(-(4*a*c - b^2)^3)^(
1/2) + b^4 - 2*a^2*b^2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*
c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^
2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c - 64*b^4*
c - 128*a^2*b^3 + 64*a^3*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^3 + 128
*b^3*c^2 + 192*a*b^2*c^2 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c) - 256
*a*b*c^2 + 64*a*b^2*c - 256*a^2*b*c))*((8*a^3*c + b*(-(4*a*c - b^2)^3)^(1/2
) + b^4 - 2*a^2*b^2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4
```

+ 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b
^2*c^2 + 10*a*b^4*c)))^(1/2)*1i)/(64*a*c - 64*a*b + 64*a^2 - (tan(x/2)*(96*
a*b^2 - 128*a^2*b - 64*a*c^2 + 32*b^2*c + 64*a^3 - 32*b^3) + ((8*a^3*c + b*
(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2
*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 -
8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^3 + 128*a*c^3 +
128*a^3*c + 64*b^3*c - 32*b^4 - 32*a^2*b^2 + 256*a^2*c^2 - 32*b^2*c^2 + ta
n(x/2)*((8*a^3*c + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^2*c^2
- 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^
4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(6
4*a*b^4 + 256*a*c^4 - 256*a^4*c - 64*b^4*c - 128*a^2*b^3 + 64*a^3*b^2 + 256
*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^3 + 128*b^3*c^2 + 192*a*b^2*c^2 - 192*a^2
*b^2*c - 512*a*b*c^3 + 512*a^3*b*c) - 256*a*b*c^2 + 64*a*b^2*c - 256*a^2*b*
c))*((8*a^3*c + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^2*c^2 -
6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c
^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2) + (ta
n(x/2)*(96*a*b^2 - 128*a^2*b - 64*a*c^2 + 32*b^2*c + 64*a^3 - 32*b^3) - ((8
*a^3*c + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^2*c^2 - 6*a*b^2
c)/(2(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*
a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^3 +
128*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32*a^2*b^2 + 256*a^2*c^2 - 32*b
^2*c^2 - tan(x/2)*((8*a^3*c + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2
+ 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a
^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)
))^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c - 64*b^4*c - 128*a^2*b^3 + 64*a^
3*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^3 + 128*b^3*c^2 + 192*a*b^2*c^
2 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c) - 256*a*b*c^2 + 64*a*b^2*c -
256*a^2*b*c))*((8*a^3*c + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8
*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*
c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^
(1/2)))*((8*a^3*c + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^2*c^
2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b
^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*2
i + atan(((tan(x/2)*(96*a*b^2 - 128*a^2*b - 64*a*c^2 + 32*b^2*c + 64*a^3 -
32*b^3) + ((8*a^3*c - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^2*
c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 +
b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)
*(64*a*b^3 + 128*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32*a^2*b^2 + 256*a
^2*c^2 - 32*b^2*c^2 + tan(x/2)*((8*a^3*c - b*(-(4*a*c - b^2)^3)^(1/2) + b^4
- 2*a^2*b^2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a
^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2
+ 10*a*b^4*c)))^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c - 64*b^4*c - 128*a^
2*b^3 + 64*a^3*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^3 + 128*b^3*c^2 +
192*a*b^2*c^2 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c) - 256*a*b*c^2 +
64*a*b^2*c - 256*a^2*b*c))*((8*a^3*c - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 -

```
2*a^2*b^2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*
c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 1
0*a*b^4*c)))^(1/2)*1i + (tan(x/2)*(96*a*b^2 - 128*a^2*b - 64*a*c^2 + 32*b^2
*c + 64*a^3 - 32*b^3) - ((8*a^3*c - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^
2*b^2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3
+ 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*
b^4*c)))^(1/2)*(64*a*b^3 + 128*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32*a
^2*b^2 + 256*a^2*c^2 - 32*b^2*c^2 - tan(x/2)*((8*a^3*c - b*(-(4*a*c - b^2)^
3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*
a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 3
2*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c - 64*
b^4*c - 128*a^2*b^3 + 64*a^3*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^3 +
 128*b^3*c^2 + 192*a*b^2*c^2 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c) -
 256*a*b*c^2 + 64*a*b^2*c - 256*a^2*b*c))*((8*a^3*c - b*(-(4*a*c - b^2)^3)^
(1/2) + b^4 - 2*a^2*b^2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2
*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a
^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*1i)/(64*a*c - 64*a*b + 64*a^2 - (tan(x/2)*
(96*a*b^2 - 128*a^2*b - 64*a*c^2 + 32*b^2*c + 64*a^3 - 32*b^3) + ((8*a^3*c
- b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*
(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c
^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^3 + 128*a*c
^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32*a^2*b^2 + 256*a^2*c^2 - 32*b^2*c^2
+ tan(x/2)*((8*a^3*c - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^2
*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2
+ b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2
)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c - 64*b^4*c - 128*a^2*b^3 + 64*a^3*b^2 +
 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^3 + 128*b^3*c^2 + 192*a*b^2*c^2 - 192
*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c) - 256*a*b*c^2 + 64*a*b^2*c - 256*a^
2*b*c))*((8*a^3*c - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^2*c^
2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b
^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2) +
 (tan(x/2)*(96*a*b^2 - 128*a^2*b - 64*a*c^2 + 32*b^2*c + 64*a^3 - 32*b^3) -
 ((8*a^3*c - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^2*c^2 - 6*a
*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2
- 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^
3 + 128*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32*a^2*b^2 + 256*a^2*c^2 -
32*b^2*c^2 - tan(x/2)*((8*a^3*c - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*
b^2 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 +
16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^
4*c)))^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c - 64*b^4*c - 128*a^2*b^3 + 6
4*a^3*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^3 + 128*b^3*c^2 + 192*a*b^
2*c^2 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c) - 256*a*b*c^2 + 64*a*b^2
*c - 256*a^2*b*c))*((8*a^3*c - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2
 + 8*a^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*
a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c
```

```
)))^(1/2)))*((8*a^3*c - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 - 2*a^2*b^2 + 8*a^
2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2
 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/
2)*2i
```

3.17 $\int \frac{1}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 14, antiderivative size = 223

$$\int \frac{1}{a+b\cos(x)+c\cos^2(x)}\,dx=\frac{4c\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}}-\frac{4c\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}$$

```
[Out] 4*c*arctan((b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c-(-4*a*c+b^2)^
(1/2))^(1/2))/(-4*a*c+b^2)^(1/2)/(b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)/(b+2*c-(-
4*a*c+b^2)^(1/2))^(1/2)-4*c*arctan((b-2*c+(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2
*x)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2))/(-4*a*c+b^2)^(1/2)/(b-2*c+(-4*a*c+b^2
)^(1/2))^(1/2)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2)
```

Rubi [A] (verified)

Time = 0.36 (sec) , antiderivative size = 223, normalized size of antiderivative = 1.00, number of steps used = 5, number of rules used = 3, $\frac{\text{number of rules}}{\text{integrand size}}$ = 0.214, Rules used = {3330, 2738, 211}

$$\int \frac{1}{a+b\cos(x)+c\cos^2(x)}\,dx=\frac{4c\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{-\sqrt{b^2-4ac}+b-2c}}{\sqrt{-\sqrt{b^2-4ac}+b+2c}}\right)}{\sqrt{b^2-4ac}\sqrt{-\sqrt{b^2-4ac}+b-2c}\sqrt{-\sqrt{b^2-4ac}+b+2c}}-\frac{4c\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{\sqrt{b^2-4ac}+b-2c}}{\sqrt{\sqrt{b^2-4ac}+b+2c}}\right)}{\sqrt{b^2-4ac}\sqrt{\sqrt{b^2-4ac}+b-2c}\sqrt{\sqrt{b^2-4ac}+b+2c}}$$

```
[In] Int[(a + b*Cos[x] + c*Cos[x]^2)^(-1),x]
```

```
[Out] (4*c*ArcTan[(Sqrt[b - 2*c - Sqrt[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c - Sqr
t[b^2 - 4*a*c]]])/(Sqrt[b^2 - 4*a*c]*Sqrt[b - 2*c - Sqrt[b^2 - 4*a*c]]*Sqrt
[b + 2*c - Sqrt[b^2 - 4*a*c]]) - (4*c*ArcTan[(Sqrt[b - 2*c + Sqrt[b^2 - 4*a
*c]]*Tan[x/2])/Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]]])/(Sqrt[b^2 - 4*a*c]*Sqrt[
b - 2*c + Sqrt[b^2 - 4*a*c]]*Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]])
```

Rule 211

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(Rt[a/b, 2]/a)*ArcTan[x/R
t[a/b, 2]], x] /; FreeQ[{a, b}, x] && PosQ[a/b]
```

Rule 2738

```
Int[((a_) + (b_.)*sin[Pi/2 + (c_.) + (d_.)*(x_)])^(-1), x_Symbol] :> With[{
e = FreeFactors[Tan[(c + d*x)/2], x]}, Dist[2*(e/d), Subst[Int[1/(a + b + (
a - b)*e^2*x^2), x], x, Tan[(c + d*x)/2]/e], x]] /; FreeQ[{a, b, c, d}, x]
&& NeQ[a^2 - b^2, 0]
```

Rule 3330

```
Int[((a_.) + cos[(d_.) + (e_.)*(x_)]^(n_.)*(b_.) + cos[(d_.) + (e_.)*(x_)]^
(n2_.)*(c_.))^(-1), x_Symbol] :> Module[{q = Rt[b^2 - 4*a*c, 2]}, Dist[2*(c
/q), Int[1/(b - q + 2*c*Cos[d + e*x]^n), x], x] - Dist[2*(c/q), Int[1/(b +
q + 2*c*Cos[d + e*x]^n), x], x]] /; FreeQ[{a, b, c, d, e, n}, x] && EqQ[n2,
 2*n] && NeQ[b^2 - 4*a*c, 0]
```

Rubi steps

$$
\begin{aligned}
\text{integral} &= \frac{(2c)\int \frac{1}{b-\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{\sqrt{b^2-4ac}} - \frac{(2c)\int \frac{1}{b+\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{\sqrt{b^2-4ac}}\\
&= \frac{(4c)\text{Subst}\left(\int \frac{1}{b+2c-\sqrt{b^2-4ac}+\left(b-2c-\sqrt{b^2-4ac}\right)x^2}\,dx, x, \tan\left(\frac{x}{2}\right)\right)}{\sqrt{b^2-4ac}}\\
&\quad - \frac{(4c)\text{Subst}\left(\int \frac{1}{b+2c+\sqrt{b^2-4ac}+\left(b-2c+\sqrt{b^2-4ac}\right)x^2}\,dx, x, \tan\left(\frac{x}{2}\right)\right)}{\sqrt{b^2-4ac}}\\
&= \frac{4c\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}}\\
&\quad - \frac{4c\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}
\end{aligned}
$$

Mathematica [A] (verified)

Time = 0.66 (sec) , antiderivative size = 198, normalized size of antiderivative = 0.89

$$\int \frac{1}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{2\sqrt{2}c\left(\frac{\operatorname{arctanh}\left(\frac{\left(b-2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)-2b\sqrt{b^2-4ac}}}\right)}{\sqrt{-b^2+2c(a+c)-b\sqrt{b^2-4ac}}}+\frac{\operatorname{arctanh}\left(\frac{\left(-b+2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)+2b\sqrt{b^2-4ac}}}\right)}{\sqrt{-b^2+2c(a+c)+b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}}$$

```
[In] Integrate[(a + b*Cos[x] + c*Cos[x]^2)^(-1),x]
```

```
[Out] (2*Sqrt[2]*c*(ArcTanh[((b - 2*c + Sqrt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2
+ 4*c*(a + c) - 2*b*Sqrt[b^2 - 4*a*c]]]/Sqrt[-b^2 + 2*c*(a + c) - b*Sqrt[b^
2 - 4*a*c]] + ArcTanh[((-b + 2*c + Sqrt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2
 + 4*c*(a + c) + 2*b*Sqrt[b^2 - 4*a*c]]]/Sqrt[-b^2 + 2*c*(a + c) + b*Sqrt[b
^2 - 4*a*c]]))/Sqrt[b^2 - 4*a*c]
```

Maple [A] (verified)

Time = 1.12 (sec) , antiderivative size = 204, normalized size of antiderivative = 0.91

method	result
default	$2(a-b+c)\left(\frac{\left(\sqrt{-4ac+b^2}+2c-b\right)\operatorname{arctanh}\left(\frac{(-a+b-c)\tan\left(\frac{x}{2}\right)}{\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}\right)}{2\sqrt{-4ac+b^2}\,(a-b+c)\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}+\frac{\left(b-2c+\sqrt{-4ac+b^2}\right)\arctan\left(\frac{(a-b+c)\,\mathrm{ta}}{\sqrt{\left(\sqrt{-4ac+b^2}+a\right.}}\right.}{2\sqrt{-4ac+b^2}\,(a-b+c)\sqrt{\left(\sqrt{-4ac+b^2}+a-c\right.}}\right.$
risch	$\sum_{\_R=\operatorname{RootOf}\left(\left(16a^4c^2-8a^3b^2c+32c^3a^3+a^2b^4-32a^2b^2c^2+16a^2c^4+10a\,b^4c-8a\,b^2c^3-b^6+b^4c^2\right)\_Z^4+\left(8a^2c^2-6a\,b^2c+8a\,c^3+b^4-2b^2c\right.\right.}$

```
[In] int(1/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] 2*(a-b+c)*(1/2*((-4*a*c+b^2)^(1/2)+2*c-b)/(-4*a*c+b^2)^(1/2)/(a-b+c)/(((-4*
a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2)*arctanh((-a+b-c)*tan(1/2*x)/(((-4*a*c+b^
2)^(1/2)-a+c)*(a-b+c))^(1/2))+1/2*(b-2*c+(-4*a*c+b^2)^(1/2))/(-4*a*c+b^2)^(
1/2)/(a-b+c)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)*arctan((a-b+c)*tan(1/
2*x)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)))
```

Fricas [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 3493 vs. $2(183) = 366$.

Time = 0.52 (sec) , antiderivative size = 3493, normalized size of antiderivative = 15.66

$$\int \frac{1}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(1/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
```

```
[Out] 1/4*sqrt(2)*sqrt(-(b^2 - 2*a*c - 2*c^2 - (a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2
- b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4
*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^
2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c)))/(a^2*b^2 - b^4 - 4*a*c^3
- (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c))*log(b^2*c*cos(x) + 2*b*c^2 -
(4*a*c^4 + (8*a^2 - b^2)*c^3 + 2*(2*a^3 - 3*a*b^2)*c^2 - (a^2*b^2 - b^4)*c)
*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2
*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 +
 2*a*b^4)*c))*cos(x) + 1/2*sqrt(2)*((a^2*b^4 - b^6 + 8*a*c^5 + 2*(12*a^2 -
b^2)*c^4 + 6*(4*a^3 - 3*a*b^2)*c^3 + (8*a^4 - 22*a^2*b^2 + 3*b^4)*c^2 - 2*(
3*a^3*b^2 - 4*a*b^4)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16
*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2
 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c))*sin(x) + (b^4 - 4*a*b^2*c)*sin(x))*sqr
t(-(b^2 - 2*a*c - 2*c^2 - (a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*
(2*a^3 - 3*a*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^
2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 -
4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c)))/(a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)
*c^2 - 2*(2*a^3 - 3*a*b^2)*c))) - 1/4*sqrt(2)*sqrt(-(b^2 - 2*a*c - 2*c^2 -
(a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)*sqrt(
b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 -
 a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b
^4)*c)))/(a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)
*c))*log(b^2*c*cos(x) + 2*b*c^2 - (4*a*c^4 + (8*a^2 - b^2)*c^3 + 2*(2*a^3 -
 3*a*b^2)*c^2 - (a^2*b^2 - b^4)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*
a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2
 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c))*cos(x) - 1/2*sqrt(2)*((a^2*
b^4 - b^6 + 8*a*c^5 + 2*(12*a^2 - b^2)*c^4 + 6*(4*a^3 - 3*a*b^2)*c^3 + (8*a
^4 - 22*a^2*b^2 + 3*b^4)*c^2 - 2*(3*a^3*b^2 - 4*a*b^4)*c)*sqrt(b^2/(a^4*b^2
 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3
- 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c))*sin(
x) + (b^4 - 4*a*b^2*c)*sin(x))*sqrt(-(b^2 - 2*a*c - 2*c^2 - (a^2*b^2 - b^4
- 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)*sqrt(b^2/(a^4*b^2 -
2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2
*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c)))/(a^2*b
^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c))) + 1/4*sqr
```

```
t(2)*sqrt(-(b^2 - 2*a*c - 2*c^2 + (a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*
c^2 - 2*(2*a^3 - 3*a*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5
- (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4
)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c)))/(a^2*b^2 - b^4 - 4*a*c^3 - (8*a^
2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c))*log(-b^2*c*cos(x) - 2*b*c^2 - (4*a*c
^4 + (8*a^2 - b^2)*c^3 + 2*(2*a^3 - 3*a*b^2)*c^2 - (a^2*b^2 - b^4)*c)*sqrt(
b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 -
 a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b
^4)*c))*cos(x) + 1/2*sqrt(2)*((a^2*b^4 - b^6 + 8*a*c^5 + 2*(12*a^2 - b^2)*c
^4 + 6*(4*a^3 - 3*a*b^2)*c^3 + (8*a^4 - 22*a^2*b^2 + 3*b^4)*c^2 - 2*(3*a^3*
b^2 - 4*a*b^4)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 -
 b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(
a^5 - 3*a^3*b^2 + 2*a*b^4)*c))*sin(x) - (b^4 - 4*a*b^2*c)*sin(x))*sqrt(-(b^
2 - 2*a*c - 2*c^2 + (a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3
 - 3*a*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^
2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5
 - 3*a^3*b^2 + 2*a*b^4)*c)))/(a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 -
 2*(2*a^3 - 3*a*b^2)*c))) - 1/4*sqrt(2)*sqrt(-(b^2 - 2*a*c - 2*c^2 + (a^2*b
^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)*sqrt(b^2/(a
^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2
)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c)
))/(a^2*b^2 - b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c))*l
og(-b^2*c*cos(x) - 2*b*c^2 - (4*a*c^4 + (8*a^2 - b^2)*c^3 + 2*(2*a^3 - 3*a*
b^2)*c^2 - (a^2*b^2 - b^4)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2*b^4 + b^6 - 4*a*c^5
 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a^4 - 11*a^2*b^2 + b^
4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c))*cos(x) - 1/2*sqrt(2)*((a^2*b^4 -
 b^6 + 8*a*c^5 + 2*(12*a^2 - b^2)*c^4 + 6*(4*a^3 - 3*a*b^2)*c^3 + (8*a^4 -
22*a^2*b^2 + 3*b^4)*c^2 - 2*(3*a^3*b^2 - 4*a*b^4)*c)*sqrt(b^2/(a^4*b^2 - 2*
a^2*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(
8*a^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c))*sin(x) -
(b^4 - 4*a*b^2*c)*sin(x))*sqrt(-(b^2 - 2*a*c - 2*c^2 + (a^2*b^2 - b^4 - 4*a
*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)*sqrt(b^2/(a^4*b^2 - 2*a^2
*b^4 + b^6 - 4*a*c^5 - (16*a^2 - b^2)*c^4 - 12*(2*a^3 - a*b^2)*c^3 - 2*(8*a
^4 - 11*a^2*b^2 + b^4)*c^2 - 4*(a^5 - 3*a^3*b^2 + 2*a*b^4)*c)))/(a^2*b^2 -
b^4 - 4*a*c^3 - (8*a^2 - b^2)*c^2 - 2*(2*a^3 - 3*a*b^2)*c)))
```

Sympy [F(-1)]

Timed out.

$$\int \frac{1}{a+b\cos(x)+c\cos^2(x)}\,dx=\text{Timed out}$$

```
[In] integrate(1/(a+b*cos(x)+c*cos(x)**2),x)
```

```
[Out] Timed out
```

Maxima [F]

$$\int \frac{1}{a+b\cos(x)+c\cos^2(x)}\,dx = \int \frac{1}{c\cos(x)^2+b\cos(x)+a}\,dx$$

```
[In] integrate(1/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
[Out] integrate(1/(c*cos(x)^2 + b*cos(x) + a), x)
```

Giac [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 2994 vs. 2(183) = 366.

Time = 1.46 (sec) , antiderivative size = 2994, normalized size of antiderivative = 13.43

$$\int \frac{1}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(1/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
[Out] -(2*a^2*b^3 - 8*a*b^4 + 6*b^5 - 8*a^3*b*c + 52*a^2*b^2*c - 44*a*b^3*c - 4*b
^4*c - 80*a^3*c^2 + 80*a^2*b*c^2 + 40*a*b^2*c^2 - 6*b^3*c^2 - 96*a^2*c^3 +
24*a*b*c^3 + 4*b^2*c^3 - 16*a*c^4 - 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
 - 4*a*c)*(a - b + c))*a^2*b^2 + 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
4*a*c)*(a - b + c))*a*b^3 + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c
)*(a - b + c))*b^4 + 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
 b + c))*a^3*c - 8*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*a^2*b*c - 34*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c)
)*a*b^2*c - 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b
^3*c + 56*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*c
^2 + 24*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b*c^2
 + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*b^2*c^2 -
20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*c^3 + 3*sq
rt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)
*a^2*b - 2*(b^2 - 4*a*c)*a^2*b - 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2 + 8*(b^2 - 4*a*c)*a*b^2 - 5*sqr
t(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*
b^3 - 6*(b^2 - 4*a*c)*b^3 + 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c - 20*(b^2 - 4*a*c)*a^2*c + 10*sqrt(a
^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b
*c + 20*(b^2 - 4*a*c)*a*b*c - 4*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a
*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c + 4*(b^2 - 4*a*c)*b^2*c + 28*sqrt(
a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*
c^2 - 24*(b^2 - 4*a*c)*a*c^2 + 7*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b*c^2 + 6*(b^2 - 4*a*c)*b*c^2 - 10*sqrt
```

```
(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*c
^3 - 4*(b^2 - 4*a*c)*c^3)*(pi*floor(1/2*x/pi + 1/2) + arctan(2*sqrt(1/2)*ta
n(1/2*x)/sqrt((2*a - 2*c + sqrt(-4*(a + b + c)*(a - b + c) + 4*(a - c)^2))/
(a - b + c))))*abs(a - b + c)/(3*a^5*b^2 - 5*a^4*b^3 - 6*a^3*b^4 + 10*a^2*b
^5 + 3*a*b^6 - 5*b^7 - 12*a^6*c + 20*a^5*b*c + 47*a^4*b^2*c - 60*a^3*b^3*c
 - 46*a^2*b^4*c + 40*a*b^5*c + 11*b^6*c - 92*a^5*c^2 + 80*a^4*b*c^2 + 182*a^
3*b^2*c^2 - 94*a^2*b^3*c^2 - 78*a*b^4*c^2 - 6*b^5*c^2 - 184*a^4*c^3 + 56*a^
3*b*c^3 + 166*a^2*b^2*c^3 + 36*a*b^3*c^3 - 6*b^4*c^3 - 120*a^3*c^4 - 48*a^2
*b*c^4 + 23*a*b^2*c^4 + 11*b^3*c^4 + 4*a^2*c^5 - 44*a*b*c^5 - 5*b^2*c^5 + 2
0*a*c^6) + (2*a^2*b^3 - 8*a*b^4 + 6*b^5 - 8*a^3*b*c + 52*a^2*b^2*c - 44*a*b
^3*c - 4*b^4*c - 80*a^3*c^2 + 80*a^2*b*c^2 + 40*a*b^2*c^2 - 6*b^3*c^2 - 96*
a^2*c^3 + 24*a*b*c^3 + 4*b^2*c^3 - 16*a*c^4 + 3*sqrt(a^2 - a*b + b*c - c^2
 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^2 - 2*sqrt(a^2 - a*b + b*c - c^2 - s
qrt(b^2 - 4*a*c)*(a - b + c))*a*b^3 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b
^2 - 4*a*c)*(a - b + c))*b^4 - 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4
*a*c)*(a - b + c))*a^3*c + 8*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)
*(a - b + c))*a^2*b*c + 34*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(
a - b + c))*a*b^2*c + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
 b + c))*b^3*c - 56*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
 c))*a^2*c^2 - 24*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c
))*a*b*c^2 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*
b^2*c^2 + 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*
c^3 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^
2 - 4*a*c)*a^2*b - 2*(b^2 - 4*a*c)*a^2*b - 2*sqrt(a^2 - a*b + b*c - c^2 - s
qrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2 + 8*(b^2 - 4*a*c)*a*b
^2 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
 - 4*a*c)*b^3 - 6*(b^2 - 4*a*c)*b^3 + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b
^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c - 20*(b^2 - 4*a*c)*a^2*c +
 10*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
4*a*c)*a*b*c + 20*(b^2 - 4*a*c)*a*b*c - 4*sqrt(a^2 - a*b + b*c - c^2 - sqrt
(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c + 4*(b^2 - 4*a*c)*b^2*c
+ 28*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
 4*a*c)*a*c^2 - 24*(b^2 - 4*a*c)*a*c^2 + 7*sqrt(a^2 - a*b + b*c - c^2 - sqr
t(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b*c^2 + 6*(b^2 - 4*a*c)*b*c^2
 - 10*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*c^3 - 4*(b^2 - 4*a*c)*c^3)*(pi*floor(1/2*x/pi + 1/2) + arctan(2*sq
rt(1/2)*tan(1/2*x)/sqrt((2*a - 2*c - sqrt(-4*(a + b + c)*(a - b + c) + 4*(a
 - c)^2))/(a - b + c))))*abs(a - b + c)/(3*a^5*b^2 - 5*a^4*b^3 - 6*a^3*b^4
+ 10*a^2*b^5 + 3*a*b^6 - 5*b^7 - 12*a^6*c + 20*a^5*b*c + 47*a^4*b^2*c - 60*
a^3*b^3*c - 46*a^2*b^4*c + 40*a*b^5*c + 11*b^6*c - 92*a^5*c^2 + 80*a^4*b*c^
2 + 182*a^3*b^2*c^2 - 94*a^2*b^3*c^2 - 78*a*b^4*c^2 - 6*b^5*c^2 - 184*a^4*c
^3 + 56*a^3*b*c^3 + 166*a^2*b^2*c^3 + 36*a*b^3*c^3 - 6*b^4*c^3 - 120*a^3*c^
4 - 48*a^2*b*c^4 + 23*a*b^2*c^4 + 11*b^3*c^4 + 4*a^2*c^5 - 44*a*b*c^5 - 5*b
^2*c^5 + 20*a*c^6)
```

Mupad [B] (verification not implemented)

Time = 13.76 (sec) , antiderivative size = 5514, normalized size of antiderivative = 24.73

$$\int \frac{1}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] int(1/(a + b*cos(x) + c*cos(x)^2),x)
```

```
[Out] - atan(((tan(x/2)*(32*a*b^2 - 64*a^2*c - 128*b*c^2 + 96*b^2*c - 32*b^3 + 64
*c^3) + (-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c
^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 +
b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*
(64*a*b^3 + 128*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32*a^2*b^2 + 256*a^
2*c^2 - 32*b^2*c^2 + tan(x/2)*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4
 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a
^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2
+ 10*a*b^4*c)))^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c - 64*b^4*c - 128*a^
2*b^3 + 64*a^3*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^3 + 128*b^3*c^2 +
 192*a*b^2*c^2 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c) - 256*a*b*c^2 +
 64*a*b^2*c - 256*a^2*b*c))*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 +
 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3
*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 +
10*a*b^4*c)))^(1/2)*1i + (tan(x/2)*(32*a*b^2 - 64*a^2*c - 128*b*c^2 + 96*b^
2*c - 32*b^3 + 64*c^3) - (-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*
a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^
3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*
a*b^4*c)))^(1/2)*(64*a*b^3 + 128*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32
*a^2*b^2 + 256*a^2*c^2 - 32*b^2*c^2 - tan(x/2)*(-(8*a*c^3 + b*(-(4*a*c - b^
2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 +
16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c
- 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c -
64*b^4*c - 128*a^2*b^3 + 64*a^3*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^
3 + 128*b^3*c^2 + 192*a*b^2*c^2 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c
) - 256*a*b*c^2 + 64*a*b^2*c - 256*a^2*b*c))*(-(8*a*c^3 + b*(-(4*a*c - b^2)
^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16
*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c -
32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*1i)/(64*a*c - 64*b*c + 64*c^2 + (tan(x
/2)*(32*a*b^2 - 64*a^2*c - 128*b*c^2 + 96*b^2*c - 32*b^3 + 64*c^3) + (-(8*a
*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c
)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*
b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^3 + 12
8*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32*a^2*b^2 + 256*a^2*c^2 - 32*b^2
*c^2 + tan(x/2)*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 -
 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^
```

```
4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c))
)^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c - 64*b^4*c - 128*a^2*b^3 + 64*a^3
*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^3 + 128*b^3*c^2 + 192*a*b^2*c^2
 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c) - 256*a*b*c^2 + 64*a*b^2*c -
256*a^2*b*c))*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2
*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*
c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^
(1/2) - (tan(x/2)*(32*a*b^2 - 64*a^2*c - 128*b*c^2 + 96*b^2*c - 32*b^3 + 64
*c^3) - (-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c
^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 +
b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*
(64*a*b^3 + 128*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32*a^2*b^2 + 256*a^
2*c^2 - 32*b^2*c^2 - tan(x/2)*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4
 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a
^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2
+ 10*a*b^4*c)))^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c - 64*b^4*c - 128*a^
2*b^3 + 64*a^3*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^3 + 128*b^3*c^2 +
 192*a*b^2*c^2 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c) - 256*a*b*c^2 +
 64*a*b^2*c - 256*a^2*b*c))*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 +
 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3
*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 +
10*a*b^4*c)))^(1/2)))*(-(8*a*c^3 + b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2
*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 +
 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b
^4*c)))^(1/2)*2i - atan(((tan(x/2)*(32*a*b^2 - 64*a^2*c - 128*b*c^2 + 96*b^
2*c - 32*b^3 + 64*c^3) + (-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*
a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^
3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*
a*b^4*c)))^(1/2)*(64*a*b^3 + 128*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32
*a^2*b^2 + 256*a^2*c^2 - 32*b^2*c^2 + tan(x/2)*(-(8*a*c^3 - b*(-(4*a*c - b^
2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 +
16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c
- 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c -
64*b^4*c - 128*a^2*b^3 + 64*a^3*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^
3 + 128*b^3*c^2 + 192*a*b^2*c^2 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c
) - 256*a*b*c^2 + 64*a*b^2*c - 256*a^2*b*c))*(-(8*a*c^3 - b*(-(4*a*c - b^2)
^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16
*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c -
32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*1i + (tan(x/2)*(32*a*b^2 - 64*a^2*c -
128*b*c^2 + 96*b^2*c - 32*b^3 + 64*c^3) - (-(8*a*c^3 - b*(-(4*a*c - b^2)^3)
^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^
2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*
a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^3 + 128*a*c^3 + 128*a^3*c + 64*b^
3*c - 32*b^4 - 32*a^2*b^2 + 256*a^2*c^2 - 32*b^2*c^2 - tan(x/2)*(-(8*a*c^3
- b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*
```

```
(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c
^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^4 + 256*a*c
^4 - 256*a^4*c - 64*b^4*c - 128*a^2*b^3 + 64*a^3*b^2 + 256*a^2*c^3 - 256*a^
3*c^2 - 64*b^2*c^3 + 128*b^3*c^2 + 192*a*b^2*c^2 - 192*a^2*b^2*c - 512*a*b*
c^3 + 512*a^3*b*c) - 256*a*b*c^2 + 64*a*b^2*c - 256*a^2*b*c))*(-(8*a*c^3 -
b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a
^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3
 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*1i)/(64*a*c - 64*b*c
+ 64*c^2 + (tan(x/2)*(32*a*b^2 - 64*a^2*c - 128*b*c^2 + 96*b^2*c - 32*b^3 +
 64*c^3) + (-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^
2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2
 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/
2)*(64*a*b^3 + 128*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32*a^2*b^2 + 256
*a^2*c^2 - 32*b^2*c^2 + tan(x/2)*(-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) +
b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 3
2*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c
^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c - 64*b^4*c - 128
*a^2*b^3 + 64*a^3*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^3 + 128*b^3*c^
2 + 192*a*b^2*c^2 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c) - 256*a*b*c^
2 + 64*a*b^2*c - 256*a^2*b*c))*(-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^
4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*
a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2
 + 10*a*b^4*c)))^(1/2) - (tan(x/2)*(32*a*b^2 - 64*a^2*c - 128*b*c^2 + 96*b^
2*c - 32*b^3 + 64*c^3) - (-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1/2) + b^4 + 8*
a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c^4 + 32*a^3*c^
3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2*b^2*c^2 + 10*
a*b^4*c)))^(1/2)*(64*a*b^3 + 128*a*c^3 + 128*a^3*c + 64*b^3*c - 32*b^4 - 32
*a^2*b^2 + 256*a^2*c^2 - 32*b^2*c^2 - tan(x/2)*(-(8*a*c^3 - b*(-(4*a*c - b^
2)^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 +
16*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c
- 32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)*(64*a*b^4 + 256*a*c^4 - 256*a^4*c -
64*b^4*c - 128*a^2*b^3 + 64*a^3*b^2 + 256*a^2*c^3 - 256*a^3*c^2 - 64*b^2*c^
3 + 128*b^3*c^2 + 192*a*b^2*c^2 - 192*a^2*b^2*c - 512*a*b*c^3 + 512*a^3*b*c
) - 256*a*b*c^2 + 64*a*b^2*c - 256*a^2*b*c))*(-(8*a*c^3 - b*(-(4*a*c - b^2)
^3)^(1/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16
*a^2*c^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c -
32*a^2*b^2*c^2 + 10*a*b^4*c)))^(1/2)))*(-(8*a*c^3 - b*(-(4*a*c - b^2)^3)^(1
/2) + b^4 + 8*a^2*c^2 - 2*b^2*c^2 - 6*a*b^2*c)/(2*(a^2*b^4 - b^6 + 16*a^2*c
^4 + 32*a^3*c^3 + 16*a^4*c^2 + b^4*c^2 - 8*a*b^2*c^3 - 8*a^3*b^2*c - 32*a^2
*b^2*c^2 + 10*a*b^4*c)))^(1/2)*2i
```

3.18 $\int \frac{\sec(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 17, antiderivative size = 245

$$\int \frac{\sec(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{2c\left(1+\frac{b}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{a\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} - \frac{2c\left(1-\frac{b}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{a\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}} + \frac{\operatorname{arctanh}(\sin(x))}{a}$$

```
[Out] arctanh(sin(x))/a-2*c*arctan((b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b
+2*c-(-4*a*c+b^2)^(1/2))^(1/2))*(1+b/(-4*a*c+b^2)^(1/2))/a/(b-2*c-(-4*a*c+b
^2)^(1/2))^(1/2)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2)-2*c*arctan((b-2*c+(-4*a*c
+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2))*(1-b/(-4*a*
c+b^2)^(1/2))/a/(b-2*c+(-4*a*c+b^2)^(1/2))^(1/2)/(b+2*c+(-4*a*c+b^2)^(1/2))
^(1/2)
```

Rubi [A] (verified)

Time = 0.80 (sec) , antiderivative size = 245, normalized size of antiderivative = 1.00, number of steps used = 8, number of rules used = 5, $\frac{\text{number of rules}}{\text{integrand size}} = 0.294$, Rules used

$= \{3338, 3374, 2738, 211, 3855\}$

$$\int \frac{\sec(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{2c\left(\frac{b}{\sqrt{b^2-4ac}}+1\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{-\sqrt{b^2-4ac}+b-2c}}{\sqrt{-\sqrt{b^2-4ac}+b+2c}}\right)}{a\sqrt{-\sqrt{b^2-4ac}+b-2c}\sqrt{-\sqrt{b^2-4ac}+b+2c}} - \frac{2c\left(1-\frac{b}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{\sqrt{b^2-4ac}+b-2c}}{\sqrt{\sqrt{b^2-4ac}+b+2c}}\right)}{a\sqrt{\sqrt{b^2-4ac}+b-2c}\sqrt{\sqrt{b^2-4ac}+b+2c}} + \frac{\operatorname{arctanh}(\sin(x))}{a}$$

```
[In] Int[Sec[x]/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (-2*c*(1 + b/Sqrt[b^2 - 4*a*c])*ArcTan[(Sqrt[b - 2*c - Sqrt[b^2 - 4*a*c]]*T
an[x/2])/Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]])/(a*Sqrt[b - 2*c - Sqrt[b^2 - 4
*a*c]]*Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]) - (2*c*(1 - b/Sqrt[b^2 - 4*a*c])*
ArcTan[(Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c + Sqrt[b^2
 - 4*a*c]]])/(a*Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]*Sqrt[b + 2*c + Sqrt[b^2 -
 4*a*c]]) + ArcTanh[Sin[x]]/a
```

Rule 211

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(Rt[a/b, 2]/a)*ArcTan[x/R
t[a/b, 2]], x] /; FreeQ[{a, b}, x] && PosQ[a/b]
```

Rule 2738

```
Int[((a_) + (b_.)*sin[Pi/2 + (c_.) + (d_.)*(x_)])^(-1), x_Symbol] :> With[{
e = FreeFactors[Tan[(c + d*x)/2], x]}, Dist[2*(e/d), Subst[Int[1/(a + b + (
a - b)*e^2*x^2), x], x, Tan[(c + d*x)/2]/e], x]] /; FreeQ[{a, b, c, d}, x]
&& NeQ[a^2 - b^2, 0]
```

Rule 3338

```
Int[cos[(d_.) + (e_.)*(x_)]^(m_.)*((a_.) + cos[(d_.) + (e_.)*(x_)]^(n_.)*(b
_.) + cos[(d_.) + (e_.)*(x_)]^(n2_.)*(c_.))^(p_), x_Symbol] :> Int[ExpandTr
ig[cos[d + e*x]^m*(a + b*cos[d + e*x]^n + c*cos[d + e*x]^(2*n))^p, x], x] /
; FreeQ[{a, b, c, d, e}, x] && EqQ[n2, 2*n] && NeQ[b^2 - 4*a*c, 0] && Integ
ersQ[m, n, p]
```

Rule 3374

```
Int[(cos[(d_.) + (e_.)*(x_)]*(B_.) + (A_))/((a_.) + cos[(d_.) + (e_.)*(x_)]
*(b_.) + cos[(d_.) + (e_.)*(x_)]^2*(c_.)), x_Symbol] :> Module[{q = Rt[b^2
- 4*a*c, 2]}, Dist[B + (b*B - 2*A*c)/q, Int[1/(b + q + 2*c*Cos[d + e*x]), x
], x] + Dist[B - (b*B - 2*A*c)/q, Int[1/(b - q + 2*c*Cos[d + e*x]), x], x]]
```

```
/; FreeQ[{a, b, c, d, e, A, B}, x] && NeQ[b^2 - 4*a*c, 0]
```

Rule 3855

```
Int[csc[(c_.) + (d_.)*(x_)], x_Symbol] :> Simp[-ArcTanh[Cos[c + d*x]]/d, x]
 /; FreeQ[{c, d}, x]
```

Rubi steps

$$\begin{aligned}
\text{integral} &= \int \left(\frac{-b-c\cos(x)}{a\left(a+b\cos(x)+c\cos^2(x)\right)}+\frac{\sec(x)}{a}\right)\,dx\\
&= \frac{\int \frac{-b-c\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx}{a}+\frac{\int \sec(x)\,dx}{a}\\
&= \frac{\operatorname{arctanh}(\sin(x))}{a}-\frac{\left(c\left(1-\frac{b}{\sqrt{b^2-4ac}}\right)\right)\int \frac{1}{b+\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{a}\\
&\quad -\frac{\left(c\left(1+\frac{b}{\sqrt{b^2-4ac}}\right)\right)\int \frac{1}{b-\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{a}\\
&= \frac{\operatorname{arctanh}(\sin(x))}{a}\\
&\quad -\frac{\left(2c\left(1-\frac{b}{\sqrt{b^2-4ac}}\right)\right)\operatorname{Subst}\left(\int \frac{1}{b+2c+\sqrt{b^2-4ac}+\left(b-2c+\sqrt{b^2-4ac}\right)x^2}\,dx,x,\tan\left(\frac{x}{2}\right)\right)}{a}\\
&\quad -\frac{\left(2c\left(1+\frac{b}{\sqrt{b^2-4ac}}\right)\right)\operatorname{Subst}\left(\int \frac{1}{b+2c-\sqrt{b^2-4ac}+\left(b-2c-\sqrt{b^2-4ac}\right)x^2}\,dx,x,\tan\left(\frac{x}{2}\right)\right)}{a}\\
&= -\frac{2c\left(1+\frac{b}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{a\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}}\\
&\quad -\frac{2c\left(1-\frac{b}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{a\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}}+\frac{\operatorname{arctanh}(\sin(x))}{a}
\end{aligned}$$

Mathematica [A] (verified)

Time = 1.03 (sec) , antiderivative size = 281, normalized size of antiderivative = 1.15

$$\int \frac{\sec(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{\frac{\sqrt{2}c\left(-b+\sqrt{b^2-4ac}\right)\operatorname{arctanh}\left(\frac{\left(b-2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)-2b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{-b^2+2c(a+c)-b\sqrt{b^2-4ac}}}-\frac{\sqrt{2}c\left(b+\sqrt{b^2-4ac}\right)\operatorname{arctanh}\left(\frac{\left(-b+2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)+2b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{-b^2+2c(a+c)+b\sqrt{b^2-4ac}}}-\log(\cos(}{a}$$

```
[In] Integrate[Sec[x]/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] ((Sqrt[2]*c*(-b + Sqrt[b^2 - 4*a*c])*ArcTanh[((b - 2*c + Sqrt[b^2 - 4*a*c])
*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) - 2*b*Sqrt[b^2 - 4*a*c]]])/(Sqrt[b^2 -
 4*a*c]*Sqrt[-b^2 + 2*c*(a + c) - b*Sqrt[b^2 - 4*a*c]]) - (Sqrt[2]*c*(b + S
qrt[b^2 - 4*a*c])*ArcTanh[((-b + 2*c + Sqrt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2
*b^2 + 4*c*(a + c) + 2*b*Sqrt[b^2 - 4*a*c]]])/(Sqrt[b^2 - 4*a*c]*Sqrt[-b^2
+ 2*c*(a + c) + b*Sqrt[b^2 - 4*a*c]]) - Log[Cos[x/2] - Sin[x/2]] + Log[Cos[
x/2] + Sin[x/2]])/a
```

Maple [A] (verified)

Time = 2.75 (sec) , antiderivative size = 274, normalized size of antiderivative = 1.12

method	result
default	$\frac{2(a-b+c)\left(\frac{\left(-b\sqrt{-4ac+b^2}+c\sqrt{-4ac+b^2}-2ac+b^2-cb\right)\operatorname{arctanh}\left(\frac{(-a+b-c)\tan\left(\frac{x}{2}\right)}{\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}\right)}{2\sqrt{-4ac+b^2}\,(a-b+c)\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}+\frac{\left(-b\sqrt{-4ac+b^2}+c\sqrt{-4ac+b^2}+2ac-b^2+cb\right)\operatorname{ar}}{2\sqrt{-4ac+b^2}\,(a-b+c)\sqrt{\left(\sqrt{-4\right.}}}\right.}{a}$
risch	Expression too large to display

```
[In] int(sec(x)/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] 2/a*(a-b+c)*(1/2*(-b*(-4*a*c+b^2)^(1/2)+c*(-4*a*c+b^2)^(1/2)-2*a*c+b^2-c*b)
/(-4*a*c+b^2)^(1/2)/(a-b+c)/(((-4*a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2)*arctan
h((-a+b-c)*tan(1/2*x)/(((-4*a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2))+1/2*(-b*(-4
*a*c+b^2)^(1/2)+c*(-4*a*c+b^2)^(1/2)+2*a*c-b^2+c*b)/(-4*a*c+b^2)^(1/2)/(a-b
+c)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)*arctan((a-b+c)*tan(1/2*x)/(((-
4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)))+1/a*ln(tan(1/2*x)+1)-1/a*ln(tan(1/2*
x)-1)
```

Fricas [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 5292 vs. 2(204) = 408.

Time = 74.29 (sec) , antiderivative size = 5292, normalized size of antiderivative = 21.60

$$\int \frac{\sec(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(sec(x)/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
[Out] Too large to include
```

Sympy [F]

$$\int \frac{\sec(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \int \frac{\sec(x)}{a + b\cos(x) + c\cos^2(x)}\,dx$$

```
[In] integrate(sec(x)/(a+b*cos(x)+c*cos(x)**2),x)
[Out] Integral(sec(x)/(a + b*cos(x) + c*cos(x)**2), x)
```

Maxima [F]

$$\int \frac{\sec(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \int \frac{\sec(x)}{c\cos(x)^2 + b\cos(x) + a}\,dx$$

```
[In] integrate(sec(x)/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
[Out] -1/2*(2*a*integrate(2*(2*b*c*cos(3*x)^2 + 2*b*c*cos(x)^2 + 2*b*c*sin(3*x)^2
 + 2*b*c*sin(x)^2 + 4*(2*a*b + b*c)*cos(2*x)^2 + c^2*cos(x) + 4*(2*a*b + b*
c)*sin(2*x)^2 + 2*(2*b^2 + 2*a*c + c^2)*sin(2*x)*sin(x) + (c^2*cos(3*x) + 2
*b*c*cos(2*x) + c^2*cos(x))*cos(4*x) + (4*b*c*cos(x) + c^2 + 2*(2*b^2 + 2*a
*c + c^2)*cos(2*x))*cos(3*x) + 2*(b*c + (2*b^2 + 2*a*c + c^2)*cos(x))*cos(2
*x) + (c^2*sin(3*x) + 2*b*c*sin(2*x) + c^2*sin(x))*sin(4*x) + 2*(2*b*c*sin(
x) + (2*b^2 + 2*a*c + c^2)*sin(2*x))*sin(3*x))/(a*c^2*cos(4*x)^2 + 4*a*b^2*
cos(3*x)^2 + 4*a*b^2*cos(x)^2 + a*c^2*sin(4*x)^2 + 4*a*b^2*sin(3*x)^2 + 4*a
*b^2*sin(x)^2 + 4*a*b*c*cos(x) + a*c^2 + 4*(4*a^3 + 4*a^2*c + a*c^2)*cos(2*
x)^2 + 4*(4*a^3 + 4*a^2*c + a*c^2)*sin(2*x)^2 + 8*(2*a^2*b + a*b*c)*sin(2*x
)*sin(x) + 2*(2*a*b*c*cos(3*x) + 2*a*b*c*cos(x) + a*c^2 + 2*(2*a^2*c + a*c^
2)*cos(2*x))*cos(4*x) + 4*(2*a*b^2*cos(x) + a*b*c + 2*(2*a^2*b + a*b*c)*cos
(2*x))*cos(3*x) + 4*(2*a^2*c + a*c^2 + 2*(2*a^2*b + a*b*c)*cos(x))*cos(2*x)
 + 4*(a*b*c*sin(3*x) + a*b*c*sin(x) + (2*a^2*c + a*c^2)*sin(2*x))*sin(4*x)
+ 8*(a*b^2*sin(x) + (2*a^2*b + a*b*c)*sin(2*x))*sin(3*x)), x) - log(cos(x)^
2 + sin(x)^2 + 2*sin(x) + 1) + log(cos(x)^2 + sin(x)^2 - 2*sin(x) + 1))/a
```

Giac [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 8954 vs. $2(204) = 408$.

Time = 2.38 (sec) , antiderivative size = 8954, normalized size of antiderivative = 36.55

$$\int \frac{\sec(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(sec(x)/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] ((2*a^2*b^5 - 4*a*b^6 + 2*b^7 - 16*a^3*b^3*c + 30*a^2*b^4*c - 8*a*b^5*c - 6
*b^6*c + 32*a^4*b*c^2 - 48*a^3*b^2*c^2 - 32*a^2*b^3*c^2 + 44*a*b^4*c^2 + 6*
b^5*c^2 - 32*a^4*c^3 + 128*a^3*b*c^3 - 64*a^2*b^2*c^3 - 48*a*b^3*c^3 - 2*b^
4*c^3 - 64*a^3*c^4 + 96*a^2*b*c^4 + 16*a*b^2*c^4 - 32*a^2*c^5 + 3*sqrt(a^2
- a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^
3 - 2*(b^2 - 4*a*c)*a^2*b^3 - 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a
*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4 + 4*(b^2 - 4*a*c)*a*b^4 - 5*sqrt(a
^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5
 - 2*(b^2 - 4*a*c)*b^5 - 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*
(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c + 8*(b^2 - 4*a*c)*a^3*b*c + 5*sqrt(a
^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2
*b^2*c - 14*(b^2 - 4*a*c)*a^2*b^2*c + 36*sqrt(a^2 - a*b + b*c - c^2 + sqrt(
b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c + 11*sqrt(a^2 - a*b + b
*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c + 6*(b^2
- 4*a*c)*b^4*c + 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
 c))*sqrt(b^2 - 4*a*c)*a^3*c^2 - 8*(b^2 - 4*a*c)*a^3*c^2 - 64*sqrt(a^2 - a*
b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^2
+ 32*(b^2 - 4*a*c)*a^2*b*c^2 - 58*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4
*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^2 - 20*(b^2 - 4*a*c)*a*b^2*c^2
 - 11*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*b^3*c^2 - 6*(b^2 - 4*a*c)*b^3*c^2 + 56*sqrt(a^2 - a*b + b*c - c^2
+ sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^3 - 16*(b^2 - 4*a*
c)*a^2*c^3 + 44*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))
*sqrt(b^2 - 4*a*c)*a*b*c^3 + 24*(b^2 - 4*a*c)*a*b*c^3 + 5*sqrt(a^2 - a*b +
b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c^3 + 2*(b
^2 - 4*a*c)*b^2*c^3 - 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
- b + c))*sqrt(b^2 - 4*a*c)*a*c^4 - 8*(b^2 - 4*a*c)*a*c^4)*a^2*abs(a - b +
c) - (4*a^3*b^5 - 8*a^2*b^6 + 4*a*b^7 - 32*a^4*b^3*c + 68*a^3*b^4*c - 32*a^
2*b^5*c - 4*a*b^6*c + 64*a^5*b*c^2 - 160*a^4*b^2*c^2 + 64*a^3*b^3*c^2 + 40*
a^2*b^4*c^2 - 4*a*b^5*c^2 + 64*a^5*c^3 - 128*a^3*b^2*c^3 + 32*a^2*b^3*c^3 +
 4*a*b^4*c^3 + 128*a^4*c^4 - 64*a^3*b*c^4 - 32*a^2*b^2*c^4 + 64*a^3*c^5 + 3
*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*b^3 - 5*sq
rt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^4 - 3*sqrt(
a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^5 + 5*sqrt(a^2
 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^6 - 12*sqrt(a^2 - a
```

```
*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^5*b*c + 23*sqrt(a^2 - a*b
 + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*b^2*c + 24*sqrt(a^2 - a*b
 + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^3*c - 33*sqrt(a^2 - a*b
 + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^4*c - 6*sqrt(a^2 - a*b
+ b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^5*c - 12*sqrt(a^2 - a*b +
b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^5*c^2 - 48*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*b*c^2 + 69*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b^2*c^2 + 23*sqrt(a^2 - a*b +
b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^3*c^2 - 68*sqrt(a^2 - a*b
+ b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4*c^3 + 4*sqrt(a^2 - a*b + b
*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*b*c^3 + 9*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^2*c^3 + 6*sqrt(a^2 - a*b + b
*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^3*c^3 - 36*sqrt(a^2 - a*b + b
*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3*c^4 - 24*sqrt(a^2 - a*b + b*c
 - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b*c^4 - 5*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^2*c^4 + 20*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*c^5 - 4*(b^2 - 4*a*c)*a^3*b^3 +
8*(b^2 - 4*a*c)*a^2*b^4 - 4*(b^2 - 4*a*c)*a*b^5 + 16*(b^2 - 4*a*c)*a^4*b*c
- 36*(b^2 - 4*a*c)*a^3*b^2*c + 16*(b^2 - 4*a*c)*a^2*b^3*c + 4*(b^2 - 4*a*c)
*a*b^4*c + 16*(b^2 - 4*a*c)*a^4*c^2 - 24*(b^2 - 4*a*c)*a^2*b^2*c^2 + 4*(b^2
 - 4*a*c)*a*b^3*c^2 + 32*(b^2 - 4*a*c)*a^3*c^3 - 16*(b^2 - 4*a*c)*a^2*b*c^3
 - 4*(b^2 - 4*a*c)*a*b^2*c^3 + 16*(b^2 - 4*a*c)*a^2*c^4)*abs(a - b + c)*abs
(a) - (2*a^5*b^4 - 4*a^4*b^5 + 2*a^3*b^6 - 12*a^6*b^2*c + 22*a^5*b^3*c - 6*
a^4*b^4*c - 2*a^3*b^5*c - 2*a^2*b^6*c + 16*a^7*c^2 - 24*a^6*b*c^2 - 12*a^5*
b^2*c^2 + 6*a^4*b^3*c^2 + 10*a^3*b^4*c^2 + 6*a^2*b^5*c^2 + 16*a^6*c^3 + 8*a
^5*b*c^3 - 4*a^4*b^2*c^3 - 30*a^3*b^3*c^3 - 6*a^2*b^4*c^3 - 16*a^5*c^4 + 24
*a^4*b*c^4 + 28*a^3*b^2*c^4 + 2*a^2*b^3*c^4 - 16*a^4*c^5 - 8*a^3*b*c^5 + 3*
sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*
c)*a^5*b^2 - 2*(b^2 - 4*a*c)*a^5*b^2 - 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(
b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^3 + 4*(b^2 - 4*a*c)*a^4*b
^3 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
 - 4*a*c)*a^3*b^4 - 2*(b^2 - 4*a*c)*a^3*b^4 - 6*sqrt(a^2 - a*b + b*c - c^2
+ sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^6*c + 4*(b^2 - 4*a*c)*
a^6*c + sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^
2 - 4*a*c)*a^5*b*c - 6*(b^2 - 4*a*c)*a^5*b*c + 23*sqrt(a^2 - a*b + b*c - c^
2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^2*c - 2*(b^2 - 4
*a*c)*a^4*b^2*c + 13*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*sqrt(b^2 - 4*a*c)*a^3*b^3*c + 2*(b^2 - 4*a*c)*a^3*b^3*c + 5*sqrt(a^2
- a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^
4*c + 2*(b^2 - 4*a*c)*a^2*b^4*c - 22*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
- 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*c^2 + 4*(b^2 - 4*a*c)*a^5*c^2 -
 27*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
4*a*c)*a^4*b*c^2 + 2*(b^2 - 4*a*c)*a^4*b*c^2 - 37*sqrt(a^2 - a*b + b*c - c^
2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c^2 - 2*(b^2 -
 4*a*c)*a^3*b^2*c^2 - 11*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
```

```
- b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3*c^2 - 6*(b^2 - 4*a*c)*a^2*b^3*c^2 + 38*
sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*
c)*a^4*c^3 - 4*(b^2 - 4*a*c)*a^4*c^3 + 31*sqrt(a^2 - a*b + b*c - c^2 + sqrt
(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^3 + 6*(b^2 - 4*a*c)*a^
3*b*c^3 + 11*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sq
rt(b^2 - 4*a*c)*a^2*b^2*c^3 + 6*(b^2 - 4*a*c)*a^2*b^2*c^3 - 10*sqrt(a^2 - a
*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^4 -
 4*(b^2 - 4*a*c)*a^3*c^4 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)
*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^4 - 2*(b^2 - 4*a*c)*a^2*b*c^4)*abs(
a - b + c))*(pi*floor(1/2*x/pi + 1/2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt(
(2*a^2 - 2*a*c + sqrt(-4*(a^2 + a*b + a*c)*(a^2 - a*b + a*c) + 4*(a^2 - a*c
)^2))/(a^2 - a*b + a*c))))/((3*a^8*b^2 - 8*a^7*b^3 - a^6*b^4 + 16*a^5*b^5 -
 7*a^4*b^6 - 8*a^3*b^7 + 5*a^2*b^8 - 12*a^9*c + 32*a^8*b*c + 30*a^7*b^2*c -
 112*a^6*b^3*c + 8*a^5*b^4*c + 96*a^4*b^5*c - 26*a^3*b^6*c - 16*a^2*b^7*c -
 104*a^8*c^2 + 192*a^7*b*c^2 + 149*a^6*b^2*c^2 - 336*a^5*b^3*c^2 - 30*a^4*b
^4*c^2 + 112*a^3*b^5*c^2 + 17*a^2*b^6*c^2 - 276*a^7*c^3 + 320*a^6*b*c^3 + 2
92*a^5*b^2*c^3 - 224*a^4*b^3*c^3 - 120*a^3*b^4*c^3 - 304*a^6*c^4 + 128*a^5*
b*c^4 + 237*a^4*b^2*c^4 + 24*a^3*b^3*c^4 - 17*a^2*b^4*c^4 - 116*a^5*c^5 - 9
6*a^4*b*c^5 + 62*a^3*b^2*c^5 + 16*a^2*b^3*c^5 + 24*a^4*c^6 - 64*a^3*b*c^6 -
 5*a^2*b^2*c^6 + 20*a^3*c^7)*abs(a)) - ((2*a^2*b^5 - 4*a*b^6 + 2*b^7 - 16*a
^3*b^3*c + 30*a^2*b^4*c - 8*a*b^5*c - 6*b^6*c + 32*a^4*b*c^2 - 48*a^3*b^2*c
^2 - 32*a^2*b^3*c^2 + 44*a*b^4*c^2 + 6*b^5*c^2 - 32*a^4*c^3 + 128*a^3*b*c^3
 - 64*a^2*b^2*c^3 - 48*a*b^3*c^3 - 2*b^4*c^3 - 64*a^3*c^4 + 96*a^2*b*c^4 +
16*a*b^2*c^4 - 32*a^2*c^5 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3 - 2*(b^2 - 4*a*c)*a^2*b^3 - 2*sqrt
(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a
*b^4 + 4*(b^2 - 4*a*c)*a*b^4 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5 - 2*(b^2 - 4*a*c)*b^5 - 12*sqrt(a^2
 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b
*c + 8*(b^2 - 4*a*c)*a^3*b*c + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c - 14*(b^2 - 4*a*c)*a^2*b^2*c
+ 36*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
 4*a*c)*a*b^3*c + 11*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*sqrt(b^2 - 4*a*c)*b^4*c + 6*(b^2 - 4*a*c)*b^4*c + 12*sqrt(a^2 - a*b +
 b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^2 - 8*(
b^2 - 4*a*c)*a^3*c^2 - 64*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*sqrt(b^2 - 4*a*c)*a^2*b*c^2 + 32*(b^2 - 4*a*c)*a^2*b*c^2 - 58*sq
rt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)
*a*b^2*c^2 - 20*(b^2 - 4*a*c)*a*b^2*c^2 - 11*sqrt(a^2 - a*b + b*c - c^2 - s
qrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^3*c^2 - 6*(b^2 - 4*a*c)*b
^3*c^2 + 56*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*a^2*c^3 - 16*(b^2 - 4*a*c)*a^2*c^3 + 44*sqrt(a^2 - a*b + b*c
 - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*c^3 + 24*(b^2
 - 4*a*c)*a*b*c^3 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
 + c))*sqrt(b^2 - 4*a*c)*b^2*c^3 + 2*(b^2 - 4*a*c)*b^2*c^3 - 20*sqrt(a^2 -
```

```
a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*c^4 -
8*(b^2 - 4*a*c)*a*c^4)*a^2*abs(a - b + c) - (4*a^3*b^5 - 8*a^2*b^6 + 4*a*b^
7 - 32*a^4*b^3*c + 68*a^3*b^4*c - 32*a^2*b^5*c - 4*a*b^6*c + 64*a^5*b*c^2 -
 160*a^4*b^2*c^2 + 64*a^3*b^3*c^2 + 40*a^2*b^4*c^2 - 4*a*b^5*c^2 + 64*a^5*c
^3 - 128*a^3*b^2*c^3 + 32*a^2*b^3*c^3 + 4*a*b^4*c^3 + 128*a^4*c^4 - 64*a^3*
b*c^4 - 32*a^2*b^2*c^4 + 64*a^3*c^5 - 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b
^2 - 4*a*c)*(a - b + c))*a^4*b^3 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
- 4*a*c)*(a - b + c))*a^3*b^4 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4
*a*c)*(a - b + c))*a^2*b^5 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*
c)*(a - b + c))*a*b^6 + 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(
a - b + c))*a^5*b*c - 23*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*a^4*b^2*c - 24*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*a^3*b^3*c + 33*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*a^2*b^4*c + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
 b + c))*a*b^5*c + 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
 + c))*a^5*c^2 + 48*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
 c))*a^4*b*c^2 - 69*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
 c))*a^3*b^2*c^2 - 23*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
 + c))*a^2*b^3*c^2 + 68*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
 b + c))*a^4*c^3 - 4*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^3*b*c^3 - 9*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
 c))*a^2*b^2*c^3 - 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*a*b^3*c^3 + 36*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^3*c^4 + 24*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
c))*a^2*b*c^4 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c
))*a*b^2*c^4 - 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c
))*a^2*c^5 - 4*(b^2 - 4*a*c)*a^3*b^3 + 8*(b^2 - 4*a*c)*a^2*b^4 - 4*(b^2 - 4
*a*c)*a*b^5 + 16*(b^2 - 4*a*c)*a^4*b*c - 36*(b^2 - 4*a*c)*a^3*b^2*c + 16*(b
^2 - 4*a*c)*a^2*b^3*c + 4*(b^2 - 4*a*c)*a*b^4*c + 16*(b^2 - 4*a*c)*a^4*c^2
- 24*(b^2 - 4*a*c)*a^2*b^2*c^2 + 4*(b^2 - 4*a*c)*a*b^3*c^2 + 32*(b^2 - 4*a*
c)*a^3*c^3 - 16*(b^2 - 4*a*c)*a^2*b*c^3 - 4*(b^2 - 4*a*c)*a*b^2*c^3 + 16*(b
^2 - 4*a*c)*a^2*c^4)*abs(a - b + c)*abs(a) - (2*a^5*b^4 - 4*a^4*b^5 + 2*a^3
*b^6 - 12*a^6*b^2*c + 22*a^5*b^3*c - 6*a^4*b^4*c - 2*a^3*b^5*c - 2*a^2*b^6*
c + 16*a^7*c^2 - 24*a^6*b*c^2 - 12*a^5*b^2*c^2 + 6*a^4*b^3*c^2 + 10*a^3*b^4
*c^2 + 6*a^2*b^5*c^2 + 16*a^6*c^3 + 8*a^5*b*c^3 - 4*a^4*b^2*c^3 - 30*a^3*b^
3*c^3 - 6*a^2*b^4*c^3 - 16*a^5*c^4 + 24*a^4*b*c^4 + 28*a^3*b^2*c^4 + 2*a^2*
b^3*c^4 - 16*a^4*c^5 - 8*a^3*b*c^5 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^
2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*b^2 - 2*(b^2 - 4*a*c)*a^5*b^2
 - 2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
 4*a*c)*a^4*b^3 + 4*(b^2 - 4*a*c)*a^4*b^3 - 5*sqrt(a^2 - a*b + b*c - c^2 -
sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^4 - 2*(b^2 - 4*a*c)*
a^3*b^4 - 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*a^6*c + 4*(b^2 - 4*a*c)*a^6*c + sqrt(a^2 - a*b + b*c - c^2 -
 sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*b*c - 6*(b^2 - 4*a*c)
*a^5*b*c + 23*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*s
```

```
qrt(b^2 - 4*a*c)*a^4*b^2*c - 2*(b^2 - 4*a*c)*a^4*b^2*c + 13*sqrt(a^2 - a*b
+ b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^3*c +
2*(b^2 - 4*a*c)*a^3*b^3*c + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^4*c + 2*(b^2 - 4*a*c)*a^2*b^4*c - 22
*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a
*c)*a^5*c^2 + 4*(b^2 - 4*a*c)*a^5*c^2 - 27*sqrt(a^2 - a*b + b*c - c^2 - sqr
t(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b*c^2 + 2*(b^2 - 4*a*c)*a
^4*b*c^2 - 37*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*s
qrt(b^2 - 4*a*c)*a^3*b^2*c^2 - 2*(b^2 - 4*a*c)*a^3*b^2*c^2 - 11*sqrt(a^2 -
a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3*
c^2 - 6*(b^2 - 4*a*c)*a^2*b^3*c^2 + 38*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^
2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*c^3 - 4*(b^2 - 4*a*c)*a^4*c^3
 + 31*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*a^3*b*c^3 + 6*(b^2 - 4*a*c)*a^3*b*c^3 + 11*sqrt(a^2 - a*b + b*c -
c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c^3 + 6*(b^2
 - 4*a*c)*a^2*b^2*c^3 - 10*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(
a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^4 - 4*(b^2 - 4*a*c)*a^3*c^4 - 5*sqrt(a^
2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*
b*c^4 - 2*(b^2 - 4*a*c)*a^2*b*c^4)*abs(a - b + c))*(pi*floor(1/2*x/pi + 1/2
) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a^2 - 2*a*c - sqrt(-4*(a^2 + a*b
+ a*c)*(a^2 - a*b + a*c) + 4*(a^2 - a*c)^2))/(a^2 - a*b + a*c)))))/((3*a^8*b
^2 - 8*a^7*b^3 - a^6*b^4 + 16*a^5*b^5 - 7*a^4*b^6 - 8*a^3*b^7 + 5*a^2*b^8 -
 12*a^9*c + 32*a^8*b*c + 30*a^7*b^2*c - 112*a^6*b^3*c + 8*a^5*b^4*c + 96*a^
4*b^5*c - 26*a^3*b^6*c - 16*a^2*b^7*c - 104*a^8*c^2 + 192*a^7*b*c^2 + 149*a
^6*b^2*c^2 - 336*a^5*b^3*c^2 - 30*a^4*b^4*c^2 + 112*a^3*b^5*c^2 + 17*a^2*b^
6*c^2 - 276*a^7*c^3 + 320*a^6*b*c^3 + 292*a^5*b^2*c^3 - 224*a^4*b^3*c^3 - 1
20*a^3*b^4*c^3 - 304*a^6*c^4 + 128*a^5*b*c^4 + 237*a^4*b^2*c^4 + 24*a^3*b^3
*c^4 - 17*a^2*b^4*c^4 - 116*a^5*c^5 - 96*a^4*b*c^5 + 62*a^3*b^2*c^5 + 16*a^
2*b^3*c^5 + 24*a^4*c^6 - 64*a^3*b*c^6 - 5*a^2*b^2*c^6 + 20*a^3*c^7)*abs(a))
 + log(abs(tan(1/2*x) + 1))/a - log(abs(tan(1/2*x) - 1))/a
```

Mupad [B] (verification not implemented)

Time = 15.39 (sec) , antiderivative size = 20126, normalized size of antiderivative = 82.15

$$\int \frac{\sec(x)}{a + b\cos(x) + c\cos^2(x)}\, dx = \text{Too large to display}$$

```
[In] int(1/(cos(x)*(a + b*cos(x) + c*cos(x)^2)),x)
```

```
[Out] (2*atanh((16384*b^4*tan(x/2))/(655360*a*c^3 - 393216*b*c^3 + 131072*b^3*c +
 16384*b^4 + 540672*c^4 - (16384*b^5)/a + 262144*a^2*c^2 + (147456*c^5)/a -
 360448*b^2*c^2 - (147456*b*c^4)/a + (49152*b^4*c)/a - (32768*b^5*c)/a^2 -
(262144*b^2*c^3)/a + (229376*b^3*c^2)/a - (32768*b^2*c^4)/a^2 + (32768*b^3*
c^3)/a^2 + (32768*b^4*c^2)/a^2 - 262144*a*b*c^2 - 131072*a*b^2*c) + (540672
```

*c^4*tan(x/2))/(655360*a*c^3 - 393216*b*c^3 + 131072*b^3*c + 16384*b^4 + 54
0672*c^4 - (16384*b^5)/a + 262144*a^2*c^2 + (147456*c^5)/a - 360448*b^2*c^2
 - (147456*b*c^4)/a + (49152*b^4*c)/a - (32768*b^5*c)/a^2 - (262144*b^2*c^3
)/a + (229376*b^3*c^2)/a - (32768*b^2*c^4)/a^2 + (32768*b^3*c^3)/a^2 + (327
68*b^4*c^2)/a^2 - 262144*a*b*c^2 - 131072*a*b^2*c) - (16384*b^5*tan(x/2))/(
16384*a*b^4 + 540672*a*c^4 - 147456*b*c^4 + 49152*b^4*c - 16384*b^5 + 14745
6*c^5 + 655360*a^2*c^3 + 262144*a^3*c^2 - 262144*b^2*c^3 + 229376*b^3*c^2 -
 360448*a*b^2*c^2 - 262144*a^2*b*c^2 - 131072*a^2*b^2*c - (32768*b^5*c)/a -
 (32768*b^2*c^4)/a + (32768*b^3*c^3)/a + (32768*b^4*c^2)/a - 393216*a*b*c^3
 + 131072*a*b^3*c) + (147456*c^5*tan(x/2))/(16384*a*b^4 + 540672*a*c^4 - 14
7456*b*c^4 + 49152*b^4*c - 16384*b^5 + 147456*c^5 + 655360*a^2*c^3 + 262144
*a^3*c^2 - 262144*b^2*c^3 + 229376*b^3*c^2 - 360448*a*b^2*c^2 - 262144*a^2*
b*c^2 - 131072*a^2*b^2*c - (32768*b^5*c)/a - (32768*b^2*c^4)/a + (32768*b^3
*c^3)/a + (32768*b^4*c^2)/a - 393216*a*b*c^3 + 131072*a*b^3*c) + (262144*a^
2*c^2*tan(x/2))/(655360*a*c^3 - 393216*b*c^3 + 131072*b^3*c + 16384*b^4 + 5
40672*c^4 - (16384*b^5)/a + 262144*a^2*c^2 + (147456*c^5)/a - 360448*b^2*c^
2 - (147456*b*c^4)/a + (49152*b^4*c)/a - (32768*b^5*c)/a^2 - (262144*b^2*c^
3)/a + (229376*b^3*c^2)/a - (32768*b^2*c^4)/a^2 + (32768*b^3*c^3)/a^2 + (32
768*b^4*c^2)/a^2 - 262144*a*b*c^2 - 131072*a*b^2*c) - (360448*b^2*c^2*tan(x
/2))/(655360*a*c^3 - 393216*b*c^3 + 131072*b^3*c + 16384*b^4 + 540672*c^4 -
 (16384*b^5)/a + 262144*a^2*c^2 + (147456*c^5)/a - 360448*b^2*c^2 - (147456
*b*c^4)/a + (49152*b^4*c)/a - (32768*b^5*c)/a^2 - (262144*b^2*c^3)/a + (229
376*b^3*c^2)/a - (32768*b^2*c^4)/a^2 + (32768*b^3*c^3)/a^2 + (32768*b^4*c^2
)/a^2 - 262144*a*b*c^2 - 131072*a*b^2*c) - (147456*b*c^4*tan(x/2))/(16384*a
*b^4 + 540672*a*c^4 - 147456*b*c^4 + 49152*b^4*c - 16384*b^5 + 147456*c^5 +
 655360*a^2*c^3 + 262144*a^3*c^2 - 262144*b^2*c^3 + 229376*b^3*c^2 - 360448
*a*b^2*c^2 - 262144*a^2*b*c^2 - 131072*a^2*b^2*c - (32768*b^5*c)/a - (32768
*b^2*c^4)/a + (32768*b^3*c^3)/a + (32768*b^4*c^2)/a - 393216*a*b*c^3 + 1310
72*a*b^3*c) + (49152*b^4*c*tan(x/2))/(16384*a*b^4 + 540672*a*c^4 - 147456*b
*c^4 + 49152*b^4*c - 16384*b^5 + 147456*c^5 + 655360*a^2*c^3 + 262144*a^3*c
^2 - 262144*b^2*c^3 + 229376*b^3*c^2 - 360448*a*b^2*c^2 - 262144*a^2*b*c^2
 - 131072*a^2*b^2*c - (32768*b^5*c)/a - (32768*b^2*c^4)/a + (32768*b^3*c^3)/
a + (32768*b^4*c^2)/a - 393216*a*b*c^3 + 131072*a*b^3*c) - (32768*b^5*c*tan
(x/2))/(147456*a*c^5 - 16384*a*b^5 - 32768*b^5*c + 16384*a^2*b^4 + 540672*a
^2*c^4 + 655360*a^3*c^3 + 262144*a^4*c^2 - 32768*b^2*c^4 + 32768*b^3*c^3 +
32768*b^4*c^2 - 262144*a*b^2*c^3 + 229376*a*b^3*c^2 - 393216*a^2*b*c^3 + 13
1072*a^2*b^3*c - 262144*a^3*b*c^2 - 131072*a^3*b^2*c - 360448*a^2*b^2*c^2 -
 147456*a*b*c^4 + 49152*a*b^4*c) - (262144*b^2*c^3*tan(x/2))/(16384*a*b^4 +
 540672*a*c^4 - 147456*b*c^4 + 49152*b^4*c - 16384*b^5 + 147456*c^5 + 65536
0*a^2*c^3 + 262144*a^3*c^2 - 262144*b^2*c^3 + 229376*b^3*c^2 - 360448*a*b^2
*c^2 - 262144*a^2*b*c^2 - 131072*a^2*b^2*c - (32768*b^5*c)/a - (32768*b^2*c
^4)/a + (32768*b^3*c^3)/a + (32768*b^4*c^2)/a - 393216*a*b*c^3 + 131072*a*b
^3*c) + (229376*b^3*c^2*tan(x/2))/(16384*a*b^4 + 540672*a*c^4 - 147456*b*c^
4 + 49152*b^4*c - 16384*b^5 + 147456*c^5 + 655360*a^2*c^3 + 262144*a^3*c^2
 - 262144*b^2*c^3 + 229376*b^3*c^2 - 360448*a*b^2*c^2 - 262144*a^2*b*c^2 - 1

31072*a^2*b^2*c - (32768*b^5*c)/a - (32768*b^2*c^4)/a + (32768*b^3*c^3)/a +
 (32768*b^4*c^2)/a - 393216*a*b*c^3 + 131072*a*b^3*c) + (655360*a*c^3*tan(x
/2))/(655360*a*c^3 - 393216*b*c^3 + 131072*b^3*c + 16384*b^4 + 540672*c^4 -
 (16384*b^5)/a + 262144*a^2*c^2 + (147456*c^5)/a - 360448*b^2*c^2 - (147456
*b*c^4)/a + (49152*b^4*c)/a - (32768*b^5*c)/a^2 - (262144*b^2*c^3)/a + (229
376*b^3*c^2)/a - (32768*b^2*c^4)/a^2 + (32768*b^3*c^3)/a^2 + (32768*b^4*c^2
)/a^2 - 262144*a*b*c^2 - 131072*a*b^2*c) - (393216*b*c^3*tan(x/2))/(655360*
a*c^3 - 393216*b*c^3 + 131072*b^3*c + 16384*b^4 + 540672*c^4 - (16384*b^5)/
a + 262144*a^2*c^2 + (147456*c^5)/a - 360448*b^2*c^2 - (147456*b*c^4)/a + (
49152*b^4*c)/a - (32768*b^5*c)/a^2 - (262144*b^2*c^3)/a + (229376*b^3*c^2)/
a - (32768*b^2*c^4)/a^2 + (32768*b^3*c^3)/a^2 + (32768*b^4*c^2)/a^2 - 26214
4*a*b*c^2 - 131072*a*b^2*c) + (131072*b^3*c*tan(x/2))/(655360*a*c^3 - 39321
6*b*c^3 + 131072*b^3*c + 16384*b^4 + 540672*c^4 - (16384*b^5)/a + 262144*a^
2*c^2 + (147456*c^5)/a - 360448*b^2*c^2 - (147456*b*c^4)/a + (49152*b^4*c)/
a - (32768*b^5*c)/a^2 - (262144*b^2*c^3)/a + (229376*b^3*c^2)/a - (32768*b^
2*c^4)/a^2 + (32768*b^3*c^3)/a^2 + (32768*b^4*c^2)/a^2 - 262144*a*b*c^2 - 1
31072*a*b^2*c) - (32768*b^2*c^4*tan(x/2))/(147456*a*c^5 - 16384*a*b^5 - 327
68*b^5*c + 16384*a^2*b^4 + 540672*a^2*c^4 + 655360*a^3*c^3 + 262144*a^4*c^2
 - 32768*b^2*c^4 + 32768*b^3*c^3 + 32768*b^4*c^2 - 262144*a*b^2*c^3 + 22937
6*a*b^3*c^2 - 393216*a^2*b*c^3 + 131072*a^2*b^3*c - 262144*a^3*b*c^2 - 1310
72*a^3*b^2*c - 360448*a^2*b^2*c^2 - 147456*a*b*c^4 + 49152*a*b^4*c) + (3276
8*b^3*c^3*tan(x/2))/(147456*a*c^5 - 16384*a*b^5 - 32768*b^5*c + 16384*a^2*b
^4 + 540672*a^2*c^4 + 655360*a^3*c^3 + 262144*a^4*c^2 - 32768*b^2*c^4 + 327
68*b^3*c^3 + 32768*b^4*c^2 - 262144*a*b^2*c^3 + 229376*a*b^3*c^2 - 393216*a
^2*b*c^3 + 131072*a^2*b^3*c - 262144*a^3*b*c^2 - 131072*a^3*b^2*c - 360448*
a^2*b^2*c^2 - 147456*a*b*c^4 + 49152*a*b^4*c) + (32768*b^4*c^2*tan(x/2))/(1
47456*a*c^5 - 16384*a*b^5 - 32768*b^5*c + 16384*a^2*b^4 + 540672*a^2*c^4 +
655360*a^3*c^3 + 262144*a^4*c^2 - 32768*b^2*c^4 + 32768*b^3*c^3 + 32768*b^4
*c^2 - 262144*a*b^2*c^3 + 229376*a*b^3*c^2 - 393216*a^2*b*c^3 + 131072*a^2*
b^3*c - 262144*a^3*b*c^2 - 131072*a^3*b^2*c - 360448*a^2*b^2*c^2 - 147456*a
*b*c^4 + 49152*a*b^4*c) - (262144*a*b*c^2*tan(x/2))/(655360*a*c^3 - 393216*
b*c^3 + 131072*b^3*c + 16384*b^4 + 540672*c^4 - (16384*b^5)/a + 262144*a^2*
c^2 + (147456*c^5)/a - 360448*b^2*c^2 - (147456*b*c^4)/a + (49152*b^4*c)/a
 - (32768*b^5*c)/a^2 - (262144*b^2*c^3)/a + (229376*b^3*c^2)/a - (32768*b^2*
c^4)/a^2 + (32768*b^3*c^3)/a^2 + (32768*b^4*c^2)/a^2 - 262144*a*b*c^2 - 131
072*a*b^2*c) - (131072*a*b^2*c*tan(x/2))/(655360*a*c^3 - 393216*b*c^3 + 131
072*b^3*c + 16384*b^4 + 540672*c^4 - (16384*b^5)/a + 262144*a^2*c^2 + (1474
56*c^5)/a - 360448*b^2*c^2 - (147456*b*c^4)/a + (49152*b^4*c)/a - (32768*b^
5*c)/a^2 - (262144*b^2*c^3)/a + (229376*b^3*c^2)/a - (32768*b^2*c^4)/a^2 +
(32768*b^3*c^3)/a^2 + (32768*b^4*c^2)/a^2 - 262144*a*b*c^2 - 131072*a*b^2*c
)))/a - atan(-((((8*a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2
) + b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2
 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16
*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c
^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(24576*a*c^5 - 49152*a*b^5 - 3

2768*b^5*c + 24576*b^6 + 32768*a^2*b^4 - 8192*a^3*b^3 + 180224*a^2*c^4 + 25
3952*a^3*c^3 + 98304*a^4*c^2 - 8192*b^2*c^4 + 32768*b^3*c^3 - 16384*b^4*c^2
 + (tan(x/2)*(49152*a*b^6 - 65536*a^6*c + 16384*b^6*c - 16384*b^7 - 65536*a
^2*b^5 + 65536*a^3*b^4 - 49152*a^4*b^3 + 16384*a^5*b^2 - 147456*a^2*c^5 + 2
12992*a^3*c^4 + 671744*a^4*c^3 + 245760*a^5*c^2 - 16384*b^4*c^3 + 16384*b^5
*c^2 + 98304*a*b^2*c^4 - 65536*a*b^3*c^3 - 180224*a*b^4*c^2 + 49152*a^2*b*c
^4 - 393216*a^2*b^4*c - 1081344*a^3*b*c^3 + 475136*a^3*b^3*c - 802816*a^4*b
*c^2 - 327680*a^4*b^2*c + 344064*a^2*b^2*c^3 + 180224*a^2*b^3*c^2 + 557056*
a^3*b^2*c^2 + 98304*a*b^5*c + 196608*a^5*b*c) + ((8*a^2*c^4 - b^6 + 8*a^3*c
^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c
 - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(
1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3
*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2
)*(57344*a^3*b^5 - 24576*a^2*b^6 - 40960*a^4*b^4 + 8192*a^5*b^3 - 98304*a^3
*c^5 - 425984*a^4*c^4 - 557056*a^5*c^3 - 229376*a^6*c^2 + 49152*a^2*b^5*c +
 196608*a^3*b*c^4 + 90112*a^3*b^4*c + 622592*a^4*b*c^3 - 327680*a^4*b^3*c +
 393216*a^5*b*c^2 + 221184*a^5*b^2*c + tan(x/2)*((8*a^2*c^4 - b^6 + 8*a^3*c
^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c
 - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(
1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3
*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2
)*(65536*a^8*c + 16384*a^2*b^7 - 49152*a^3*b^6 + 65536*a^4*b^5 - 65536*a^5*
b^4 + 49152*a^6*b^3 - 16384*a^7*b^2 + 196608*a^4*c^5 + 131072*a^5*c^4 - 262
144*a^6*c^3 - 131072*a^7*c^2 - 16384*a^2*b^6*c - 114688*a^3*b^5*c - 65536*a
^4*b*c^4 + 376832*a^4*b^4*c + 720896*a^5*b*c^3 - 409600*a^5*b^3*c + 589824*
a^6*b*c^2 + 294912*a^6*b^2*c + 16384*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 1146
88*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 196608*a^3*b^4*c^2 - 557056*a^4*b^2*c^
3 + 16384*a^4*b^3*c^2 - 655360*a^5*b^2*c^2 - 196608*a^7*b*c) + 24576*a^2*b^
2*c^4 - 49152*a^2*b^3*c^3 + 106496*a^3*b^2*c^3 - 352256*a^3*b^3*c^2 + 17203
2*a^4*b^2*c^2 - 32768*a^6*b*c))*((8*a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4*a*
c - b^2)^3)^(1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2)
 - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^
4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b
^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2) + 24576*a*b^2*
c^3 + 237568*a*b^3*c^2 - 458752*a^2*b*c^3 + 262144*a^2*b^3*c - 270336*a^3*b
*c^2 - 155648*a^3*b^2*c + 16384*a^2*b^2*c^2 - 122880*a*b*c^4 - 114688*a*b^4
*c + 32768*a^4*b*c) + tan(x/2)*(8192*a*b^4 - 73728*a*c^4 - 57344*b*c^4 + 40
960*b^4*c - 8192*b^5 + 24576*c^5 - 81920*a^2*c^3 + 16384*a^3*c^2 + 81920*b^
2*c^3 - 81920*b^3*c^2 - 81920*a*b^2*c^2 + 81920*a^2*b*c^2 - 32768*a^2*b^2*c
 + 163840*a*b*c^3))*((8*a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^
(1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2
*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6
+ 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b
^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*1i - (((8*a^2*c^4 - b^6 +
8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-

(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^
2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 +
 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)
))^(1/2)*(24576*a*c^5 - 49152*a*b^5 - 32768*b^5*c + 24576*b^6 + 32768*a^2*b
^4 - 8192*a^3*b^3 + 180224*a^2*c^4 + 253952*a^3*c^3 + 98304*a^4*c^2 - 8192*
b^2*c^4 + 32768*b^3*c^3 - 16384*b^4*c^2 - (tan(x/2)*(49152*a*b^6 - 65536*a^
6*c + 16384*b^6*c - 16384*b^7 - 65536*a^2*b^5 + 65536*a^3*b^4 - 49152*a^4*b
^3 + 16384*a^5*b^2 - 147456*a^2*c^5 + 212992*a^3*c^4 + 671744*a^4*c^3 + 245
760*a^5*c^2 - 16384*b^4*c^3 + 16384*b^5*c^2 + 98304*a*b^2*c^4 - 65536*a*b^3
*c^3 - 180224*a*b^4*c^2 + 49152*a^2*b*c^4 - 393216*a^2*b^4*c - 1081344*a^3*
b*c^3 + 475136*a^3*b^3*c - 802816*a^4*b*c^2 - 327680*a^4*b^2*c + 344064*a^2
*b^2*c^3 + 180224*a^2*b^3*c^2 + 557056*a^3*b^2*c^2 + 98304*a*b^5*c + 196608
*a^5*b*c) + ((8*a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) +
b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8
*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4
*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 -
 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(24576*a^2*b^6 - 57344*a^3*b^5 + 4
0960*a^4*b^4 - 8192*a^5*b^3 + 98304*a^3*c^5 + 425984*a^4*c^4 + 557056*a^5*c
^3 + 229376*a^6*c^2 - 49152*a^2*b^5*c - 196608*a^3*b*c^4 - 90112*a^3*b^4*c
- 622592*a^4*b*c^3 + 327680*a^4*b^3*c - 393216*a^5*b*c^2 - 221184*a^5*b^2*c
 + tan(x/2)*((8*a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) +
b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8
*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4
*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 -
 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(65536*a^8*c + 16384*a^2*b^7 - 491
52*a^3*b^6 + 65536*a^4*b^5 - 65536*a^5*b^4 + 49152*a^6*b^3 - 16384*a^7*b^2
+ 196608*a^4*c^5 + 131072*a^5*c^4 - 262144*a^6*c^3 - 131072*a^7*c^2 - 16384
*a^2*b^6*c - 114688*a^3*b^5*c - 65536*a^4*b*c^4 + 376832*a^4*b^4*c + 720896
*a^5*b*c^3 - 409600*a^5*b^3*c + 589824*a^6*b*c^2 + 294912*a^6*b^2*c + 16384
*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 114688*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 +
 196608*a^3*b^4*c^2 - 557056*a^4*b^2*c^3 + 16384*a^4*b^3*c^2 - 655360*a^5*b
^2*c^2 - 196608*a^7*b*c) - 24576*a^2*b^2*c^4 + 49152*a^2*b^3*c^3 - 106496*a
^3*b^2*c^3 + 352256*a^3*b^3*c^2 - 172032*a^4*b^2*c^2 + 32768*a^6*b*c))*((8*
a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) + b^4*c^2 - 6*a*b^
2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b
c(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^
3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 -
 32*a^4*b^2*c^2)))^(1/2) + 24576*a*b^2*c^3 + 237568*a*b^3*c^2 - 458752*a^2*
b*c^3 + 262144*a^2*b^3*c - 270336*a^3*b*c^2 - 155648*a^3*b^2*c + 16384*a^2*
b^2*c^2 - 122880*a*b*c^4 - 114688*a*b^4*c + 32768*a^4*b*c) - tan(x/2)*(8192
*a*b^4 - 73728*a*c^4 - 57344*b*c^4 + 40960*b^4*c - 8192*b^5 + 24576*c^5 - 8
1920*a^2*c^3 + 16384*a^3*c^2 + 81920*b^2*c^3 - 81920*b^3*c^2 - 81920*a*b^2*
c^2 + 81920*a^2*b*c^2 - 32768*a^2*b^2*c + 163840*a*b*c^3))*((8*a^2*c^4 - b^
6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^
2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c

- b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c
^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*
c^2)))^(1/2)*1i)/((((8*a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(
1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*
c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 +
16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^
4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(24576*a*c^5 - 49152*a*b^5
- 32768*b^5*c + 24576*b^6 + 32768*a^2*b^4 - 8192*a^3*b^3 + 180224*a^2*c^4 +
253952*a^3*c^3 + 98304*a^4*c^2 - 8192*b^2*c^4 + 32768*b^3*c^3 - 16384*b^4*
c^2 + (tan(x/2)*(49152*a*b^6 - 65536*a^6*c + 16384*b^6*c - 16384*b^7 - 6553
6*a^2*b^5 + 65536*a^3*b^4 - 49152*a^4*b^3 + 16384*a^5*b^2 - 147456*a^2*c^5
+ 212992*a^3*c^4 + 671744*a^4*c^3 + 245760*a^5*c^2 - 16384*b^4*c^3 + 16384*
b^5*c^2 + 98304*a*b^2*c^4 - 65536*a*b^3*c^3 - 180224*a*b^4*c^2 + 49152*a^2*
b*c^4 - 393216*a^2*b^4*c - 1081344*a^3*b*c^3 + 475136*a^3*b^3*c - 802816*a^
4*b*c^2 - 327680*a^4*b^2*c + 344064*a^2*b^2*c^3 + 180224*a^2*b^3*c^2 + 5570
56*a^3*b^2*c^2 + 98304*a*b^5*c + 196608*a^5*b*c) + ((8*a^2*c^4 - b^6 + 8*a^
3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a
*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3
)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*
a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(
1/2)*(57344*a^3*b^5 - 24576*a^2*b^6 - 40960*a^4*b^4 + 8192*a^5*b^3 - 98304*
a^3*c^5 - 425984*a^4*c^4 - 557056*a^5*c^3 - 229376*a^6*c^2 + 49152*a^2*b^5*
c + 196608*a^3*b*c^4 + 90112*a^3*b^4*c + 622592*a^4*b*c^3 - 327680*a^4*b^3*
c + 393216*a^5*b*c^2 + 221184*a^5*b^2*c + tan(x/2)*((8*a^2*c^4 - b^6 + 8*a^
3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a
*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3
)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*
a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(
1/2)*(65536*a^8*c + 16384*a^2*b^7 - 49152*a^3*b^6 + 65536*a^4*b^5 - 65536*a
^5*b^4 + 49152*a^6*b^3 - 16384*a^7*b^2 + 196608*a^4*c^5 + 131072*a^5*c^4 -
262144*a^6*c^3 - 131072*a^7*c^2 - 16384*a^2*b^6*c - 114688*a^3*b^5*c - 6553
6*a^4*b*c^4 + 376832*a^4*b^4*c + 720896*a^5*b*c^3 - 409600*a^5*b^3*c + 5898
24*a^6*b*c^2 + 294912*a^6*b^2*c + 16384*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 1
14688*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 196608*a^3*b^4*c^2 - 557056*a^4*b^2
*c^3 + 16384*a^4*b^3*c^2 - 655360*a^5*b^2*c^2 - 196608*a^7*b*c) + 24576*a^2
*b^2*c^4 - 49152*a^2*b^3*c^3 + 106496*a^3*b^2*c^3 - 352256*a^3*b^3*c^2 + 17
2032*a^4*b^2*c^2 - 32768*a^6*b*c))*((8*a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4
*a*c - b^2)^3)^(1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/
2) - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4
*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^
5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2) + 24576*a*b
^2*c^3 + 237568*a*b^3*c^2 - 458752*a^2*b*c^3 + 262144*a^2*b^3*c - 270336*a^
3*b*c^2 - 155648*a^3*b^2*c + 16384*a^2*b^2*c^2 - 122880*a*b*c^4 - 114688*a*
b^4*c + 32768*a^4*b*c) + tan(x/2)*(8192*a*b^4 - 73728*a*c^4 - 57344*b*c^4 +
40960*b^4*c - 8192*b^5 + 24576*c^5 - 81920*a^2*c^3 + 16384*a^3*c^2 + 81920

*b^2*c^3 - 81920*b^3*c^2 - 81920*a*b^2*c^2 + 81920*a^2*b*c^2 - 32768*a^2*b^
2*c + 163840*a*b*c^3))*((8*a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^
3)^(1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*
b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b
^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^
2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2) + (((8*a^2*c^4 - b^6 +
8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-
(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^
2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 +
10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)
))^(1/2)*(24576*a*c^5 - 49152*a*b^5 - 32768*b^5*c + 24576*b^6 + 32768*a^2*b
^4 - 8192*a^3*b^3 + 180224*a^2*c^4 + 253952*a^3*c^3 + 98304*a^4*c^2 - 8192*
b^2*c^4 + 32768*b^3*c^3 - 16384*b^4*c^2 - (tan(x/2)*(49152*a*b^6 - 65536*a^
6*c + 16384*b^6*c - 16384*b^7 - 65536*a^2*b^5 + 65536*a^3*b^4 - 49152*a^4*b
^3 + 16384*a^5*b^2 - 147456*a^2*c^5 + 212992*a^3*c^4 + 671744*a^4*c^3 + 245
760*a^5*c^2 - 16384*b^4*c^3 + 16384*b^5*c^2 + 98304*a*b^2*c^4 - 65536*a*b^3
*c^3 - 180224*a*b^4*c^2 + 49152*a^2*b*c^4 - 393216*a^2*b^4*c - 1081344*a^3*
b*c^3 + 475136*a^3*b^3*c - 802816*a^4*b*c^2 - 327680*a^4*b^2*c + 344064*a^2
*b^2*c^3 + 180224*a^2*b^3*c^2 + 557056*a^3*b^2*c^2 + 98304*a*b^5*c + 196608
*a^5*b*c) + ((8*a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) +
b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8
*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4
*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 -
8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(24576*a^2*b^6 - 57344*a^3*b^5 + 4
0960*a^4*b^4 - 8192*a^5*b^3 + 98304*a^3*c^5 + 425984*a^4*c^4 + 557056*a^5*c
^3 + 229376*a^6*c^2 - 49152*a^2*b^5*c - 196608*a^3*b*c^4 - 90112*a^3*b^4*c
- 622592*a^4*b*c^3 + 327680*a^4*b^3*c - 393216*a^5*b*c^2 - 221184*a^5*b^2*c
+ tan(x/2)*((8*a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) +
b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8
*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4
*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 -
8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(65536*a^8*c + 16384*a^2*b^7 - 491
52*a^3*b^6 + 65536*a^4*b^5 - 65536*a^5*b^4 + 49152*a^6*b^3 - 16384*a^7*b^2
+ 196608*a^4*c^5 + 131072*a^5*c^4 - 262144*a^6*c^3 - 131072*a^7*c^2 - 16384
*a^2*b^6*c - 114688*a^3*b^5*c - 65536*a^4*b*c^4 + 376832*a^4*b^4*c + 720896
*a^5*b*c^3 - 409600*a^5*b^3*c + 589824*a^6*b*c^2 + 294912*a^6*b^2*c + 16384
*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 114688*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 +
196608*a^3*b^4*c^2 - 557056*a^4*b^2*c^3 + 16384*a^4*b^3*c^2 - 655360*a^5*b
^2*c^2 - 196608*a^7*b*c) - 24576*a^2*b^2*c^4 + 49152*a^2*b^3*c^3 - 106496*a
^3*b^2*c^3 + 352256*a^3*b^3*c^2 - 172032*a^4*b^2*c^2 + 32768*a^6*b*c))*((8*
a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) + b^4*c^2 - 6*a*b^
2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b
c(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^
3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 -
32*a^4*b^2*c^2)))^(1/2) + 24576*a*b^2*c^3 + 237568*a*b^3*c^2 - 458752*a^2*

b*c^3 + 262144*a^2*b^3*c - 270336*a^3*b*c^2 - 155648*a^3*b^2*c + 16384*a^2*
b^2*c^2 - 122880*a*b*c^4 - 114688*a*b^4*c + 32768*a^4*b*c) - tan(x/2)*(8192
*a*b^4 - 73728*a*c^4 - 57344*b*c^4 + 40960*b^4*c - 8192*b^5 + 24576*c^5 - 8
1920*a^2*c^3 + 16384*a^3*c^2 + 81920*b^2*c^3 - 81920*b^3*c^2 - 81920*a*b^2*
c^2 + 81920*a^2*b*c^2 - 32768*a^2*b^2*c + 163840*a*b*c^3))*((8*a^2*c^4 - b^
6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) + b^4*c^2 - 6*a*b^2*c^3 + b*c^
2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8*a*b^4*c + 2*a*b*c*(-(4*a*c
- b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c
^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*
c^2)))^(1/2) - 49152*a*c^3 + 65536*b*c^3 - 49152*c^4 - 16384*b^2*c^2 + 1638
4*a*b*c^2))*((8*a^2*c^4 - b^6 + 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) +
b^4*c^2 - 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 18*a^2*b^2*c^2 + 8
*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4
*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 -
 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*2i + atan(-((tan(x/2)*(8192*a*b^4
- 73728*a*c^4 - 57344*b*c^4 + 40960*b^4*c - 8192*b^5 + 24576*c^5 - 81920*a^
2*c^3 + 16384*a^3*c^2 + 81920*b^2*c^3 - 81920*b^3*c^2 - 81920*a*b^2*c^2 + 8
1920*a^2*b*c^2 - 32768*a^2*b^2*c + 163840*a*b*c^3) + (-(b^6 - 8*a^2*c^4 - 8
*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3 + b*c^2*(-(
4*a*c - b^2)^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2
)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 +
10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2))
)^(1/2)*(24576*a*c^5 - 49152*a*b^5 - 32768*b^5*c + 24576*b^6 + ((-(b^6 - 8*
a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3
+ b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(
4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16
*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^
4*b^2*c^2)))^(1/2)*(57344*a^3*b^5 - 24576*a^2*b^6 - 40960*a^4*b^4 + 8192*a^
5*b^3 - 98304*a^3*c^5 - 425984*a^4*c^4 - 557056*a^5*c^3 - 229376*a^6*c^2 +
49152*a^2*b^5*c + 196608*a^3*b*c^4 + 90112*a^3*b^4*c + 622592*a^4*b*c^3 - 3
27680*a^4*b^3*c + 393216*a^5*b*c^2 + 221184*a^5*b^2*c + 24576*a^2*b^2*c^4 -
 49152*a^2*b^3*c^3 + 106496*a^3*b^2*c^3 - 352256*a^3*b^3*c^2 + 172032*a^4*b
^2*c^2 - 32768*a^6*b*c + tan(x/2)*(-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4
*a*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/
2) + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4
*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^
5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(65536*a^8*
c + 16384*a^2*b^7 - 49152*a^3*b^6 + 65536*a^4*b^5 - 65536*a^5*b^4 + 49152*a
^6*b^3 - 16384*a^7*b^2 + 196608*a^4*c^5 + 131072*a^5*c^4 - 262144*a^6*c^3 -
 131072*a^7*c^2 - 16384*a^2*b^6*c - 114688*a^3*b^5*c - 65536*a^4*b*c^4 + 37
6832*a^4*b^4*c + 720896*a^5*b*c^3 - 409600*a^5*b^3*c + 589824*a^6*b*c^2 + 2
94912*a^6*b^2*c + 16384*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 114688*a^3*b^2*c^
4 + 81920*a^3*b^3*c^3 + 196608*a^3*b^4*c^2 - 557056*a^4*b^2*c^3 + 16384*a^4
*b^3*c^2 - 655360*a^5*b^2*c^2 - 196608*a^7*b*c)) + tan(x/2)*(49152*a*b^6 -
65536*a^6*c + 16384*b^6*c - 16384*b^7 - 65536*a^2*b^5 + 65536*a^3*b^4 - 491

52*a^4*b^3 + 16384*a^5*b^2 - 147456*a^2*c^5 + 212992*a^3*c^4 + 671744*a^4*c
^3 + 245760*a^5*c^2 - 16384*b^4*c^3 + 16384*b^5*c^2 + 98304*a*b^2*c^4 - 655
36*a*b^3*c^3 - 180224*a*b^4*c^2 + 49152*a^2*b*c^4 - 393216*a^2*b^4*c - 1081
344*a^3*b*c^3 + 475136*a^3*b^3*c - 802816*a^4*b*c^2 - 327680*a^4*b^2*c + 34
4064*a^2*b^2*c^3 + 180224*a^2*b^3*c^2 + 557056*a^3*b^2*c^2 + 98304*a*b^5*c
+ 196608*a^5*b*c))*(-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^
(1/2) - b^4*c^2 + 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 18*a^2*b^2
*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6
+ 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b
^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2) + 32768*a^2*b^4 - 8192*a^3
*b^3 + 180224*a^2*c^4 + 253952*a^3*c^3 + 98304*a^4*c^2 - 8192*b^2*c^4 + 327
68*b^3*c^3 - 16384*b^4*c^2 + 24576*a*b^2*c^3 + 237568*a*b^3*c^2 - 458752*a^
2*b*c^3 + 262144*a^2*b^3*c - 270336*a^3*b*c^2 - 155648*a^3*b^2*c + 16384*a^
2*b^2*c^2 - 122880*a*b*c^4 - 114688*a*b^4*c + 32768*a^4*b*c))*(-(b^6 - 8*a^
2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3 +
b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*
a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a
^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*
b^2*c^2)))^(1/2)*1i + (tan(x/2)*(8192*a*b^4 - 73728*a*c^4 - 57344*b*c^4 + 4
0960*b^4*c - 8192*b^5 + 24576*c^5 - 81920*a^2*c^3 + 16384*a^3*c^2 + 81920*b
^2*c^3 - 81920*b^3*c^2 - 81920*a*b^2*c^2 + 81920*a^2*b*c^2 - 32768*a^2*b^2*
c + 163840*a*b*c^3) - (-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^
3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 18*a^2*
b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b
^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^
2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(24576*a*c^5 - 49152*a*
b^5 - 32768*b^5*c + 24576*b^6 - ((-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*
a*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2
) + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*
b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5
*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(24576*a^2*b
^6 - 57344*a^3*b^5 + 40960*a^4*b^4 - 8192*a^5*b^3 + 98304*a^3*c^5 + 425984*
a^4*c^4 + 557056*a^5*c^3 + 229376*a^6*c^2 - 49152*a^2*b^5*c - 196608*a^3*b*
c^4 - 90112*a^3*b^4*c - 622592*a^4*b*c^3 + 327680*a^4*b^3*c - 393216*a^5*b*
c^2 - 221184*a^5*b^2*c - 24576*a^2*b^2*c^4 + 49152*a^2*b^3*c^3 - 106496*a^3
*b^2*c^3 + 352256*a^3*b^3*c^2 - 172032*a^4*b^2*c^2 + 32768*a^6*b*c + tan(x/
2)*(-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2
+ 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c
 + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 3
2*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b
^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(65536*a^8*c + 16384*a^2*b^7 - 49152*a^3*b
^6 + 65536*a^4*b^5 - 65536*a^5*b^4 + 49152*a^6*b^3 - 16384*a^7*b^2 + 196608
*a^4*c^5 + 131072*a^5*c^4 - 262144*a^6*c^3 - 131072*a^7*c^2 - 16384*a^2*b^6
*c - 114688*a^3*b^5*c - 65536*a^4*b*c^4 + 376832*a^4*b^4*c + 720896*a^5*b*c
^3 - 409600*a^5*b^3*c + 589824*a^6*b*c^2 + 294912*a^6*b^2*c + 16384*a^2*b^4

*c^3 - 16384*a^2*b^5*c^2 - 114688*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 196608*
a^3*b^4*c^2 - 557056*a^4*b^2*c^3 + 16384*a^4*b^3*c^2 - 655360*a^5*b^2*c^2 -
196608*a^7*b*c)) + tan(x/2)*(49152*a*b^6 - 65536*a^6*c + 16384*b^6*c - 163
84*b^7 - 65536*a^2*b^5 + 65536*a^3*b^4 - 49152*a^4*b^3 + 16384*a^5*b^2 - 14
7456*a^2*c^5 + 212992*a^3*c^4 + 671744*a^4*c^3 + 245760*a^5*c^2 - 16384*b^4
*c^3 + 16384*b^5*c^2 + 98304*a*b^2*c^4 - 65536*a*b^3*c^3 - 180224*a*b^4*c^2
+ 49152*a^2*b*c^4 - 393216*a^2*b^4*c - 1081344*a^3*b*c^3 + 475136*a^3*b^3*
c - 802816*a^4*b*c^2 - 327680*a^4*b^2*c + 344064*a^2*b^2*c^3 + 180224*a^2*b
^3*c^2 + 557056*a^3*b^2*c^2 + 98304*a*b^5*c + 196608*a^5*b*c))*(-(b^6 - 8*a
^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3 +
b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4
*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*
a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4
*b^2*c^2)))^(1/2) + 32768*a^2*b^4 - 8192*a^3*b^3 + 180224*a^2*c^4 + 253952*
a^3*c^3 + 98304*a^4*c^2 - 8192*b^2*c^4 + 32768*b^3*c^3 - 16384*b^4*c^2 + 24
576*a*b^2*c^3 + 237568*a*b^3*c^2 - 458752*a^2*b*c^3 + 262144*a^2*b^3*c - 27
0336*a^3*b*c^2 - 155648*a^3*b^2*c + 16384*a^2*b^2*c^2 - 122880*a*b*c^4 - 11
4688*a*b^4*c + 32768*a^4*b*c))*(-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*
c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2)
+ 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^
4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b
^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*1i)/((tan(x/2)
*(8192*a*b^4 - 73728*a*c^4 - 57344*b*c^4 + 40960*b^4*c - 8192*b^5 + 24576*c
^5 - 81920*a^2*c^3 + 16384*a^3*c^2 + 81920*b^2*c^3 - 81920*b^3*c^2 - 81920*
a*b^2*c^2 + 81920*a^2*b*c^2 - 32768*a^2*b^2*c + 163840*a*b*c^3) - (-(b^6 -
8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^
3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(
-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 +
16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*
a^4*b^2*c^2)))^(1/2)*(24576*a*c^5 - 49152*a*b^5 - 32768*b^5*c + 24576*b^6 -
((-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2 +
6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c
+ 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32
*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^
2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(24576*a^2*b^6 - 57344*a^3*b^5 + 40960*a^4*
b^4 - 8192*a^5*b^3 + 98304*a^3*c^5 + 425984*a^4*c^4 + 557056*a^5*c^3 + 2293
76*a^6*c^2 - 49152*a^2*b^5*c - 196608*a^3*b*c^4 - 90112*a^3*b^4*c - 622592*
a^4*b*c^3 + 327680*a^4*b^3*c - 393216*a^5*b*c^2 - 221184*a^5*b^2*c - 24576*
a^2*b^2*c^4 + 49152*a^2*b^3*c^3 - 106496*a^3*b^2*c^3 + 352256*a^3*b^3*c^2 -
172032*a^4*b^2*c^2 + 32768*a^6*b*c + tan(x/2)*(-(b^6 - 8*a^2*c^4 - 8*a^3*c
^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3 + b*c^2*(-(4*a*c
- b^2)^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(
1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3
*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2
)*(65536*a^8*c + 16384*a^2*b^7 - 49152*a^3*b^6 + 65536*a^4*b^5 - 65536*a^5*

b^4 + 49152*a^6*b^3 - 16384*a^7*b^2 + 196608*a^4*c^5 + 131072*a^5*c^4 - 262
144*a^6*c^3 - 131072*a^7*c^2 - 16384*a^2*b^6*c - 114688*a^3*b^5*c - 65536*a
^4*b*c^4 + 376832*a^4*b^4*c + 720896*a^5*b*c^3 - 409600*a^5*b^3*c + 589824*
a^6*b*c^2 + 294912*a^6*b^2*c + 16384*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 1146
88*a^3*b^2*c^4 + 81920*a^3*b^3*c^3 + 196608*a^3*b^4*c^2 - 557056*a^4*b^2*c^
3 + 16384*a^4*b^3*c^2 - 655360*a^5*b^2*c^2 - 196608*a^7*b*c)) + tan(x/2)*(4
9152*a*b^6 - 65536*a^6*c + 16384*b^6*c - 16384*b^7 - 65536*a^2*b^5 + 65536*
a^3*b^4 - 49152*a^4*b^3 + 16384*a^5*b^2 - 147456*a^2*c^5 + 212992*a^3*c^4 +
671744*a^4*c^3 + 245760*a^5*c^2 - 16384*b^4*c^3 + 16384*b^5*c^2 + 98304*a*
b^2*c^4 - 65536*a*b^3*c^3 - 180224*a*b^4*c^2 + 49152*a^2*b*c^4 - 393216*a^2
*b^4*c - 1081344*a^3*b*c^3 + 475136*a^3*b^3*c - 802816*a^4*b*c^2 - 327680*a
^4*b^2*c + 344064*a^2*b^2*c^3 + 180224*a^2*b^3*c^2 + 557056*a^3*b^2*c^2 + 9
8304*a*b^5*c + 196608*a^5*b*c))*(-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a
*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2)
+ 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b
^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*
b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2) + 32768*a^2*b
^4 - 8192*a^3*b^3 + 180224*a^2*c^4 + 253952*a^3*c^3 + 98304*a^4*c^2 - 8192*
b^2*c^4 + 32768*b^3*c^3 - 16384*b^4*c^2 + 24576*a*b^2*c^3 + 237568*a*b^3*c^
2 - 458752*a^2*b*c^3 + 262144*a^2*b^3*c - 270336*a^3*b*c^2 - 155648*a^3*b^2
*c + 16384*a^2*b^2*c^2 - 122880*a*b*c^4 - 114688*a*b^4*c + 32768*a^4*b*c))*
(-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2 + 6
*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c +
2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a
^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*
c^3 - 32*a^4*b^2*c^2)))^(1/2) - (tan(x/2)*(8192*a*b^4 - 73728*a*c^4 - 57344
*b*c^4 + 40960*b^4*c - 8192*b^5 + 24576*c^5 - 81920*a^2*c^3 + 16384*a^3*c^2
+ 81920*b^2*c^3 - 81920*b^3*c^2 - 81920*a*b^2*c^2 + 81920*a^2*b*c^2 - 3276
8*a^2*b^2*c + 163840*a*b*c^3) + (-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a
*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2)
+ 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b
^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*
b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(24576*a*c^5
- 49152*a*b^5 - 32768*b^5*c + 24576*b^6 + ((-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 -
b^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^
2)^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2)
)/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4
*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(5
7344*a^3*b^5 - 24576*a^2*b^6 - 40960*a^4*b^4 + 8192*a^5*b^3 - 98304*a^3*c^5
- 425984*a^4*c^4 - 557056*a^5*c^3 - 229376*a^6*c^2 + 49152*a^2*b^5*c + 196
608*a^3*b*c^4 + 90112*a^3*b^4*c + 622592*a^4*b*c^3 - 327680*a^4*b^3*c + 393
216*a^5*b*c^2 + 221184*a^5*b^2*c + 24576*a^2*b^2*c^4 - 49152*a^2*b^3*c^3 +
106496*a^3*b^2*c^3 - 352256*a^3*b^3*c^2 + 172032*a^4*b^2*c^2 - 32768*a^6*b*
c + tan(x/2)*(-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2)
- b^4*c^2 + 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 18*a^2*b^2*c^2 -

```
8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a
^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2
 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2)*(65536*a^8*c + 16384*a^2*b^7 - 4
9152*a^3*b^6 + 65536*a^4*b^5 - 65536*a^5*b^4 + 49152*a^6*b^3 - 16384*a^7*b^
2 + 196608*a^4*c^5 + 131072*a^5*c^4 - 262144*a^6*c^3 - 131072*a^7*c^2 - 163
84*a^2*b^6*c - 114688*a^3*b^5*c - 65536*a^4*b*c^4 + 376832*a^4*b^4*c + 7208
96*a^5*b*c^3 - 409600*a^5*b^3*c + 589824*a^6*b*c^2 + 294912*a^6*b^2*c + 163
84*a^2*b^4*c^3 - 16384*a^2*b^5*c^2 - 114688*a^3*b^2*c^4 + 81920*a^3*b^3*c^3
 + 196608*a^3*b^4*c^2 - 557056*a^4*b^2*c^3 + 16384*a^4*b^3*c^2 - 655360*a^5
*b^2*c^2 - 196608*a^7*b*c)) + tan(x/2)*(49152*a*b^6 - 65536*a^6*c + 16384*b
^6*c - 16384*b^7 - 65536*a^2*b^5 + 65536*a^3*b^4 - 49152*a^4*b^3 + 16384*a^
5*b^2 - 147456*a^2*c^5 + 212992*a^3*c^4 + 671744*a^4*c^3 + 245760*a^5*c^2 -
 16384*b^4*c^3 + 16384*b^5*c^2 + 98304*a*b^2*c^4 - 65536*a*b^3*c^3 - 180224
*a*b^4*c^2 + 49152*a^2*b*c^4 - 393216*a^2*b^4*c - 1081344*a^3*b*c^3 + 47513
6*a^3*b^3*c - 802816*a^4*b*c^2 - 327680*a^4*b^2*c + 344064*a^2*b^2*c^3 + 18
0224*a^2*b^3*c^2 + 557056*a^3*b^2*c^2 + 98304*a*b^5*c + 196608*a^5*b*c))*(-
(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a
*b^2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*
a*b*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5
*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^
3 - 32*a^4*b^2*c^2)))^(1/2) + 32768*a^2*b^4 - 8192*a^3*b^3 + 180224*a^2*c^4
 + 253952*a^3*c^3 + 98304*a^4*c^2 - 8192*b^2*c^4 + 32768*b^3*c^3 - 16384*b^
4*c^2 + 24576*a*b^2*c^3 + 237568*a*b^3*c^2 - 458752*a^2*b*c^3 + 262144*a^2*
b^3*c - 270336*a^3*b*c^2 - 155648*a^3*b^2*c + 16384*a^2*b^2*c^2 - 122880*a*
b*c^4 - 114688*a*b^4*c + 32768*a^4*b*c))*(-(b^6 - 8*a^2*c^4 - 8*a^3*c^3 - b
^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^2*c^3 + b*c^2*(-(4*a*c - b^2)
^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b*c*(-(4*a*c - b^2)^3)^(1/2))/
(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^3 + 16*a^6*c^2 + 10*a^3*b^4*c
 - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 - 32*a^4*b^2*c^2)))^(1/2) + 49
152*a*c^3 - 65536*b*c^3 + 49152*c^4 + 16384*b^2*c^2 - 16384*a*b*c^2))*(-(b^
6 - 8*a^2*c^4 - 8*a^3*c^3 - b^3*(-(4*a*c - b^2)^3)^(1/2) - b^4*c^2 + 6*a*b^
2*c^3 + b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 18*a^2*b^2*c^2 - 8*a*b^4*c + 2*a*b
*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^4*b^4 - a^2*b^6 + 16*a^4*c^4 + 32*a^5*c^
3 + 16*a^6*c^2 + 10*a^3*b^4*c - 8*a^5*b^2*c + a^2*b^4*c^2 - 8*a^3*b^2*c^3 -
 32*a^4*b^2*c^2)))^(1/2)*2i
```

3.19 $\int \frac{\sec^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 19, antiderivative size = 275

$$\int \frac{\sec^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{2bc\left(1+\frac{b^2-2ac}{b\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{a^2\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} + \frac{2bc\left(1-\frac{b^2-2ac}{b\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{a^2\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}} - \frac{b\,\text{arctanh}(\sin(x))}{a^2} + \frac{\tan(x)}{a}$$

```
[Out] -b*arctanh(sin(x))/a^2+2*b*c*arctan((b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/
2*x)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2))*(1+(-2*a*c+b^2)/b/(-4*a*c+b^2)^(1/2)
)/a^2/(b-2*c-(-4*a*c+b^2)^(1/2))^(1/2)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2)+2*b
*c*arctan((b-2*c+(-4*a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c+(-4*a*c+b^2)^(
1/2))^(1/2))*(1+(2*a*c-b^2)/b/(-4*a*c+b^2)^(1/2))/a^2/(b-2*c+(-4*a*c+b^2)^(
1/2))^(1/2)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2)+tan(x)/a
```

Rubi [A] (verified)

Time = 1.30 (sec) , antiderivative size = 275, normalized size of antiderivative = 1.00, number of steps used = 10, number of rules used = 7, $\frac{\text{number of rules}}{\text{integrand size}} = 0.368$, Rules used

= {3338, 3374, 2738, 211, 3855, 3852, 8}

$$\int \frac{\sec^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{2bc\left(\frac{b^2-2ac}{b\sqrt{b^2-4ac}}+1\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{-\sqrt{b^2-4ac}+b-2c}}{\sqrt{-\sqrt{b^2-4ac}+b+2c}}\right)}{a^2\sqrt{-\sqrt{b^2-4ac}+b-2c}\sqrt{-\sqrt{b^2-4ac}+b+2c}} + \frac{2bc\left(1-\frac{b^2-2ac}{b\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{\sqrt{b^2-4ac}+b-2c}}{\sqrt{\sqrt{b^2-4ac}+b+2c}}\right)}{a^2\sqrt{\sqrt{b^2-4ac}+b-2c}\sqrt{\sqrt{b^2-4ac}+b+2c}} - \frac{b\operatorname{arctanh}(\sin(x))}{a^2} + \frac{\tan(x)}{a}$$

```
[In] Int[Sec[x]^2/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (2*b*c*(1 + (b^2 - 2*a*c)/(b*Sqrt[b^2 - 4*a*c]))*ArcTan[(Sqrt[b - 2*c - Sqr
t[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]])/(a^2*Sqrt[b -
 2*c - Sqrt[b^2 - 4*a*c]]*Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]) + (2*b*c*(1 -
(b^2 - 2*a*c)/(b*Sqrt[b^2 - 4*a*c]))*ArcTan[(Sqrt[b - 2*c + Sqrt[b^2 - 4*a*
c]]*Tan[x/2])/Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]]])/(a^2*Sqrt[b - 2*c + Sqrt[
b^2 - 4*a*c]]*Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]]) - (b*ArcTanh[Sin[x]])/a^2
+ Tan[x]/a
```

Rule 8

```
Int[a_, x_Symbol] :> Simp[a*x, x] /; FreeQ[a, x]
```

Rule 211

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(Rt[a/b, 2]/a)*ArcTan[x/R
t[a/b, 2]], x] /; FreeQ[{a, b}, x] && PosQ[a/b]
```

Rule 2738

```
Int[((a_) + (b_.)*sin[Pi/2 + (c_.) + (d_.)*(x_)])^(-1), x_Symbol] :> With[{
e = FreeFactors[Tan[(c + d*x)/2], x]}, Dist[2*(e/d), Subst[Int[1/(a + b + (
a - b)*e^2*x^2), x], x, Tan[(c + d*x)/2]/e], x]] /; FreeQ[{a, b, c, d}, x]
&& NeQ[a^2 - b^2, 0]
```

Rule 3338

```
Int[cos[(d_.) + (e_.)*(x_)]^(m_.)*((a_.) + cos[(d_.) + (e_.)*(x_)]^(n_.)*(b
_.) + cos[(d_.) + (e_.)*(x_)]^(n2_.)*(c_.))^(p_), x_Symbol] :> Int[ExpandTr
ig[cos[d + e*x]^m*(a + b*cos[d + e*x]^n + c*cos[d + e*x]^(2*n))^p, x], x] /
; FreeQ[{a, b, c, d, e}, x] && EqQ[n2, 2*n] && NeQ[b^2 - 4*a*c, 0] && Integ
ersQ[m, n, p]
```

Rule 3374

```
Int[(cos[(d_.) + (e_.)*(x_)]*(B_.) + (A_))/((a_.) + cos[(d_.) + (e_.)*(x_)]
*(b_.) + cos[(d_.) + (e_.)*(x_)]^2*(c_.)), x_Symbol] :> Module[{q = Rt[b^2
- 4*a*c, 2]}, Dist[B + (b*B - 2*A*c)/q, Int[1/(b + q + 2*c*Cos[d + e*x]), x
], x] + Dist[B - (b*B - 2*A*c)/q, Int[1/(b - q + 2*c*Cos[d + e*x]), x], x]]
 /; FreeQ[{a, b, c, d, e, A, B}, x] && NeQ[b^2 - 4*a*c, 0]
```

Rule 3852

```
Int[csc[(c_.) + (d_.)*(x_)]^(n_), x_Symbol] :> Dist[-d^(-1), Subst[Int[Expa
ndIntegrand[(1 + x^2)^(n/2 - 1), x], x], x, Cot[c + d*x]], x] /; FreeQ[{c,
d}, x] && IGtQ[n/2, 0]
```

Rule 3855

```
Int[csc[(c_.) + (d_.)*(x_)], x_Symbol] :> Simp[-ArcTanh[Cos[c + d*x]]/d, x]
 /; FreeQ[{c, d}, x]
```

Rubi steps

$$\text{integral} = \int \left(\frac{b^2\left(1-\frac{ac}{b^2}\right)+bc\cos(x)}{a^2\left(a+b\cos(x)+c\cos^2(x)\right)} - \frac{b\sec(x)}{a^2} + \frac{\sec^2(x)}{a} \right) dx$$

$$= \frac{\int \frac{b^2\left(1-\frac{ac}{b^2}\right)+bc\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx}{a^2} + \frac{\int \sec^2(x)\,dx}{a} - \frac{b\int \sec(x)\,dx}{a^2}$$

$$= -\frac{b\text{arctanh}(\sin(x))}{a^2} - \frac{\text{Subst}\left(\int 1\,dx, x, -\tan(x)\right)}{a} + \frac{\left(c\left(b-\frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\right)\int \frac{1}{b+\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{a^2} + \frac{\left(c\left(b+\frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\right)\int \frac{1}{b-\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{a^2}$$

$$= -\frac{b\text{arctanh}(\sin(x))}{a^2} + \frac{\tan(x)}{a} + \frac{\left(2c\left(b-\frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\right)\text{Subst}\left(\int \frac{1}{b+2c+\sqrt{b^2-4ac}+\left(b-2c+\sqrt{b^2-4ac}\right)x^2}\,dx, x, \tan\left(\frac{x}{2}\right)\right)}{a^2} + \frac{\left(2c\left(b+\frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\right)\text{Subst}\left(\int \frac{1}{b+2c-\sqrt{b^2-4ac}+\left(b-2c-\sqrt{b^2-4ac}\right)x^2}\,dx, x, \tan\left(\frac{x}{2}\right)\right)}{a^2}$$

$$= \frac{2c\left(b+\frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{a^2\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} + \frac{2c\left(b-\frac{b^2-2ac}{\sqrt{b^2-4ac}}\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{a^2\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}} - \frac{b\text{arctanh}(\sin(x))}{a^2} + \frac{\tan(x)}{a}$$

Mathematica [A] (verified)

Time = 1.90 (sec) , antiderivative size = 348, normalized size of antiderivative = 1.27

$$\int \frac{\sec^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \frac{-\frac{\sqrt{2}c\left(-b^2+2ac+b\sqrt{b^2-4ac}\right)\text{arctanh}\left(\frac{\left(b-2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)-2b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{-b^2+2c(a+c)-b\sqrt{b^2-4ac}}} + \frac{\sqrt{2}c\left(b^2-2ac+b\sqrt{b^2-4ac}\right)\text{arctanh}\left(\frac{\left(-b+2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)+2b\sqrt{b^2-4a}}}\right)}{\sqrt{b^2-4ac}\sqrt{-b^2+2c(a+c)+b\sqrt{b^2-4ac}}}}{a^2}$$

```
[In] Integrate[Sec[x]^2/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (-((Sqrt[2]*c*(-b^2 + 2*a*c + b*Sqrt[b^2 - 4*a*c])*ArcTanh[((b - 2*c + Sqrt
[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) - 2*b*Sqrt[b^2 - 4*a*c]]
])/(Sqrt[b^2 - 4*a*c]*Sqrt[-b^2 + 2*c*(a + c) - b*Sqrt[b^2 - 4*a*c]])) + (S
qrt[2]*c*(b^2 - 2*a*c + b*Sqrt[b^2 - 4*a*c])*ArcTanh[((-b + 2*c + Sqrt[b^2
- 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*(a + c) + 2*b*Sqrt[b^2 - 4*a*c]]])/(S
qrt[b^2 - 4*a*c]*Sqrt[-b^2 + 2*c*(a + c) + b*Sqrt[b^2 - 4*a*c]]) + b*Log[Co
s[x/2] - Sin[x/2]] - b*Log[Cos[x/2] + Sin[x/2]] + (a*Sin[x/2])/(Cos[x/2] -
Sin[x/2]) + (a*Sin[x/2])/(Cos[x/2] + Sin[x/2]))/a^2
```

Maple [A] (verified)

Time = 4.66 (sec) , antiderivative size = 354, normalized size of antiderivative = 1.29

method	result
default	$\frac{2(a-b+c)\left(\frac{\left(-\sqrt{-4ac+b^2}\,ac+\sqrt{-4ac+b^2}\,b^2-\sqrt{-4ac+b^2}\,bc+3cab-2a\,c^2-b^3+b^2c\right)\text{arctanh}\left(\frac{(-a+b-c)\tan\left(\frac{x}{2}\right)}{\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}\right)}{2\sqrt{-4ac+b^2}\,(a-b+c)\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}+\frac{\left(-\sqrt{-4ac+b^2}\,a\right.}{}\right.}{a^2}$
risch	Expression too large to display

```
[In] int(sec(x)^2/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] 2/a^2*(a-b+c)*(1/2*(-(-4*a*c+b^2)^(1/2)*a*c+(-4*a*c+b^2)^(1/2)*b^2-(-4*a*c+
b^2)^(1/2)*b*c+3*c*a*b-2*a*c^2-b^3+b^2*c)/(-4*a*c+b^2)^(1/2)/(a-b+c)/(((-4*
a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2)*arctanh((-a+b-c)*tan(1/2*x)/(((-4*a*c+b^
2)^(1/2)-a+c)*(a-b+c))^(1/2))+1/2*(-(-4*a*c+b^2)^(1/2)*a*c+(-4*a*c+b^2)^(1/
2)*b^2-(-4*a*c+b^2)^(1/2)*b*c-3*c*a*b+2*a*c^2+b^3-b^2*c)/(-4*a*c+b^2)^(1/2)
/(a-b+c)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)*arctan((a-b+c)*tan(1/2*x)
/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)))-1/a/(tan(1/2*x)+1)-b/a^2*ln(tan
(1/2*x)+1)-1/a/(tan(1/2*x)-1)+b/a^2*ln(tan(1/2*x)-1)
```

Fricas [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 6851 vs. $2(234) = 468$.

Time = 257.77 (sec) , antiderivative size = 6851, normalized size of antiderivative = 24.91

$$\int \frac{\sec^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(sec(x)^2/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
```

```
[Out] Too large to include
```

Sympy [F]

$$\int \frac{\sec^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \int \frac{\sec^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$$

```
[In] integrate(sec(x)**2/(a+b*cos(x)+c*cos(x)**2),x)
```

```
[Out] Integral(sec(x)**2/(a + b*cos(x) + c*cos(x)**2), x)
```

Maxima [F]

$$\int \frac{\sec^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \int \frac{\sec(x)^2}{c\cos(x)^2+b\cos(x)+a}\,dx$$

```
[In] integrate(sec(x)^2/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
```

```
[Out] 1/2*(2*(a^2*cos(2*x)^2 + a^2*sin(2*x)^2 + 2*a^2*cos(2*x) + a^2)*integrate(2
*(2*b^2*c*cos(3*x)^2 + 2*b^2*c*cos(x)^2 + 2*b^2*c*sin(3*x)^2 + 2*b^2*c*sin(
x)^2 + b*c^2*cos(x) + 4*(2*a*b^2 - a*c^2 - (2*a^2 - b^2)*c)*cos(2*x)^2 + 4*
(2*a*b^2 - a*c^2 - (2*a^2 - b^2)*c)*sin(2*x)^2 + 2*(2*b^3 + b*c^2)*sin(2*x)
*sin(x) + (b*c^2*cos(3*x) + b*c^2*cos(x) + 2*(b^2*c - a*c^2)*cos(2*x))*cos(
```

```
4*x) + (4*b^2*c*cos(x) + b*c^2 + 2*(2*b^3 + b*c^2)*cos(2*x))*cos(3*x) + 2*(
b^2*c - a*c^2 + (2*b^3 + b*c^2)*cos(x))*cos(2*x) + (b*c^2*sin(3*x) + b*c^2*
sin(x) + 2*(b^2*c - a*c^2)*sin(2*x))*sin(4*x) + 2*(2*b^2*c*sin(x) + (2*b^3
+ b*c^2)*sin(2*x))*sin(3*x))/(a^2*c^2*cos(4*x)^2 + 4*a^2*b^2*cos(3*x)^2 + 4
*a^2*b^2*cos(x)^2 + a^2*c^2*sin(4*x)^2 + 4*a^2*b^2*sin(3*x)^2 + 4*a^2*b^2*s
in(x)^2 + 4*a^2*b*c*cos(x) + a^2*c^2 + 4*(4*a^4 + 4*a^3*c + a^2*c^2)*cos(2*
x)^2 + 4*(4*a^4 + 4*a^3*c + a^2*c^2)*sin(2*x)^2 + 8*(2*a^3*b + a^2*b*c)*sin
(2*x)*sin(x) + 2*(2*a^2*b*c*cos(3*x) + 2*a^2*b*c*cos(x) + a^2*c^2 + 2*(2*a^
3*c + a^2*c^2)*cos(2*x))*cos(4*x) + 4*(2*a^2*b^2*cos(x) + a^2*b*c + 2*(2*a^
3*b + a^2*b*c)*cos(2*x))*cos(3*x) + 4*(2*a^3*c + a^2*c^2 + 2*(2*a^3*b + a^2
*b*c)*cos(x))*cos(2*x) + 4*(a^2*b*c*sin(3*x) + a^2*b*c*sin(x) + (2*a^3*c +
a^2*c^2)*sin(2*x))*sin(4*x) + 8*(a^2*b^2*sin(x) + (2*a^3*b + a^2*b*c)*sin(2
*x))*sin(3*x)), x) - (b*cos(2*x)^2 + b*sin(2*x)^2 + 2*b*cos(2*x) + b)*log(c
os(x)^2 + sin(x)^2 + 2*sin(x) + 1) + (b*cos(2*x)^2 + b*sin(2*x)^2 + 2*b*cos
(2*x) + b)*log(cos(x)^2 + sin(x)^2 - 2*sin(x) + 1) + 4*a*sin(2*x))/(a^2*cos
(2*x)^2 + a^2*sin(2*x)^2 + 2*a^2*cos(2*x) + a^2)
```

Giac [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 4784 vs. $2(234) = 468$.

Time = 2.90 (sec) , antiderivative size = 4784, normalized size of antiderivative = 17.40

$$\int \frac{\sec^2(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(sec(x)^2/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] (2*a^2*b^5 - 2*b^7 - 14*a^3*b^3*c - 6*a^2*b^4*c + 26*a*b^5*c - 2*b^6*c + 24
*a^4*b*c^2 + 36*a^3*b^2*c^2 - 108*a^2*b^3*c^2 + 16*a*b^4*c^2 + 2*b^5*c^2 -
48*a^4*c^3 + 144*a^3*b*c^3 - 24*a^2*b^2*c^3 - 22*a*b^3*c^3 + 2*b^4*c^3 - 32
*a^3*c^4 + 56*a^2*b*c^4 - 12*a*b^2*c^4 + 16*a^2*c^5 + 3*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*b^4 - 2*sqrt(a^2 - a*b + b*c -
 c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^5 - 5*sqrt(a^2 - a*b + b*c - c^2
+ sqrt(b^2 - 4*a*c)*(a - b + c))*b^6 - 15*sqrt(a^2 - a*b + b*c - c^2 + sqrt
(b^2 - 4*a*c)*(a - b + c))*a^3*b^2*c + 13*sqrt(a^2 - a*b + b*c - c^2 + sqrt
(b^2 - 4*a*c)*(a - b + c))*a^2*b^3*c + 37*sqrt(a^2 - a*b + b*c - c^2 + sqrt
(b^2 - 4*a*c)*(a - b + c))*a*b^4*c + sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
- 4*a*c)*(a - b + c))*b^5*c + 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*
a*c)*(a - b + c))*a^4*c^2 - 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*
c)*(a - b + c))*a^3*b*c^2 - 82*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*
c)*(a - b + c))*a^2*b^2*c^2 + 4*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a
*c)*(a - b + c))*a*b^3*c^2 + sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)
*(a - b + c))*b^4*c^2 + 56*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(
a - b + c))*a^3*c^3 - 32*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
- b + c))*a^2*b*c^3 + sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
```

+ c))*a*b^2*c^3 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*b^3*c^3 - 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*a^2*c^4 + 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c)
)*a*b*c^4 + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*s
qrt(b^2 - 4*a*c)*a^2*b^3 - 2*(b^2 - 4*a*c)*a^2*b^3 - 2*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4 - 5*sqrt(a^
2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5
+ 2*(b^2 - 4*a*c)*b^5 - 9*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
- b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c + 6*(b^2 - 4*a*c)*a^3*b*c + 9*sqrt(a^2
- a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b
^2*c + 6*(b^2 - 4*a*c)*a^2*b^2*c + 27*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2
- 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c - 18*(b^2 - 4*a*c)*a*b^3*c
+ sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4
*a*c)*b^4*c + 2*(b^2 - 4*a*c)*b^4*c - 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b
^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^2 - 12*(b^2 - 4*a*c)*a^3*c
^2 - 38*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^
2 - 4*a*c)*a^2*b*c^2 + 36*(b^2 - 4*a*c)*a^2*b*c^2 + 6*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^2 - 8*(b^2
- 4*a*c)*a*b^2*c^2 + sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*sqrt(b^2 - 4*a*c)*b^3*c^2 - 2*(b^2 - 4*a*c)*b^3*c^2 - 28*sqrt(a^2 -
a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^3
- 8*(b^2 - 4*a*c)*a^2*c^3 + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b*c^3 + 14*(b^2 - 4*a*c)*a*b*c^3 - 5*sqr
t(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*
b^2*c^3 - 2*(b^2 - 4*a*c)*b^2*c^3 + 10*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^
2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*c^4 + 4*(b^2 - 4*a*c)*a*c^4)*(p
i*floor(1/2*x/pi + 1/2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a^3 - 2*a^2
c + sqrt(-4(a^3 + a^2*b + a^2*c)*(a^3 - a^2*b + a^2*c) + 4*(a^3 - a^2*c)^
2))/(a^3 - a^2*b + a^2*c))))*abs(a - b + c)/(3*a^7*b^2 - 5*a^6*b^3 - 6*a^5*
b^4 + 10*a^4*b^5 + 3*a^3*b^6 - 5*a^2*b^7 - 12*a^8*c + 20*a^7*b*c + 47*a^6*b
^2*c - 60*a^5*b^3*c - 46*a^4*b^4*c + 40*a^3*b^5*c + 11*a^2*b^6*c - 92*a^7*c
^2 + 80*a^6*b*c^2 + 182*a^5*b^2*c^2 - 94*a^4*b^3*c^2 - 78*a^3*b^4*c^2 - 6*a
^2*b^5*c^2 - 184*a^6*c^3 + 56*a^5*b*c^3 + 166*a^4*b^2*c^3 + 36*a^3*b^3*c^3
- 6*a^2*b^4*c^3 - 120*a^5*c^4 - 48*a^4*b*c^4 + 23*a^3*b^2*c^4 + 11*a^2*b^3*
c^4 + 4*a^4*c^5 - 44*a^3*b*c^5 - 5*a^2*b^2*c^5 + 20*a^3*c^6) - (2*a^2*b^5 -
2*b^7 - 14*a^3*b^3*c - 6*a^2*b^4*c + 26*a*b^5*c - 2*b^6*c + 24*a^4*b*c^2 +
36*a^3*b^2*c^2 - 108*a^2*b^3*c^2 + 16*a*b^4*c^2 + 2*b^5*c^2 - 48*a^4*c^3 +
144*a^3*b*c^3 - 24*a^2*b^2*c^3 - 22*a*b^3*c^3 + 2*b^4*c^3 - 32*a^3*c^4 + 5
6*a^2*b*c^4 - 12*a*b^2*c^4 + 16*a^2*c^5 - 3*sqrt(a^2 - a*b + b*c - c^2 - sq
rt(b^2 - 4*a*c)*(a - b + c))*a^2*b^4 + 2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(
b^2 - 4*a*c)*(a - b + c))*a*b^5 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
4*a*c)*(a - b + c))*b^6 + 15*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*a^3*b^2*c - 13*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*a^2*b^3*c - 37*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*a*b^4*c - sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a

```
- b + c))*b^5*c - 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^4*c^2 + 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
c))*a^3*b*c^2 + 82*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
c))*a^2*b^2*c^2 - 4*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
 c))*a*b^3*c^2 - sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c)
)*b^4*c^2 - 56*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*
a^3*c^3 + 32*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^
2*b*c^3 - sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^2
*c^3 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*b^3*c^
3 + 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a^2*c^4
- 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a*b*c^4 +
3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*a^2*b^3 - 2*(b^2 - 4*a*c)*a^2*b^3 - 2*sqrt(a^2 - a*b + b*c - c^2 - sqr
t(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^4 - 5*sqrt(a^2 - a*b + b*
c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5 + 2*(b^2 - 4
*a*c)*b^5 - 9*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*s
qrt(b^2 - 4*a*c)*a^3*b*c + 6*(b^2 - 4*a*c)*a^3*b*c + 9*sqrt(a^2 - a*b + b*c
 - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c + 6*(b^
2 - 4*a*c)*a^2*b^2*c + 27*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*sqrt(b^2 - 4*a*c)*a*b^3*c - 18*(b^2 - 4*a*c)*a*b^3*c + sqrt(a^2
- a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c
+ 2*(b^2 - 4*a*c)*b^4*c - 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*
(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^2 - 12*(b^2 - 4*a*c)*a^3*c^2 - 38*sqrt
(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a
^2*b*c^2 + 36*(b^2 - 4*a*c)*a^2*b*c^2 + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt
(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^2 - 8*(b^2 - 4*a*c)*a*
b^2*c^2 + sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(
b^2 - 4*a*c)*b^3*c^2 - 2*(b^2 - 4*a*c)*b^3*c^2 - 28*sqrt(a^2 - a*b + b*c -
c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*c^3 - 8*(b^2 - 4
*a*c)*a^2*c^3 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c
))*sqrt(b^2 - 4*a*c)*a*b*c^3 + 14*(b^2 - 4*a*c)*a*b*c^3 - 5*sqrt(a^2 - a*b
+ b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^2*c^3 - 2*
(b^2 - 4*a*c)*b^2*c^3 + 10*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(
a - b + c))*sqrt(b^2 - 4*a*c)*a*c^4 + 4*(b^2 - 4*a*c)*a*c^4)*(pi*floor(1/2*
x/pi + 1/2) + arctan(2*sqrt(1/2)*tan(1/2*x)/sqrt((2*a^3 - 2*a^2*c - sqrt(-4
*(a^3 + a^2*b + a^2*c)*(a^3 - a^2*b + a^2*c) + 4*(a^3 - a^2*c)^2))/(a^3 - a
^2*b + a^2*c))))*abs(a - b + c)/(3*a^7*b^2 - 5*a^6*b^3 - 6*a^5*b^4 + 10*a^4
*b^5 + 3*a^3*b^6 - 5*a^2*b^7 - 12*a^8*c + 20*a^7*b*c + 47*a^6*b^2*c - 60*a^
5*b^3*c - 46*a^4*b^4*c + 40*a^3*b^5*c + 11*a^2*b^6*c - 92*a^7*c^2 + 80*a^6*
b*c^2 + 182*a^5*b^2*c^2 - 94*a^4*b^3*c^2 - 78*a^3*b^4*c^2 - 6*a^2*b^5*c^2 -
 184*a^6*c^3 + 56*a^5*b*c^3 + 166*a^4*b^2*c^3 + 36*a^3*b^3*c^3 - 6*a^2*b^4*
c^3 - 120*a^5*c^4 - 48*a^4*b*c^4 + 23*a^3*b^2*c^4 + 11*a^2*b^3*c^4 + 4*a^4*
c^5 - 44*a^3*b*c^5 - 5*a^2*b^2*c^5 + 20*a^3*c^6) - b*log(abs(tan(1/2*x) + 1
))/a^2 + b*log(abs(tan(1/2*x) - 1))/a^2 - 2*tan(1/2*x)/((tan(1/2*x)^2 - 1)*
a)
```

Mupad [B] (verification not implemented)

Time = 14.79 (sec) , antiderivative size = 29417, normalized size of antiderivative = 106.97

$$\int \frac{\sec^2(x)}{a + b\cos(x) + c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] int(1/(cos(x)^2*(a + b*cos(x) + c*cos(x)^2)),x)
```

```
[Out] (b*atan(((b*((8192*tan(x/2)*(a*b^8 + 5*b^8*c - b^9 + a^2*c^7 + a^3*c^6 + b^
4*c^5 - 5*b^5*c^4 + 10*b^6*c^3 - 10*b^7*c^2 - 2*a*b^2*c^6 + 14*a*b^3*c^5 -
35*a*b^4*c^4 + 40*a*b^5*c^3 - 20*a*b^6*c^2 - a^2*b*c^6 - 6*a^2*b^6*c + 10*a
^2*b^2*c^5 - 20*a^2*b^3*c^4 + 5*a^2*b^4*c^3 + 11*a^2*b^5*c^2 + 10*a^3*b^2*c
^4 - 18*a^3*b^3*c^3 + 9*a^3*b^4*c^2 - 2*a^4*b^2*c^3 + 2*a*b^7*c))/a^4 + (b*
((8192*(6*a^2*b^8 - 3*a*b^9 - 4*a^3*b^7 + a^4*b^6 + 3*a^4*c^6 + 2*a^5*c^5 -
 a^6*c^4 + 2*a*b^5*c^4 - 5*a*b^6*c^3 + a*b^7*c^2 + 16*a^2*b^7*c + 8*a^3*b*c
^6 - 38*a^3*b^6*c + 10*a^4*b*c^5 + 23*a^4*b^5*c + 6*a^5*b*c^4 - 5*a^5*b^4*c
 - 10*a^2*b^3*c^5 + 25*a^2*b^4*c^4 + 4*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 20*a^
3*b^2*c^5 - 36*a^3*b^3*c^4 + 91*a^3*b^4*c^3 - 3*a^3*b^5*c^2 - 24*a^4*b^2*c^
4 - 55*a^4*b^3*c^3 + 57*a^4*b^4*c^2 - 3*a^5*b^2*c^3 - 28*a^5*b^3*c^2 + 4*a^
6*b^2*c^2 + 5*a*b^8*c))/a^4 + (b*((b*((8192*(3*a^5*b^7 - 7*a^6*b^6 + 5*a^7*
b^5 - a^8*b^4 + 12*a^7*c^5 + 20*a^8*c^4 + 4*a^9*c^3 - 4*a^10*c^2 - 5*a^5*b^
6*c + 8*a^6*b*c^5 - 15*a^6*b^5*c + 28*a^7*b*c^4 + 46*a^7*b^4*c + 64*a^8*b*c
^3 - 31*a^8*b^3*c + 44*a^9*b*c^2 + 5*a^9*b^2*c - 2*a^5*b^3*c^4 + 5*a^5*b^4*
c^3 - a^5*b^5*c^2 - 23*a^6*b^2*c^4 - 3*a^6*b^3*c^3 + 40*a^6*b^4*c^2 - 85*a^
7*b^2*c^3 - 4*a^7*b^3*c^2 - 73*a^8*b^2*c^2))/a^4 + (8192*b*tan(x/2)*(8*a^12
*c + 2*a^6*b^7 - 6*a^7*b^6 + 8*a^8*b^5 - 8*a^9*b^4 + 6*a^10*b^3 - 2*a^11*b^
2 + 24*a^8*c^5 + 16*a^9*c^4 - 32*a^10*c^3 - 16*a^11*c^2 - 2*a^6*b^6*c - 14*
a^7*b^5*c - 8*a^8*b*c^4 + 46*a^8*b^4*c + 88*a^9*b*c^3 - 50*a^9*b^3*c + 72*a
^10*b*c^2 + 36*a^10*b^2*c + 2*a^6*b^4*c^3 - 2*a^6*b^5*c^2 - 14*a^7*b^2*c^4
+ 10*a^7*b^3*c^3 + 24*a^7*b^4*c^2 - 68*a^8*b^2*c^3 + 2*a^8*b^3*c^2 - 80*a^9
*b^2*c^2 - 24*a^11*b*c))/a^6))/a^2 + (8192*tan(x/2)*(6*a^3*b^8 - 2*a^2*b^9
- 8*a^4*b^7 + 8*a^5*b^6 - 6*a^6*b^5 + 2*a^7*b^4 + 10*a^5*c^6 + 6*a^6*c^5 -
2*a^7*c^4 + 2*a^8*c^3 + 2*a^2*b^8*c + 14*a^3*b^7*c - 50*a^4*b^6*c - 22*a^5*
b*c^5 + 56*a^5*b^5*c + 12*a^6*b*c^4 - 38*a^6*b^4*c + 18*a^7*b*c^3 + 24*a^7*
b^3*c - 8*a^8*b^2*c - 2*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 14*a^3*b^4*c^4 - 10*a
^3*b^5*c^3 - 24*a^3*b^6*c^2 - 27*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 59*a^4*b^4*
c^3 + 7*a^4*b^5*c^2 + 11*a^5*b^2*c^4 - 122*a^5*b^3*c^3 + 93*a^5*b^4*c^2 + 3
7*a^6*b^2*c^3 - 99*a^6*b^3*c^2 + 23*a^7*b^2*c^2))/a^4))/a^2))/a^2)*1i)/a^2
+ (b*((8192*tan(x/2)*(a*b^8 + 5*b^8*c - b^9 + a^2*c^7 + a^3*c^6 + b^4*c^5 -
 5*b^5*c^4 + 10*b^6*c^3 - 10*b^7*c^2 - 2*a*b^2*c^6 + 14*a*b^3*c^5 - 35*a*b^
4*c^4 + 40*a*b^5*c^3 - 20*a*b^6*c^2 - a^2*b*c^6 - 6*a^2*b^6*c + 10*a^2*b^2*
c^5 - 20*a^2*b^3*c^4 + 5*a^2*b^4*c^3 + 11*a^2*b^5*c^2 + 10*a^3*b^2*c^4 - 18
*a^3*b^3*c^3 + 9*a^3*b^4*c^2 - 2*a^4*b^2*c^3 + 2*a*b^7*c))/a^4 - (b*((8192*
(6*a^2*b^8 - 3*a*b^9 - 4*a^3*b^7 + a^4*b^6 + 3*a^4*c^6 + 2*a^5*c^5 - a^6*c^
```

4 + 2*a*b^5*c^4 - 5*a*b^6*c^3 + a*b^7*c^2 + 16*a^2*b^7*c + 8*a^3*b*c^6 - 38
*a^3*b^6*c + 10*a^4*b*c^5 + 23*a^4*b^5*c + 6*a^5*b*c^4 - 5*a^5*b^4*c - 10*a
^2*b^3*c^5 + 25*a^2*b^4*c^4 + 4*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 20*a^3*b^2*c
^5 - 36*a^3*b^3*c^4 + 91*a^3*b^4*c^3 - 3*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 55*
a^4*b^3*c^3 + 57*a^4*b^4*c^2 - 3*a^5*b^2*c^3 - 28*a^5*b^3*c^2 + 4*a^6*b^2*c
^2 + 5*a*b^8*c))/a^4 + (b*((b*((8192*(3*a^5*b^7 - 7*a^6*b^6 + 5*a^7*b^5 - a
^8*b^4 + 12*a^7*c^5 + 20*a^8*c^4 + 4*a^9*c^3 - 4*a^10*c^2 - 5*a^5*b^6*c + 8
*a^6*b*c^5 - 15*a^6*b^5*c + 28*a^7*b*c^4 + 46*a^7*b^4*c + 64*a^8*b*c^3 - 31
*a^8*b^3*c + 44*a^9*b*c^2 + 5*a^9*b^2*c - 2*a^5*b^3*c^4 + 5*a^5*b^4*c^3 - a
^5*b^5*c^2 - 23*a^6*b^2*c^4 - 3*a^6*b^3*c^3 + 40*a^6*b^4*c^2 - 85*a^7*b^2*c
^3 - 4*a^7*b^3*c^2 - 73*a^8*b^2*c^2))/a^4 - (8192*b*tan(x/2)*(8*a^12*c + 2*
a^6*b^7 - 6*a^7*b^6 + 8*a^8*b^5 - 8*a^9*b^4 + 6*a^10*b^3 - 2*a^11*b^2 + 24*
a^8*c^5 + 16*a^9*c^4 - 32*a^10*c^3 - 16*a^11*c^2 - 2*a^6*b^6*c - 14*a^7*b^5
*c - 8*a^8*b*c^4 + 46*a^8*b^4*c + 88*a^9*b*c^3 - 50*a^9*b^3*c + 72*a^10*b*c
^2 + 36*a^10*b^2*c + 2*a^6*b^4*c^3 - 2*a^6*b^5*c^2 - 14*a^7*b^2*c^4 + 10*a^
7*b^3*c^3 + 24*a^7*b^4*c^2 - 68*a^8*b^2*c^3 + 2*a^8*b^3*c^2 - 80*a^9*b^2*c^
2 - 24*a^11*b*c))/a^6))/a^2 - (8192*tan(x/2)*(6*a^3*b^8 - 2*a^2*b^9 - 8*a^4
*b^7 + 8*a^5*b^6 - 6*a^6*b^5 + 2*a^7*b^4 + 10*a^5*c^6 + 6*a^6*c^5 - 2*a^7*c
^4 + 2*a^8*c^3 + 2*a^2*b^8*c + 14*a^3*b^7*c - 50*a^4*b^6*c - 22*a^5*b*c^5 +
56*a^5*b^5*c + 12*a^6*b*c^4 - 38*a^6*b^4*c + 18*a^7*b*c^3 + 24*a^7*b^3*c -
8*a^8*b^2*c - 2*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 14*a^3*b^4*c^4 - 10*a^3*b^5*
c^3 - 24*a^3*b^6*c^2 - 27*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 59*a^4*b^4*c^3 + 7
*a^4*b^5*c^2 + 11*a^5*b^2*c^4 - 122*a^5*b^3*c^3 + 93*a^5*b^4*c^2 + 37*a^6*b
^2*c^3 - 99*a^6*b^3*c^2 + 23*a^7*b^2*c^2))/a^4))/a^2))/a^2)*1i)/a^2)/((1638
4*(b*c^7 - 4*b^2*c^6 + 6*b^3*c^5 - 4*b^4*c^4 + b^5*c^3 - 2*a*b^2*c^5 + 2*a*
b^3*c^4 - a*b^4*c^3 + a^2*b^2*c^4 + a*b*c^6))/a^4 + (b*((8192*tan(x/2)*(a*b
^8 + 5*b^8*c - b^9 + a^2*c^7 + a^3*c^6 + b^4*c^5 - 5*b^5*c^4 + 10*b^6*c^3 -
10*b^7*c^2 - 2*a*b^2*c^6 + 14*a*b^3*c^5 - 35*a*b^4*c^4 + 40*a*b^5*c^3 - 20
*a*b^6*c^2 - a^2*b*c^6 - 6*a^2*b^6*c + 10*a^2*b^2*c^5 - 20*a^2*b^3*c^4 + 5*
a^2*b^4*c^3 + 11*a^2*b^5*c^2 + 10*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 9*a^3*b^4*
c^2 - 2*a^4*b^2*c^3 + 2*a*b^7*c))/a^4 + (b*((8192*(6*a^2*b^8 - 3*a*b^9 - 4*
a^3*b^7 + a^4*b^6 + 3*a^4*c^6 + 2*a^5*c^5 - a^6*c^4 + 2*a*b^5*c^4 - 5*a*b^6
*c^3 + a*b^7*c^2 + 16*a^2*b^7*c + 8*a^3*b*c^6 - 38*a^3*b^6*c + 10*a^4*b*c^5
+ 23*a^4*b^5*c + 6*a^5*b*c^4 - 5*a^5*b^4*c - 10*a^2*b^3*c^5 + 25*a^2*b^4*c
^4 + 4*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 20*a^3*b^2*c^5 - 36*a^3*b^3*c^4 + 91*
a^3*b^4*c^3 - 3*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 55*a^4*b^3*c^3 + 57*a^4*b^4*
c^2 - 3*a^5*b^2*c^3 - 28*a^5*b^3*c^2 + 4*a^6*b^2*c^2 + 5*a*b^8*c))/a^4 + (b
((b((8192*(3*a^5*b^7 - 7*a^6*b^6 + 5*a^7*b^5 - a^8*b^4 + 12*a^7*c^5 + 20*
a^8*c^4 + 4*a^9*c^3 - 4*a^10*c^2 - 5*a^5*b^6*c + 8*a^6*b*c^5 - 15*a^6*b^5*c
+ 28*a^7*b*c^4 + 46*a^7*b^4*c + 64*a^8*b*c^3 - 31*a^8*b^3*c + 44*a^9*b*c^2
+ 5*a^9*b^2*c - 2*a^5*b^3*c^4 + 5*a^5*b^4*c^3 - a^5*b^5*c^2 - 23*a^6*b^2*c
^4 - 3*a^6*b^3*c^3 + 40*a^6*b^4*c^2 - 85*a^7*b^2*c^3 - 4*a^7*b^3*c^2 - 73*a
^8*b^2*c^2))/a^4 + (8192*b*tan(x/2)*(8*a^12*c + 2*a^6*b^7 - 6*a^7*b^6 + 8*a
^8*b^5 - 8*a^9*b^4 + 6*a^10*b^3 - 2*a^11*b^2 + 24*a^8*c^5 + 16*a^9*c^4 - 32
*a^10*c^3 - 16*a^11*c^2 - 2*a^6*b^6*c - 14*a^7*b^5*c - 8*a^8*b*c^4 + 46*a^8

*b^4*c + 88*a^9*b*c^3 - 50*a^9*b^3*c + 72*a^10*b*c^2 + 36*a^10*b^2*c + 2*a^6*b^4*c^3 - 2*a^6*b^5*c^2 - 14*a^7*b^2*c^4 + 10*a^7*b^3*c^3 + 24*a^7*b^4*c^2 - 68*a^8*b^2*c^3 + 2*a^8*b^3*c^2 - 80*a^9*b^2*c^2 - 24*a^11*b*c))/a^6))/a^2 + (8192*tan(x/2)*(6*a^3*b^8 - 2*a^2*b^9 - 8*a^4*b^7 + 8*a^5*b^6 - 6*a^6*b^5 + 2*a^7*b^4 + 10*a^5*c^6 + 6*a^6*c^5 - 2*a^7*c^4 + 2*a^8*c^3 + 2*a^2*b^8*c + 14*a^3*b^7*c - 50*a^4*b^6*c - 22*a^5*b*c^5 + 56*a^5*b^5*c + 12*a^6*b*c^4 - 38*a^6*b^4*c + 18*a^7*b*c^3 + 24*a^7*b^3*c - 8*a^8*b^2*c - 2*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 14*a^3*b^4*c^4 - 10*a^3*b^5*c^3 - 24*a^3*b^6*c^2 - 27*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 59*a^4*b^4*c^3 + 7*a^4*b^5*c^2 + 11*a^5*b^2*c^4 - 122*a^5*b^3*c^3 + 93*a^5*b^4*c^2 + 37*a^6*b^2*c^3 - 99*a^6*b^3*c^2 + 23*a^7*b^2*c^2))/a^4))/a^2))/a^2))/a^2 - (b*((8192*tan(x/2)*(a*b^8 + 5*b^8*c - b^9 + a^2*c^7 + a^3*c^6 + b^4*c^5 - 5*b^5*c^4 + 10*b^6*c^3 - 10*b^7*c^2 - 2*a*b^2*c^6 + 14*a*b^3*c^5 - 35*a*b^4*c^4 + 40*a*b^5*c^3 - 20*a*b^6*c^2 - a^2*b*c^6 - 6*a^2*b^6*c + 10*a^2*b^2*c^5 - 20*a^2*b^3*c^4 + 5*a^2*b^4*c^3 + 11*a^2*b^5*c^2 + 10*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 9*a^3*b^4*c^2 - 2*a^4*b^2*c^3 + 2*a*b^7*c))/a^4 - (b*((8192*(6*a^2*b^8 - 3*a*b^9 - 4*a^3*b^7 + a^4*b^6 + 3*a^4*c^6 + 2*a^5*c^5 - a^6*c^4 + 2*a*b^5*c^4 - 5*a*b^6*c^3 + a*b^7*c^2 + 16*a^2*b^7*c + 8*a^3*b*c^6 - 38*a^3*b^6*c + 10*a^4*b*c^5 + 23*a^4*b^5*c + 6*a^5*b*c^4 - 5*a^5*b^4*c - 10*a^2*b^3*c^5 + 25*a^2*b^4*c^4 + 4*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 20*a^3*b^2*c^5 - 36*a^3*b^3*c^4 + 91*a^3*b^4*c^3 - 3*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 55*a^4*b^3*c^3 + 57*a^4*b^4*c^2 - 3*a^5*b^2*c^3 - 28*a^5*b^3*c^2 + 4*a^6*b^2*c^2 + 5*a*b^8*c))/a^4 + (b*((b*((8192*(3*a^5*b^7 - 7*a^6*b^6 + 5*a^7*b^5 - a^8*b^4 + 12*a^7*c^5 + 20*a^8*c^4 + 4*a^9*c^3 - 4*a^10*c^2 - 5*a^5*b^6*c + 8*a^6*b*c^5 - 15*a^6*b^5*c + 28*a^7*b*c^4 + 46*a^7*b^4*c + 64*a^8*b*c^3 - 31*a^8*b^3*c + 44*a^9*b*c^2 + 5*a^9*b^2*c - 2*a^5*b^3*c^4 + 5*a^5*b^4*c^3 - a^5*b^5*c^2 - 23*a^6*b^2*c^4 - 3*a^6*b^3*c^3 + 40*a^6*b^4*c^2 - 85*a^7*b^2*c^3 - 4*a^7*b^3*c^2 - 73*a^8*b^2*c^2))/a^4 - (8192*b*tan(x/2)*(8*a^12*c + 2*a^6*b^7 - 6*a^7*b^6 + 8*a^8*b^5 - 8*a^9*b^4 + 6*a^10*b^3 - 2*a^11*b^2 + 24*a^8*c^5 + 16*a^9*c^4 - 32*a^10*c^3 - 16*a^11*c^2 - 2*a^6*b^6*c - 14*a^7*b^5*c - 8*a^8*b*c^4 + 46*a^8*b^4*c + 88*a^9*b*c^3 - 50*a^9*b^3*c + 72*a^10*b*c^2 + 36*a^10*b^2*c + 2*a^6*b^4*c^3 - 2*a^6*b^5*c^2 - 14*a^7*b^2*c^4 + 10*a^7*b^3*c^3 + 24*a^7*b^4*c^2 - 68*a^8*b^2*c^3 + 2*a^8*b^3*c^2 - 80*a^9*b^2*c^2 - 24*a^11*b*c))/a^6))/a^2 - (8192*tan(x/2)*(6*a^3*b^8 - 2*a^2*b^9 - 8*a^4*b^7 + 8*a^5*b^6 - 6*a^6*b^5 + 2*a^7*b^4 + 10*a^5*c^6 + 6*a^6*c^5 - 2*a^7*c^4 + 2*a^8*c^3 + 2*a^2*b^8*c + 14*a^3*b^7*c - 50*a^4*b^6*c - 22*a^5*b*c^5 + 56*a^5*b^5*c + 12*a^6*b*c^4 - 38*a^6*b^4*c + 18*a^7*b*c^3 + 24*a^7*b^3*c - 8*a^8*b^2*c - 2*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 14*a^3*b^4*c^4 - 10*a^3*b^5*c^3 - 24*a^3*b^6*c^2 - 27*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 59*a^4*b^4*c^3 + 7*a^4*b^5*c^2 + 11*a^5*b^2*c^4 - 122*a^5*b^3*c^3 + 93*a^5*b^4*c^2 + 37*a^6*b^2*c^3 - 99*a^6*b^3*c^2 + 23*a^7*b^2*c^2))/a^4))/a^2))/a^2))/a^2))*2i)/a^2 - atan((((((8192*(3*a^5*b^7 - 7*a^6*b^6 + 5*a^7*b^5 - a^8*b^4 + 12*a^7*c^5 + 20*a^8*c^4 + 4*a^9*c^3 - 4*a^10*c^2 - 5*a^5*b^6*c + 8*a^6*b*c^5 - 15*a^6*b^5*c + 28*a^7*b*c^4 + 46*a^7*b^4*c + 64*a^8*b*c^3 - 31*a^8*b^3*c + 44*a^9*b*c^2 + 5*a^9*b^2*c - 2*a^5*b^3*c^4 + 5*a^5*b^4*c^3 - a^5*b^5*c^2 - 23*a^6*b^2*c^4 - 3*a^6*b^3*c^3 + 40*a^6*b

```
^4*c^2 - 85*a^7*b^2*c^3 - 4*a^7*b^3*c^2 - 73*a^8*b^2*c^2))/a^4 - (8192*tan(
x/2)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^
2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^
2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1
/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/
2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b
^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2)*
(8*a^12*c + 2*a^6*b^7 - 6*a^7*b^6 + 8*a^8*b^5 - 8*a^9*b^4 + 6*a^10*b^3 - 2*
a^11*b^2 + 24*a^8*c^5 + 16*a^9*c^4 - 32*a^10*c^3 - 16*a^11*c^2 - 2*a^6*b^6*
c - 14*a^7*b^5*c - 8*a^8*b*c^4 + 46*a^8*b^4*c + 88*a^9*b*c^3 - 50*a^9*b^3*c
 + 72*a^10*b*c^2 + 36*a^10*b^2*c + 2*a^6*b^4*c^3 - 2*a^6*b^5*c^2 - 14*a^7*b
^2*c^4 + 10*a^7*b^3*c^3 + 24*a^7*b^4*c^2 - 68*a^8*b^2*c^3 + 2*a^8*b^3*c^2 -
 80*a^9*b^2*c^2 - 24*a^11*b*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(
-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b
^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3
*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) +
4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32
*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^
2*c^3 - 32*a^6*b^2*c^2)))^(1/2) - (8192*tan(x/2)*(6*a^3*b^8 - 2*a^2*b^9 - 8
*a^4*b^7 + 8*a^5*b^6 - 6*a^6*b^5 + 2*a^7*b^4 + 10*a^5*c^6 + 6*a^6*c^5 - 2*a
^7*c^4 + 2*a^8*c^3 + 2*a^2*b^8*c + 14*a^3*b^7*c - 50*a^4*b^6*c - 22*a^5*b*c
^5 + 56*a^5*b^5*c + 12*a^6*b*c^4 - 38*a^6*b^4*c + 18*a^7*b*c^3 + 24*a^7*b^3
*c - 8*a^8*b^2*c - 2*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 14*a^3*b^4*c^4 - 10*a^3*
b^5*c^3 - 24*a^3*b^6*c^2 - 27*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 59*a^4*b^4*c^3
 + 7*a^4*b^5*c^2 + 11*a^5*b^2*c^4 - 122*a^5*b^3*c^3 + 93*a^5*b^4*c^2 + 37*a
^6*b^2*c^3 - 99*a^6*b^3*c^2 + 23*a^7*b^2*c^2))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*
a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2
*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) -
 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b
^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 +
16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4
*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) + (8192*(6*a^2*b^8 - 3*a*b^9
 - 4*a^3*b^7 + a^4*b^6 + 3*a^4*c^6 + 2*a^5*c^5 - a^6*c^4 + 2*a*b^5*c^4 - 5*
a*b^6*c^3 + a*b^7*c^2 + 16*a^2*b^7*c + 8*a^3*b*c^6 - 38*a^3*b^6*c + 10*a^4*
b*c^5 + 23*a^4*b^5*c + 6*a^5*b*c^4 - 5*a^5*b^4*c - 10*a^2*b^3*c^5 + 25*a^2*
b^4*c^4 + 4*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 20*a^3*b^2*c^5 - 36*a^3*b^3*c^4
+ 91*a^3*b^4*c^3 - 3*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 55*a^4*b^3*c^3 + 57*a^4
*b^4*c^2 - 3*a^5*b^2*c^3 - 28*a^5*b^3*c^2 + 4*a^6*b^2*c^2 + 5*a*b^8*c))/a^4
)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 +
 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(
-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2)
 - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))
/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*
c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) - (
8192*tan(x/2)*(a*b^8 + 5*b^8*c - b^9 + a^2*c^7 + a^3*c^6 + b^4*c^5 - 5*b^5*
```

c^4 + 10*b^6*c^3 - 10*b^7*c^2 - 2*a*b^2*c^6 + 14*a*b^3*c^5 - 35*a*b^4*c^4 + 40*a*b^5*c^3 - 20*a*b^6*c^2 - a^2*b*c^6 - 6*a^2*b^6*c + 10*a^2*b^2*c^5 - 20*a^2*b^3*c^4 + 5*a^2*b^4*c^3 + 11*a^2*b^5*c^2 + 10*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 9*a^3*b^4*c^2 - 2*a^4*b^2*c^3 + 2*a*b^7*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2)*1i - (((((8192*(3*a^5*b^7 - 7*a^6*b^6 + 5*a^7*b^5 - a^8*b^4 + 12*a^7*c^5 + 20*a^8*c^4 + 4*a^9*c^3 - 4*a^10*c^2 - 5*a^5*b^6*c + 8*a^6*b*c^5 - 15*a^6*b^5*c + 28*a^7*b*c^4 + 46*a^7*b^4*c + 64*a^8*b*c^3 - 31*a^8*b^3*c + 44*a^9*b*c^2 + 5*a^9*b^2*c - 2*a^5*b^3*c^4 + 5*a^5*b^4*c^3 - a^5*b^5*c^2 - 23*a^6*b^2*c^4 - 3*a^6*b^3*c^3 + 40*a^6*b^4*c^2 - 85*a^7*b^2*c^3 - 4*a^7*b^3*c^2 - 73*a^8*b^2*c^2))/a^4 + (8192*tan(x/2)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2)*(8*a^12*c + 2*a^6*b^7 - 6*a^7*b^6 + 8*a^8*b^5 - 8*a^9*b^4 + 6*a^10*b^3 - 2*a^11*b^2 + 24*a^8*c^5 + 16*a^9*c^4 - 32*a^10*c^3 - 16*a^11*c^2 - 2*a^6*b^6*c - 14*a^7*b^5*c - 8*a^8*b*c^4 + 46*a^8*b^4*c + 88*a^9*b*c^3 - 50*a^9*b^3*c + 72*a^10*b*c^2 + 36*a^10*b^2*c + 2*a^6*b^4*c^3 - 2*a^6*b^5*c^2 - 14*a^7*b^2*c^4 + 10*a^7*b^3*c^3 + 24*a^7*b^4*c^2 - 68*a^8*b^2*c^3 + 2*a^8*b^3*c^2 - 80*a^9*b^2*c^2 - 24*a^11*b*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) + (8192*tan(x/2)*(6*a^3*b^8 - 2*a^2*b^9 - 8*a^4*b^7 + 8*a^5*b^6 - 6*a^6*b^5 + 2*a^7*b^4 + 10*a^5*c^6 + 6*a^6*c^5 - 2*a^7*c^4 + 2*a^8*c^3 + 2*a^2*b^8*c + 14*a^3*b^7*c - 50*a^4*b^6*c - 22*a^5*b*c^5 + 56*a^5*b^5*c + 12*a^6*b*c^4 - 38*a^6*b^4*c + 18*a^7*b*c^3 + 24*a^7*b^3*c - 8*a^8*b^2*c - 2*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 14*a^3*b^4*c^4 - 10*a^3*b^5*c^3 - 24*a^3*b^6*c^2 - 27*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 59*a^4*b^4*c^3 + 7*a^4*b^5*c^2 + 11*a^5*b^2*c^4 - 122*a^5*b^3*c^3 + 93*a^5*b^4*c^2 + 37*a^6*b^2*c^3 - 99*a^6*b^3*c^2 + 23*a^7*b^2*c^2))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c

+ a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) + (8192*(6*a^2*b^8 - 3*a*b^9 - 4*a^3*b^7 + a^4*b^6 + 3*a^4*c^6 + 2*a^5*c^5 - a^6*c^4 + 2*a*b^5*c^4 - 5*a*b^6*c^3 + a*b^7*c^2 + 16*a^2*b^7*c + 8*a^3*b*c^6 - 38*a^3*b^6*c + 10*a^4*b*c^5 + 23*a^4*b^5*c + 6*a^5*b*c^4 - 5*a^5*b^4*c - 10*a^2*b^3*c^5 + 25*a^2*b^4*c^4 + 4*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 20*a^3*b^2*c^5 - 36*a^3*b^3*c^4 + 91*a^3*b^4*c^3 - 3*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 55*a^4*b^3*c^3 + 57*a^4*b^4*c^2 - 3*a^5*b^2*c^3 - 28*a^5*b^3*c^2 + 4*a^6*b^2*c^2 + 5*a*b^8*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) + (8192*tan(x/2)*(a*b^8 + 5*b^8*c - b^9 + a^2*c^7 + a^3*c^6 + b^4*c^5 - 5*b^5*c^4 + 10*b^6*c^3 - 10*b^7*c^2 - 2*a*b^2*c^6 + 14*a*b^3*c^5 - 35*a*b^4*c^4 + 40*a*b^5*c^3 - 20*a*b^6*c^2 - a^2*b*c^6 - 6*a^2*b^6*c + 10*a^2*b^2*c^5 - 20*a^2*b^3*c^4 + 5*a^2*b^4*c^3 + 11*a^2*b^5*c^2 + 10*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 9*a^3*b^4*c^2 - 2*a^4*b^2*c^3 + 2*a*b^7*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2)*1i)/((((((8192*(3*a^5*b^7 - 7*a^6*b^6 + 5*a^7*b^5 - a^8*b^4 + 12*a^7*c^5 + 20*a^8*c^4 + 4*a^9*c^3 - 4*a^10*c^2 - 5*a^5*b^6*c + 8*a^6*b*c^5 - 15*a^6*b^5*c + 28*a^7*b*c^4 + 46*a^7*b^4*c + 64*a^8*b*c^3 - 31*a^8*b^3*c + 44*a^9*b*c^2 + 5*a^9*b^2*c - 2*a^5*b^3*c^4 + 5*a^5*b^4*c^3 - a^5*b^5*c^2 - 23*a^6*b^2*c^4 - 3*a^6*b^3*c^3 + 40*a^6*b^4*c^2 - 85*a^7*b^2*c^3 - 4*a^7*b^3*c^2 - 73*a^8*b^2*c^2))/a^4 - (8192*tan(x/2)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2)*(8*a^12*c + 2*a^6*b^7 - 6*a^7*b^6 + 8*a^8*b^5 - 8*a^9*b^4 + 6*a^10*b^3 - 2*a^11*b^2 + 24*a^8*c^5 + 16*a^9*c^4 - 32*a^10*c^3 - 16*a^11*c^2 - 2*a^6*b^6*c - 14*a^7*b^5*c - 8*a^8*b*c^4 + 46*a^8*b^4*c + 88*a^9*b*c^3 - 50*a^9*b^3*c + 72*a^10*b*c^2 + 36*a^10*b^2*c + 2*a^6*b^4*c^3 - 2*a^6*b^5*c^2 - 14*a^7*b^2*c^4 + 10*a^7*b^3*c^3 + 24*a^7*b^4*c^2 - 68*a^8*b^2*c^3 + 2*a^8*b^3*c^2 - 80*a^9*b^2*c^2 - 24*a^11*b*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6

+ 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*
b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) - (8192*tan(x/2)*(6*a^3*b
^8 - 2*a^2*b^9 - 8*a^4*b^7 + 8*a^5*b^6 - 6*a^6*b^5 + 2*a^7*b^4 + 10*a^5*c^6
+ 6*a^6*c^5 - 2*a^7*c^4 + 2*a^8*c^3 + 2*a^2*b^8*c + 14*a^3*b^7*c - 50*a^4*
b^6*c - 22*a^5*b*c^5 + 56*a^5*b^5*c + 12*a^6*b*c^4 - 38*a^6*b^4*c + 18*a^7*
b*c^3 + 24*a^7*b^3*c - 8*a^8*b^2*c - 2*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 14*a^3
*b^4*c^4 - 10*a^3*b^5*c^3 - 24*a^3*b^6*c^2 - 27*a^4*b^2*c^5 + 15*a^4*b^3*c^
4 + 59*a^4*b^4*c^3 + 7*a^4*b^5*c^2 + 11*a^5*b^2*c^4 - 122*a^5*b^3*c^3 + 93*
a^5*b^4*c^2 + 37*a^6*b^2*c^3 - 99*a^6*b^3*c^2 + 23*a^7*b^2*c^2))/a^4)*(-(b^
8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^
4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c
- b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*
b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^
6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a
^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) + (8192*(6
*a^2*b^8 - 3*a*b^9 - 4*a^3*b^7 + a^4*b^6 + 3*a^4*c^6 + 2*a^5*c^5 - a^6*c^4
+ 2*a*b^5*c^4 - 5*a*b^6*c^3 + a*b^7*c^2 + 16*a^2*b^7*c + 8*a^3*b*c^6 - 38*a
^3*b^6*c + 10*a^4*b*c^5 + 23*a^4*b^5*c + 6*a^5*b*c^4 - 5*a^5*b^4*c - 10*a^2
*b^3*c^5 + 25*a^2*b^4*c^4 + 4*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 20*a^3*b^2*c^5
- 36*a^3*b^3*c^4 + 91*a^3*b^4*c^3 - 3*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 55*a^
4*b^3*c^3 + 57*a^4*b^4*c^2 - 3*a^5*b^2*c^3 - 28*a^5*b^3*c^2 + 4*a^6*b^2*c^2
+ 5*a*b^8*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)
^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b
^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a
*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*
c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8
*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^
2*c^2)))^(1/2) - (8192*tan(x/2)*(a*b^8 + 5*b^8*c - b^9 + a^2*c^7 + a^3*c^6
+ b^4*c^5 - 5*b^5*c^4 + 10*b^6*c^3 - 10*b^7*c^2 - 2*a*b^2*c^6 + 14*a*b^3*c^
5 - 35*a*b^4*c^4 + 40*a*b^5*c^3 - 20*a*b^6*c^2 - a^2*b*c^6 - 6*a^2*b^6*c +
10*a^2*b^2*c^5 - 20*a^2*b^3*c^4 + 5*a^2*b^4*c^3 + 11*a^2*b^5*c^2 + 10*a^3*b
^2*c^4 - 18*a^3*b^3*c^3 + 9*a^3*b^4*c^2 - 2*a^4*b^2*c^3 + 2*a*b^7*c))/a^4)*
(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8
*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(
4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) -
2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(
2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c
- 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) + (((
((8192*(3*a^5*b^7 - 7*a^6*b^6 + 5*a^7*b^5 - a^8*b^4 + 12*a^7*c^5 + 20*a^8*c
^4 + 4*a^9*c^3 - 4*a^10*c^2 - 5*a^5*b^6*c + 8*a^6*b*c^5 - 15*a^6*b^5*c + 28
*a^7*b*c^4 + 46*a^7*b^4*c + 64*a^8*b*c^3 - 31*a^8*b^3*c + 44*a^9*b*c^2 + 5*
a^9*b^2*c - 2*a^5*b^3*c^4 + 5*a^5*b^4*c^3 - a^5*b^5*c^2 - 23*a^6*b^2*c^4 -
3*a^6*b^3*c^3 + 40*a^6*b^4*c^2 - 85*a^7*b^2*c^3 - 4*a^7*b^3*c^2 - 73*a^8*b^
2*c^2))/a^4 + (8192*tan(x/2)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c
- b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 -

38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c
^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*
c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3
 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 -
32*a^6*b^2*c^2)))^(1/2)*(8*a^12*c + 2*a^6*b^7 - 6*a^7*b^6 + 8*a^8*b^5 - 8*a
^9*b^4 + 6*a^10*b^3 - 2*a^11*b^2 + 24*a^8*c^5 + 16*a^9*c^4 - 32*a^10*c^3 -
16*a^11*c^2 - 2*a^6*b^6*c - 14*a^7*b^5*c - 8*a^8*b*c^4 + 46*a^8*b^4*c + 88*
a^9*b*c^3 - 50*a^9*b^3*c + 72*a^10*b*c^2 + 36*a^10*b^2*c + 2*a^6*b^4*c^3 -
2*a^6*b^5*c^2 - 14*a^7*b^2*c^4 + 10*a^7*b^3*c^3 + 24*a^7*b^4*c^2 - 68*a^8*b
^2*c^3 + 2*a^8*b^3*c^2 - 80*a^9*b^2*c^2 - 24*a^11*b*c))/a^4)*(-(b^8 + 8*a^3
*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 1
8*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3
)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(
4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a
^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c
 + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) + (8192*tan(x/2)*(6
*a^3*b^8 - 2*a^2*b^9 - 8*a^4*b^7 + 8*a^5*b^6 - 6*a^6*b^5 + 2*a^7*b^4 + 10*a
^5*c^6 + 6*a^6*c^5 - 2*a^7*c^4 + 2*a^8*c^3 + 2*a^2*b^8*c + 14*a^3*b^7*c - 5
0*a^4*b^6*c - 22*a^5*b*c^5 + 56*a^5*b^5*c + 12*a^6*b*c^4 - 38*a^6*b^4*c + 1
8*a^7*b*c^3 + 24*a^7*b^3*c - 8*a^8*b^2*c - 2*a^2*b^6*c^3 + 2*a^2*b^7*c^2 +
14*a^3*b^4*c^4 - 10*a^3*b^5*c^3 - 24*a^3*b^6*c^2 - 27*a^4*b^2*c^5 + 15*a^4*
b^3*c^4 + 59*a^4*b^4*c^3 + 7*a^4*b^5*c^2 + 11*a^5*b^2*c^4 - 122*a^5*b^3*c^3
 + 93*a^5*b^4*c^2 + 37*a^6*b^2*c^3 - 99*a^6*b^3*c^2 + 23*a^7*b^2*c^2))/a^4)
*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 +
8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-
(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2)
 - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/
(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c
 - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) + (8
192*(6*a^2*b^8 - 3*a*b^9 - 4*a^3*b^7 + a^4*b^6 + 3*a^4*c^6 + 2*a^5*c^5 - a^
6*c^4 + 2*a*b^5*c^4 - 5*a*b^6*c^3 + a*b^7*c^2 + 16*a^2*b^7*c + 8*a^3*b*c^6
 - 38*a^3*b^6*c + 10*a^4*b*c^5 + 23*a^4*b^5*c + 6*a^5*b*c^4 - 5*a^5*b^4*c -
10*a^2*b^3*c^5 + 25*a^2*b^4*c^4 + 4*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 20*a^3*b
^2*c^5 - 36*a^3*b^3*c^4 + 91*a^3*b^4*c^3 - 3*a^3*b^5*c^2 - 24*a^4*b^2*c^4 -
 55*a^4*b^3*c^3 + 57*a^4*b^4*c^2 - 3*a^5*b^2*c^3 - 28*a^5*b^3*c^2 + 4*a^6*b
^2*c^2 + 5*a*b^8*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b
^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38
*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*
(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(
-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 +
16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*
a^6*b^2*c^2)))^(1/2) + (8192*tan(x/2)*(a*b^8 + 5*b^8*c - b^9 + a^2*c^7 + a^
3*c^6 + b^4*c^5 - 5*b^5*c^4 + 10*b^6*c^3 - 10*b^7*c^2 - 2*a*b^2*c^6 + 14*a*
b^3*c^5 - 35*a*b^4*c^4 + 40*a*b^5*c^3 - 20*a*b^6*c^2 - a^2*b*c^6 - 6*a^2*b^
6*c + 10*a^2*b^2*c^5 - 20*a^2*b^3*c^4 + 5*a^2*b^4*c^3 + 11*a^2*b^5*c^2 + 10

*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 9*a^3*b^4*c^2 - 2*a^4*b^2*c^3 + 2*a*b^7*c))
/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c
^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c
^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(
1/2) - 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1
/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*
b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2)
 + (16384*(b*c^7 - 4*b^2*c^6 + 6*b^3*c^5 - 4*b^4*c^4 + b^5*c^3 - 2*a*b^2*c^
5 + 2*a*b^3*c^4 - a*b^4*c^3 + a^2*b^2*c^4 + a*b*c^6))/a^4))*(-(b^8 + 8*a^3*
c^5 + 8*a^4*c^4 - b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18
*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 + b^3*c^2*(-(4*a*c - b^2)^3)
^(1/2) - 10*a*b^6*c - 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) - 2*a*b*c^3*(-(4
*a*c - b^2)^3)^(1/2) + 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^
4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c +
 a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2)*2i - atan((((((8192
*(3*a^5*b^7 - 7*a^6*b^6 + 5*a^7*b^5 - a^8*b^4 + 12*a^7*c^5 + 20*a^8*c^4 + 4
*a^9*c^3 - 4*a^10*c^2 - 5*a^5*b^6*c + 8*a^6*b*c^5 - 15*a^6*b^5*c + 28*a^7*b
*c^4 + 46*a^7*b^4*c + 64*a^8*b*c^3 - 31*a^8*b^3*c + 44*a^9*b*c^2 + 5*a^9*b^
2*c - 2*a^5*b^3*c^4 + 5*a^5*b^4*c^3 - a^5*b^5*c^2 - 23*a^6*b^2*c^4 - 3*a^6*
b^3*c^3 + 40*a^6*b^4*c^2 - 85*a^7*b^2*c^3 - 4*a^7*b^3*c^2 - 73*a^8*b^2*c^2)
)/a^4 - (8192*tan(x/2)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)
^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^
3*b^2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(
4*a*c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4
*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*
a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6
*b^2*c^2)))^(1/2)*(8*a^12*c + 2*a^6*b^7 - 6*a^7*b^6 + 8*a^8*b^5 - 8*a^9*b^4
 + 6*a^10*b^3 - 2*a^11*b^2 + 24*a^8*c^5 + 16*a^9*c^4 - 32*a^10*c^3 - 16*a^1
1*c^2 - 2*a^6*b^6*c - 14*a^7*b^5*c - 8*a^8*b*c^4 + 46*a^8*b^4*c + 88*a^9*b*
c^3 - 50*a^9*b^3*c + 72*a^10*b*c^2 + 36*a^10*b^2*c + 2*a^6*b^4*c^3 - 2*a^6*
b^5*c^2 - 14*a^7*b^2*c^4 + 10*a^7*b^3*c^3 + 24*a^7*b^4*c^2 - 68*a^8*b^2*c^3
 + 2*a^8*b^3*c^2 - 80*a^9*b^2*c^2 - 24*a^11*b*c))/a^4)*(-(b^8 + 8*a^3*c^5 +
 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*
b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2
) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c
- b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6
 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*
b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) - (8192*tan(x/2)*(6*a^3*b
^8 - 2*a^2*b^9 - 8*a^4*b^7 + 8*a^5*b^6 - 6*a^6*b^5 + 2*a^7*b^4 + 10*a^5*c^6
 + 6*a^6*c^5 - 2*a^7*c^4 + 2*a^8*c^3 + 2*a^2*b^8*c + 14*a^3*b^7*c - 50*a^4*
b^6*c - 22*a^5*b*c^5 + 56*a^5*b^5*c + 12*a^6*b*c^4 - 38*a^6*b^4*c + 18*a^7*
b*c^3 + 24*a^7*b^3*c - 8*a^8*b^2*c - 2*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 14*a^3
*b^4*c^4 - 10*a^3*b^5*c^3 - 24*a^3*b^6*c^2 - 27*a^4*b^2*c^5 + 15*a^4*b^3*c^
4 + 59*a^4*b^4*c^3 + 7*a^4*b^5*c^2 + 11*a^5*b^2*c^4 - 122*a^5*b^3*c^3 + 93*
a^5*b^4*c^2 + 37*a^6*b^2*c^3 - 99*a^6*b^3*c^2 + 23*a^7*b^2*c^2))/a^4)*(-(b^

8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^
4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2*(-(4*a*c
 - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 2*a*
b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^
6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a
^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) + (8192*(6
*a^2*b^8 - 3*a*b^9 - 4*a^3*b^7 + a^4*b^6 + 3*a^4*c^6 + 2*a^5*c^5 - a^6*c^4
 + 2*a*b^5*c^4 - 5*a*b^6*c^3 + a*b^7*c^2 + 16*a^2*b^7*c + 8*a^3*b*c^6 - 38*a
^3*b^6*c + 10*a^4*b*c^5 + 23*a^4*b^5*c + 6*a^5*b*c^4 - 5*a^5*b^4*c - 10*a^2
*b^3*c^5 + 25*a^2*b^4*c^4 + 4*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 20*a^3*b^2*c^5
 - 36*a^3*b^3*c^4 + 91*a^3*b^4*c^3 - 3*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 55*a^
4*b^3*c^3 + 57*a^4*b^4*c^2 - 3*a^5*b^2*c^3 - 28*a^5*b^3*c^2 + 4*a^6*b^2*c^2
 + 5*a*b^8*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)
^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b
^2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a
*c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*
c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8
*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^
2*c^2)))^(1/2) - (8192*tan(x/2)*(a*b^8 + 5*b^8*c - b^9 + a^2*c^7 + a^3*c^6
 + b^4*c^5 - 5*b^5*c^4 + 10*b^6*c^3 - 10*b^7*c^2 - 2*a*b^2*c^6 + 14*a*b^3*c^
5 - 35*a*b^4*c^4 + 40*a*b^5*c^3 - 20*a*b^6*c^2 - a^2*b*c^6 - 6*a^2*b^6*c +
10*a^2*b^2*c^5 - 20*a^2*b^3*c^4 + 5*a^2*b^4*c^3 + 11*a^2*b^5*c^2 + 10*a^3*b
^2*c^4 - 18*a^3*b^3*c^3 + 9*a^3*b^4*c^2 - 2*a^4*b^2*c^3 + 2*a*b^7*c))/a^4)*
(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8
*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2*(-(
4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) +
 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(
2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c
 - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2)*1i -
(((((8192*(3*a^5*b^7 - 7*a^6*b^6 + 5*a^7*b^5 - a^8*b^4 + 12*a^7*c^5 + 20*a^
8*c^4 + 4*a^9*c^3 - 4*a^10*c^2 - 5*a^5*b^6*c + 8*a^6*b*c^5 - 15*a^6*b^5*c +
 28*a^7*b*c^4 + 46*a^7*b^4*c + 64*a^8*b*c^3 - 31*a^8*b^3*c + 44*a^9*b*c^2 +
 5*a^9*b^2*c - 2*a^5*b^3*c^4 + 5*a^5*b^4*c^3 - a^5*b^5*c^2 - 23*a^6*b^2*c^4
 - 3*a^6*b^3*c^3 + 40*a^6*b^4*c^2 - 85*a^7*b^2*c^3 - 4*a^7*b^3*c^2 - 73*a^8
*b^2*c^2))/a^4 + (8192*tan(x/2)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a
*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^
2 - 38*a^3*b^2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*
b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b
^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*
c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3
 - 32*a^6*b^2*c^2)))^(1/2)*(8*a^12*c + 2*a^6*b^7 - 6*a^7*b^6 + 8*a^8*b^5 -
8*a^9*b^4 + 6*a^10*b^3 - 2*a^11*b^2 + 24*a^8*c^5 + 16*a^9*c^4 - 32*a^10*c^3
 - 16*a^11*c^2 - 2*a^6*b^6*c - 14*a^7*b^5*c - 8*a^8*b*c^4 + 46*a^8*b^4*c +
88*a^9*b*c^3 - 50*a^9*b^3*c + 72*a^10*b*c^2 + 36*a^10*b^2*c + 2*a^6*b^4*c^3
 - 2*a^6*b^5*c^2 - 14*a^7*b^2*c^4 + 10*a^7*b^3*c^3 + 24*a^7*b^4*c^2 - 68*a^

8*b^2*c^3 + 2*a^8*b^3*c^2 - 80*a^9*b^2*c^2 - 24*a^11*b*c))/a^4)*(-(b^8 + 8*
a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3
- 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2*(-(4*a*c - b^2
)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 2*a*b*c^3*
(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4
- a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2
*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) + (8192*tan(x/2)
*(6*a^3*b^8 - 2*a^2*b^9 - 8*a^4*b^7 + 8*a^5*b^6 - 6*a^6*b^5 + 2*a^7*b^4 + 1
0*a^5*c^6 + 6*a^6*c^5 - 2*a^7*c^4 + 2*a^8*c^3 + 2*a^2*b^8*c + 14*a^3*b^7*c
- 50*a^4*b^6*c - 22*a^5*b*c^5 + 56*a^5*b^5*c + 12*a^6*b*c^4 - 38*a^6*b^4*c
+ 18*a^7*b*c^3 + 24*a^7*b^3*c - 8*a^8*b^2*c - 2*a^2*b^6*c^3 + 2*a^2*b^7*c^2
+ 14*a^3*b^4*c^4 - 10*a^3*b^5*c^3 - 24*a^3*b^6*c^2 - 27*a^4*b^2*c^5 + 15*a
^4*b^3*c^4 + 59*a^4*b^4*c^3 + 7*a^4*b^5*c^2 + 11*a^5*b^2*c^4 - 122*a^5*b^3*
c^3 + 93*a^5*b^4*c^2 + 37*a^6*b^2*c^3 - 99*a^6*b^3*c^2 + 23*a^7*b^2*c^2))/a
^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2
+ 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2
*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/
2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2
))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^
4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) +
(8192*(6*a^2*b^8 - 3*a*b^9 - 4*a^3*b^7 + a^4*b^6 + 3*a^4*c^6 + 2*a^5*c^5 -
a^6*c^4 + 2*a*b^5*c^4 - 5*a*b^6*c^3 + a*b^7*c^2 + 16*a^2*b^7*c + 8*a^3*b*c
^6 - 38*a^3*b^6*c + 10*a^4*b*c^5 + 23*a^4*b^5*c + 6*a^5*b*c^4 - 5*a^5*b^4*c
- 10*a^2*b^3*c^5 + 25*a^2*b^4*c^4 + 4*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 20*a^
3*b^2*c^5 - 36*a^3*b^3*c^4 + 91*a^3*b^4*c^3 - 3*a^3*b^5*c^2 - 24*a^4*b^2*c^
4 - 55*a^4*b^3*c^3 + 57*a^4*b^4*c^2 - 3*a^5*b^2*c^3 - 28*a^5*b^3*c^2 + 4*a^
6*b^2*c^2 + 5*a*b^8*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c
- b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 -
38*a^3*b^2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c
^2*(-(4*a*c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*
c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3
+ 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 -
32*a^6*b^2*c^2)))^(1/2) + (8192*tan(x/2)*(a*b^8 + 5*b^8*c - b^9 + a^2*c^7 +
a^3*c^6 + b^4*c^5 - 5*b^5*c^4 + 10*b^6*c^3 - 10*b^7*c^2 - 2*a*b^2*c^6 + 14
*a*b^3*c^5 - 35*a*b^4*c^4 + 40*a*b^5*c^3 - 20*a*b^6*c^2 - a^2*b*c^6 - 6*a^2
*b^6*c + 10*a^2*b^2*c^5 - 20*a^2*b^3*c^4 + 5*a^2*b^4*c^3 + 11*a^2*b^5*c^2 +
10*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 9*a^3*b^4*c^2 - 2*a^4*b^2*c^3 + 2*a*b^7*
c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^
6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^
3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3
)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)
^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a
^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1
/2)*1i)/((((((8192*(3*a^5*b^7 - 7*a^6*b^6 + 5*a^7*b^5 - a^8*b^4 + 12*a^7*c^
5 + 20*a^8*c^4 + 4*a^9*c^3 - 4*a^10*c^2 - 5*a^5*b^6*c + 8*a^6*b*c^5 - 15*a^

6*b^5*c + 28*a^7*b*c^4 + 46*a^7*b^4*c + 64*a^8*b*c^3 - 31*a^8*b^3*c + 44*a^9*b*c^2 + 5*a^9*b^2*c - 2*a^5*b^3*c^4 + 5*a^5*b^4*c^3 - a^5*b^5*c^2 - 23*a^6*b^2*c^4 - 3*a^6*b^3*c^3 + 40*a^6*b^4*c^2 - 85*a^7*b^2*c^3 - 4*a^7*b^3*c^2 - 73*a^8*b^2*c^2))/a^4 - (8192*tan(x/2)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2)*(8*a^12*c + 2*a^6*b^7 - 6*a^7*b^6 + 8*a^8*b^5 - 8*a^9*b^4 + 6*a^10*b^3 - 2*a^11*b^2 + 24*a^8*c^5 + 16*a^9*c^4 - 32*a^10*c^3 - 16*a^11*c^2 - 2*a^6*b^6*c - 14*a^7*b^5*c - 8*a^8*b*c^4 + 46*a^8*b^4*c + 88*a^9*b*c^3 - 50*a^9*b^3*c + 72*a^10*b*c^2 + 36*a^10*b^2*c + 2*a^6*b^4*c^3 - 2*a^6*b^5*c^2 - 14*a^7*b^2*c^4 + 10*a^7*b^3*c^3 + 24*a^7*b^4*c^2 - 68*a^8*b^2*c^3 + 2*a^8*b^3*c^2 - 80*a^9*b^2*c^2 - 24*a^11*b*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) - (8192*tan(x/2)*(6*a^3*b^8 - 2*a^2*b^9 - 8*a^4*b^7 + 8*a^5*b^6 - 6*a^6*b^5 + 2*a^7*b^4 + 10*a^5*c^6 + 6*a^6*c^5 - 2*a^7*c^4 + 2*a^8*c^3 + 2*a^2*b^8*c + 14*a^3*b^7*c - 50*a^4*b^6*c - 22*a^5*b*c^5 + 56*a^5*b^5*c + 12*a^6*b*c^4 - 38*a^6*b^4*c + 18*a^7*b*c^3 + 24*a^7*b^3*c - 8*a^8*b^2*c - 2*a^2*b^6*c^3 + 2*a^2*b^7*c^2 + 14*a^3*b^4*c^4 - 10*a^3*b^5*c^3 - 24*a^3*b^6*c^2 - 27*a^4*b^2*c^5 + 15*a^4*b^3*c^4 + 59*a^4*b^4*c^3 + 7*a^4*b^5*c^2 + 11*a^5*b^2*c^4 - 122*a^5*b^3*c^3 + 93*a^5*b^4*c^2 + 37*a^6*b^2*c^3 - 99*a^6*b^3*c^2 + 23*a^7*b^2*c^2))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) + (8192*(6*a^2*b^8 - 3*a*b^9 - 4*a^3*b^7 + a^4*b^6 + 3*a^4*c^6 + 2*a^5*c^5 - a^6*c^4 + 2*a*b^5*c^4 - 5*a*b^6*c^3 + a*b^7*c^2 + 16*a^2*b^7*c + 8*a^3*b*c^6 - 38*a^3*b^6*c + 10*a^4*b*c^5 + 23*a^4*b^5*c + 6*a^5*b*c^4 - 5*a^5*b^4*c - 10*a^2*b^3*c^5 + 25*a^2*b^4*c^4 + 4*a^2*b^5*c^3 - 41*a^2*b^6*c^2 - 20*a^3*b^2*c^5 - 36*a^3*b^3*c^4 + 91*a^3*b^4*c^3 - 3*a^3*b^5*c^2 - 24*a^4*b^2*c^4 - 55*a^4*b^3*c^3 + 57*a^4*b^4*c^2 - 3*a^5*b^2*c^3 - 28*a^5*b^3*c^2 + 4*a^6*b^2*c^2 + 5*a*b^8*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 3

2*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b
^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) - (8192*tan(x/2)*(a*b^8 + 5*b^8*c - b^9 +
a^2*c^7 + a^3*c^6 + b^4*c^5 - 5*b^5*c^4 + 10*b^6*c^3 - 10*b^7*c^2 - 2*a*b^2
*c^6 + 14*a*b^3*c^5 - 35*a*b^4*c^4 + 40*a*b^5*c^3 - 20*a*b^6*c^2 - a^2*b*c^
6 - 6*a^2*b^6*c + 10*a^2*b^2*c^5 - 20*a^2*b^3*c^4 + 5*a^2*b^4*c^3 + 11*a^2*
b^5*c^2 + 10*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 9*a^3*b^4*c^2 - 2*a^4*b^2*c^3 +
2*a*b^7*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(
1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2
*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c
- b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c
- b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c
^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*
c^2)))^(1/2) + (((((8192*(3*a^5*b^7 - 7*a^6*b^6 + 5*a^7*b^5 - a^8*b^4 + 12*
a^7*c^5 + 20*a^8*c^4 + 4*a^9*c^3 - 4*a^10*c^2 - 5*a^5*b^6*c + 8*a^6*b*c^5 -
15*a^6*b^5*c + 28*a^7*b*c^4 + 46*a^7*b^4*c + 64*a^8*b*c^3 - 31*a^8*b^3*c +
44*a^9*b*c^2 + 5*a^9*b^2*c - 2*a^5*b^3*c^4 + 5*a^5*b^4*c^3 - a^5*b^5*c^2 -
23*a^6*b^2*c^4 - 3*a^6*b^3*c^3 + 40*a^6*b^4*c^2 - 85*a^7*b^2*c^3 - 4*a^7*b
^3*c^2 - 73*a^8*b^2*c^2))/a^4 + (8192*tan(x/2)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c
^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4
+ 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a
*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3
)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^
6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2
- 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2)*(8*a^12*c + 2*a^6*b^7 - 6*a^7*b^6
+ 8*a^8*b^5 - 8*a^9*b^4 + 6*a^10*b^3 - 2*a^11*b^2 + 24*a^8*c^5 + 16*a^9*c^
4 - 32*a^10*c^3 - 16*a^11*c^2 - 2*a^6*b^6*c - 14*a^7*b^5*c - 8*a^8*b*c^4 +
46*a^8*b^4*c + 88*a^9*b*c^3 - 50*a^9*b^3*c + 72*a^10*b*c^2 + 36*a^10*b^2*c
+ 2*a^6*b^4*c^3 - 2*a^6*b^5*c^2 - 14*a^7*b^2*c^4 + 10*a^7*b^3*c^3 + 24*a^7*
b^4*c^2 - 68*a^8*b^2*c^3 + 2*a^8*b^3*c^2 - 80*a^9*b^2*c^2 - 24*a^11*b*c))/a
^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2
+ 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2
*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/
2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2
))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^
4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) +
(8192*tan(x/2)*(6*a^3*b^8 - 2*a^2*b^9 - 8*a^4*b^7 + 8*a^5*b^6 - 6*a^6*b^5
+ 2*a^7*b^4 + 10*a^5*c^6 + 6*a^6*c^5 - 2*a^7*c^4 + 2*a^8*c^3 + 2*a^2*b^8*c
+ 14*a^3*b^7*c - 50*a^4*b^6*c - 22*a^5*b*c^5 + 56*a^5*b^5*c + 12*a^6*b*c^4
- 38*a^6*b^4*c + 18*a^7*b*c^3 + 24*a^7*b^3*c - 8*a^8*b^2*c - 2*a^2*b^6*c^3
+ 2*a^2*b^7*c^2 + 14*a^3*b^4*c^4 - 10*a^3*b^5*c^3 - 24*a^3*b^6*c^2 - 27*a^4
*b^2*c^5 + 15*a^4*b^3*c^4 + 59*a^4*b^4*c^3 + 7*a^4*b^5*c^2 + 11*a^5*b^2*c^4
- 122*a^5*b^3*c^3 + 93*a^5*b^4*c^2 + 37*a^6*b^2*c^3 - 99*a^6*b^3*c^2 + 23*
a^7*b^2*c^2))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^
(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^
2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*

```
c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c
 - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*
c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2
*c^2)))^(1/2) + (8192*(6*a^2*b^8 - 3*a*b^9 - 4*a^3*b^7 + a^4*b^6 + 3*a^4*c^
6 + 2*a^5*c^5 - a^6*c^4 + 2*a*b^5*c^4 - 5*a*b^6*c^3 + a*b^7*c^2 + 16*a^2*b^
7*c + 8*a^3*b*c^6 - 38*a^3*b^6*c + 10*a^4*b*c^5 + 23*a^4*b^5*c + 6*a^5*b*c^
4 - 5*a^5*b^4*c - 10*a^2*b^3*c^5 + 25*a^2*b^4*c^4 + 4*a^2*b^5*c^3 - 41*a^2*
b^6*c^2 - 20*a^3*b^2*c^5 - 36*a^3*b^3*c^4 + 91*a^3*b^4*c^3 - 3*a^3*b^5*c^2
- 24*a^4*b^2*c^4 - 55*a^4*b^3*c^3 + 57*a^4*b^4*c^2 - 3*a^5*b^2*c^3 - 28*a^5
*b^3*c^2 + 4*a^6*b^2*c^2 + 5*a*b^8*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4
+ b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 3
3*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^
6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(
1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c
^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8
*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2) + (8192*tan(x/2)*(a*b^8 + 5*b^8*c -
b^9 + a^2*c^7 + a^3*c^6 + b^4*c^5 - 5*b^5*c^4 + 10*b^6*c^3 - 10*b^7*c^2 - 2
*a*b^2*c^6 + 14*a*b^3*c^5 - 35*a*b^4*c^4 + 40*a*b^5*c^3 - 20*a*b^6*c^2 - a^
2*b*c^6 - 6*a^2*b^6*c + 10*a^2*b^2*c^5 - 20*a^2*b^3*c^4 + 5*a^2*b^4*c^3 + 1
1*a^2*b^5*c^2 + 10*a^3*b^2*c^4 - 18*a^3*b^3*c^3 + 9*a^3*b^4*c^2 - 2*a^4*b^2
*c^3 + 2*a*b^7*c))/a^4)*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2
)^3)^(1/2) - b^6*c^2 + 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a
^3*b^2*c^3 - b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-
(4*a*c - b^2)^3)^(1/2) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(
4*a*c - b^2)^3)^(1/2))/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16
*a^8*c^2 + 10*a^5*b^4*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^
6*b^2*c^2)))^(1/2) + (16384*(b*c^7 - 4*b^2*c^6 + 6*b^3*c^5 - 4*b^4*c^4 + b^
5*c^3 - 2*a*b^2*c^5 + 2*a*b^3*c^4 - a*b^4*c^3 + a^2*b^2*c^4 + a*b*c^6))/a^4
))*(-(b^8 + 8*a^3*c^5 + 8*a^4*c^4 + b^5*(-(4*a*c - b^2)^3)^(1/2) - b^6*c^2
+ 8*a*b^4*c^3 - 18*a^2*b^2*c^4 + 33*a^2*b^4*c^2 - 38*a^3*b^2*c^3 - b^3*c^2*
(-(4*a*c - b^2)^3)^(1/2) - 10*a*b^6*c + 3*a^2*b*c^2*(-(4*a*c - b^2)^3)^(1/2
) + 2*a*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 4*a*b^3*c*(-(4*a*c - b^2)^3)^(1/2)
)/(2*(a^6*b^4 - a^4*b^6 + 16*a^6*c^4 + 32*a^7*c^3 + 16*a^8*c^2 + 10*a^5*b^4
*c - 8*a^7*b^2*c + a^4*b^4*c^2 - 8*a^5*b^2*c^3 - 32*a^6*b^2*c^2)))^(1/2)*2i
 - (2*tan(x/2))/(a*(tan(x/2)^2 - 1))
```

3.20 $\int \frac{\sec^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$

Optimal result

Integrand size = 19, antiderivative size = 334

$$\begin{aligned}
\int \frac{\sec^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx
&= -\frac{2c\left(b^3-3abc+\sqrt{b^2-4ac}(b^2-ac)\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{a^3\sqrt{b^2-4ac}\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} \\
&\quad+\frac{2c\left(b^3-3abc-\sqrt{b^2-4ac}(b^2-ac)\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{a^3\sqrt{b^2-4ac}\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}} \\
&\quad+\frac{\operatorname{arctanh}(\sin(x))}{2a}+\frac{(b^2-ac)\operatorname{arctanh}(\sin(x))}{a^3}-\frac{b\tan(x)}{a^2}+\frac{\sec(x)\tan(x)}{2a}
\end{aligned}$$

```
[Out] 1/2*arctanh(sin(x))/a+(-a*c+b^2)*arctanh(sin(x))/a^3-2*c*arctan((b-2*c-(-4*
a*c+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2))*(b^3-3*a
*b*c+(-a*c+b^2)*(-4*a*c+b^2)^(1/2))/a^3/(-4*a*c+b^2)^(1/2)/(b-2*c-(-4*a*c+b
^2)^(1/2))^(1/2)/(b+2*c-(-4*a*c+b^2)^(1/2))^(1/2)+2*c*arctan((b-2*c+(-4*a*c
+b^2)^(1/2))^(1/2)*tan(1/2*x)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2))*(b^3-3*a*b*
c-(-a*c+b^2)*(-4*a*c+b^2)^(1/2))/a^3/(-4*a*c+b^2)^(1/2)/(b-2*c+(-4*a*c+b^2)
^(1/2))^(1/2)/(b+2*c+(-4*a*c+b^2)^(1/2))^(1/2)-b*tan(x)/a^2+1/2*sec(x)*tan(
x)/a
```

Rubi [A] (verified)

Time = 4.99 (sec) , antiderivative size = 334, normalized size of antiderivative = 1.00, number of steps used = 12, number of rules used = 8, $\frac{\text{number of rules}}{\text{integrand size}} = 0.421$, Rules used = {3338, 3374, 2738, 211, 3855, 3852, 8, 3853}

$$\int \frac{\sec^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$$
$$= -\frac{2c\left(\sqrt{b^2-4ac}(b^2-ac)-3abc+b^3\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{-\sqrt{b^2-4ac}+b-2c}}{\sqrt{-\sqrt{b^2-4ac}+b+2c}}\right)}{a^3\sqrt{b^2-4ac}\sqrt{-\sqrt{b^2-4ac}+b-2c}\sqrt{-\sqrt{b^2-4ac}+b+2c}}$$
$$+\frac{2c\left(-\sqrt{b^2-4ac}(b^2-ac)-3abc+b^3\right)\arctan\left(\frac{\tan\left(\frac{x}{2}\right)\sqrt{\sqrt{b^2-4ac}+b-2c}}{\sqrt{\sqrt{b^2-4ac}+b+2c}}\right)}{a^3\sqrt{b^2-4ac}\sqrt{\sqrt{b^2-4ac}+b-2c}\sqrt{\sqrt{b^2-4ac}+b+2c}}$$
$$+\frac{(b^2-ac)\operatorname{arctanh}(\sin(x))}{a^3}-\frac{b\tan(x)}{a^2}+\frac{\operatorname{arctanh}(\sin(x))}{2a}+\frac{\tan(x)\sec(x)}{2a}$$

```
[In] Int[Sec[x]^3/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] (-2*c*(b^3 - 3*a*b*c + Sqrt[b^2 - 4*a*c]*(b^2 - a*c))*ArcTan[(Sqrt[b - 2*c
- Sqrt[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c - Sqrt[b^2 - 4*a*c]]])/(a^3*Sqr
t[b^2 - 4*a*c]*Sqrt[b - 2*c - Sqrt[b^2 - 4*a*c]]*Sqrt[b + 2*c - Sqrt[b^2 -
4*a*c]]) + (2*c*(b^3 - 3*a*b*c - Sqrt[b^2 - 4*a*c]*(b^2 - a*c))*ArcTan[(Sqr
t[b - 2*c + Sqrt[b^2 - 4*a*c]]*Tan[x/2])/Sqrt[b + 2*c + Sqrt[b^2 - 4*a*c]]]
)/(a^3*Sqrt[b^2 - 4*a*c]*Sqrt[b - 2*c + Sqrt[b^2 - 4*a*c]]*Sqrt[b + 2*c + S
qrt[b^2 - 4*a*c]]) + ArcTanh[Sin[x]]/(2*a) + ((b^2 - a*c)*ArcTanh[Sin[x]])/
a^3 - (b*Tan[x])/a^2 + (Sec[x]*Tan[x])/(2*a)
```

Rule 8

```
Int[a_, x_Symbol] :> Simp[a*x, x] /; FreeQ[a, x]
```

Rule 211

```
Int[((a_) + (b_.)*(x_)^2)^(-1), x_Symbol] :> Simp[(Rt[a/b, 2]/a)*ArcTan[x/R
t[a/b, 2]], x] /; FreeQ[{a, b}, x] && PosQ[a/b]
```

Rule 2738

```
Int[((a_) + (b_.)*sin[Pi/2 + (c_.) + (d_.)*(x_)])^(-1), x_Symbol] :> With[{
e = FreeFactors[Tan[(c + d*x)/2], x]}, Dist[2*(e/d), Subst[Int[1/(a + b + (
a - b)*e^2*x^2), x], x, Tan[(c + d*x)/2]/e], x]] /; FreeQ[{a, b, c, d}, x]
&& NeQ[a^2 - b^2, 0]
```

Rule 3338

```
Int[cos[(d_.) + (e_.)*(x_)]^(m_.)*((a_.) + cos[(d_.) + (e_.)*(x_)]^(n_.)*(b
_.) + cos[(d_.) + (e_.)*(x_)]^(n2_.)*(c_.))^(p_), x_Symbol] :> Int[ExpandTr
ig[cos[d + e*x]^m*(a + b*cos[d + e*x]^n + c*cos[d + e*x]^(2*n))^p, x], x] /
; FreeQ[{a, b, c, d, e}, x] && EqQ[n2, 2*n] && NeQ[b^2 - 4*a*c, 0] && Integ
ersQ[m, n, p]
```

Rule 3374

```
Int[(cos[(d_.) + (e_.)*(x_)]*(B_.) + (A_))/((a_.) + cos[(d_.) + (e_.)*(x_)]
*(b_.) + cos[(d_.) + (e_.)*(x_)]^2*(c_.)), x_Symbol] :> Module[{q = Rt[b^2
- 4*a*c, 2]}, Dist[B + (b*B - 2*A*c)/q, Int[1/(b + q + 2*c*Cos[d + e*x]), x
], x] + Dist[B - (b*B - 2*A*c)/q, Int[1/(b - q + 2*c*Cos[d + e*x]), x], x]]
 /; FreeQ[{a, b, c, d, e, A, B}, x] && NeQ[b^2 - 4*a*c, 0]
```

Rule 3852

```
Int[csc[(c_.) + (d_.)*(x_)]^(n_), x_Symbol] :> Dist[-d^(-1), Subst[Int[Expa
ndIntegrand[(1 + x^2)^(n/2 - 1), x], x], x, Cot[c + d*x]], x] /; FreeQ[{c,
d}, x] && IGtQ[n/2, 0]
```

Rule 3853

```
Int[(csc[(c_.) + (d_.)*(x_)]*(b_.))^(n_), x_Symbol] :> Simp[(-b)*Cos[c + d*
x]*((b*Csc[c + d*x])^(n - 1)/(d*(n - 1))), x] + Dist[b^2*((n - 2)/(n - 1)),
 Int[(b*Csc[c + d*x])^(n - 2), x], x] /; FreeQ[{b, c, d}, x] && GtQ[n, 1] &
& IntegerQ[2*n]
```

Rule 3855

```
Int[csc[(c_.) + (d_.)*(x_)], x_Symbol] :> Simp[-ArcTanh[Cos[c + d*x]]/d, x]
 /; FreeQ[{c, d}, x]
```

Rubi steps

integral

$$= \int \left(\frac{-b^3\left(1-\frac{2ac}{b^2}\right)-b^2c\left(1-\frac{ac}{b^2}\right)\cos(x)}{a^3\left(a+b\cos(x)+c\cos^2(x)\right)}+\frac{\left(b^2-ac\right)\sec(x)}{a^3}-\frac{b\sec^2(x)}{a^2}+\frac{\sec^3(x)}{a}\right)dx$$

$$=\frac{\int\frac{-b^3\left(1-\frac{2ac}{b^2}\right)-b^2c\left(1-\frac{ac}{b^2}\right)\cos(x)}{a+b\cos(x)+c\cos^2(x)}\,dx}{a^3}+\frac{\int\sec^3(x)\,dx}{a}-\frac{b\int\sec^2(x)\,dx}{a^2}+\frac{\left(b^2-ac\right)\int\sec(x)\,dx}{a^3}$$

$$= \frac{(b^2-ac)\operatorname{arctanh}(\sin(x))}{a^3} + \frac{\sec(x)\tan(x)}{2a} + \frac{\int \sec(x)\,dx}{2a} + \frac{b\text{Subst}(\int 1\,dx, x, -\tan(x))}{a^2} + \frac{\left(c\left(b^3-3abc-\sqrt{b^2-4ac}(b^2-ac)\right)\right)\int \frac{1}{b+\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{a^3\sqrt{b^2-4ac}} - \frac{\left(c\left(b^3-3abc+\sqrt{b^2-4ac}(b^2-ac)\right)\right)\int \frac{1}{b-\sqrt{b^2-4ac}+2c\cos(x)}\,dx}{a^3\sqrt{b^2-4ac}}$$

$$= \frac{\operatorname{arctanh}(\sin(x))}{2a} + \frac{(b^2-ac)\operatorname{arctanh}(\sin(x))}{a^3} - \frac{b\tan(x)}{a^2} + \frac{\sec(x)\tan(x)}{2a} + \frac{\left(2c\left(b^3-3abc-\sqrt{b^2-4ac}(b^2-ac)\right)\right)\text{Subst}\left(\int \frac{1}{b+2c+\sqrt{b^2-4ac}+\left(b-2c+\sqrt{b^2-4ac}\right)x^2}\,dx, x, \tan\left(\frac{x}{2}\right)\right)}{a^3\sqrt{b^2-4ac}} - \frac{\left(2c\left(b^3-3abc+\sqrt{b^2-4ac}(b^2-ac)\right)\right)\text{Subst}\left(\int \frac{1}{b+2c-\sqrt{b^2-4ac}+\left(b-2c-\sqrt{b^2-4ac}\right)x^2}\,dx, x, \tan\left(\frac{x}{2}\right)\right)}{a^3\sqrt{b^2-4ac}}$$

$$= -\frac{2c\left(b^3-3abc+\sqrt{b^2-4ac}(b^2-ac)\right)\arctan\left(\frac{\sqrt{b-2c-\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c-\sqrt{b^2-4ac}}}\right)}{a^3\sqrt{b^2-4ac}\sqrt{b-2c-\sqrt{b^2-4ac}}\sqrt{b+2c-\sqrt{b^2-4ac}}} + \frac{2c\left(b^3-3abc-\sqrt{b^2-4ac}(b^2-ac)\right)\arctan\left(\frac{\sqrt{b-2c+\sqrt{b^2-4ac}}\tan\left(\frac{x}{2}\right)}{\sqrt{b+2c+\sqrt{b^2-4ac}}}\right)}{a^3\sqrt{b^2-4ac}\sqrt{b-2c+\sqrt{b^2-4ac}}\sqrt{b+2c+\sqrt{b^2-4ac}}} + \frac{\operatorname{arctanh}(\sin(x))}{2a} + \frac{(b^2-ac)\operatorname{arctanh}(\sin(x))}{a^3} - \frac{b\tan(x)}{a^2} + \frac{\sec(x)\tan(x)}{2a}$$

Mathematica [A] (verified)

Time = 3.61 (sec) , antiderivative size = 446, normalized size of antiderivative = 1.34

$$\int \frac{\sec^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = -\frac{\frac{4\sqrt{2}c\left(b^3-3abc-b^2\sqrt{b^2-4ac}+ac\sqrt{b^2-4ac}\right)\operatorname{arctanh}\left(\frac{\left(b-2c+\sqrt{b^2-4ac}\right)\tan\left(\frac{x}{2}\right)}{\sqrt{-2b^2+4c(a+c)-2b\sqrt{b^2-4ac}}}\right)}{\sqrt{b^2-4ac}\sqrt{-b^2+2c(a+c)-b\sqrt{b^2-4ac}}} + \frac{4\sqrt{2}c\left(b^3-3abc+b^2\sqrt{b^2-4ac}-ac\sqrt{b^2-4ac}\right)\text{arcta}}{\sqrt{b^2-4ac}\sqrt{-b^2+2c(a+c)\cdot}}}{}$$

```
[In] Integrate[Sec[x]^3/(a + b*Cos[x] + c*Cos[x]^2),x]
```

```
[Out] -1/4*((4*Sqrt[2]*c*(b^3 - 3*a*b*c - b^2*Sqrt[b^2 - 4*a*c] + a*c*Sqrt[b^2 -
4*a*c])*ArcTanh[((b - 2*c + Sqrt[b^2 - 4*a*c])*Tan[x/2])/Sqrt[-2*b^2 + 4*c*
(a + c) - 2*b*Sqrt[b^2 - 4*a*c]]])/(Sqrt[b^2 - 4*a*c]*Sqrt[-b^2 + 2*c*(a +
c) - b*Sqrt[b^2 - 4*a*c]]) + (4*Sqrt[2]*c*(b^3 - 3*a*b*c + b^2*Sqrt[b^2 - 4
*a*c] - a*c*Sqrt[b^2 - 4*a*c])*ArcTanh[((-b + 2*c + Sqrt[b^2 - 4*a*c])*Tan[
x/2])/Sqrt[-2*b^2 + 4*c*(a + c) + 2*b*Sqrt[b^2 - 4*a*c]]])/(Sqrt[b^2 - 4*a*
```

```
c]*Sqrt[-b^2 + 2*c*(a + c) + b*Sqrt[b^2 - 4*a*c]]) + 2*(a^2 + 2*b^2 - 2*a*c
)*Log[Cos[x/2] - Sin[x/2]] - 2*(a^2 + 2*b^2 - 2*a*c)*Log[Cos[x/2] + Sin[x/2
]] + (4*a*b*Sin[x/2])/(Cos[x/2] - Sin[x/2]) + a^2/(Cos[x/2] + Sin[x/2])^2 +
 (4*a*b*Sin[x/2])/(Cos[x/2] + Sin[x/2]) + a^2/(-1 + Sin[x]))/a^3
```

Maple [A] (verified)

Time = 7.69 (sec) , antiderivative size = 481, normalized size of antiderivative = 1.44

method	result
default	$\frac{2(a-b+c)\left(\frac{\left(2cab\sqrt{-4ac+b^2}-a\,c^2\sqrt{-4ac+b^2}-b^3\sqrt{-4ac+b^2}+b^2c\sqrt{-4ac+b^2}+2a^2c^2-4a\,b^2c+3ab\,c^2+b^4-b^3c\right)\operatorname{arctanh}\left(\frac{(-a+b-c)\tan\left(\frac{x}{2}\right)}{\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-}}\right.}{2\sqrt{-4ac+b^2}\,(a-b+c)\sqrt{\left(\sqrt{-4ac+b^2}-a+c\right)(a-b+c)}}\right.}{}$
risch	Expression too large to display

```
[In] int(sec(x)^3/(a+cos(x)*b+c*cos(x)^2),x,method=_RETURNVERBOSE)
```

```
[Out] 2/a^3*(a-b+c)*(1/2*(2*c*a*b*(-4*a*c+b^2)^(1/2)-a*c^2*(-4*a*c+b^2)^(1/2)-b^3
*(-4*a*c+b^2)^(1/2)+b^2*c*(-4*a*c+b^2)^(1/2)+2*a^2*c^2-4*a*b^2*c+3*a*b*c^2+
b^4-b^3*c)/(-4*a*c+b^2)^(1/2)/(a-b+c)/(((-4*a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1
/2)*arctanh((-a+b-c)*tan(1/2*x)/(((-4*a*c+b^2)^(1/2)-a+c)*(a-b+c))^(1/2))+1
/2*(2*c*a*b*(-4*a*c+b^2)^(1/2)-a*c^2*(-4*a*c+b^2)^(1/2)-b^3*(-4*a*c+b^2)^(1
/2)+b^2*c*(-4*a*c+b^2)^(1/2)-2*a^2*c^2+4*a*b^2*c-3*a*b*c^2-b^4+b^3*c)/(-4*a
*c+b^2)^(1/2)/(a-b+c)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)*arctan((a-b+
c)*tan(1/2*x)/(((-4*a*c+b^2)^(1/2)+a-c)*(a-b+c))^(1/2)))-1/2/a/(tan(1/2*x)+
1)^2-1/2*(-a-2*b)/a^2/(tan(1/2*x)+1)+1/2*(a^2-2*a*c+2*b^2)/a^3*ln(tan(1/2*x
)+1)+1/2/a/(tan(1/2*x)-1)^2-1/2*(-a-2*b)/a^2/(tan(1/2*x)-1)+1/2/a^3*(-a^2+2
*a*c-2*b^2)*ln(tan(1/2*x)-1)
```

Fricas [F(-1)]

Timed out.

$$\int \frac{\sec^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Timed out}$$

```
[In] integrate(sec(x)^3/(a+b*cos(x)+c*cos(x)^2),x, algorithm="fricas")
```

```
[Out] Timed out
```

Sympy [F]

$$\int \frac{\sec^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \int \frac{\sec^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx$$

```
[In] integrate(sec(x)**3/(a+b*cos(x)+c*cos(x)**2),x)
[Out] Integral(sec(x)**3/(a + b*cos(x) + c*cos(x)**2), x)
```

Maxima [F]

$$\int \frac{\sec^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \int \frac{\sec(x)^3}{c\cos(x)^2+b\cos(x)+a}\,dx$$

```
[In] integrate(sec(x)^3/(a+b*cos(x)+c*cos(x)^2),x, algorithm="maxima")
[Out] -1/4*(8*a^2*cos(3*x)*sin(2*x) + 8*a^2*cos(2*x)*sin(x) + 4*a^2*sin(x) - 4*(a
^2*sin(3*x) + 2*a*b*sin(2*x) - a^2*sin(x))*cos(4*x) - 4*(a^3*cos(4*x)^2 + 4
*a^3*cos(2*x)^2 + a^3*sin(4*x)^2 + 4*a^3*sin(4*x)*sin(2*x) + 4*a^3*sin(2*x)
^2 + 4*a^3*cos(2*x) + a^3 + 2*(2*a^3*cos(2*x) + a^3)*cos(4*x))*integrate(-2
*(2*(b^3*c - a*b*c^2)*cos(3*x)^2 + 4*(2*a*b^3 - 2*a*b*c^2 - (4*a^2*b - b^3)
*c)*cos(2*x)^2 + 2*(b^3*c - a*b*c^2)*cos(x)^2 + 2*(b^3*c - a*b*c^2)*sin(3*x
)^2 + 4*(2*a*b^3 - 2*a*b*c^2 - (4*a^2*b - b^3)*c)*sin(2*x)^2 + 2*(2*b^4 - 2
*a*b^2*c - a*c^3 - (2*a^2 - b^2)*c^2)*sin(2*x)*sin(x) + 2*(b^3*c - a*b*c^2)
*sin(x)^2 + ((b^2*c^2 - a*c^3)*cos(3*x) + 2*(b^3*c - 2*a*b*c^2)*cos(2*x) +
(b^2*c^2 - a*c^3)*cos(x))*cos(4*x) + (b^2*c^2 - a*c^3 + 2*(2*b^4 - 2*a*b^2*
c - a*c^3 - (2*a^2 - b^2)*c^2)*cos(2*x) + 4*(b^3*c - a*b*c^2)*cos(x))*cos(3
*x) + 2*(b^3*c - 2*a*b*c^2 + (2*b^4 - 2*a*b^2*c - a*c^3 - (2*a^2 - b^2)*c^2
)*cos(x))*cos(2*x) + (b^2*c^2 - a*c^3)*cos(x) + ((b^2*c^2 - a*c^3)*sin(3*x)
 + 2*(b^3*c - 2*a*b*c^2)*sin(2*x) + (b^2*c^2 - a*c^3)*sin(x))*sin(4*x) + 2*
((2*b^4 - 2*a*b^2*c - a*c^3 - (2*a^2 - b^2)*c^2)*sin(2*x) + 2*(b^3*c - a*b*
c^2)*sin(x))*sin(3*x))/(a^3*c^2*cos(4*x)^2 + 4*a^3*b^2*cos(3*x)^2 + 4*a^3*b
^2*cos(x)^2 + a^3*c^2*sin(4*x)^2 + 4*a^3*b^2*sin(3*x)^2 + 4*a^3*b^2*sin(x)^
2 + 4*a^3*b*c*cos(x) + a^3*c^2 + 4*(4*a^5 + 4*a^4*c + a^3*c^2)*cos(2*x)^2 +
 4*(4*a^5 + 4*a^4*c + a^3*c^2)*sin(2*x)^2 + 8*(2*a^4*b + a^3*b*c)*sin(2*x)*
sin(x) + 2*(2*a^3*b*c*cos(3*x) + 2*a^3*b*c*cos(x) + a^3*c^2 + 2*(2*a^4*c +
a^3*c^2)*cos(2*x))*cos(4*x) + 4*(2*a^3*b^2*cos(x) + a^3*b*c + 2*(2*a^4*b +
a^3*b*c)*cos(2*x))*cos(3*x) + 4*(2*a^4*c + a^3*c^2 + 2*(2*a^4*b + a^3*b*c)*
cos(x))*cos(2*x) + 4*(a^3*b*c*sin(3*x) + a^3*b*c*sin(x) + (2*a^4*c + a^3*c^
2)*sin(2*x))*sin(4*x) + 8*(a^3*b^2*sin(x) + (2*a^4*b + a^3*b*c)*sin(2*x))*s
in(3*x)), x) - ((a^2 + 2*b^2 - 2*a*c)*cos(4*x)^2 + 4*(a^2 + 2*b^2 - 2*a*c)*
cos(2*x)^2 + (a^2 + 2*b^2 - 2*a*c)*sin(4*x)^2 + 4*(a^2 + 2*b^2 - 2*a*c)*sin
(4*x)*sin(2*x) + 4*(a^2 + 2*b^2 - 2*a*c)*sin(2*x)^2 + a^2 + 2*b^2 - 2*a*c +
```

```
 2*(a^2 + 2*b^2 - 2*a*c + 2*(a^2 + 2*b^2 - 2*a*c)*cos(2*x))*cos(4*x) + 4*(a
^2 + 2*b^2 - 2*a*c)*cos(2*x))*log(cos(x)^2 + sin(x)^2 + 2*sin(x) + 1) + ((a
^2 + 2*b^2 - 2*a*c)*cos(4*x)^2 + 4*(a^2 + 2*b^2 - 2*a*c)*cos(2*x)^2 + (a^2
+ 2*b^2 - 2*a*c)*sin(4*x)^2 + 4*(a^2 + 2*b^2 - 2*a*c)*sin(4*x)*sin(2*x) + 4
*(a^2 + 2*b^2 - 2*a*c)*sin(2*x)^2 + a^2 + 2*b^2 - 2*a*c + 2*(a^2 + 2*b^2 -
2*a*c + 2*(a^2 + 2*b^2 - 2*a*c)*cos(2*x))*cos(4*x) + 4*(a^2 + 2*b^2 - 2*a*c
)*cos(2*x))*log(cos(x)^2 + sin(x)^2 - 2*sin(x) + 1) + 4*(a^2*cos(3*x) + 2*a
*b*cos(2*x) - a^2*cos(x) + 2*a*b)*sin(4*x) - 4*(2*a^2*cos(2*x) + a^2)*sin(3
*x) - 8*(a^2*cos(x) - a*b)*sin(2*x))/(a^3*cos(4*x)^2 + 4*a^3*cos(2*x)^2 + a
^3*sin(4*x)^2 + 4*a^3*sin(4*x)*sin(2*x) + 4*a^3*sin(2*x)^2 + 4*a^3*cos(2*x)
 + a^3 + 2*(2*a^3*cos(2*x) + a^3)*cos(4*x))
```

Giac [B] (verification not implemented)

Leaf count of result is larger than twice the leaf count of optimal. 12615 vs. $2(286) = 572$.

Time = 2.84 (sec) , antiderivative size = 12615, normalized size of antiderivative = 37.77

$$\int \frac{\sec^3(x)}{a+b\cos(x)+c\cos^2(x)}\,dx = \text{Too large to display}$$

```
[In] integrate(sec(x)^3/(a+b*cos(x)+c*cos(x)^2),x, algorithm="giac")
```

```
[Out] ((2*a^2*b^7 - 4*a*b^8 + 2*b^9 - 20*a^3*b^5*c + 38*a^2*b^6*c - 12*a*b^7*c -
6*b^8*c + 64*a^4*b^3*c^2 - 110*a^3*b^4*c^2 - 12*a^2*b^5*c^2 + 54*a*b^6*c^2
+ 6*b^7*c^2 - 64*a^5*b*c^3 + 80*a^4*b^2*c^3 + 160*a^3*b^3*c^3 - 140*a^2*b^4
*c^3 - 56*a*b^5*c^3 - 2*b^6*c^3 + 32*a^5*c^4 - 192*a^4*b*c^4 + 64*a^3*b^2*c
^4 + 160*a^2*b^3*c^4 + 18*a*b^4*c^4 + 64*a^4*c^5 - 128*a^3*b*c^5 - 48*a^2*b
^2*c^5 + 32*a^3*c^6 + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
 b + c))*sqrt(b^2 - 4*a*c)*a^2*b^5 - 2*(b^2 - 4*a*c)*a^2*b^5 - 2*sqrt(a^2 -
 a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^6 +
 4*(b^2 - 4*a*c)*a*b^6 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(
a - b + c))*sqrt(b^2 - 4*a*c)*b^7 - 2*(b^2 - 4*a*c)*b^7 - 18*sqrt(a^2 - a*b
 + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^3*c +
 12*(b^2 - 4*a*c)*a^3*b^3*c + 9*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a
*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^4*c - 22*(b^2 - 4*a*c)*a^2*b^4*c +
 46*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
4*a*c)*a*b^5*c + 4*(b^2 - 4*a*c)*a*b^5*c + 11*sqrt(a^2 - a*b + b*c - c^2 +
sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^6*c + 6*(b^2 - 4*a*c)*b^
6*c + 24*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b
^2 - 4*a*c)*a^4*b*c^2 - 16*(b^2 - 4*a*c)*a^4*b*c^2 - sqrt(a^2 - a*b + b*c -
 c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c^2 + 22*(b
^2 - 4*a*c)*a^3*b^2*c^2 - 134*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3*c^2 + 28*(b^2 - 4*a*c)*a^2*b^3*c^2
 - 75*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2
- 4*a*c)*a*b^4*c^2 - 30*(b^2 - 4*a*c)*a*b^4*c^2 - 11*sqrt(a^2 - a*b + b*c -
```

c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5*c^2 - 6*(b^2 -
4*a*c)*b^5*c^2 - 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*sqrt(b^2 - 4*a*c)*a^4*c^3 + 8*(b^2 - 4*a*c)*a^4*c^3 + 120*sqrt(a^2 - a
*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^3
- 48*(b^2 - 4*a*c)*a^3*b*c^3 + 138*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c^3 + 20*(b^2 - 4*a*c)*a^2*b
^2*c^3 + 60*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*a*b^3*c^3 + 32*(b^2 - 4*a*c)*a*b^3*c^3 + 5*sqrt(a^2 - a*b +
b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c^3 + 2*(b
^2 - 4*a*c)*b^4*c^3 - 56*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
- b + c))*sqrt(b^2 - 4*a*c)*a^3*c^4 + 16*(b^2 - 4*a*c)*a^3*c^4 - 64*sqrt(a^
2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*
b*c^4 - 32*(b^2 - 4*a*c)*a^2*b*c^4 - 25*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b
^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^4 - 10*(b^2 - 4*a*c)*a*b
^2*c^4 + 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*a^2*c^5 + 8*(b^2 - 4*a*c)*a^2*c^5)*a^2*abs(a - b + c) - (4*a
^3*b^7 - 8*a^2*b^8 + 4*a*b^9 - 40*a^4*b^5*c + 84*a^3*b^6*c - 40*a^2*b^7*c -
4*a*b^8*c + 128*a^5*b^3*c^2 - 292*a^4*b^4*c^2 + 120*a^3*b^5*c^2 + 52*a^2*b
^6*c^2 - 4*a*b^7*c^2 - 128*a^6*b*c^3 + 352*a^5*b^2*c^3 - 64*a^4*b^3*c^3 - 2
32*a^3*b^4*c^3 + 32*a^2*b^5*c^3 + 4*a*b^6*c^3 - 64*a^6*c^4 - 128*a^5*b*c^4
+ 384*a^4*b^2*c^4 - 64*a^3*b^3*c^4 - 36*a^2*b^4*c^4 - 128*a^5*c^5 + 96*a^3*
b^2*c^5 - 64*a^4*c^6 + 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a
- b + c))*a^4*b^5 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^3*b^6 - 3*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*a^2*b^7 + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))
*a*b^8 - 18*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^5
*b^3*c + 33*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4
*b^4*c + 30*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^3
*b^5*c - 43*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^2
*b^6*c - 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a*b^
7*c + 24*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^6*b*
c^2 - 55*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^5*b^
2*c^2 - 101*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*a^4
*b^3*c^2 + 132*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*
a^3*b^4*c^2 + 40*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c)
)*a^2*b^5*c^2 + 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*a^6*c^3 + 116*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
))*a^5*b*c^3 - 177*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*a^4*b^2*c^3 - 72*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^3*b^3*c^3 - 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^2*b^4*c^3 + 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
b + c))*a*b^5*c^3 + 68*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a -
b + c))*a^5*c^4 + 32*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^4*b*c^4 - sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
))*a^3*b^2*c^4 - 25*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +

c))*a^2*b^3*c^4 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*a*b^4*c^4 + 36*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^4*c^5 + 4*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
))*a^3*b*c^5 + 25*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c
))*a^2*b^2*c^5 - 20*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*a^3*c^6 - 4*(b^2 - 4*a*c)*a^3*b^5 + 8*(b^2 - 4*a*c)*a^2*b^6 - 4*(b^2 -
4*a*c)*a*b^7 + 24*(b^2 - 4*a*c)*a^4*b^3*c - 52*(b^2 - 4*a*c)*a^3*b^4*c + 2
4*(b^2 - 4*a*c)*a^2*b^5*c + 4*(b^2 - 4*a*c)*a*b^6*c - 32*(b^2 - 4*a*c)*a^5*
b*c^2 + 84*(b^2 - 4*a*c)*a^4*b^2*c^2 - 24*(b^2 - 4*a*c)*a^3*b^3*c^2 - 36*(b
^2 - 4*a*c)*a^2*b^4*c^2 + 4*(b^2 - 4*a*c)*a*b^5*c^2 - 16*(b^2 - 4*a*c)*a^5*
c^3 - 32*(b^2 - 4*a*c)*a^4*b*c^3 + 88*(b^2 - 4*a*c)*a^3*b^2*c^3 - 16*(b^2 -
4*a*c)*a^2*b^3*c^3 - 4*(b^2 - 4*a*c)*a*b^4*c^3 - 32*(b^2 - 4*a*c)*a^4*c^4
+ 20*(b^2 - 4*a*c)*a^2*b^2*c^4 - 16*(b^2 - 4*a*c)*a^3*c^5)*abs(a - b + c)*a
bs(a) - (2*a^5*b^6 - 4*a^4*b^7 + 2*a^3*b^8 - 16*a^6*b^4*c + 30*a^5*b^5*c -
10*a^4*b^6*c - 2*a^3*b^7*c - 2*a^2*b^8*c + 36*a^7*b^2*c^2 - 58*a^6*b^3*c^2
- 8*a^5*b^4*c^2 + 12*a^4*b^5*c^2 + 14*a^3*b^6*c^2 + 6*a^2*b^7*c^2 - 16*a^8*
c^3 + 8*a^7*b*c^3 + 68*a^6*b^2*c^3 - 10*a^5*b^3*c^3 - 20*a^4*b^4*c^3 - 44*a
^3*b^5*c^3 - 6*a^2*b^6*c^3 - 16*a^7*c^4 - 24*a^6*b*c^4 - 20*a^5*b^2*c^4 + 8
2*a^4*b^3*c^4 + 44*a^3*b^4*c^4 + 2*a^2*b^5*c^4 + 16*a^6*c^5 - 8*a^5*b*c^5 -
84*a^4*b^2*c^5 - 14*a^3*b^3*c^5 + 16*a^5*c^6 + 24*a^4*b*c^6 + 3*sqrt(a^2 -
a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*b^4
- 2*(b^2 - 4*a*c)*a^5*b^4 - 2*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*
c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^5 + 4*(b^2 - 4*a*c)*a^4*b^5 - 5*sqr
t(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*
a^3*b^6 - 2*(b^2 - 4*a*c)*a^3*b^6 - 12*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^
2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^6*b^2*c + 8*(b^2 - 4*a*c)*a^6*b
^2*c + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b
^2 - 4*a*c)*a^5*b^3*c - 14*(b^2 - 4*a*c)*a^5*b^3*c + 33*sqrt(a^2 - a*b + b*
c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^4*c + 2*(b
^2 - 4*a*c)*a^4*b^4*c + 13*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(
a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^5*c + 2*(b^2 - 4*a*c)*a^3*b^5*c + 5*sqr
t(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*
a^2*b^6*c + 2*(b^2 - 4*a*c)*a^2*b^6*c + 6*sqrt(a^2 - a*b + b*c - c^2 + sqrt
(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^7*c^2 - 4*(b^2 - 4*a*c)*a^7*
c^2 + 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^
2 - 4*a*c)*a^6*b*c^2 + 2*(b^2 - 4*a*c)*a^6*b*c^2 - 60*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*b^2*c^2 + 16*(
b^2 - 4*a*c)*a^5*b^2*c^2 - 58*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^3*c^2 - 4*(b^2 - 4*a*c)*a^4*b^3*c^2
- 47*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 -
4*a*c)*a^3*b^4*c^2 - 6*(b^2 - 4*a*c)*a^3*b^4*c^2 - 11*sqrt(a^2 - a*b + b*c
- c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^5*c^2 - 6*(
b^2 - 4*a*c)*a^2*b^5*c^2 + 22*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^6*c^3 - 4*(b^2 - 4*a*c)*a^6*c^3 + 49*sqr
t(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*

a^5*b*c^3 - 6*(b^2 - 4*a*c)*a^5*b*c^3 + 110*sqrt(a^2 - a*b + b*c - c^2 + sq
rt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^2*c^3 - 4*(b^2 - 4*a*c
)*a^4*b^2*c^3 + 58*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b +
c))*sqrt(b^2 - 4*a*c)*a^3*b^3*c^3 + 20*(b^2 - 4*a*c)*a^3*b^3*c^3 + 11*sqrt(
a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^
2*b^4*c^3 + 6*(b^2 - 4*a*c)*a^2*b^4*c^3 - 38*sqrt(a^2 - a*b + b*c - c^2 + s
qrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*c^4 + 4*(b^2 - 4*a*c)*a
^5*c^4 - 69*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqr
t(b^2 - 4*a*c)*a^4*b*c^4 - 2*(b^2 - 4*a*c)*a^4*b*c^4 - 38*sqrt(a^2 - a*b +
b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c^4 -
20*(b^2 - 4*a*c)*a^3*b^2*c^4 - 5*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*
a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3*c^4 - 2*(b^2 - 4*a*c)*a^2*b^3*c
^4 + 10*sqrt(a^2 - a*b + b*c - c^2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^
2 - 4*a*c)*a^4*c^5 + 4*(b^2 - 4*a*c)*a^4*c^5 + 15*sqrt(a^2 - a*b + b*c - c^
2 + sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^5 + 6*(b^2 - 4
*a*c)*a^3*b*c^5)*abs(a - b + c))*(pi*floor(1/2*x/pi + 1/2) + arctan(2*sqrt(
1/2)*tan(1/2*x)/sqrt((2*a^4 - 2*a^3*c + sqrt(-4*(a^4 + a^3*b + a^3*c)*(a^4
- a^3*b + a^3*c) + 4*(a^4 - a^3*c)^2))/(a^4 - a^3*b + a^3*c))))/((3*a^10*b^
2 - 8*a^9*b^3 - a^8*b^4 + 16*a^7*b^5 - 7*a^6*b^6 - 8*a^5*b^7 + 5*a^4*b^8 -
12*a^11*c + 32*a^10*b*c + 30*a^9*b^2*c - 112*a^8*b^3*c + 8*a^7*b^4*c + 96*a
^6*b^5*c - 26*a^5*b^6*c - 16*a^4*b^7*c - 104*a^10*c^2 + 192*a^9*b*c^2 + 149
*a^8*b^2*c^2 - 336*a^7*b^3*c^2 - 30*a^6*b^4*c^2 + 112*a^5*b^5*c^2 + 17*a^4*
b^6*c^2 - 276*a^9*c^3 + 320*a^8*b*c^3 + 292*a^7*b^2*c^3 - 224*a^6*b^3*c^3 -
 120*a^5*b^4*c^3 - 304*a^8*c^4 + 128*a^7*b*c^4 + 237*a^6*b^2*c^4 + 24*a^5*b
^3*c^4 - 17*a^4*b^4*c^4 - 116*a^7*c^5 - 96*a^6*b*c^5 + 62*a^5*b^2*c^5 + 16*
a^4*b^3*c^5 + 24*a^6*c^6 - 64*a^5*b*c^6 - 5*a^4*b^2*c^6 + 20*a^5*c^7)*abs(a
)) - ((2*a^2*b^7 - 4*a*b^8 + 2*b^9 - 20*a^3*b^5*c + 38*a^2*b^6*c - 12*a*b^7
*c - 6*b^8*c + 64*a^4*b^3*c^2 - 110*a^3*b^4*c^2 - 12*a^2*b^5*c^2 + 54*a*b^6
*c^2 + 6*b^7*c^2 - 64*a^5*b*c^3 + 80*a^4*b^2*c^3 + 160*a^3*b^3*c^3 - 140*a^
2*b^4*c^3 - 56*a*b^5*c^3 - 2*b^6*c^3 + 32*a^5*c^4 - 192*a^4*b*c^4 + 64*a^3*
b^2*c^4 + 160*a^2*b^3*c^4 + 18*a*b^4*c^4 + 64*a^4*c^5 - 128*a^3*b*c^5 - 48*
a^2*b^2*c^5 + 32*a^3*c^6 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)
*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^5 - 2*(b^2 - 4*a*c)*a^2*b^5 - 2*sqrt(
a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*
b^6 + 4*(b^2 - 4*a*c)*a*b^6 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a
c)(a - b + c))*sqrt(b^2 - 4*a*c)*b^7 - 2*(b^2 - 4*a*c)*b^7 - 18*sqrt(a^2
- a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^
3*c + 12*(b^2 - 4*a*c)*a^3*b^3*c + 9*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
- 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^4*c - 22*(b^2 - 4*a*c)*a^2*b^
4*c + 46*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b
^2 - 4*a*c)*a*b^5*c + 4*(b^2 - 4*a*c)*a*b^5*c + 11*sqrt(a^2 - a*b + b*c - c
^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^6*c + 6*(b^2 - 4*a*
c)*b^6*c + 24*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*s
qrt(b^2 - 4*a*c)*a^4*b*c^2 - 16*(b^2 - 4*a*c)*a^4*b*c^2 - sqrt(a^2 - a*b +
b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c^2 +

22*(b^2 - 4*a*c)*a^3*b^2*c^2 - 134*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3*c^2 + 28*(b^2 - 4*a*c)*a^2*b^
3*c^2 - 75*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt
(b^2 - 4*a*c)*a*b^4*c^2 - 30*(b^2 - 4*a*c)*a*b^4*c^2 - 11*sqrt(a^2 - a*b +
b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^5*c^2 - 6*(b
^2 - 4*a*c)*b^5*c^2 - 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*sqrt(b^2 - 4*a*c)*a^4*c^3 + 8*(b^2 - 4*a*c)*a^4*c^3 + 120*sqrt(a^
2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*
b*c^3 - 48*(b^2 - 4*a*c)*a^3*b*c^3 + 138*sqrt(a^2 - a*b + b*c - c^2 - sqrt(
b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^2*c^3 + 20*(b^2 - 4*a*c)*
a^2*b^2*c^3 + 60*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c)
)*sqrt(b^2 - 4*a*c)*a*b^3*c^3 + 32*(b^2 - 4*a*c)*a*b^3*c^3 + 5*sqrt(a^2 - a
*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*b^4*c^3 +
2*(b^2 - 4*a*c)*b^4*c^3 - 56*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*c^4 + 16*(b^2 - 4*a*c)*a^3*c^4 - 64*sq
rt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)
*a^2*b*c^4 - 32*(b^2 - 4*a*c)*a^2*b*c^4 - 25*sqrt(a^2 - a*b + b*c - c^2 - s
qrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a*b^2*c^4 - 10*(b^2 - 4*a*c
)*a*b^2*c^4 + 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c)
)*sqrt(b^2 - 4*a*c)*a^2*c^5 + 8*(b^2 - 4*a*c)*a^2*c^5)*a^2*abs(a - b + c) -
(4*a^3*b^7 - 8*a^2*b^8 + 4*a*b^9 - 40*a^4*b^5*c + 84*a^3*b^6*c - 40*a^2*b^
7*c - 4*a*b^8*c + 128*a^5*b^3*c^2 - 292*a^4*b^4*c^2 + 120*a^3*b^5*c^2 + 52*
a^2*b^6*c^2 - 4*a*b^7*c^2 - 128*a^6*b*c^3 + 352*a^5*b^2*c^3 - 64*a^4*b^3*c^
3 - 232*a^3*b^4*c^3 + 32*a^2*b^5*c^3 + 4*a*b^6*c^3 - 64*a^6*c^4 - 128*a^5*b
*c^4 + 384*a^4*b^2*c^4 - 64*a^3*b^3*c^4 - 36*a^2*b^4*c^4 - 128*a^5*c^5 + 96
*a^3*b^2*c^5 - 64*a^4*c^6 - 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c
)*(a - b + c))*a^4*b^5 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(
a - b + c))*a^3*b^6 + 3*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*a^2*b^7 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*a*b^8 + 18*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c)
)*a^5*b^3*c - 33*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c)
)*a^4*b^4*c - 30*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c)
)*a^3*b^5*c + 43*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c)
)*a^2*b^6*c + 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))
*a*b^7*c - 24*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a
^6*b*c^2 + 55*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*a
^5*b^2*c^2 + 101*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c)
)*a^4*b^3*c^2 - 132*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b +
c))*a^3*b^4*c^2 - 40*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b
+ c))*a^2*b^5*c^2 - 12*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*a^6*c^3 - 116*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*a^5*b*c^3 + 177*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*a^4*b^2*c^3 + 72*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*a^3*b^3*c^3 + 2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(
a - b + c))*a^2*b^4*c^3 - 6*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*

```
(a - b + c))*a*b^5*c^3 - 68*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*
(a - b + c))*a^5*c^4 - 32*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*a^4*b*c^4 + sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*a^3*b^2*c^4 + 25*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*a^2*b^3*c^4 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*a*b^4*c^4 - 36*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
 - b + c))*a^4*c^5 - 4*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*a^3*b*c^5 - 25*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
b + c))*a^2*b^2*c^5 + 20*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a
- b + c))*a^3*c^6 - 4*(b^2 - 4*a*c)*a^3*b^5 + 8*(b^2 - 4*a*c)*a^2*b^6 - 4*(
b^2 - 4*a*c)*a*b^7 + 24*(b^2 - 4*a*c)*a^4*b^3*c - 52*(b^2 - 4*a*c)*a^3*b^4*
c + 24*(b^2 - 4*a*c)*a^2*b^5*c + 4*(b^2 - 4*a*c)*a*b^6*c - 32*(b^2 - 4*a*c)
*a^5*b*c^2 + 84*(b^2 - 4*a*c)*a^4*b^2*c^2 - 24*(b^2 - 4*a*c)*a^3*b^3*c^2 -
36*(b^2 - 4*a*c)*a^2*b^4*c^2 + 4*(b^2 - 4*a*c)*a*b^5*c^2 - 16*(b^2 - 4*a*c)
*a^5*c^3 - 32*(b^2 - 4*a*c)*a^4*b*c^3 + 88*(b^2 - 4*a*c)*a^3*b^2*c^3 - 16*(
b^2 - 4*a*c)*a^2*b^3*c^3 - 4*(b^2 - 4*a*c)*a*b^4*c^3 - 32*(b^2 - 4*a*c)*a^4
*c^4 + 20*(b^2 - 4*a*c)*a^2*b^2*c^4 - 16*(b^2 - 4*a*c)*a^3*c^5)*abs(a - b +
 c)*abs(a) - (2*a^5*b^6 - 4*a^4*b^7 + 2*a^3*b^8 - 16*a^6*b^4*c + 30*a^5*b^5
*c - 10*a^4*b^6*c - 2*a^3*b^7*c - 2*a^2*b^8*c + 36*a^7*b^2*c^2 - 58*a^6*b^3
*c^2 - 8*a^5*b^4*c^2 + 12*a^4*b^5*c^2 + 14*a^3*b^6*c^2 + 6*a^2*b^7*c^2 - 16
*a^8*c^3 + 8*a^7*b*c^3 + 68*a^6*b^2*c^3 - 10*a^5*b^3*c^3 - 20*a^4*b^4*c^3 -
 44*a^3*b^5*c^3 - 6*a^2*b^6*c^3 - 16*a^7*c^4 - 24*a^6*b*c^4 - 20*a^5*b^2*c^
4 + 82*a^4*b^3*c^4 + 44*a^3*b^4*c^4 + 2*a^2*b^5*c^4 + 16*a^6*c^5 - 8*a^5*b*
c^5 - 84*a^4*b^2*c^5 - 14*a^3*b^3*c^5 + 16*a^5*c^6 + 24*a^4*b*c^6 + 3*sqrt(
a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^
5*b^4 - 2*(b^2 - 4*a*c)*a^5*b^4 - 2*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^5 + 4*(b^2 - 4*a*c)*a^4*b^5 -
5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*a^3*b^6 - 2*(b^2 - 4*a*c)*a^3*b^6 - 12*sqrt(a^2 - a*b + b*c - c^2 - sq
rt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^6*b^2*c + 8*(b^2 - 4*a*c)*
a^6*b^2*c + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*s
qrt(b^2 - 4*a*c)*a^5*b^3*c - 14*(b^2 - 4*a*c)*a^5*b^3*c + 33*sqrt(a^2 - a*b
 + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^4*c +
 2*(b^2 - 4*a*c)*a^4*b^4*c + 13*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a
*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^5*c + 2*(b^2 - 4*a*c)*a^3*b^5*c +
5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*a^2*b^6*c + 2*(b^2 - 4*a*c)*a^2*b^6*c + 6*sqrt(a^2 - a*b + b*c - c^2 -
 sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^7*c^2 - 4*(b^2 - 4*a*c)
*a^7*c^2 + 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sq
rt(b^2 - 4*a*c)*a^6*b*c^2 + 2*(b^2 - 4*a*c)*a^6*b*c^2 - 60*sqrt(a^2 - a*b +
 b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*b^2*c^2 +
 16*(b^2 - 4*a*c)*a^5*b^2*c^2 - 58*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^3*c^2 - 4*(b^2 - 4*a*c)*a^4*b^3
*c^2 - 47*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(
b^2 - 4*a*c)*a^3*b^4*c^2 - 6*(b^2 - 4*a*c)*a^3*b^4*c^2 - 11*sqrt(a^2 - a*b
```

```
+ b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^5*c^2
- 6*(b^2 - 4*a*c)*a^2*b^5*c^2 + 22*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 -
4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^6*c^3 - 4*(b^2 - 4*a*c)*a^6*c^3 + 4
9*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*
a*c)*a^5*b*c^3 - 6*(b^2 - 4*a*c)*a^5*b*c^3 + 110*sqrt(a^2 - a*b + b*c - c^2
 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^4*b^2*c^3 - 4*(b^2 -
4*a*c)*a^4*b^2*c^3 + 58*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a -
 b + c))*sqrt(b^2 - 4*a*c)*a^3*b^3*c^3 + 20*(b^2 - 4*a*c)*a^3*b^3*c^3 + 11*
sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*
c)*a^2*b^4*c^3 + 6*(b^2 - 4*a*c)*a^2*b^4*c^3 - 38*sqrt(a^2 - a*b + b*c - c^
2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^5*c^4 + 4*(b^2 - 4*a
*c)*a^5*c^4 - 69*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c)
)*sqrt(b^2 - 4*a*c)*a^4*b*c^4 - 2*(b^2 - 4*a*c)*a^4*b*c^4 - 38*sqrt(a^2 - a
*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b^2*c
^4 - 20*(b^2 - 4*a*c)*a^3*b^2*c^4 - 5*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2
 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^2*b^3*c^4 - 2*(b^2 - 4*a*c)*a^2*
b^3*c^4 + 10*sqrt(a^2 - a*b + b*c - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sq
rt(b^2 - 4*a*c)*a^4*c^5 + 4*(b^2 - 4*a*c)*a^4*c^5 + 15*sqrt(a^2 - a*b + b*c
 - c^2 - sqrt(b^2 - 4*a*c)*(a - b + c))*sqrt(b^2 - 4*a*c)*a^3*b*c^5 + 6*(b^
2 - 4*a*c)*a^3*b*c^5)*abs(a - b + c))*(pi*floor(1/2*x/pi + 1/2) + arctan(2*
sqrt(1/2)*tan(1/2*x)/sqrt((2*a^4 - 2*a^3*c - sqrt(-4*(a^4 + a^3*b + a^3*c)*
(a^4 - a^3*b + a^3*c) + 4*(a^4 - a^3*c)^2))/(a^4 - a^3*b + a^3*c))))/((3*a^
10*b^2 - 8*a^9*b^3 - a^8*b^4 + 16*a^7*b^5 - 7*a^6*b^6 - 8*a^5*b^7 + 5*a^4*b
^8 - 12*a^11*c + 32*a^10*b*c + 30*a^9*b^2*c - 112*a^8*b^3*c + 8*a^7*b^4*c +
 96*a^6*b^5*c - 26*a^5*b^6*c - 16*a^4*b^7*c - 104*a^10*c^2 + 192*a^9*b*c^2
+ 149*a^8*b^2*c^2 - 336*a^7*b^3*c^2 - 30*a^6*b^4*c^2 + 112*a^5*b^5*c^2 + 17
*a^4*b^6*c^2 - 276*a^9*c^3 + 320*a^8*b*c^3 + 292*a^7*b^2*c^3 - 224*a^6*b^3*
c^3 - 120*a^5*b^4*c^3 - 304*a^8*c^4 + 128*a^7*b*c^4 + 237*a^6*b^2*c^4 + 24*
a^5*b^3*c^4 - 17*a^4*b^4*c^4 - 116*a^7*c^5 - 96*a^6*b*c^5 + 62*a^5*b^2*c^5
+ 16*a^4*b^3*c^5 + 24*a^6*c^6 - 64*a^5*b*c^6 - 5*a^4*b^2*c^6 + 20*a^5*c^7)*
abs(a)) + 1/2*(a^2 + 2*b^2 - 2*a*c)*log(abs(tan(1/2*x) + 1))/a^3 - 1/2*(a^2
 + 2*b^2 - 2*a*c)*log(abs(tan(1/2*x) - 1))/a^3 + (a*tan(1/2*x)^3 + 2*b*tan(
1/2*x)^3 + a*tan(1/2*x) - 2*b*tan(1/2*x))/((tan(1/2*x)^2 - 1)^2*a^2)
```

Mupad [B] (verification not implemented)

Time = 16.99 (sec) , antiderivative size = 45255, normalized size of antiderivative = 135.49

$$\int \frac{\sec^3(x)}{a + b\cos(x) + c\cos^2(x)}\, dx = \text{Too large to display}$$

```
[In] int(1/(cos(x)^3*(a + b*cos(x) + c*cos(x)^2)),x)
```

```
[Out] ((tan(x/2)^3*(a + 2*b))/a^2 + (tan(x/2)*(a - 2*b))/a^2)/(tan(x/2)^4 - 2*tan
(x/2)^2 + 1) - atan((((((2048*(26*a^9*b^7 - 12*a^8*b^8 - 18*a^10*b^6 + 6*a
```

^11*b^5 - 2*a^12*b^4 + 48*a^10*c^6 + 176*a^11*c^5 + 176*a^12*c^4 + 16*a^13*
c^3 - 32*a^14*c^2 + 20*a^8*b^7*c + 74*a^9*b^6*c - 144*a^10*b*c^5 - 192*a^10
*b^5*c - 352*a^11*b*c^4 + 122*a^11*b^4*c - 144*a^12*b*c^3 - 40*a^12*b^3*c +
64*a^13*b*c^2 + 16*a^13*b^2*c + 8*a^8*b^4*c^4 - 20*a^8*b^5*c^3 + 4*a^8*b^6
*c^2 - 44*a^9*b^2*c^5 + 116*a^9*b^3*c^4 + 10*a^9*b^4*c^3 - 182*a^9*b^5*c^2
- 148*a^10*b^2*c^4 + 496*a^10*b^3*c^3 - 50*a^10*b^4*c^2 - 260*a^11*b^2*c^3
+ 388*a^11*b^3*c^2 - 204*a^12*b^2*c^2))/a^8 - (2048*tan(x/2)*((8*a^4*c^6 -
b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 +
33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*
b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*
a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-
(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*
c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3
+ 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 -
32*a^8*b^2*c^2)))^(1/2)*(32*a^16*c + 8*a^10*b^7 - 24*a^11*b^6 + 32*a^12*b^
5 - 32*a^13*b^4 + 24*a^14*b^3 - 8*a^15*b^2 + 96*a^12*c^5 + 64*a^13*c^4 - 12
8*a^14*c^3 - 64*a^15*c^2 - 8*a^10*b^6*c - 56*a^11*b^5*c - 32*a^12*b*c^4 + 1
84*a^12*b^4*c + 352*a^13*b*c^3 - 200*a^13*b^3*c + 288*a^14*b*c^2 + 144*a^14
*b^2*c + 8*a^10*b^4*c^3 - 8*a^10*b^5*c^2 - 56*a^11*b^2*c^4 + 40*a^11*b^3*c^
3 + 96*a^11*b^4*c^2 - 272*a^12*b^2*c^3 + 8*a^12*b^3*c^2 - 320*a^13*b^2*c^2
- 96*a^15*b*c))/a^8)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3
)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3
*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/
2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a
*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*
(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4
- a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^
2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) + (2048*tan(x/2
)*(8*a^14*c + 8*a^4*b^11 - 24*a^5*b^10 + 36*a^6*b^9 - 52*a^7*b^8 + 61*a^8*b
^7 - 49*a^9*b^6 + 33*a^10*b^5 - 17*a^11*b^4 + 6*a^12*b^3 - 2*a^13*b^2 + 72*
a^8*c^7 - 136*a^9*c^6 - 192*a^10*c^5 + 168*a^11*c^4 + 80*a^12*c^3 - 64*a^13
*c^2 - 8*a^4*b^10*c - 72*a^5*b^9*c + 244*a^6*b^8*c - 308*a^7*b^7*c - 88*a^8
*b*c^6 + 375*a^8*b^6*c + 496*a^9*b*c^5 - 416*a^9*b^5*c - 16*a^10*b*c^4 + 29
5*a^10*b^4*c - 328*a^11*b*c^3 - 178*a^11*b^3*c + 184*a^12*b*c^2 + 84*a^12*b
^2*c + 8*a^4*b^8*c^3 - 8*a^4*b^9*c^2 - 72*a^5*b^6*c^4 + 56*a^5*b^7*c^3 + 11
2*a^5*b^8*c^2 + 220*a^6*b^4*c^5 - 140*a^6*b^5*c^4 - 424*a^6*b^6*c^3 + 80*a^
6*b^7*c^2 - 256*a^7*b^2*c^6 + 192*a^7*b^3*c^5 + 416*a^7*b^4*c^4 + 572*a^7*b
^5*c^3 - 732*a^7*b^6*c^2 + 64*a^8*b^2*c^5 - 1152*a^8*b^3*c^4 + 521*a^8*b^4*
c^3 + 779*a^8*b^5*c^2 + 234*a^9*b^2*c^4 - 494*a^9*b^3*c^3 - 723*a^9*b^4*c^2
+ 180*a^10*b^2*c^3 + 770*a^10*b^3*c^2 - 416*a^11*b^2*c^2 - 24*a^13*b*c))/a
^8)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2
- 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3
*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8*c -
4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2
) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3

)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) - (2048*(26*a^3*b^11 - 12*a^2*b^12 - 30*a^4*b^10 + 29*a^5*b^9 - 20*a^6*b^8 + 10*a^7*b^7 - 4*a^8*b^6 + a^9*b^5 + 12*a^6*c^8 + 88*a^7*c^7 + 72*a^8*c^6 - 44*a^9*c^5 - 28*a^10*c^4 + 12*a^11*c^3 + 20*a^2*b^11*c + 98*a^3*b^10*c - 228*a^4*b^9*c + 251*a^5*b^8*c - 96*a^6*b*c^7 - 238*a^6*b^7*c - 200*a^7*b*c^6 + 154*a^7*b^6*c + 100*a^8*b*c^5 - 72*a^8*b^5*c + 112*a^9*b*c^4 + 27*a^9*b^4*c - 68*a^10*b*c^3 - 6*a^10*b^3*c + 8*a^11*b*c^2 + 8*a^2*b^8*c^4 - 20*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 60*a^3*b^6*c^5 + 156*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 222*a^3*b^9*c^2 + 136*a^4*b^4*c^6 - 388*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 856*a^4*b^7*c^3 - 202*a^4*b^8*c^2 - 100*a^5*b^2*c^7 + 364*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 1362*a^5*b^5*c^4 - 115*a^5*b^6*c^3 + 635*a^5*b^7*c^2 - 340*a^6*b^2*c^6 + 904*a^6*b^3*c^5 + 583*a^6*b^4*c^4 - 564*a^6*b^5*c^3 - 655*a^6*b^6*c^2 - 399*a^7*b^2*c^5 + 9*a^7*b^3*c^4 + 536*a^7*b^4*c^3 + 612*a^7*b^5*c^2 - 37*a^8*b^2*c^4 - 524*a^8*b^3*c^3 - 354*a^8*b^4*c^2 + 239*a^9*b^2*c^3 + 145*a^9*b^3*c^2 - 47*a^10*b^2*c^2))/a^8)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) - (2048*tan(x/2)*(4*a*b^12 + 20*b^12*c - 4*b^13 - 4*a^2*b^11 + 4*a^3*b^10 - a^4*b^9 + a^5*b^8 + 12*a^4*c^9 - 44*a^5*c^8 + 2*a^6*c^7 + 38*a^7*c^6 - 18*a^8*c^5 + 2*a^9*c^4 + 4*b^8*c^5 - 20*b^9*c^4 + 40*b^10*c^3 - 40*b^11*c^2 - 24*a*b^6*c^6 + 136*a*b^7*c^5 - 300*a*b^8*c^4 + 320*a*b^9*c^3 - 160*a*b^10*c^2 - 20*a^2*b^10*c + 20*a^3*b^9*c - 92*a^4*b*c^8 - 31*a^4*b^8*c + 168*a^5*b*c^7 + 4*a^5*b^7*c + 2*a^6*b*c^6 - 8*a^6*b^6*c - 84*a^7*b*c^5 + 26*a^8*b*c^4 + 44*a^2*b^4*c^7 - 300*a^2*b^5*c^6 + 764*a^2*b^6*c^5 - 900*a^2*b^7*c^4 + 460*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 32*a^3*b^2*c^8 + 272*a^3*b^3*c^7 - 840*a^3*b^4*c^6 + 1156*a^3*b^5*c^5 - 660*a^3*b^6*c^4 + 72*a^3*b^7*c^3 + 8*a^3*b^8*c^2 + 384*a^4*b^2*c^7 - 704*a^4*b^3*c^6 + 541*a^4*b^4*c^5 - 149*a^4*b^5*c^4 + 34*a^4*b^6*c^3 + 6*a^4*b^7*c^2 - 204*a^5*b^2*c^6 + 96*a^5*b^3*c^5 + 41*a^5*b^4*c^4 - 132*a^5*b^5*c^3 + 82*a^5*b^6*c^2 - 90*a^6*b^2*c^5 + 174*a^6*b^3*c^4 - 104*a^6*b^4*c^3 + 8*a^6*b^5*c^2 + 82*a^7*b^2*c^4 - 40*a^7*b^3*c^3 + 20*a^7*b^4*c^2 - 16*a^8*b^2*c^3 + 24*a*b^11*c))/a^8)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2)*1i - (

((((2048*(26*a^9*b^7 - 12*a^8*b^8 - 18*a^10*b^6 + 6*a^11*b^5 - 2*a^12*b^4 +
48*a^10*c^6 + 176*a^11*c^5 + 176*a^12*c^4 + 16*a^13*c^3 - 32*a^14*c^2 + 20
*a^8*b^7*c + 74*a^9*b^6*c - 144*a^10*b*c^5 - 192*a^10*b^5*c - 352*a^11*b*c^
4 + 122*a^11*b^4*c - 144*a^12*b*c^3 - 40*a^12*b^3*c + 64*a^13*b*c^2 + 16*a^
13*b^2*c + 8*a^8*b^4*c^4 - 20*a^8*b^5*c^3 + 4*a^8*b^6*c^2 - 44*a^9*b^2*c^5
+ 116*a^9*b^3*c^4 + 10*a^9*b^4*c^3 - 182*a^9*b^5*c^2 - 148*a^10*b^2*c^4 + 4
96*a^10*b^3*c^3 - 50*a^10*b^4*c^2 - 260*a^11*b^2*c^3 + 388*a^11*b^3*c^2 - 2
04*a^12*b^2*c^2))/a^8 + (2048*tan(x/2)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7
*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^
2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4
*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) +
3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2)
- 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(
1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^
7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/
2)*(32*a^16*c + 8*a^10*b^7 - 24*a^11*b^6 + 32*a^12*b^5 - 32*a^13*b^4 + 24*a
^14*b^3 - 8*a^15*b^2 + 96*a^12*c^5 + 64*a^13*c^4 - 128*a^14*c^3 - 64*a^15*c
^2 - 8*a^10*b^6*c - 56*a^11*b^5*c - 32*a^12*b*c^4 + 184*a^12*b^4*c + 352*a^
13*b*c^3 - 200*a^13*b^3*c + 288*a^14*b*c^2 + 144*a^14*b^2*c + 8*a^10*b^4*c^
3 - 8*a^10*b^5*c^2 - 56*a^11*b^2*c^4 + 40*a^11*b^3*c^3 + 96*a^11*b^4*c^2 -
272*a^12*b^2*c^3 + 8*a^12*b^3*c^2 - 320*a^13*b^2*c^2 - 96*a^15*b*c))/a^8)*(
(8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10
*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*
c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*
b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4
*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/
2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4
+ 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*
a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) - (2048*tan(x/2)*(8*a^14*c + 8*a^4*b^
11 - 24*a^5*b^10 + 36*a^6*b^9 - 52*a^7*b^8 + 61*a^8*b^7 - 49*a^9*b^6 + 33*a
^10*b^5 - 17*a^11*b^4 + 6*a^12*b^3 - 2*a^13*b^2 + 72*a^8*c^7 - 136*a^9*c^6
- 192*a^10*c^5 + 168*a^11*c^4 + 80*a^12*c^3 - 64*a^13*c^2 - 8*a^4*b^10*c -
72*a^5*b^9*c + 244*a^6*b^8*c - 308*a^7*b^7*c - 88*a^8*b*c^6 + 375*a^8*b^6*c
+ 496*a^9*b*c^5 - 416*a^9*b^5*c - 16*a^10*b*c^4 + 295*a^10*b^4*c - 328*a^1
1*b*c^3 - 178*a^11*b^3*c + 184*a^12*b*c^2 + 84*a^12*b^2*c + 8*a^4*b^8*c^3 -
8*a^4*b^9*c^2 - 72*a^5*b^6*c^4 + 56*a^5*b^7*c^3 + 112*a^5*b^8*c^2 + 220*a^
6*b^4*c^5 - 140*a^6*b^5*c^4 - 424*a^6*b^6*c^3 + 80*a^6*b^7*c^2 - 256*a^7*b^
2*c^6 + 192*a^7*b^3*c^5 + 416*a^7*b^4*c^4 + 572*a^7*b^5*c^3 - 732*a^7*b^6*c
^2 + 64*a^8*b^2*c^5 - 1152*a^8*b^3*c^4 + 521*a^8*b^4*c^3 + 779*a^8*b^5*c^2
+ 234*a^9*b^2*c^4 - 494*a^9*b^3*c^3 - 723*a^9*b^4*c^2 + 180*a^10*b^2*c^3 +
770*a^10*b^3*c^2 - 416*a^11*b^2*c^2 - 24*a^13*b*c))/a^8)*((8*a^4*c^6 - b^10
+ 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*a
^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*
c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c
- b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a

*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-
(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 1
6*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*
a^8*b^2*c^2)))^(1/2) - (2048*(26*a^3*b^11 - 12*a^2*b^12 - 30*a^4*b^10 + 29*
a^5*b^9 - 20*a^6*b^8 + 10*a^7*b^7 - 4*a^8*b^6 + a^9*b^5 + 12*a^6*c^8 + 88*a
^7*c^7 + 72*a^8*c^6 - 44*a^9*c^5 - 28*a^10*c^4 + 12*a^11*c^3 + 20*a^2*b^11*
c + 98*a^3*b^10*c - 228*a^4*b^9*c + 251*a^5*b^8*c - 96*a^6*b*c^7 - 238*a^6*
b^7*c - 200*a^7*b*c^6 + 154*a^7*b^6*c + 100*a^8*b*c^5 - 72*a^8*b^5*c + 112*
a^9*b*c^4 + 27*a^9*b^4*c - 68*a^10*b*c^3 - 6*a^10*b^3*c + 8*a^11*b*c^2 + 8*
a^2*b^8*c^4 - 20*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 60*a^3*b^6*c^5 + 156*a^3*b^
7*c^4 + 2*a^3*b^8*c^3 - 222*a^3*b^9*c^2 + 136*a^4*b^4*c^6 - 388*a^4*b^5*c^5
 - 152*a^4*b^6*c^4 + 856*a^4*b^7*c^3 - 202*a^4*b^8*c^2 - 100*a^5*b^2*c^7 +
364*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 1362*a^5*b^5*c^4 - 115*a^5*b^6*c^3 + 63
5*a^5*b^7*c^2 - 340*a^6*b^2*c^6 + 904*a^6*b^3*c^5 + 583*a^6*b^4*c^4 - 564*a
^6*b^5*c^3 - 655*a^6*b^6*c^2 - 399*a^7*b^2*c^5 + 9*a^7*b^3*c^4 + 536*a^7*b^
4*c^3 + 612*a^7*b^5*c^2 - 37*a^8*b^2*c^4 - 524*a^8*b^3*c^3 - 354*a^8*b^4*c^
2 + 239*a^9*b^2*c^3 + 145*a^9*b^3*c^2 - 47*a^10*b^2*c^2))/a^8)*((8*a^4*c^6
 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3
 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^
4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(
4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*
(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^
5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c
^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3
 - 32*a^8*b^2*c^2)))^(1/2) + (2048*tan(x/2)*(4*a*b^12 + 20*b^12*c - 4*b^13
 - 4*a^2*b^11 + 4*a^3*b^10 - a^4*b^9 + a^5*b^8 + 12*a^4*c^9 - 44*a^5*c^8 + 2
*a^6*c^7 + 38*a^7*c^6 - 18*a^8*c^5 + 2*a^9*c^4 + 4*b^8*c^5 - 20*b^9*c^4 + 4
0*b^10*c^3 - 40*b^11*c^2 - 24*a*b^6*c^6 + 136*a*b^7*c^5 - 300*a*b^8*c^4 + 3
20*a*b^9*c^3 - 160*a*b^10*c^2 - 20*a^2*b^10*c + 20*a^3*b^9*c - 92*a^4*b*c^8
 - 31*a^4*b^8*c + 168*a^5*b*c^7 + 4*a^5*b^7*c + 2*a^6*b*c^6 - 8*a^6*b^6*c -
 84*a^7*b*c^5 + 26*a^8*b*c^4 + 44*a^2*b^4*c^7 - 300*a^2*b^5*c^6 + 764*a^2*b
^6*c^5 - 900*a^2*b^7*c^4 + 460*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 32*a^3*b^2*c^
8 + 272*a^3*b^3*c^7 - 840*a^3*b^4*c^6 + 1156*a^3*b^5*c^5 - 660*a^3*b^6*c^4
 + 72*a^3*b^7*c^3 + 8*a^3*b^8*c^2 + 384*a^4*b^2*c^7 - 704*a^4*b^3*c^6 + 541*
a^4*b^4*c^5 - 149*a^4*b^5*c^4 + 34*a^4*b^6*c^3 + 6*a^4*b^7*c^2 - 204*a^5*b^
2*c^6 + 96*a^5*b^3*c^5 + 41*a^5*b^4*c^4 - 132*a^5*b^5*c^3 + 82*a^5*b^6*c^2
 - 90*a^6*b^2*c^5 + 174*a^6*b^3*c^4 - 104*a^6*b^4*c^3 + 8*a^6*b^5*c^2 + 82*a
^7*b^2*c^4 - 40*a^7*b^3*c^3 + 20*a^7*b^4*c^2 - 16*a^8*b^2*c^3 + 24*a*b^11*c
))/a^8)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8
*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96
*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8
*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^
(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^
2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 1
6*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4

*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2)*1i)/((4096*(14*a^3*c^9 + a^4
*c^8 - 10*a^5*c^7 + 3*a^6*c^6 - 4*b^4*c^8 + 16*b^5*c^7 - 24*b^6*c^6 + 16*b^
7*c^5 - 4*b^8*c^4 + 4*a*b^2*c^9 - 28*a*b^3*c^8 + 56*a*b^4*c^7 - 40*a*b^5*c^
6 + 4*a*b^6*c^5 + 4*a*b^7*c^4 + 12*a^2*b*c^9 - 22*a^3*b*c^8 + 4*a^4*b*c^7 +
 6*a^5*b*c^6 - 2*a^6*b*c^5 - 48*a^2*b^2*c^8 + 48*a^2*b^3*c^7 - 8*a^2*b^4*c^
6 - 4*a^2*b^6*c^4 + 4*a^3*b^2*c^7 - 4*a^3*b^3*c^6 + 4*a^3*b^5*c^4 + 10*a^4*
b^2*c^6 - 8*a^4*b^3*c^5 - a^4*b^4*c^4 - a^5*b^2*c^5 + a^5*b^3*c^4))/a^8 + (
((((2048*(26*a^9*b^7 - 12*a^8*b^8 - 18*a^10*b^6 + 6*a^11*b^5 - 2*a^12*b^4 +
 48*a^10*c^6 + 176*a^11*c^5 + 176*a^12*c^4 + 16*a^13*c^3 - 32*a^14*c^2 + 20
*a^8*b^7*c + 74*a^9*b^6*c - 144*a^10*b*c^5 - 192*a^10*b^5*c - 352*a^11*b*c^
4 + 122*a^11*b^4*c - 144*a^12*b*c^3 - 40*a^12*b^3*c + 64*a^13*b*c^2 + 16*a^
13*b^2*c + 8*a^8*b^4*c^4 - 20*a^8*b^5*c^3 + 4*a^8*b^6*c^2 - 44*a^9*b^2*c^5
 + 116*a^9*b^3*c^4 + 10*a^9*b^4*c^3 - 182*a^9*b^5*c^2 - 148*a^10*b^2*c^4 + 4
96*a^10*b^3*c^3 - 50*a^10*b^4*c^2 - 260*a^11*b^2*c^3 + 388*a^11*b^3*c^2 - 2
04*a^12*b^2*c^2))/a^8 - (2048*tan(x/2)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7
*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^
2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4
*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) +
3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2)
 - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(
1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^
7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/
2)*(32*a^16*c + 8*a^10*b^7 - 24*a^11*b^6 + 32*a^12*b^5 - 32*a^13*b^4 + 24*a
^14*b^3 - 8*a^15*b^2 + 96*a^12*c^5 + 64*a^13*c^4 - 128*a^14*c^3 - 64*a^15*c
^2 - 8*a^10*b^6*c - 56*a^11*b^5*c - 32*a^12*b*c^4 + 184*a^12*b^4*c + 352*a^
13*b*c^3 - 200*a^13*b^3*c + 288*a^14*b*c^2 + 144*a^14*b^2*c + 8*a^10*b^4*c^
3 - 8*a^10*b^5*c^2 - 56*a^11*b^2*c^4 + 40*a^11*b^3*c^3 + 96*a^11*b^4*c^2 -
272*a^12*b^2*c^3 + 8*a^12*b^3*c^2 - 320*a^13*b^2*c^2 - 96*a^15*b*c))/a^8)*(
(8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10
*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*
c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*
b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4
*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/
2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4
 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*
a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) + (2048*tan(x/2)*(8*a^14*c + 8*a^4*b^
11 - 24*a^5*b^10 + 36*a^6*b^9 - 52*a^7*b^8 + 61*a^8*b^7 - 49*a^9*b^6 + 33*a
^10*b^5 - 17*a^11*b^4 + 6*a^12*b^3 - 2*a^13*b^2 + 72*a^8*c^7 - 136*a^9*c^6
 - 192*a^10*c^5 + 168*a^11*c^4 + 80*a^12*c^3 - 64*a^13*c^2 - 8*a^4*b^10*c -
72*a^5*b^9*c + 244*a^6*b^8*c - 308*a^7*b^7*c - 88*a^8*b*c^6 + 375*a^8*b^6*c
 + 496*a^9*b*c^5 - 416*a^9*b^5*c - 16*a^10*b*c^4 + 295*a^10*b^4*c - 328*a^1
1*b*c^3 - 178*a^11*b^3*c + 184*a^12*b*c^2 + 84*a^12*b^2*c + 8*a^4*b^8*c^3 -
 8*a^4*b^9*c^2 - 72*a^5*b^6*c^4 + 56*a^5*b^7*c^3 + 112*a^5*b^8*c^2 + 220*a^
6*b^4*c^5 - 140*a^6*b^5*c^4 - 424*a^6*b^6*c^3 + 80*a^6*b^7*c^2 - 256*a^7*b^
2*c^6 + 192*a^7*b^3*c^5 + 416*a^7*b^4*c^4 + 572*a^7*b^5*c^3 - 732*a^7*b^6*c

^2 + 64*a^8*b^2*c^5 - 1152*a^8*b^3*c^4 + 521*a^8*b^4*c^3 + 779*a^8*b^5*c^2 + 234*a^9*b^2*c^4 - 494*a^9*b^3*c^3 - 723*a^9*b^4*c^2 + 180*a^10*b^2*c^3 + 770*a^10*b^3*c^2 - 416*a^11*b^2*c^2 - 24*a^13*b*c))/a^8)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) - (2048*(26*a^3*b^11 - 12*a^2*b^12 - 30*a^4*b^10 + 29*a^5*b^9 - 20*a^6*b^8 + 10*a^7*b^7 - 4*a^8*b^6 + a^9*b^5 + 12*a^6*c^8 + 88*a^7*c^7 + 72*a^8*c^6 - 44*a^9*c^5 - 28*a^10*c^4 + 12*a^11*c^3 + 20*a^2*b^11*c + 98*a^3*b^10*c - 228*a^4*b^9*c + 251*a^5*b^8*c - 96*a^6*b*c^7 - 238*a^6*b^7*c - 200*a^7*b*c^6 + 154*a^7*b^6*c + 100*a^8*b*c^5 - 72*a^8*b^5*c + 112*a^9*b*c^4 + 27*a^9*b^4*c - 68*a^10*b*c^3 - 6*a^10*b^3*c + 8*a^11*b*c^2 + 8*a^2*b^8*c^4 - 20*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 60*a^3*b^6*c^5 + 156*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 222*a^3*b^9*c^2 + 136*a^4*b^4*c^6 - 388*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 856*a^4*b^7*c^3 - 202*a^4*b^8*c^2 - 100*a^5*b^2*c^7 + 364*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 1362*a^5*b^5*c^4 - 115*a^5*b^6*c^3 + 635*a^5*b^7*c^2 - 340*a^6*b^2*c^6 + 904*a^6*b^3*c^5 + 583*a^6*b^4*c^4 - 564*a^6*b^5*c^3 - 655*a^6*b^6*c^2 - 399*a^7*b^2*c^5 + 9*a^7*b^3*c^4 + 536*a^7*b^4*c^3 + 612*a^7*b^5*c^2 - 37*a^8*b^2*c^4 - 524*a^8*b^3*c^3 - 354*a^8*b^4*c^2 + 239*a^9*b^2*c^3 + 145*a^9*b^3*c^2 - 47*a^10*b^2*c^2))/a^8)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) - (2048*tan(x/2)*(4*a*b^12 + 20*b^12*c - 4*b^13 - 4*a^2*b^11 + 4*a^3*b^10 - a^4*b^9 + a^5*b^8 + 12*a^4*c^9 - 44*a^5*c^8 + 2*a^6*c^7 + 38*a^7*c^6 - 18*a^8*c^5 + 2*a^9*c^4 + 4*b^8*c^5 - 20*b^9*c^4 + 40*b^10*c^3 - 40*b^11*c^2 - 24*a*b^6*c^6 + 136*a*b^7*c^5 - 300*a*b^8*c^4 + 320*a*b^9*c^3 - 160*a*b^10*c^2 - 20*a^2*b^10*c + 20*a^3*b^9*c - 92*a^4*b*c^8 - 31*a^4*b^8*c + 168*a^5*b*c^7 + 4*a^5*b^7*c + 2*a^6*b*c^6 - 8*a^6*b^6*c - 84*a^7*b*c^5 + 26*a^8*b*c^4 + 44*a^2*b^4*c^7 - 300*a^2*b^5*c^6 + 764*a^2*b^6*c^5 - 900*a^2*b^7*c^4 + 460*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 32*a^3*b^2*c^8 + 272*a^3*b^3*c^7 - 840*a^3*b^4*c^6 + 1156*a^3*b^5*c^5 - 660*a^3*b^6*c^4 + 72*a^3*b^7*c^3 + 8*a^3*b^8*c^2 + 384*a^4*b^2*c^7 - 704*a^4*b^3*c^6 + 541*a^4*b^4*c^5 - 149*a^4*b^5*c^4 + 34*a^4*b^6*c^3 + 6*a^4*b^7*c^2 - 204*a^5*b^2*c^6 + 96*a^5*b^3*c^5 + 41*a^5*b^4*c^4 - 132*a^5*b^5*c^3 + 82*a^5*b^6*c^2 - 90*a^6*b^2*c^5 + 174*a^6*b^3*c^4 - 104*a^6*b^4*c^3 + 8*a^6*b^5*c^2 + 82*a^7*b^2*c^4 - 40*a^7*b^3*c^3 + 20*a^7*b^4*c^2 - 16*a^8*b^2*c^3 + 24*a*b^11*c

))/a^8)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8
*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96
*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8
*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^
(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^
2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 1
6*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4
*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) + (((((2048*(26*a^9*b^7 - 12
*a^8*b^8 - 18*a^10*b^6 + 6*a^11*b^5 - 2*a^12*b^4 + 48*a^10*c^6 + 176*a^11*c
^5 + 176*a^12*c^4 + 16*a^13*c^3 - 32*a^14*c^2 + 20*a^8*b^7*c + 74*a^9*b^6*c
 - 144*a^10*b*c^5 - 192*a^10*b^5*c - 352*a^11*b*c^4 + 122*a^11*b^4*c - 144*
a^12*b*c^3 - 40*a^12*b^3*c + 64*a^13*b*c^2 + 16*a^13*b^2*c + 8*a^8*b^4*c^4
 - 20*a^8*b^5*c^3 + 4*a^8*b^6*c^2 - 44*a^9*b^2*c^5 + 116*a^9*b^3*c^4 + 10*a^
9*b^4*c^3 - 182*a^9*b^5*c^2 - 148*a^10*b^2*c^4 + 496*a^10*b^3*c^3 - 50*a^10
*b^4*c^2 - 260*a^11*b^2*c^3 + 388*a^11*b^3*c^2 - 204*a^12*b^2*c^2))/a^8 + (
2048*tan(x/2)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2)
 + b^8*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^
5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12
*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^
2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*
c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b
^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a
^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2)*(32*a^16*c + 8*a^10*b^
7 - 24*a^11*b^6 + 32*a^12*b^5 - 32*a^13*b^4 + 24*a^14*b^3 - 8*a^15*b^2 + 96
*a^12*c^5 + 64*a^13*c^4 - 128*a^14*c^3 - 64*a^15*c^2 - 8*a^10*b^6*c - 56*a^
11*b^5*c - 32*a^12*b*c^4 + 184*a^12*b^4*c + 352*a^13*b*c^3 - 200*a^13*b^3*c
 + 288*a^14*b*c^2 + 144*a^14*b^2*c + 8*a^10*b^4*c^3 - 8*a^10*b^5*c^2 - 56*a
^11*b^2*c^4 + 40*a^11*b^3*c^3 + 96*a^11*b^4*c^2 - 272*a^12*b^2*c^3 + 8*a^12
*b^3*c^2 - 320*a^13*b^2*c^2 - 96*a^15*b*c))/a^8)*((8*a^4*c^6 - b^10 + 8*a^5
c^5 - b^7(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c
^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^
5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3
)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2
)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c -
 b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c
^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*
c^2)))^(1/2) - (2048*tan(x/2)*(8*a^14*c + 8*a^4*b^11 - 24*a^5*b^10 + 36*a^6
*b^9 - 52*a^7*b^8 + 61*a^8*b^7 - 49*a^9*b^6 + 33*a^10*b^5 - 17*a^11*b^4 + 6
*a^12*b^3 - 2*a^13*b^2 + 72*a^8*c^7 - 136*a^9*c^6 - 192*a^10*c^5 + 168*a^11
*c^4 + 80*a^12*c^3 - 64*a^13*c^2 - 8*a^4*b^10*c - 72*a^5*b^9*c + 244*a^6*b^
8*c - 308*a^7*b^7*c - 88*a^8*b*c^6 + 375*a^8*b^6*c + 496*a^9*b*c^5 - 416*a^
9*b^5*c - 16*a^10*b*c^4 + 295*a^10*b^4*c - 328*a^11*b*c^3 - 178*a^11*b^3*c
+ 184*a^12*b*c^2 + 84*a^12*b^2*c + 8*a^4*b^8*c^3 - 8*a^4*b^9*c^2 - 72*a^5*b
^6*c^4 + 56*a^5*b^7*c^3 + 112*a^5*b^8*c^2 + 220*a^6*b^4*c^5 - 140*a^6*b^5*c
^4 - 424*a^6*b^6*c^3 + 80*a^6*b^7*c^2 - 256*a^7*b^2*c^6 + 192*a^7*b^3*c^5 +

416*a^7*b^4*c^4 + 572*a^7*b^5*c^3 - 732*a^7*b^6*c^2 + 64*a^8*b^2*c^5 - 115
2*a^8*b^3*c^4 + 521*a^8*b^4*c^3 + 779*a^8*b^5*c^2 + 234*a^9*b^2*c^4 - 494*a
^9*b^3*c^3 - 723*a^9*b^4*c^2 + 180*a^10*b^2*c^3 + 770*a^10*b^3*c^2 - 416*a^
11*b^2*c^2 - 24*a^13*b*c))/a^8)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a
*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c
^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c -
b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b
c^4(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a
^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(
2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c
 - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) - (2
048*(26*a^3*b^11 - 12*a^2*b^12 - 30*a^4*b^10 + 29*a^5*b^9 - 20*a^6*b^8 + 10
*a^7*b^7 - 4*a^8*b^6 + a^9*b^5 + 12*a^6*c^8 + 88*a^7*c^7 + 72*a^8*c^6 - 44*
a^9*c^5 - 28*a^10*c^4 + 12*a^11*c^3 + 20*a^2*b^11*c + 98*a^3*b^10*c - 228*a
^4*b^9*c + 251*a^5*b^8*c - 96*a^6*b*c^7 - 238*a^6*b^7*c - 200*a^7*b*c^6 + 1
54*a^7*b^6*c + 100*a^8*b*c^5 - 72*a^8*b^5*c + 112*a^9*b*c^4 + 27*a^9*b^4*c
 - 68*a^10*b*c^3 - 6*a^10*b^3*c + 8*a^11*b*c^2 + 8*a^2*b^8*c^4 - 20*a^2*b^9*
c^3 + 4*a^2*b^10*c^2 - 60*a^3*b^6*c^5 + 156*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 2
22*a^3*b^9*c^2 + 136*a^4*b^4*c^6 - 388*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 856*
a^4*b^7*c^3 - 202*a^4*b^8*c^2 - 100*a^5*b^2*c^7 + 364*a^5*b^3*c^6 + 394*a^5
*b^4*c^5 - 1362*a^5*b^5*c^4 - 115*a^5*b^6*c^3 + 635*a^5*b^7*c^2 - 340*a^6*b
^2*c^6 + 904*a^6*b^3*c^5 + 583*a^6*b^4*c^4 - 564*a^6*b^5*c^3 - 655*a^6*b^6*
c^2 - 399*a^7*b^2*c^5 + 9*a^7*b^3*c^4 + 536*a^7*b^4*c^3 + 612*a^7*b^5*c^2 -
 37*a^8*b^2*c^4 - 524*a^8*b^3*c^3 - 354*a^8*b^4*c^2 + 239*a^9*b^2*c^3 + 145
*a^9*b^3*c^2 - 47*a^10*b^2*c^2))/a^8)*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*
(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2
*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*
a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3
*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2)
 - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1
/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7
*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2
) + (2048*tan(x/2)*(4*a*b^12 + 20*b^12*c - 4*b^13 - 4*a^2*b^11 + 4*a^3*b^10
 - a^4*b^9 + a^5*b^8 + 12*a^4*c^9 - 44*a^5*c^8 + 2*a^6*c^7 + 38*a^7*c^6 - 1
8*a^8*c^5 + 2*a^9*c^4 + 4*b^8*c^5 - 20*b^9*c^4 + 40*b^10*c^3 - 40*b^11*c^2
 - 24*a*b^6*c^6 + 136*a*b^7*c^5 - 300*a*b^8*c^4 + 320*a*b^9*c^3 - 160*a*b^10
*c^2 - 20*a^2*b^10*c + 20*a^3*b^9*c - 92*a^4*b*c^8 - 31*a^4*b^8*c + 168*a^5
*b*c^7 + 4*a^5*b^7*c + 2*a^6*b*c^6 - 8*a^6*b^6*c - 84*a^7*b*c^5 + 26*a^8*b*
c^4 + 44*a^2*b^4*c^7 - 300*a^2*b^5*c^6 + 764*a^2*b^6*c^5 - 900*a^2*b^7*c^4
 + 460*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 32*a^3*b^2*c^8 + 272*a^3*b^3*c^7 - 840
*a^3*b^4*c^6 + 1156*a^3*b^5*c^5 - 660*a^3*b^6*c^4 + 72*a^3*b^7*c^3 + 8*a^3*
b^8*c^2 + 384*a^4*b^2*c^7 - 704*a^4*b^3*c^6 + 541*a^4*b^4*c^5 - 149*a^4*b^5
*c^4 + 34*a^4*b^6*c^3 + 6*a^4*b^7*c^2 - 204*a^5*b^2*c^6 + 96*a^5*b^3*c^5 +
41*a^5*b^4*c^4 - 132*a^5*b^5*c^3 + 82*a^5*b^6*c^2 - 90*a^6*b^2*c^5 + 174*a^
6*b^3*c^4 - 104*a^6*b^4*c^3 + 8*a^6*b^5*c^2 + 82*a^7*b^2*c^4 - 40*a^7*b^3*c

^3 + 20*a^7*b^4*c^2 - 16*a^8*b^2*c^3 + 24*a*b^11*c))/a^8)*((8*a^4*c^6 - b^1
0 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*
a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2
*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c
 - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*
a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(
-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 +
16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32
*a^8*b^2*c^2)))^(1/2)))*((8*a^4*c^6 - b^10 + 8*a^5*c^5 - b^7*(-(4*a*c - b^2
)^3)^(1/2) + b^8*c^2 - 10*a*b^6*c^3 + 33*a^2*b^4*c^4 - 52*a^2*b^6*c^2 - 38*
a^3*b^2*c^5 + 96*a^3*b^4*c^3 - 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^
(1/2) + 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(
4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c
^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b
^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9
*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2)*2i - atan(((
((((2048*(26*a^9*b^7 - 12*a^8*b^8 - 18*a^10*b^6 + 6*a^11*b^5 - 2*a^12*b^4 +
 48*a^10*c^6 + 176*a^11*c^5 + 176*a^12*c^4 + 16*a^13*c^3 - 32*a^14*c^2 + 20
*a^8*b^7*c + 74*a^9*b^6*c - 144*a^10*b*c^5 - 192*a^10*b^5*c - 352*a^11*b*c^
4 + 122*a^11*b^4*c - 144*a^12*b*c^3 - 40*a^12*b^3*c + 64*a^13*b*c^2 + 16*a^
13*b^2*c + 8*a^8*b^4*c^4 - 20*a^8*b^5*c^3 + 4*a^8*b^6*c^2 - 44*a^9*b^2*c^5
+ 116*a^9*b^3*c^4 + 10*a^9*b^4*c^3 - 182*a^9*b^5*c^2 - 148*a^10*b^2*c^4 + 4
96*a^10*b^3*c^3 - 50*a^10*b^4*c^2 - 260*a^11*b^2*c^3 + 388*a^11*b^3*c^2 - 2
04*a^12*b^2*c^2))/a^8 - (2048*tan(x/2)*(-(b^10 - 8*a^4*c^6 - 8*a^5*c^5 - b^
7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33*a^2*b^4*c^4 + 52*a
^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 + b^5*c^2*(-(
4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) +
 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2
) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^
(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a
^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1
/2)*(32*a^16*c + 8*a^10*b^7 - 24*a^11*b^6 + 32*a^12*b^5 - 32*a^13*b^4 + 24*
a^14*b^3 - 8*a^15*b^2 + 96*a^12*c^5 + 64*a^13*c^4 - 128*a^14*c^3 - 64*a^15*
c^2 - 8*a^10*b^6*c - 56*a^11*b^5*c - 32*a^12*b*c^4 + 184*a^12*b^4*c + 352*a
^13*b*c^3 - 200*a^13*b^3*c + 288*a^14*b*c^2 + 144*a^14*b^2*c + 8*a^10*b^4*c
^3 - 8*a^10*b^5*c^2 - 56*a^11*b^2*c^4 + 40*a^11*b^3*c^3 + 96*a^11*b^4*c^2 -
 272*a^12*b^2*c^3 + 8*a^12*b^3*c^2 - 320*a^13*b^2*c^2 - 96*a^15*b*c))/a^8)*
(-(b^10 - 8*a^4*c^6 - 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 +
10*a*b^6*c^3 - 33*a^2*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^
4*c^3 + 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*
a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) +
 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(
1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c
^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 -
8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) + (2048*tan(x/2)*(8*a^14*c + 8*a^4*

b^11 - 24*a^5*b^10 + 36*a^6*b^9 - 52*a^7*b^8 + 61*a^8*b^7 - 49*a^9*b^6 + 33
*a^10*b^5 - 17*a^11*b^4 + 6*a^12*b^3 - 2*a^13*b^2 + 72*a^8*c^7 - 136*a^9*c^
6 - 192*a^10*c^5 + 168*a^11*c^4 + 80*a^12*c^3 - 64*a^13*c^2 - 8*a^4*b^10*c
- 72*a^5*b^9*c + 244*a^6*b^8*c - 308*a^7*b^7*c - 88*a^8*b*c^6 + 375*a^8*b^6
*c + 496*a^9*b*c^5 - 416*a^9*b^5*c - 16*a^10*b*c^4 + 295*a^10*b^4*c - 328*a
^11*b*c^3 - 178*a^11*b^3*c + 184*a^12*b*c^2 + 84*a^12*b^2*c + 8*a^4*b^8*c^3
 - 8*a^4*b^9*c^2 - 72*a^5*b^6*c^4 + 56*a^5*b^7*c^3 + 112*a^5*b^8*c^2 + 220*
a^6*b^4*c^5 - 140*a^6*b^5*c^4 - 424*a^6*b^6*c^3 + 80*a^6*b^7*c^2 - 256*a^7*
b^2*c^6 + 192*a^7*b^3*c^5 + 416*a^7*b^4*c^4 + 572*a^7*b^5*c^3 - 732*a^7*b^6
*c^2 + 64*a^8*b^2*c^5 - 1152*a^8*b^3*c^4 + 521*a^8*b^4*c^3 + 779*a^8*b^5*c^
2 + 234*a^9*b^2*c^4 - 494*a^9*b^3*c^3 - 723*a^9*b^4*c^2 + 180*a^10*b^2*c^3
+ 770*a^10*b^3*c^2 - 416*a^11*b^2*c^2 - 24*a^13*b*c))/a^8)*(-(b^10 - 8*a^4*
c^6 - 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 3
3*a^2*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b
^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a
*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(
4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c
*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3
+ 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 -
32*a^8*b^2*c^2)))^(1/2) - (2048*(26*a^3*b^11 - 12*a^2*b^12 - 30*a^4*b^10 +
29*a^5*b^9 - 20*a^6*b^8 + 10*a^7*b^7 - 4*a^8*b^6 + a^9*b^5 + 12*a^6*c^8 + 8
8*a^7*c^7 + 72*a^8*c^6 - 44*a^9*c^5 - 28*a^10*c^4 + 12*a^11*c^3 + 20*a^2*b^
11*c + 98*a^3*b^10*c - 228*a^4*b^9*c + 251*a^5*b^8*c - 96*a^6*b*c^7 - 238*a
^6*b^7*c - 200*a^7*b*c^6 + 154*a^7*b^6*c + 100*a^8*b*c^5 - 72*a^8*b^5*c + 1
12*a^9*b*c^4 + 27*a^9*b^4*c - 68*a^10*b*c^3 - 6*a^10*b^3*c + 8*a^11*b*c^2 +
 8*a^2*b^8*c^4 - 20*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 60*a^3*b^6*c^5 + 156*a^3
*b^7*c^4 + 2*a^3*b^8*c^3 - 222*a^3*b^9*c^2 + 136*a^4*b^4*c^6 - 388*a^4*b^5*
c^5 - 152*a^4*b^6*c^4 + 856*a^4*b^7*c^3 - 202*a^4*b^8*c^2 - 100*a^5*b^2*c^7
 + 364*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 1362*a^5*b^5*c^4 - 115*a^5*b^6*c^3 +
 635*a^5*b^7*c^2 - 340*a^6*b^2*c^6 + 904*a^6*b^3*c^5 + 583*a^6*b^4*c^4 - 56
4*a^6*b^5*c^3 - 655*a^6*b^6*c^2 - 399*a^7*b^2*c^5 + 9*a^7*b^3*c^4 + 536*a^7
*b^4*c^3 + 612*a^7*b^5*c^2 - 37*a^8*b^2*c^4 - 524*a^8*b^3*c^3 - 354*a^8*b^4
*c^2 + 239*a^9*b^2*c^3 + 145*a^9*b^3*c^2 - 47*a^10*b^2*c^2))/a^8)*(-(b^10 -
 8*a^4*c^6 - 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*
c^3 - 33*a^2*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 6
6*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3
*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*
c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*
a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a
^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2
*c^3 - 32*a^8*b^2*c^2)))^(1/2) - (2048*tan(x/2)*(4*a*b^12 + 20*b^12*c - 4*b
^13 - 4*a^2*b^11 + 4*a^3*b^10 - a^4*b^9 + a^5*b^8 + 12*a^4*c^9 - 44*a^5*c^8
 + 2*a^6*c^7 + 38*a^7*c^6 - 18*a^8*c^5 + 2*a^9*c^4 + 4*b^8*c^5 - 20*b^9*c^4
 + 40*b^10*c^3 - 40*b^11*c^2 - 24*a*b^6*c^6 + 136*a*b^7*c^5 - 300*a*b^8*c^4
 + 320*a*b^9*c^3 - 160*a*b^10*c^2 - 20*a^2*b^10*c + 20*a^3*b^9*c - 92*a^4*b

```
*c^8 - 31*a^4*b^8*c + 168*a^5*b*c^7 + 4*a^5*b^7*c + 2*a^6*b*c^6 - 8*a^6*b^6
*c - 84*a^7*b*c^5 + 26*a^8*b*c^4 + 44*a^2*b^4*c^7 - 300*a^2*b^5*c^6 + 764*a
^2*b^6*c^5 - 900*a^2*b^7*c^4 + 460*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 32*a^3*b^
2*c^8 + 272*a^3*b^3*c^7 - 840*a^3*b^4*c^6 + 1156*a^3*b^5*c^5 - 660*a^3*b^6*
c^4 + 72*a^3*b^7*c^3 + 8*a^3*b^8*c^2 + 384*a^4*b^2*c^7 - 704*a^4*b^3*c^6 +
541*a^4*b^4*c^5 - 149*a^4*b^5*c^4 + 34*a^4*b^6*c^3 + 6*a^4*b^7*c^2 - 204*a^
5*b^2*c^6 + 96*a^5*b^3*c^5 + 41*a^5*b^4*c^4 - 132*a^5*b^5*c^3 + 82*a^5*b^6*
c^2 - 90*a^6*b^2*c^5 + 174*a^6*b^3*c^4 - 104*a^6*b^4*c^3 + 8*a^6*b^5*c^2 +
82*a^7*b^2*c^4 - 40*a^7*b^3*c^3 + 20*a^7*b^4*c^2 - 16*a^8*b^2*c^3 + 24*a*b^
11*c))/a^8)*(-(b^10 - 8*a^4*c^6 - 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2)
- b^8*c^2 + 10*a*b^6*c^3 - 33*a^2*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5
 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) - 12*
a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2
)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c
 - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^
6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^
6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2)*1i - (((((2048*(26*a^9*
b^7 - 12*a^8*b^8 - 18*a^10*b^6 + 6*a^11*b^5 - 2*a^12*b^4 + 48*a^10*c^6 + 17
6*a^11*c^5 + 176*a^12*c^4 + 16*a^13*c^3 - 32*a^14*c^2 + 20*a^8*b^7*c + 74*a
^9*b^6*c - 144*a^10*b*c^5 - 192*a^10*b^5*c - 352*a^11*b*c^4 + 122*a^11*b^4*
c - 144*a^12*b*c^3 - 40*a^12*b^3*c + 64*a^13*b*c^2 + 16*a^13*b^2*c + 8*a^8*
b^4*c^4 - 20*a^8*b^5*c^3 + 4*a^8*b^6*c^2 - 44*a^9*b^2*c^5 + 116*a^9*b^3*c^4
 + 10*a^9*b^4*c^3 - 182*a^9*b^5*c^2 - 148*a^10*b^2*c^4 + 496*a^10*b^3*c^3 -
 50*a^10*b^4*c^2 - 260*a^11*b^2*c^3 + 388*a^11*b^3*c^2 - 204*a^12*b^2*c^2))
/a^8 + (2048*tan(x/2)*(-(b^10 - 8*a^4*c^6 - 8*a^5*c^5 - b^7*(-(4*a*c - b^2)
^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33*a^2*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a
^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(
1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4
*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^
2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^
4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*
b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2)*(32*a^16*c +
8*a^10*b^7 - 24*a^11*b^6 + 32*a^12*b^5 - 32*a^13*b^4 + 24*a^14*b^3 - 8*a^15
*b^2 + 96*a^12*c^5 + 64*a^13*c^4 - 128*a^14*c^3 - 64*a^15*c^2 - 8*a^10*b^6*
c - 56*a^11*b^5*c - 32*a^12*b*c^4 + 184*a^12*b^4*c + 352*a^13*b*c^3 - 200*a
^13*b^3*c + 288*a^14*b*c^2 + 144*a^14*b^2*c + 8*a^10*b^4*c^3 - 8*a^10*b^5*c
^2 - 56*a^11*b^2*c^4 + 40*a^11*b^3*c^3 + 96*a^11*b^4*c^2 - 272*a^12*b^2*c^3
 + 8*a^12*b^3*c^2 - 320*a^13*b^2*c^2 - 96*a^15*b*c))/a^8)*(-(b^10 - 8*a^4*c
^6 - 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33
*a^2*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^
2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*
c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4
*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*
(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 +
 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 3
```

2*a^8*b^2*c^2)))^(1/2) - (2048*tan(x/2)*(8*a^14*c + 8*a^4*b^11 - 24*a^5*b^1
0 + 36*a^6*b^9 - 52*a^7*b^8 + 61*a^8*b^7 - 49*a^9*b^6 + 33*a^10*b^5 - 17*a^
11*b^4 + 6*a^12*b^3 - 2*a^13*b^2 + 72*a^8*c^7 - 136*a^9*c^6 - 192*a^10*c^5
+ 168*a^11*c^4 + 80*a^12*c^3 - 64*a^13*c^2 - 8*a^4*b^10*c - 72*a^5*b^9*c +
244*a^6*b^8*c - 308*a^7*b^7*c - 88*a^8*b*c^6 + 375*a^8*b^6*c + 496*a^9*b*c^
5 - 416*a^9*b^5*c - 16*a^10*b*c^4 + 295*a^10*b^4*c - 328*a^11*b*c^3 - 178*a
^11*b^3*c + 184*a^12*b*c^2 + 84*a^12*b^2*c + 8*a^4*b^8*c^3 - 8*a^4*b^9*c^2
- 72*a^5*b^6*c^4 + 56*a^5*b^7*c^3 + 112*a^5*b^8*c^2 + 220*a^6*b^4*c^5 - 140
*a^6*b^5*c^4 - 424*a^6*b^6*c^3 + 80*a^6*b^7*c^2 - 256*a^7*b^2*c^6 + 192*a^7
*b^3*c^5 + 416*a^7*b^4*c^4 + 572*a^7*b^5*c^3 - 732*a^7*b^6*c^2 + 64*a^8*b^2
*c^5 - 1152*a^8*b^3*c^4 + 521*a^8*b^4*c^3 + 779*a^8*b^5*c^2 + 234*a^9*b^2*c
^4 - 494*a^9*b^3*c^3 - 723*a^9*b^4*c^2 + 180*a^10*b^2*c^3 + 770*a^10*b^3*c^
2 - 416*a^11*b^2*c^2 - 24*a^13*b*c))/a^8)*(-(b^10 - 8*a^4*c^6 - 8*a^5*c^5 -
b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33*a^2*b^4*c^4 + 5
2*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 + b^5*c^2*
(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2
) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(
1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^
3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 1
0*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))
^(1/2) - (2048*(26*a^3*b^11 - 12*a^2*b^12 - 30*a^4*b^10 + 29*a^5*b^9 - 20*a
^6*b^8 + 10*a^7*b^7 - 4*a^8*b^6 + a^9*b^5 + 12*a^6*c^8 + 88*a^7*c^7 + 72*a^
8*c^6 - 44*a^9*c^5 - 28*a^10*c^4 + 12*a^11*c^3 + 20*a^2*b^11*c + 98*a^3*b^1
0*c - 228*a^4*b^9*c + 251*a^5*b^8*c - 96*a^6*b*c^7 - 238*a^6*b^7*c - 200*a^
7*b*c^6 + 154*a^7*b^6*c + 100*a^8*b*c^5 - 72*a^8*b^5*c + 112*a^9*b*c^4 + 27
*a^9*b^4*c - 68*a^10*b*c^3 - 6*a^10*b^3*c + 8*a^11*b*c^2 + 8*a^2*b^8*c^4 -
20*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 60*a^3*b^6*c^5 + 156*a^3*b^7*c^4 + 2*a^3*
b^8*c^3 - 222*a^3*b^9*c^2 + 136*a^4*b^4*c^6 - 388*a^4*b^5*c^5 - 152*a^4*b^6
*c^4 + 856*a^4*b^7*c^3 - 202*a^4*b^8*c^2 - 100*a^5*b^2*c^7 + 364*a^5*b^3*c^
6 + 394*a^5*b^4*c^5 - 1362*a^5*b^5*c^4 - 115*a^5*b^6*c^3 + 635*a^5*b^7*c^2
- 340*a^6*b^2*c^6 + 904*a^6*b^3*c^5 + 583*a^6*b^4*c^4 - 564*a^6*b^5*c^3 - 6
55*a^6*b^6*c^2 - 399*a^7*b^2*c^5 + 9*a^7*b^3*c^4 + 536*a^7*b^4*c^3 + 612*a^
7*b^5*c^2 - 37*a^8*b^2*c^4 - 524*a^8*b^3*c^3 - 354*a^8*b^4*c^2 + 239*a^9*b^
2*c^3 + 145*a^9*b^3*c^2 - 47*a^10*b^2*c^2))/a^8)*(-(b^10 - 8*a^4*c^6 - 8*a^
5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33*a^2*b^4*
c^4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 + b
^5*c^2*(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^
3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^
2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c
- b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*
c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2
*c^2)))^(1/2) + (2048*tan(x/2)*(4*a*b^12 + 20*b^12*c - 4*b^13 - 4*a^2*b^11
+ 4*a^3*b^10 - a^4*b^9 + a^5*b^8 + 12*a^4*c^9 - 44*a^5*c^8 + 2*a^6*c^7 + 38
*a^7*c^6 - 18*a^8*c^5 + 2*a^9*c^4 + 4*b^8*c^5 - 20*b^9*c^4 + 40*b^10*c^3 -
40*b^11*c^2 - 24*a*b^6*c^6 + 136*a*b^7*c^5 - 300*a*b^8*c^4 + 320*a*b^9*c^3

- 160*a*b^10*c^2 - 20*a^2*b^10*c + 20*a^3*b^9*c - 92*a^4*b*c^8 - 31*a^4*b^8
*c + 168*a^5*b*c^7 + 4*a^5*b^7*c + 2*a^6*b*c^6 - 8*a^6*b^6*c - 84*a^7*b*c^5
 + 26*a^8*b*c^4 + 44*a^2*b^4*c^7 - 300*a^2*b^5*c^6 + 764*a^2*b^6*c^5 - 900*
a^2*b^7*c^4 + 460*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 32*a^3*b^2*c^8 + 272*a^3*b
^3*c^7 - 840*a^3*b^4*c^6 + 1156*a^3*b^5*c^5 - 660*a^3*b^6*c^4 + 72*a^3*b^7*
c^3 + 8*a^3*b^8*c^2 + 384*a^4*b^2*c^7 - 704*a^4*b^3*c^6 + 541*a^4*b^4*c^5 -
 149*a^4*b^5*c^4 + 34*a^4*b^6*c^3 + 6*a^4*b^7*c^2 - 204*a^5*b^2*c^6 + 96*a^
5*b^3*c^5 + 41*a^5*b^4*c^4 - 132*a^5*b^5*c^3 + 82*a^5*b^6*c^2 - 90*a^6*b^2*
c^5 + 174*a^6*b^3*c^4 - 104*a^6*b^4*c^3 + 8*a^6*b^5*c^2 + 82*a^7*b^2*c^4 -
40*a^7*b^3*c^3 + 20*a^7*b^4*c^2 - 16*a^8*b^2*c^3 + 24*a*b^11*c))/a^8)*(-(b^
10 - 8*a^4*c^6 - 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*
b^6*c^3 - 33*a^2*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3
 + 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3
c^3(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^
3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2)
+ 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 +
32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7
*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2)*1i)/((4096*(14*a^3*c^9 + a^4*c^8 - 10*a^
5*c^7 + 3*a^6*c^6 - 4*b^4*c^8 + 16*b^5*c^7 - 24*b^6*c^6 + 16*b^7*c^5 - 4*b^
8*c^4 + 4*a*b^2*c^9 - 28*a*b^3*c^8 + 56*a*b^4*c^7 - 40*a*b^5*c^6 + 4*a*b^6*
c^5 + 4*a*b^7*c^4 + 12*a^2*b*c^9 - 22*a^3*b*c^8 + 4*a^4*b*c^7 + 6*a^5*b*c^6
 - 2*a^6*b*c^5 - 48*a^2*b^2*c^8 + 48*a^2*b^3*c^7 - 8*a^2*b^4*c^6 - 4*a^2*b^
6*c^4 + 4*a^3*b^2*c^7 - 4*a^3*b^3*c^6 + 4*a^3*b^5*c^4 + 10*a^4*b^2*c^6 - 8*
a^4*b^3*c^5 - a^4*b^4*c^4 - a^5*b^2*c^5 + a^5*b^3*c^4))/a^8 + (((((2048*(26
*a^9*b^7 - 12*a^8*b^8 - 18*a^10*b^6 + 6*a^11*b^5 - 2*a^12*b^4 + 48*a^10*c^6
 + 176*a^11*c^5 + 176*a^12*c^4 + 16*a^13*c^3 - 32*a^14*c^2 + 20*a^8*b^7*c +
 74*a^9*b^6*c - 144*a^10*b*c^5 - 192*a^10*b^5*c - 352*a^11*b*c^4 + 122*a^11
*b^4*c - 144*a^12*b*c^3 - 40*a^12*b^3*c + 64*a^13*b*c^2 + 16*a^13*b^2*c + 8
*a^8*b^4*c^4 - 20*a^8*b^5*c^3 + 4*a^8*b^6*c^2 - 44*a^9*b^2*c^5 + 116*a^9*b^
3*c^4 + 10*a^9*b^4*c^3 - 182*a^9*b^5*c^2 - 148*a^10*b^2*c^4 + 496*a^10*b^3*
c^3 - 50*a^10*b^4*c^2 - 260*a^11*b^2*c^3 + 388*a^11*b^3*c^2 - 204*a^12*b^2*
c^2))/a^8 - (2048*tan(x/2)*(-(b^10 - 8*a^4*c^6 - 8*a^5*c^5 - b^7*(-(4*a*c -
 b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33*a^2*b^4*c^4 + 52*a^2*b^6*c^2 +
 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)
^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4
*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b
^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a
^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8
*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2)*(32*a^16
*c + 8*a^10*b^7 - 24*a^11*b^6 + 32*a^12*b^5 - 32*a^13*b^4 + 24*a^14*b^3 - 8
*a^15*b^2 + 96*a^12*c^5 + 64*a^13*c^4 - 128*a^14*c^3 - 64*a^15*c^2 - 8*a^10
*b^6*c - 56*a^11*b^5*c - 32*a^12*b*c^4 + 184*a^12*b^4*c + 352*a^13*b*c^3 -
200*a^13*b^3*c + 288*a^14*b*c^2 + 144*a^14*b^2*c + 8*a^10*b^4*c^3 - 8*a^10*
b^5*c^2 - 56*a^11*b^2*c^4 + 40*a^11*b^3*c^3 + 96*a^11*b^4*c^2 - 272*a^12*b^
2*c^3 + 8*a^12*b^3*c^2 - 320*a^13*b^2*c^2 - 96*a^15*b*c))/a^8)*(-(b^10 - 8*

a^4*c^6 - 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3
 - 33*a^2*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a
^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-
(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3
*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b
^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*
c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^
3 - 32*a^8*b^2*c^2)))^(1/2) + (2048*tan(x/2)*(8*a^14*c + 8*a^4*b^11 - 24*a^
5*b^10 + 36*a^6*b^9 - 52*a^7*b^8 + 61*a^8*b^7 - 49*a^9*b^6 + 33*a^10*b^5 -
17*a^11*b^4 + 6*a^12*b^3 - 2*a^13*b^2 + 72*a^8*c^7 - 136*a^9*c^6 - 192*a^10
*c^5 + 168*a^11*c^4 + 80*a^12*c^3 - 64*a^13*c^2 - 8*a^4*b^10*c - 72*a^5*b^9
*c + 244*a^6*b^8*c - 308*a^7*b^7*c - 88*a^8*b*c^6 + 375*a^8*b^6*c + 496*a^9
*b*c^5 - 416*a^9*b^5*c - 16*a^10*b*c^4 + 295*a^10*b^4*c - 328*a^11*b*c^3 -
178*a^11*b^3*c + 184*a^12*b*c^2 + 84*a^12*b^2*c + 8*a^4*b^8*c^3 - 8*a^4*b^9
*c^2 - 72*a^5*b^6*c^4 + 56*a^5*b^7*c^3 + 112*a^5*b^8*c^2 + 220*a^6*b^4*c^5
 - 140*a^6*b^5*c^4 - 424*a^6*b^6*c^3 + 80*a^6*b^7*c^2 - 256*a^7*b^2*c^6 + 19
2*a^7*b^3*c^5 + 416*a^7*b^4*c^4 + 572*a^7*b^5*c^3 - 732*a^7*b^6*c^2 + 64*a^
8*b^2*c^5 - 1152*a^8*b^3*c^4 + 521*a^8*b^4*c^3 + 779*a^8*b^5*c^2 + 234*a^9*
b^2*c^4 - 494*a^9*b^3*c^3 - 723*a^9*b^4*c^2 + 180*a^10*b^2*c^3 + 770*a^10*b
^3*c^2 - 416*a^11*b^2*c^2 - 24*a^13*b*c))/a^8)*(-(b^10 - 8*a^4*c^6 - 8*a^5*
c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33*a^2*b^4*c^
4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 + b^5
c^2(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)
^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)
^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c -
b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^
2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c
^2)))^(1/2) - (2048*(26*a^3*b^11 - 12*a^2*b^12 - 30*a^4*b^10 + 29*a^5*b^9 -
 20*a^6*b^8 + 10*a^7*b^7 - 4*a^8*b^6 + a^9*b^5 + 12*a^6*c^8 + 88*a^7*c^7 +
72*a^8*c^6 - 44*a^9*c^5 - 28*a^10*c^4 + 12*a^11*c^3 + 20*a^2*b^11*c + 98*a^
3*b^10*c - 228*a^4*b^9*c + 251*a^5*b^8*c - 96*a^6*b*c^7 - 238*a^6*b^7*c - 2
00*a^7*b*c^6 + 154*a^7*b^6*c + 100*a^8*b*c^5 - 72*a^8*b^5*c + 112*a^9*b*c^4
 + 27*a^9*b^4*c - 68*a^10*b*c^3 - 6*a^10*b^3*c + 8*a^11*b*c^2 + 8*a^2*b^8*c
^4 - 20*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 60*a^3*b^6*c^5 + 156*a^3*b^7*c^4 + 2
*a^3*b^8*c^3 - 222*a^3*b^9*c^2 + 136*a^4*b^4*c^6 - 388*a^4*b^5*c^5 - 152*a^
4*b^6*c^4 + 856*a^4*b^7*c^3 - 202*a^4*b^8*c^2 - 100*a^5*b^2*c^7 + 364*a^5*b
^3*c^6 + 394*a^5*b^4*c^5 - 1362*a^5*b^5*c^4 - 115*a^5*b^6*c^3 + 635*a^5*b^7
*c^2 - 340*a^6*b^2*c^6 + 904*a^6*b^3*c^5 + 583*a^6*b^4*c^4 - 564*a^6*b^5*c^
3 - 655*a^6*b^6*c^2 - 399*a^7*b^2*c^5 + 9*a^7*b^3*c^4 + 536*a^7*b^4*c^3 + 6
12*a^7*b^5*c^2 - 37*a^8*b^2*c^4 - 524*a^8*b^3*c^3 - 354*a^8*b^4*c^2 + 239*a
^9*b^2*c^3 + 145*a^9*b^3*c^2 - 47*a^10*b^2*c^2))/a^8)*(-(b^10 - 8*a^4*c^6 -
 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33*a^2
*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^
4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c -
b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c

- b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4
*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*
a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^
8*b^2*c^2)))^(1/2) - (2048*tan(x/2)*(4*a*b^12 + 20*b^12*c - 4*b^13 - 4*a^2*
b^11 + 4*a^3*b^10 - a^4*b^9 + a^5*b^8 + 12*a^4*c^9 - 44*a^5*c^8 + 2*a^6*c^7
+ 38*a^7*c^6 - 18*a^8*c^5 + 2*a^9*c^4 + 4*b^8*c^5 - 20*b^9*c^4 + 40*b^10*c
^3 - 40*b^11*c^2 - 24*a*b^6*c^6 + 136*a*b^7*c^5 - 300*a*b^8*c^4 + 320*a*b^9
*c^3 - 160*a*b^10*c^2 - 20*a^2*b^10*c + 20*a^3*b^9*c - 92*a^4*b*c^8 - 31*a^
4*b^8*c + 168*a^5*b*c^7 + 4*a^5*b^7*c + 2*a^6*b*c^6 - 8*a^6*b^6*c - 84*a^7*
b*c^5 + 26*a^8*b*c^4 + 44*a^2*b^4*c^7 - 300*a^2*b^5*c^6 + 764*a^2*b^6*c^5 -
900*a^2*b^7*c^4 + 460*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 32*a^3*b^2*c^8 + 272*
a^3*b^3*c^7 - 840*a^3*b^4*c^6 + 1156*a^3*b^5*c^5 - 660*a^3*b^6*c^4 + 72*a^3
*b^7*c^3 + 8*a^3*b^8*c^2 + 384*a^4*b^2*c^7 - 704*a^4*b^3*c^6 + 541*a^4*b^4*
c^5 - 149*a^4*b^5*c^4 + 34*a^4*b^6*c^3 + 6*a^4*b^7*c^2 - 204*a^5*b^2*c^6 +
96*a^5*b^3*c^5 + 41*a^5*b^4*c^4 - 132*a^5*b^5*c^3 + 82*a^5*b^6*c^2 - 90*a^6
*b^2*c^5 + 174*a^6*b^3*c^4 - 104*a^6*b^4*c^3 + 8*a^6*b^5*c^2 + 82*a^7*b^2*c
^4 - 40*a^7*b^3*c^3 + 20*a^7*b^4*c^2 - 16*a^8*b^2*c^3 + 24*a*b^11*c))/a^8)*
(-(b^10 - 8*a^4*c^6 - 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 +
10*a*b^6*c^3 - 33*a^2*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^
4*c^3 + 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*
a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) +
4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(
1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c
^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 -
8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) + (((((2048*(26*a^9*b^7 - 12*a^8*b^
8 - 18*a^10*b^6 + 6*a^11*b^5 - 2*a^12*b^4 + 48*a^10*c^6 + 176*a^11*c^5 + 17
6*a^12*c^4 + 16*a^13*c^3 - 32*a^14*c^2 + 20*a^8*b^7*c + 74*a^9*b^6*c - 144*
a^10*b*c^5 - 192*a^10*b^5*c - 352*a^11*b*c^4 + 122*a^11*b^4*c - 144*a^12*b*
c^3 - 40*a^12*b^3*c + 64*a^13*b*c^2 + 16*a^13*b^2*c + 8*a^8*b^4*c^4 - 20*a^
8*b^5*c^3 + 4*a^8*b^6*c^2 - 44*a^9*b^2*c^5 + 116*a^9*b^3*c^4 + 10*a^9*b^4*c
^3 - 182*a^9*b^5*c^2 - 148*a^10*b^2*c^4 + 496*a^10*b^3*c^3 - 50*a^10*b^4*c^
2 - 260*a^11*b^2*c^3 + 388*a^11*b^3*c^2 - 204*a^12*b^2*c^2))/a^8 + (2048*ta
n(x/2)*(-(b^10 - 8*a^4*c^6 - 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8
*c^2 + 10*a*b^6*c^3 - 33*a^2*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96
*a^3*b^4*c^3 + 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8
*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^
(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^
2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 1
6*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4
*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2)*(32*a^16*c + 8*a^10*b^7 - 24
*a^11*b^6 + 32*a^12*b^5 - 32*a^13*b^4 + 24*a^14*b^3 - 8*a^15*b^2 + 96*a^12*
c^5 + 64*a^13*c^4 - 128*a^14*c^3 - 64*a^15*c^2 - 8*a^10*b^6*c - 56*a^11*b^5
*c - 32*a^12*b*c^4 + 184*a^12*b^4*c + 352*a^13*b*c^3 - 200*a^13*b^3*c + 288
*a^14*b*c^2 + 144*a^14*b^2*c + 8*a^10*b^4*c^3 - 8*a^10*b^5*c^2 - 56*a^11*b^
2*c^4 + 40*a^11*b^3*c^3 + 96*a^11*b^4*c^2 - 272*a^12*b^2*c^3 + 8*a^12*b^3*c

^2 - 320*a^13*b^2*c^2 - 96*a^15*b*c))/a^8)*(-(b^10 - 8*a^4*c^6 - 8*a^5*c^5
- b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33*a^2*b^4*c^4 +
52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 + b^5*c^2
*(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/
2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^
(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)
^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 +
10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2))
)^(1/2) - (2048*tan(x/2)*(8*a^14*c + 8*a^4*b^11 - 24*a^5*b^10 + 36*a^6*b^9
- 52*a^7*b^8 + 61*a^8*b^7 - 49*a^9*b^6 + 33*a^10*b^5 - 17*a^11*b^4 + 6*a^12
*b^3 - 2*a^13*b^2 + 72*a^8*c^7 - 136*a^9*c^6 - 192*a^10*c^5 + 168*a^11*c^4
+ 80*a^12*c^3 - 64*a^13*c^2 - 8*a^4*b^10*c - 72*a^5*b^9*c + 244*a^6*b^8*c -
308*a^7*b^7*c - 88*a^8*b*c^6 + 375*a^8*b^6*c + 496*a^9*b*c^5 - 416*a^9*b^5
*c - 16*a^10*b*c^4 + 295*a^10*b^4*c - 328*a^11*b*c^3 - 178*a^11*b^3*c + 184
*a^12*b*c^2 + 84*a^12*b^2*c + 8*a^4*b^8*c^3 - 8*a^4*b^9*c^2 - 72*a^5*b^6*c^
4 + 56*a^5*b^7*c^3 + 112*a^5*b^8*c^2 + 220*a^6*b^4*c^5 - 140*a^6*b^5*c^4 -
424*a^6*b^6*c^3 + 80*a^6*b^7*c^2 - 256*a^7*b^2*c^6 + 192*a^7*b^3*c^5 + 416*
a^7*b^4*c^4 + 572*a^7*b^5*c^3 - 732*a^7*b^6*c^2 + 64*a^8*b^2*c^5 - 1152*a^8
*b^3*c^4 + 521*a^8*b^4*c^3 + 779*a^8*b^5*c^2 + 234*a^9*b^2*c^4 - 494*a^9*b^
3*c^3 - 723*a^9*b^4*c^2 + 180*a^10*b^2*c^3 + 770*a^10*b^3*c^2 - 416*a^11*b^
2*c^2 - 24*a^13*b*c))/a^8)*(-(b^10 - 8*a^4*c^6 - 8*a^5*c^5 - b^7*(-(4*a*c -
b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33*a^2*b^4*c^4 + 52*a^2*b^6*c^2 +
38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)
^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4
*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b
^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a
^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8
*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) - (2048*
(26*a^3*b^11 - 12*a^2*b^12 - 30*a^4*b^10 + 29*a^5*b^9 - 20*a^6*b^8 + 10*a^7
*b^7 - 4*a^8*b^6 + a^9*b^5 + 12*a^6*c^8 + 88*a^7*c^7 + 72*a^8*c^6 - 44*a^9*
c^5 - 28*a^10*c^4 + 12*a^11*c^3 + 20*a^2*b^11*c + 98*a^3*b^10*c - 228*a^4*b
^9*c + 251*a^5*b^8*c - 96*a^6*b*c^7 - 238*a^6*b^7*c - 200*a^7*b*c^6 + 154*a
^7*b^6*c + 100*a^8*b*c^5 - 72*a^8*b^5*c + 112*a^9*b*c^4 + 27*a^9*b^4*c - 68
*a^10*b*c^3 - 6*a^10*b^3*c + 8*a^11*b*c^2 + 8*a^2*b^8*c^4 - 20*a^2*b^9*c^3
+ 4*a^2*b^10*c^2 - 60*a^3*b^6*c^5 + 156*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 222*a
^3*b^9*c^2 + 136*a^4*b^4*c^6 - 388*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 856*a^4*
b^7*c^3 - 202*a^4*b^8*c^2 - 100*a^5*b^2*c^7 + 364*a^5*b^3*c^6 + 394*a^5*b^4
*c^5 - 1362*a^5*b^5*c^4 - 115*a^5*b^6*c^3 + 635*a^5*b^7*c^2 - 340*a^6*b^2*c
^6 + 904*a^6*b^3*c^5 + 583*a^6*b^4*c^4 - 564*a^6*b^5*c^3 - 655*a^6*b^6*c^2
- 399*a^7*b^2*c^5 + 9*a^7*b^3*c^4 + 536*a^7*b^4*c^3 + 612*a^7*b^5*c^2 - 37*
a^8*b^2*c^4 - 524*a^8*b^3*c^3 - 354*a^8*b^4*c^2 + 239*a^9*b^2*c^3 + 145*a^9
*b^3*c^2 - 47*a^10*b^2*c^2))/a^8)*(-(b^10 - 8*a^4*c^6 - 8*a^5*c^5 - b^7*(-(
4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33*a^2*b^4*c^4 + 52*a^2*b^
6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c
- b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^

2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 1
0*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2)
)/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^
4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2) +
(2048*tan(x/2)*(4*a*b^12 + 20*b^12*c - 4*b^13 - 4*a^2*b^11 + 4*a^3*b^10 -
a^4*b^9 + a^5*b^8 + 12*a^4*c^9 - 44*a^5*c^8 + 2*a^6*c^7 + 38*a^7*c^6 - 18*a
^8*c^5 + 2*a^9*c^4 + 4*b^8*c^5 - 20*b^9*c^4 + 40*b^10*c^3 - 40*b^11*c^2 - 2
4*a*b^6*c^6 + 136*a*b^7*c^5 - 300*a*b^8*c^4 + 320*a*b^9*c^3 - 160*a*b^10*c^
2 - 20*a^2*b^10*c + 20*a^3*b^9*c - 92*a^4*b*c^8 - 31*a^4*b^8*c + 168*a^5*b*
c^7 + 4*a^5*b^7*c + 2*a^6*b*c^6 - 8*a^6*b^6*c - 84*a^7*b*c^5 + 26*a^8*b*c^4
+ 44*a^2*b^4*c^7 - 300*a^2*b^5*c^6 + 764*a^2*b^6*c^5 - 900*a^2*b^7*c^4 + 4
60*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 32*a^3*b^2*c^8 + 272*a^3*b^3*c^7 - 840*a^
3*b^4*c^6 + 1156*a^3*b^5*c^5 - 660*a^3*b^6*c^4 + 72*a^3*b^7*c^3 + 8*a^3*b^8
*c^2 + 384*a^4*b^2*c^7 - 704*a^4*b^3*c^6 + 541*a^4*b^4*c^5 - 149*a^4*b^5*c^
4 + 34*a^4*b^6*c^3 + 6*a^4*b^7*c^2 - 204*a^5*b^2*c^6 + 96*a^5*b^3*c^5 + 41*
a^5*b^4*c^4 - 132*a^5*b^5*c^3 + 82*a^5*b^6*c^2 - 90*a^6*b^2*c^5 + 174*a^6*b
^3*c^4 - 104*a^6*b^4*c^3 + 8*a^6*b^5*c^2 + 82*a^7*b^2*c^4 - 40*a^7*b^3*c^3
+ 20*a^7*b^4*c^2 - 16*a^8*b^2*c^3 + 24*a*b^11*c))/a^8)*(-(b^10 - 8*a^4*c^6
- 8*a^5*c^5 - b^7*(-(4*a*c - b^2)^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33*a^
2*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c
^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c -
b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*
c - b^2)^3)^(1/2) - 10*a^2*b^3*c^2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(
4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16
*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a
^8*b^2*c^2)))^(1/2)))*(-(b^10 - 8*a^4*c^6 - 8*a^5*c^5 - b^7*(-(4*a*c - b^2)
^3)^(1/2) - b^8*c^2 + 10*a*b^6*c^3 - 33*a^2*b^4*c^4 + 52*a^2*b^6*c^2 + 38*a
^3*b^2*c^5 - 96*a^3*b^4*c^3 + 66*a^4*b^2*c^4 + b^5*c^2*(-(4*a*c - b^2)^3)^(
1/2) - 12*a*b^8*c - 4*a*b^3*c^3*(-(4*a*c - b^2)^3)^(1/2) + 3*a^2*b*c^4*(-(4
*a*c - b^2)^3)^(1/2) + 4*a^3*b*c^3*(-(4*a*c - b^2)^3)^(1/2) - 10*a^2*b^3*c^
2*(-(4*a*c - b^2)^3)^(1/2) + 6*a*b^5*c*(-(4*a*c - b^2)^3)^(1/2))/(2*(a^8*b^
4 - a^6*b^6 + 16*a^8*c^4 + 32*a^9*c^3 + 16*a^10*c^2 + 10*a^7*b^4*c - 8*a^9*
b^2*c + a^6*b^4*c^2 - 8*a^7*b^2*c^3 - 32*a^8*b^2*c^2)))^(1/2)*2i + (atan(((
((2048*tan(x/2)*(4*a*b^12 + 20*b^12*c - 4*b^13 - 4*a^2*b^11 + 4*a^3*b^10 -
a^4*b^9 + a^5*b^8 + 12*a^4*c^9 - 44*a^5*c^8 + 2*a^6*c^7 + 38*a^7*c^6 - 18*a
^8*c^5 + 2*a^9*c^4 + 4*b^8*c^5 - 20*b^9*c^4 + 40*b^10*c^3 - 40*b^11*c^2 - 2
4*a*b^6*c^6 + 136*a*b^7*c^5 - 300*a*b^8*c^4 + 320*a*b^9*c^3 - 160*a*b^10*c^
2 - 20*a^2*b^10*c + 20*a^3*b^9*c - 92*a^4*b*c^8 - 31*a^4*b^8*c + 168*a^5*b*
c^7 + 4*a^5*b^7*c + 2*a^6*b*c^6 - 8*a^6*b^6*c - 84*a^7*b*c^5 + 26*a^8*b*c^4
+ 44*a^2*b^4*c^7 - 300*a^2*b^5*c^6 + 764*a^2*b^6*c^5 - 900*a^2*b^7*c^4 + 4
60*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 32*a^3*b^2*c^8 + 272*a^3*b^3*c^7 - 840*a^
3*b^4*c^6 + 1156*a^3*b^5*c^5 - 660*a^3*b^6*c^4 + 72*a^3*b^7*c^3 + 8*a^3*b^8
*c^2 + 384*a^4*b^2*c^7 - 704*a^4*b^3*c^6 + 541*a^4*b^4*c^5 - 149*a^4*b^5*c^
4 + 34*a^4*b^6*c^3 + 6*a^4*b^7*c^2 - 204*a^5*b^2*c^6 + 96*a^5*b^3*c^5 + 41*
a^5*b^4*c^4 - 132*a^5*b^5*c^3 + 82*a^5*b^6*c^2 - 90*a^6*b^2*c^5 + 174*a^6*b

^3*c^4 - 104*a^6*b^4*c^3 + 8*a^6*b^5*c^2 + 82*a^7*b^2*c^4 - 40*a^7*b^3*c^3
+ 20*a^7*b^4*c^2 - 16*a^8*b^2*c^3 + 24*a*b^11*c))/a^8 + (((2048*(26*a^3*b^1
1 - 12*a^2*b^12 - 30*a^4*b^10 + 29*a^5*b^9 - 20*a^6*b^8 + 10*a^7*b^7 - 4*a^
8*b^6 + a^9*b^5 + 12*a^6*c^8 + 88*a^7*c^7 + 72*a^8*c^6 - 44*a^9*c^5 - 28*a^
10*c^4 + 12*a^11*c^3 + 20*a^2*b^11*c + 98*a^3*b^10*c - 228*a^4*b^9*c + 251*
a^5*b^8*c - 96*a^6*b*c^7 - 238*a^6*b^7*c - 200*a^7*b*c^6 + 154*a^7*b^6*c +
100*a^8*b*c^5 - 72*a^8*b^5*c + 112*a^9*b*c^4 + 27*a^9*b^4*c - 68*a^10*b*c^3
 - 6*a^10*b^3*c + 8*a^11*b*c^2 + 8*a^2*b^8*c^4 - 20*a^2*b^9*c^3 + 4*a^2*b^1
0*c^2 - 60*a^3*b^6*c^5 + 156*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 222*a^3*b^9*c^2
+ 136*a^4*b^4*c^6 - 388*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 856*a^4*b^7*c^3 - 2
02*a^4*b^8*c^2 - 100*a^5*b^2*c^7 + 364*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 1362
*a^5*b^5*c^4 - 115*a^5*b^6*c^3 + 635*a^5*b^7*c^2 - 340*a^6*b^2*c^6 + 904*a^
6*b^3*c^5 + 583*a^6*b^4*c^4 - 564*a^6*b^5*c^3 - 655*a^6*b^6*c^2 - 399*a^7*b
^2*c^5 + 9*a^7*b^3*c^4 + 536*a^7*b^4*c^3 + 612*a^7*b^5*c^2 - 37*a^8*b^2*c^4
 - 524*a^8*b^3*c^3 - 354*a^8*b^4*c^2 + 239*a^9*b^2*c^3 + 145*a^9*b^3*c^2 -
47*a^10*b^2*c^2))/a^8 - (((2048*tan(x/2)*(8*a^14*c + 8*a^4*b^11 - 24*a^5*b^
10 + 36*a^6*b^9 - 52*a^7*b^8 + 61*a^8*b^7 - 49*a^9*b^6 + 33*a^10*b^5 - 17*a
^11*b^4 + 6*a^12*b^3 - 2*a^13*b^2 + 72*a^8*c^7 - 136*a^9*c^6 - 192*a^10*c^5
 + 168*a^11*c^4 + 80*a^12*c^3 - 64*a^13*c^2 - 8*a^4*b^10*c - 72*a^5*b^9*c +
 244*a^6*b^8*c - 308*a^7*b^7*c - 88*a^8*b*c^6 + 375*a^8*b^6*c + 496*a^9*b*c
^5 - 416*a^9*b^5*c - 16*a^10*b*c^4 + 295*a^10*b^4*c - 328*a^11*b*c^3 - 178*
a^11*b^3*c + 184*a^12*b*c^2 + 84*a^12*b^2*c + 8*a^4*b^8*c^3 - 8*a^4*b^9*c^2
 - 72*a^5*b^6*c^4 + 56*a^5*b^7*c^3 + 112*a^5*b^8*c^2 + 220*a^6*b^4*c^5 - 14
0*a^6*b^5*c^4 - 424*a^6*b^6*c^3 + 80*a^6*b^7*c^2 - 256*a^7*b^2*c^6 + 192*a^
7*b^3*c^5 + 416*a^7*b^4*c^4 + 572*a^7*b^5*c^3 - 732*a^7*b^6*c^2 + 64*a^8*b^
2*c^5 - 1152*a^8*b^3*c^4 + 521*a^8*b^4*c^3 + 779*a^8*b^5*c^2 + 234*a^9*b^2*
c^4 - 494*a^9*b^3*c^3 - 723*a^9*b^4*c^2 + 180*a^10*b^2*c^3 + 770*a^10*b^3*c
^2 - 416*a^11*b^2*c^2 - 24*a^13*b*c))/a^8 + (((2048*(26*a^9*b^7 - 12*a^8*b^
8 - 18*a^10*b^6 + 6*a^11*b^5 - 2*a^12*b^4 + 48*a^10*c^6 + 176*a^11*c^5 + 17
6*a^12*c^4 + 16*a^13*c^3 - 32*a^14*c^2 + 20*a^8*b^7*c + 74*a^9*b^6*c - 144*
a^10*b*c^5 - 192*a^10*b^5*c - 352*a^11*b*c^4 + 122*a^11*b^4*c - 144*a^12*b*
c^3 - 40*a^12*b^3*c + 64*a^13*b*c^2 + 16*a^13*b^2*c + 8*a^8*b^4*c^4 - 20*a^
8*b^5*c^3 + 4*a^8*b^6*c^2 - 44*a^9*b^2*c^5 + 116*a^9*b^3*c^4 + 10*a^9*b^4*c
^3 - 182*a^9*b^5*c^2 - 148*a^10*b^2*c^4 + 496*a^10*b^3*c^3 - 50*a^10*b^4*c^
2 - 260*a^11*b^2*c^3 + 388*a^11*b^3*c^2 - 204*a^12*b^2*c^2))/a^8 - (1024*ta
n(x/2)*(a^2 - 2*a*c + 2*b^2)*(32*a^16*c + 8*a^10*b^7 - 24*a^11*b^6 + 32*a^1
2*b^5 - 32*a^13*b^4 + 24*a^14*b^3 - 8*a^15*b^2 + 96*a^12*c^5 + 64*a^13*c^4
- 128*a^14*c^3 - 64*a^15*c^2 - 8*a^10*b^6*c - 56*a^11*b^5*c - 32*a^12*b*c^4
 + 184*a^12*b^4*c + 352*a^13*b*c^3 - 200*a^13*b^3*c + 288*a^14*b*c^2 + 144*
a^14*b^2*c + 8*a^10*b^4*c^3 - 8*a^10*b^5*c^2 - 56*a^11*b^2*c^4 + 40*a^11*b^
3*c^3 + 96*a^11*b^4*c^2 - 272*a^12*b^2*c^3 + 8*a^12*b^3*c^2 - 320*a^13*b^2*
c^2 - 96*a^15*b*c))/a^11)*(a^2 - 2*a*c + 2*b^2))/(2*a^3))*(a^2 - 2*a*c + 2*
b^2))/(2*a^3))*(a^2 - 2*a*c + 2*b^2))/(2*a^3))*(a^2 - 2*a*c + 2*b^2)*1i)/(2
*a^3) + (((2048*tan(x/2)*(4*a*b^12 + 20*b^12*c - 4*b^13 - 4*a^2*b^11 + 4*a^
3*b^10 - a^4*b^9 + a^5*b^8 + 12*a^4*c^9 - 44*a^5*c^8 + 2*a^6*c^7 + 38*a^7*c

^6 - 18*a^8*c^5 + 2*a^9*c^4 + 4*b^8*c^5 - 20*b^9*c^4 + 40*b^10*c^3 - 40*b^1
1*c^2 - 24*a*b^6*c^6 + 136*a*b^7*c^5 - 300*a*b^8*c^4 + 320*a*b^9*c^3 - 160*
a*b^10*c^2 - 20*a^2*b^10*c + 20*a^3*b^9*c - 92*a^4*b*c^8 - 31*a^4*b^8*c + 1
68*a^5*b*c^7 + 4*a^5*b^7*c + 2*a^6*b*c^6 - 8*a^6*b^6*c - 84*a^7*b*c^5 + 26*
a^8*b*c^4 + 44*a^2*b^4*c^7 - 300*a^2*b^5*c^6 + 764*a^2*b^6*c^5 - 900*a^2*b^
7*c^4 + 460*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 32*a^3*b^2*c^8 + 272*a^3*b^3*c^7
 - 840*a^3*b^4*c^6 + 1156*a^3*b^5*c^5 - 660*a^3*b^6*c^4 + 72*a^3*b^7*c^3 +
8*a^3*b^8*c^2 + 384*a^4*b^2*c^7 - 704*a^4*b^3*c^6 + 541*a^4*b^4*c^5 - 149*a
^4*b^5*c^4 + 34*a^4*b^6*c^3 + 6*a^4*b^7*c^2 - 204*a^5*b^2*c^6 + 96*a^5*b^3*
c^5 + 41*a^5*b^4*c^4 - 132*a^5*b^5*c^3 + 82*a^5*b^6*c^2 - 90*a^6*b^2*c^5 +
174*a^6*b^3*c^4 - 104*a^6*b^4*c^3 + 8*a^6*b^5*c^2 + 82*a^7*b^2*c^4 - 40*a^7
*b^3*c^3 + 20*a^7*b^4*c^2 - 16*a^8*b^2*c^3 + 24*a*b^11*c))/a^8 - (((2048*(2
6*a^3*b^11 - 12*a^2*b^12 - 30*a^4*b^10 + 29*a^5*b^9 - 20*a^6*b^8 + 10*a^7*b
^7 - 4*a^8*b^6 + a^9*b^5 + 12*a^6*c^8 + 88*a^7*c^7 + 72*a^8*c^6 - 44*a^9*c^
5 - 28*a^10*c^4 + 12*a^11*c^3 + 20*a^2*b^11*c + 98*a^3*b^10*c - 228*a^4*b^9
*c + 251*a^5*b^8*c - 96*a^6*b*c^7 - 238*a^6*b^7*c - 200*a^7*b*c^6 + 154*a^7
*b^6*c + 100*a^8*b*c^5 - 72*a^8*b^5*c + 112*a^9*b*c^4 + 27*a^9*b^4*c - 68*a
^10*b*c^3 - 6*a^10*b^3*c + 8*a^11*b*c^2 + 8*a^2*b^8*c^4 - 20*a^2*b^9*c^3 +
4*a^2*b^10*c^2 - 60*a^3*b^6*c^5 + 156*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 222*a^3
*b^9*c^2 + 136*a^4*b^4*c^6 - 388*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 856*a^4*b^
7*c^3 - 202*a^4*b^8*c^2 - 100*a^5*b^2*c^7 + 364*a^5*b^3*c^6 + 394*a^5*b^4*c
^5 - 1362*a^5*b^5*c^4 - 115*a^5*b^6*c^3 + 635*a^5*b^7*c^2 - 340*a^6*b^2*c^6
 + 904*a^6*b^3*c^5 + 583*a^6*b^4*c^4 - 564*a^6*b^5*c^3 - 655*a^6*b^6*c^2 -
399*a^7*b^2*c^5 + 9*a^7*b^3*c^4 + 536*a^7*b^4*c^3 + 612*a^7*b^5*c^2 - 37*a^
8*b^2*c^4 - 524*a^8*b^3*c^3 - 354*a^8*b^4*c^2 + 239*a^9*b^2*c^3 + 145*a^9*b
^3*c^2 - 47*a^10*b^2*c^2))/a^8 + (((2048*tan(x/2)*(8*a^14*c + 8*a^4*b^11 -
24*a^5*b^10 + 36*a^6*b^9 - 52*a^7*b^8 + 61*a^8*b^7 - 49*a^9*b^6 + 33*a^10*b
^5 - 17*a^11*b^4 + 6*a^12*b^3 - 2*a^13*b^2 + 72*a^8*c^7 - 136*a^9*c^6 - 192
*a^10*c^5 + 168*a^11*c^4 + 80*a^12*c^3 - 64*a^13*c^2 - 8*a^4*b^10*c - 72*a^
5*b^9*c + 244*a^6*b^8*c - 308*a^7*b^7*c - 88*a^8*b*c^6 + 375*a^8*b^6*c + 49
6*a^9*b*c^5 - 416*a^9*b^5*c - 16*a^10*b*c^4 + 295*a^10*b^4*c - 328*a^11*b*c
^3 - 178*a^11*b^3*c + 184*a^12*b*c^2 + 84*a^12*b^2*c + 8*a^4*b^8*c^3 - 8*a^
4*b^9*c^2 - 72*a^5*b^6*c^4 + 56*a^5*b^7*c^3 + 112*a^5*b^8*c^2 + 220*a^6*b^4
*c^5 - 140*a^6*b^5*c^4 - 424*a^6*b^6*c^3 + 80*a^6*b^7*c^2 - 256*a^7*b^2*c^6
 + 192*a^7*b^3*c^5 + 416*a^7*b^4*c^4 + 572*a^7*b^5*c^3 - 732*a^7*b^6*c^2 +
64*a^8*b^2*c^5 - 1152*a^8*b^3*c^4 + 521*a^8*b^4*c^3 + 779*a^8*b^5*c^2 + 234
*a^9*b^2*c^4 - 494*a^9*b^3*c^3 - 723*a^9*b^4*c^2 + 180*a^10*b^2*c^3 + 770*a
^10*b^3*c^2 - 416*a^11*b^2*c^2 - 24*a^13*b*c))/a^8 - (((2048*(26*a^9*b^7 -
12*a^8*b^8 - 18*a^10*b^6 + 6*a^11*b^5 - 2*a^12*b^4 + 48*a^10*c^6 + 176*a^11
*c^5 + 176*a^12*c^4 + 16*a^13*c^3 - 32*a^14*c^2 + 20*a^8*b^7*c + 74*a^9*b^6
*c - 144*a^10*b*c^5 - 192*a^10*b^5*c - 352*a^11*b*c^4 + 122*a^11*b^4*c - 14
4*a^12*b*c^3 - 40*a^12*b^3*c + 64*a^13*b*c^2 + 16*a^13*b^2*c + 8*a^8*b^4*c^
4 - 20*a^8*b^5*c^3 + 4*a^8*b^6*c^2 - 44*a^9*b^2*c^5 + 116*a^9*b^3*c^4 + 10*
a^9*b^4*c^3 - 182*a^9*b^5*c^2 - 148*a^10*b^2*c^4 + 496*a^10*b^3*c^3 - 50*a^
10*b^4*c^2 - 260*a^11*b^2*c^3 + 388*a^11*b^3*c^2 - 204*a^12*b^2*c^2))/a^8 +

(1024*tan(x/2)*(a^2 - 2*a*c + 2*b^2)*(32*a^16*c + 8*a^10*b^7 - 24*a^11*b^6 + 32*a^12*b^5 - 32*a^13*b^4 + 24*a^14*b^3 - 8*a^15*b^2 + 96*a^12*c^5 + 64*a^13*c^4 - 128*a^14*c^3 - 64*a^15*c^2 - 8*a^10*b^6*c - 56*a^11*b^5*c - 32*a^12*b*c^4 + 184*a^12*b^4*c + 352*a^13*b*c^3 - 200*a^13*b^3*c + 288*a^14*b*c^2 + 144*a^14*b^2*c + 8*a^10*b^4*c^3 - 8*a^10*b^5*c^2 - 56*a^11*b^2*c^4 + 40*a^11*b^3*c^3 + 96*a^11*b^4*c^2 - 272*a^12*b^2*c^3 + 8*a^12*b^3*c^2 - 320*a^13*b^2*c^2 - 96*a^15*b*c))/a^11)*(a^2 - 2*a*c + 2*b^2))/(2*a^3))*(a^2 - 2*a*c + 2*b^2))/(2*a^3))*(a^2 - 2*a*c + 2*b^2))/(2*a^3))*(a^2 - 2*a*c + 2*b^2)*1i)/(2*a^3))/((4096*(14*a^3*c^9 + a^4*c^8 - 10*a^5*c^7 + 3*a^6*c^6 - 4*b^4*c^8 + 16*b^5*c^7 - 24*b^6*c^6 + 16*b^7*c^5 - 4*b^8*c^4 + 4*a*b^2*c^9 - 28*a*b^3*c^8 + 56*a*b^4*c^7 - 40*a*b^5*c^6 + 4*a*b^6*c^5 + 4*a*b^7*c^4 + 12*a^2*b*c^9 - 22*a^3*b*c^8 + 4*a^4*b*c^7 + 6*a^5*b*c^6 - 2*a^6*b*c^5 - 48*a^2*b^2*c^8 + 48*a^2*b^3*c^7 - 8*a^2*b^4*c^6 - 4*a^2*b^6*c^4 + 4*a^3*b^2*c^7 - 4*a^3*b^3*c^6 + 4*a^3*b^5*c^4 + 10*a^4*b^2*c^6 - 8*a^4*b^3*c^5 - a^4*b^4*c^4 - a^5*b^2*c^5 + a^5*b^3*c^4))/a^8 - (((2048*tan(x/2)*(4*a*b^12 + 20*b^12*c - 4*b^13 - 4*a^2*b^11 + 4*a^3*b^10 - a^4*b^9 + a^5*b^8 + 12*a^4*c^9 - 44*a^5*c^8 + 2*a^6*c^7 + 38*a^7*c^6 - 18*a^8*c^5 + 2*a^9*c^4 + 4*b^8*c^5 - 20*b^9*c^4 + 40*b^10*c^3 - 40*b^11*c^2 - 24*a*b^6*c^6 + 136*a*b^7*c^5 - 300*a*b^8*c^4 + 320*a*b^9*c^3 - 160*a*b^10*c^2 - 20*a^2*b^10*c + 20*a^3*b^9*c - 92*a^4*b*c^8 - 31*a^4*b^8*c + 168*a^5*b*c^7 + 4*a^5*b^7*c + 2*a^6*b*c^6 - 8*a^6*b^6*c - 84*a^7*b*c^5 + 26*a^8*b*c^4 + 44*a^2*b^4*c^7 - 300*a^2*b^5*c^6 + 764*a^2*b^6*c^5 - 900*a^2*b^7*c^4 + 460*a^2*b^8*c^3 - 44*a^2*b^9*c^2 - 32*a^3*b^2*c^8 + 272*a^3*b^3*c^7 - 840*a^3*b^4*c^6 + 1156*a^3*b^5*c^5 - 660*a^3*b^6*c^4 + 72*a^3*b^7*c^3 + 8*a^3*b^8*c^2 + 384*a^4*b^2*c^7 - 704*a^4*b^3*c^6 + 541*a^4*b^4*c^5 - 149*a^4*b^5*c^4 + 34*a^4*b^6*c^3 + 6*a^4*b^7*c^2 - 204*a^5*b^2*c^6 + 96*a^5*b^3*c^5 + 41*a^5*b^4*c^4 - 132*a^5*b^5*c^3 + 82*a^5*b^6*c^2 - 90*a^6*b^2*c^5 + 174*a^6*b^3*c^4 - 104*a^6*b^4*c^3 + 8*a^6*b^5*c^2 + 82*a^7*b^2*c^4 - 40*a^7*b^3*c^3 + 20*a^7*b^4*c^2 - 16*a^8*b^2*c^3 + 24*a*b^11*c))/a^8 + (((2048*(26*a^3*b^11 - 12*a^2*b^12 - 30*a^4*b^10 + 29*a^5*b^9 - 20*a^6*b^8 + 10*a^7*b^7 - 4*a^8*b^6 + a^9*b^5 + 12*a^6*c^8 + 88*a^7*c^7 + 72*a^8*c^6 - 44*a^9*c^5 - 28*a^10*c^4 + 12*a^11*c^3 + 20*a^2*b^11*c + 98*a^3*b^10*c - 228*a^4*b^9*c + 251*a^5*b^8*c - 96*a^6*b*c^7 - 238*a^6*b^7*c - 200*a^7*b*c^6 + 154*a^7*b^6*c + 100*a^8*b*c^5 - 72*a^8*b^5*c + 112*a^9*b*c^4 + 27*a^9*b^4*c - 68*a^10*b*c^3 - 6*a^10*b^3*c + 8*a^11*b*c^2 + 8*a^2*b^8*c^4 - 20*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 60*a^3*b^6*c^5 + 156*a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 222*a^3*b^9*c^2 + 136*a^4*b^4*c^6 - 388*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 856*a^4*b^7*c^3 - 202*a^4*b^8*c^2 - 100*a^5*b^2*c^7 + 364*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 1362*a^5*b^5*c^4 - 115*a^5*b^6*c^3 + 635*a^5*b^7*c^2 - 340*a^6*b^2*c^6 + 904*a^6*b^3*c^5 + 583*a^6*b^4*c^4 - 564*a^6*b^5*c^3 - 655*a^6*b^6*c^2 - 399*a^7*b^2*c^5 + 9*a^7*b^3*c^4 + 536*a^7*b^4*c^3 + 612*a^7*b^5*c^2 - 37*a^8*b^2*c^4 - 524*a^8*b^3*c^3 - 354*a^8*b^4*c^2 + 239*a^9*b^2*c^3 + 145*a^9*b^3*c^2 - 47*a^10*b^2*c^2))/a^8 - (((2048*tan(x/2)*(8*a^14*c + 8*a^4*b^11 - 24*a^5*b^10 + 36*a^6*b^9 - 52*a^7*b^8 + 61*a^8*b^7 - 49*a^9*b^6 + 33*a^10*b^5 - 17*a^11*b^4 + 6*a^12*b^3 - 2*a^13*b^2 + 72*a^8*c^7 - 136*a^9*c^6 - 192*a^10*c^5 + 168*a^11*c^4 + 80*a^12*c^3 - 64*

a^13*c^2 - 8*a^4*b^10*c - 72*a^5*b^9*c + 244*a^6*b^8*c - 308*a^7*b^7*c - 88
*a^8*b*c^6 + 375*a^8*b^6*c + 496*a^9*b*c^5 - 416*a^9*b^5*c - 16*a^10*b*c^4
+ 295*a^10*b^4*c - 328*a^11*b*c^3 - 178*a^11*b^3*c + 184*a^12*b*c^2 + 84*a^
12*b^2*c + 8*a^4*b^8*c^3 - 8*a^4*b^9*c^2 - 72*a^5*b^6*c^4 + 56*a^5*b^7*c^3
+ 112*a^5*b^8*c^2 + 220*a^6*b^4*c^5 - 140*a^6*b^5*c^4 - 424*a^6*b^6*c^3 + 8
0*a^6*b^7*c^2 - 256*a^7*b^2*c^6 + 192*a^7*b^3*c^5 + 416*a^7*b^4*c^4 + 572*a
^7*b^5*c^3 - 732*a^7*b^6*c^2 + 64*a^8*b^2*c^5 - 1152*a^8*b^3*c^4 + 521*a^8*
b^4*c^3 + 779*a^8*b^5*c^2 + 234*a^9*b^2*c^4 - 494*a^9*b^3*c^3 - 723*a^9*b^4
*c^2 + 180*a^10*b^2*c^3 + 770*a^10*b^3*c^2 - 416*a^11*b^2*c^2 - 24*a^13*b*c
))/a^8 + (((2048*(26*a^9*b^7 - 12*a^8*b^8 - 18*a^10*b^6 + 6*a^11*b^5 - 2*a^
12*b^4 + 48*a^10*c^6 + 176*a^11*c^5 + 176*a^12*c^4 + 16*a^13*c^3 - 32*a^14*
c^2 + 20*a^8*b^7*c + 74*a^9*b^6*c - 144*a^10*b*c^5 - 192*a^10*b^5*c - 352*a
^11*b*c^4 + 122*a^11*b^4*c - 144*a^12*b*c^3 - 40*a^12*b^3*c + 64*a^13*b*c^2
+ 16*a^13*b^2*c + 8*a^8*b^4*c^4 - 20*a^8*b^5*c^3 + 4*a^8*b^6*c^2 - 44*a^9*
b^2*c^5 + 116*a^9*b^3*c^4 + 10*a^9*b^4*c^3 - 182*a^9*b^5*c^2 - 148*a^10*b^2
*c^4 + 496*a^10*b^3*c^3 - 50*a^10*b^4*c^2 - 260*a^11*b^2*c^3 + 388*a^11*b^3
*c^2 - 204*a^12*b^2*c^2))/a^8 - (1024*tan(x/2)*(a^2 - 2*a*c + 2*b^2)*(32*a^
16*c + 8*a^10*b^7 - 24*a^11*b^6 + 32*a^12*b^5 - 32*a^13*b^4 + 24*a^14*b^3 -
8*a^15*b^2 + 96*a^12*c^5 + 64*a^13*c^4 - 128*a^14*c^3 - 64*a^15*c^2 - 8*a^
10*b^6*c - 56*a^11*b^5*c - 32*a^12*b*c^4 + 184*a^12*b^4*c + 352*a^13*b*c^3
- 200*a^13*b^3*c + 288*a^14*b*c^2 + 144*a^14*b^2*c + 8*a^10*b^4*c^3 - 8*a^1
0*b^5*c^2 - 56*a^11*b^2*c^4 + 40*a^11*b^3*c^3 + 96*a^11*b^4*c^2 - 272*a^12*
b^2*c^3 + 8*a^12*b^3*c^2 - 320*a^13*b^2*c^2 - 96*a^15*b*c))/a^11)*(a^2 - 2*
a*c + 2*b^2))/(2*a^3))*(a^2 - 2*a*c + 2*b^2))/(2*a^3))*(a^2 - 2*a*c + 2*b^2
))/(2*a^3))*(a^2 - 2*a*c + 2*b^2))/(2*a^3) + (((2048*tan(x/2)*(4*a*b^12 + 2
0*b^12*c - 4*b^13 - 4*a^2*b^11 + 4*a^3*b^10 - a^4*b^9 + a^5*b^8 + 12*a^4*c^
9 - 44*a^5*c^8 + 2*a^6*c^7 + 38*a^7*c^6 - 18*a^8*c^5 + 2*a^9*c^4 + 4*b^8*c^
5 - 20*b^9*c^4 + 40*b^10*c^3 - 40*b^11*c^2 - 24*a*b^6*c^6 + 136*a*b^7*c^5 -
300*a*b^8*c^4 + 320*a*b^9*c^3 - 160*a*b^10*c^2 - 20*a^2*b^10*c + 20*a^3*b^
9*c - 92*a^4*b*c^8 - 31*a^4*b^8*c + 168*a^5*b*c^7 + 4*a^5*b^7*c + 2*a^6*b*c
^6 - 8*a^6*b^6*c - 84*a^7*b*c^5 + 26*a^8*b*c^4 + 44*a^2*b^4*c^7 - 300*a^2*b
^5*c^6 + 764*a^2*b^6*c^5 - 900*a^2*b^7*c^4 + 460*a^2*b^8*c^3 - 44*a^2*b^9*c
^2 - 32*a^3*b^2*c^8 + 272*a^3*b^3*c^7 - 840*a^3*b^4*c^6 + 1156*a^3*b^5*c^5
- 660*a^3*b^6*c^4 + 72*a^3*b^7*c^3 + 8*a^3*b^8*c^2 + 384*a^4*b^2*c^7 - 704*
a^4*b^3*c^6 + 541*a^4*b^4*c^5 - 149*a^4*b^5*c^4 + 34*a^4*b^6*c^3 + 6*a^4*b^
7*c^2 - 204*a^5*b^2*c^6 + 96*a^5*b^3*c^5 + 41*a^5*b^4*c^4 - 132*a^5*b^5*c^3
+ 82*a^5*b^6*c^2 - 90*a^6*b^2*c^5 + 174*a^6*b^3*c^4 - 104*a^6*b^4*c^3 + 8*
a^6*b^5*c^2 + 82*a^7*b^2*c^4 - 40*a^7*b^3*c^3 + 20*a^7*b^4*c^2 - 16*a^8*b^2
*c^3 + 24*a*b^11*c))/a^8 - (((2048*(26*a^3*b^11 - 12*a^2*b^12 - 30*a^4*b^10
+ 29*a^5*b^9 - 20*a^6*b^8 + 10*a^7*b^7 - 4*a^8*b^6 + a^9*b^5 + 12*a^6*c^8
+ 88*a^7*c^7 + 72*a^8*c^6 - 44*a^9*c^5 - 28*a^10*c^4 + 12*a^11*c^3 + 20*a^2
*b^11*c + 98*a^3*b^10*c - 228*a^4*b^9*c + 251*a^5*b^8*c - 96*a^6*b*c^7 - 23
8*a^6*b^7*c - 200*a^7*b*c^6 + 154*a^7*b^6*c + 100*a^8*b*c^5 - 72*a^8*b^5*c
+ 112*a^9*b*c^4 + 27*a^9*b^4*c - 68*a^10*b*c^3 - 6*a^10*b^3*c + 8*a^11*b*c^
2 + 8*a^2*b^8*c^4 - 20*a^2*b^9*c^3 + 4*a^2*b^10*c^2 - 60*a^3*b^6*c^5 + 156*

a^3*b^7*c^4 + 2*a^3*b^8*c^3 - 222*a^3*b^9*c^2 + 136*a^4*b^4*c^6 - 388*a^4*b^5*c^5 - 152*a^4*b^6*c^4 + 856*a^4*b^7*c^3 - 202*a^4*b^8*c^2 - 100*a^5*b^2*c^7 + 364*a^5*b^3*c^6 + 394*a^5*b^4*c^5 - 1362*a^5*b^5*c^4 - 115*a^5*b^6*c^3 + 635*a^5*b^7*c^2 - 340*a^6*b^2*c^6 + 904*a^6*b^3*c^5 + 583*a^6*b^4*c^4 - 564*a^6*b^5*c^3 - 655*a^6*b^6*c^2 - 399*a^7*b^2*c^5 + 9*a^7*b^3*c^4 + 536*a^7*b^4*c^3 + 612*a^7*b^5*c^2 - 37*a^8*b^2*c^4 - 524*a^8*b^3*c^3 - 354*a^8*b^4*c^2 + 239*a^9*b^2*c^3 + 145*a^9*b^3*c^2 - 47*a^10*b^2*c^2))/a^8 + (((2048*tan(x/2)*(8*a^14*c + 8*a^4*b^11 - 24*a^5*b^10 + 36*a^6*b^9 - 52*a^7*b^8 + 61*a^8*b^7 - 49*a^9*b^6 + 33*a^10*b^5 - 17*a^11*b^4 + 6*a^12*b^3 - 2*a^13*b^2 + 72*a^8*c^7 - 136*a^9*c^6 - 192*a^10*c^5 + 168*a^11*c^4 + 80*a^12*c^3 - 64*a^13*c^2 - 8*a^4*b^10*c - 72*a^5*b^9*c + 244*a^6*b^8*c - 308*a^7*b^7*c - 88*a^8*b*c^6 + 375*a^8*b^6*c + 496*a^9*b*c^5 - 416*a^9*b^5*c - 16*a^10*b*c^4 + 295*a^10*b^4*c - 328*a^11*b*c^3 - 178*a^11*b^3*c + 184*a^12*b*c^2 + 84*a^12*b^2*c + 8*a^4*b^8*c^3 - 8*a^4*b^9*c^2 - 72*a^5*b^6*c^4 + 56*a^5*b^7*c^3 + 112*a^5*b^8*c^2 + 220*a^6*b^4*c^5 - 140*a^6*b^5*c^4 - 424*a^6*b^6*c^3 + 80*a^6*b^7*c^2 - 256*a^7*b^2*c^6 + 192*a^7*b^3*c^5 + 416*a^7*b^4*c^4 + 572*a^7*b^5*c^3 - 732*a^7*b^6*c^2 + 64*a^8*b^2*c^5 - 1152*a^8*b^3*c^4 + 521*a^8*b^4*c^3 + 779*a^8*b^5*c^2 + 234*a^9*b^2*c^4 - 494*a^9*b^3*c^3 - 723*a^9*b^4*c^2 + 180*a^10*b^2*c^3 + 770*a^10*b^3*c^2 - 416*a^11*b^2*c^2 - 24*a^13*b*c))/a^8 - (((2048*(26*a^9*b^7 - 12*a^8*b^8 - 18*a^10*b^6 + 6*a^11*b^5 - 2*a^12*b^4 + 48*a^10*c^6 + 176*a^11*c^5 + 176*a^12*c^4 + 16*a^13*c^3 - 32*a^14*c^2 + 20*a^8*b^7*c + 74*a^9*b^6*c - 144*a^10*b*c^5 - 192*a^10*b^5*c - 352*a^11*b*c^4 + 122*a^11*b^4*c - 144*a^12*b*c^3 - 40*a^12*b^3*c + 64*a^13*b*c^2 + 16*a^13*b^2*c + 8*a^8*b^4*c^4 - 20*a^8*b^5*c^3 + 4*a^8*b^6*c^2 - 44*a^9*b^2*c^5 + 116*a^9*b^3*c^4 + 10*a^9*b^4*c^3 - 182*a^9*b^5*c^2 - 148*a^10*b^2*c^4 + 496*a^10*b^3*c^3 - 50*a^10*b^4*c^2 - 260*a^11*b^2*c^3 + 388*a^11*b^3*c^2 - 204*a^12*b^2*c^2))/a^8 + (1024*tan(x/2)*(a^2 - 2*a*c + 2*b^2)*(32*a^16*c + 8*a^10*b^7 - 24*a^11*b^6 + 32*a^12*b^5 - 32*a^13*b^4 + 24*a^14*b^3 - 8*a^15*b^2 + 96*a^12*c^5 + 64*a^13*c^4 - 128*a^14*c^3 - 64*a^15*c^2 - 8*a^10*b^6*c - 56*a^11*b^5*c - 32*a^12*b*c^4 + 184*a^12*b^4*c + 352*a^13*b*c^3 - 200*a^13*b^3*c + 288*a^14*b*c^2 + 144*a^14*b^2*c + 8*a^10*b^4*c^3 - 8*a^10*b^5*c^2 - 56*a^11*b^2*c^4 + 40*a^11*b^3*c^3 + 96*a^11*b^4*c^2 - 272*a^12*b^2*c^3 + 8*a^12*b^3*c^2 - 320*a^13*b^2*c^2 - 96*a^15*b*c))/a^11)*(a^2 - 2*a*c + 2*b^2))/(2*a^3))*(a^2 - 2*a*c + 2*b^2))/(2*a^3))*(a^2 - 2*a*c + 2*b^2))/(2*a^3))*(a^2 - 2*a*c + 2*b^2))/(2*a^3)))*(a^2 - 2*a*c + 2*b^2)*1i)/a^3

CHAPTER 4

APPENDIX

4.1 Listing of Grading functions

The following are the current version of the grading functions used for grading the quality of the antiderivative with reference to the optimal antiderivative included in the test suite.

There is a version for Maple and for Mathematica/Rubi. There is a version for grading Sympy and version for use with Sagemath.

The following are links to the current source code.

The following are the listings of source code of the grading functions.

Mathematica and Rubi grading function

```
(* Original version thanks to Albert Rich emailed on 03/21/2017 *)
(* ::Package:: *)

(* Nasser: April 7,2022. add second output which gives reason for the grade *)
(*                       Small rewrite of logic in main function to make it*)
(*                       match Maple's logic. No change in functionality otherwise*)

(* ::Subsection:: *)
(*GradeAntiderivative[result,optimal]*)


(* ::Text:: *)
(*If result and optimal are mathematical expressions, *)
```

```
(*          GradeAntiderivative[result,optimal] returns*)
(*  "F" if the result fails to integrate an expression that*)
(*      is integrable*)
(*  "C" if result involves higher level functions than necessary*)
(*  "B" if result is more than twice the size of the optimal*)
(*      antiderivative*)
(*  "A" if result can be considered optimal*)


GradeAntiderivative[result_,optimal_] := Module[{expnResult,expnOptimal,leafCountResult,leafCou
    expnResult  = ExpnType[result];
    expnOptimal = ExpnType[optimal];
    leafCountResult = LeafCount[result];
    leafCountOptimal = LeafCount[optimal];

    (*Print["expnResult=",expnResult," expnOptimal=",expnOptimal];*)
    If[expnResult<=expnOptimal,
        If[Not[FreeQ[result,Complex]], (*result contains complex*)
            If[Not[FreeQ[optimal,Complex]], (*optimal contains complex*)
                If[leafCountResult<=2*leafCountOptimal,
                    finalresult={"A",""}
                    ,(*ELSE*)
                    finalresult={"B","Both result and optimal contain complex but leaf count is
                  ]
            ,(*ELSE*)
                finalresult={"C","Result contains complex when optimal does not."}
            ]
        ,(*ELSE*)(*result does not contains complex*)
            If[leafCountResult<=2*leafCountOptimal,
               finalresult={"A",""}
            ,(*ELSE*)
               finalresult={"B","Leaf count is larger than twice the leaf count of optimal. $"<
              ]
        ]
    ,(*ELSE*) (*expnResult>expnOptimal*)
        If[FreeQ[result,Integrate] && FreeQ[result,Int],
            finalresult={"C","Result contains higher order function than in optimal. Order "<>T
            ,
            finalresult={"F","Contains unresolved integral."}
        ]
    ];

    finalresult
]

(* ::Text:: *)
(*The following summarizes the type number assigned an *)
```

```
(*expression based on the functions it involves*)
(*1 = rational function*)
(*2 = algebraic function*)
(*3 = elementary function*)
(*4 = special function*)
(*5 = hyperpergeometric function*)
(*6 = appell function*)
(*7 = rootsum function*)
(*8 = integrate function*)
(*9 = unknown function*)


ExpnType[expn_] :=
  If[AtomQ[expn],
    1,
  If[ListQ[expn],
    Max[Map[ExpnType,expn]],
  If[Head[expn]===Power,
    If[IntegerQ[expn[[2]]],
      ExpnType[expn[[1]]],
    If[Head[expn[[2]]]===Rational,
      If[IntegerQ[expn[[1]]] || Head[expn[[1]]]===Rational,
        1,
      Max[ExpnType[expn[[1]]],2]],
    Max[ExpnType[expn[[1]]],ExpnType[expn[[2]]],3]]],
  If[Head[expn]===Plus || Head[expn]===Times,
    Max[ExpnType[First[expn]],ExpnType[Rest[expn]]],
  If[ElementaryFunctionQ[Head[expn]],
    Max[3,ExpnType[expn[[1]]]],
  If[SpecialFunctionQ[Head[expn]],
    Apply[Max,Append[Map[ExpnType,Apply[List,expn]],4]],
  If[HypergeometricFunctionQ[Head[expn]],
    Apply[Max,Append[Map[ExpnType,Apply[List,expn]],5]],
  If[AppellFunctionQ[Head[expn]],
    Apply[Max,Append[Map[ExpnType,Apply[List,expn]],6]],
  If[Head[expn]===RootSum,
    Apply[Max,Append[Map[ExpnType,Apply[List,expn]],7]],
  If[Head[expn]===Integrate || Head[expn]===Int,
    Apply[Max,Append[Map[ExpnType,Apply[List,expn]],8]],
  9]]]]]]]]]]


ElementaryFunctionQ[func_] :=
  MemberQ[{
  Exp,Log,
  Sin,Cos,Tan,Cot,Sec,Csc,
  ArcSin,ArcCos,ArcTan,ArcCot,ArcSec,ArcCsc,
```

```
  Sinh,Cosh,Tanh,Coth,Sech,Csch,
  ArcSinh,ArcCosh,ArcTanh,ArcCoth,ArcSech,ArcCsch
},func]


SpecialFunctionQ[func_] :=
  MemberQ[{
  Erf, Erfc, Erfi,
  FresnelS, FresnelC,
  ExpIntegralE, ExpIntegralEi, LogIntegral,
  SinIntegral, CosIntegral, SinhIntegral, CoshIntegral,
  Gamma, LogGamma, PolyGamma,
  Zeta, PolyLog, ProductLog,
  EllipticF, EllipticE, EllipticPi
},func]


HypergeometricFunctionQ[func_] :=
  MemberQ[{Hypergeometric1F1,Hypergeometric2F1,HypergeometricPFQ},func]


AppellFunctionQ[func_] :=
  MemberQ[{AppellF1},func]
```

Maple grading function

```
# File: GradeAntiderivative.mpl
# Original version thanks to Albert Rich emailed on 03/21/2017

#Nasser 03/22/2017  Use Maple leaf count instead since buildin
#Nasser 03/23/2017  missing 'ln' for ElementaryFunctionQ added
#Nasser 03/24/2017  corrected the check for complex result
#Nasser 10/27/2017  check for leafsize and do not call ExpnType()
#                   if leaf size is "too large". Set at 500,000
#Nasser 12/22/2019  Added debug flag, added 'dilog' to special functions
#                   see problem 156, file Apostol_Problems
#Nasser 4/07/2022   add second output which gives reason for the grade

GradeAntiderivative := proc(result,optimal)
local leaf_count_result,
        leaf_count_optimal,
        ExpnType_result,
        ExpnType_optimal,
        debug:=false;

        leaf_count_result:=leafcount(result);
```

```
        #do NOT call ExpnType() if leaf size is too large. Recursion problem
        if leaf_count_result > 500000 then
            return "B","result has leaf size over 500,000. Avoiding possible recursion issues
        fi;

        leaf_count_optimal := leafcount(optimal);
        ExpnType_result  := ExpnType(result);
        ExpnType_optimal := ExpnType(optimal);

        if debug then
            print("ExpnType_result",ExpnType_result," ExpnType_optimal=",ExpnType_optimal);
        fi;

# If result and optimal are mathematical expressions,
#  GradeAntiderivative[result,optimal] returns
#   "F" if the result fails to integrate an expression that
#       is integrable
#   "C" if result involves higher level functions than necessary
#   "B" if result is more than twice the size of the optimal
#       antiderivative
#   "A" if result can be considered optimal

    #This check below actually is not needed, since I only
    #call this grading only for passed integrals. i.e. I check
    #for "F" before calling this. But no harm of keeping it here.
    #just in case.


    if not type(result,freeof('int')) then
        return "F","Result contains unresolved integral";
    fi;


    if ExpnType_result<=ExpnType_optimal then
        if debug then
             print("ExpnType_result<=ExpnType_optimal");
        fi;
        if is_contains_complex(result) then
            if is_contains_complex(optimal) then
                if debug then
                        print("both result and optimal complex");
                fi;
                if leaf_count_result<=2*leaf_count_optimal then
                    return "A"," ";
                else
                   return "B",cat("Both result and optimal contain complex but leaf count of
                                convert(leaf_count_result,string)," vs. $2 (",
```

```
                                   convert(leaf_count_optimal,string)," ) = ",convert(2*leaf_co
                 end if
             else #result contains complex but optimal is not
                 if debug then
                         print("result contains complex but optimal is not");
                 fi;
                 return "C","Result contains complex when optimal does not.";
             fi;
         else # result do not contain complex
              # this assumes optimal do not as well. No check is needed here.
             if debug then
                    print("result do not contain complex, this assumes optimal do not as well")
             fi;
             if leaf_count_result<=2*leaf_count_optimal then
                 if debug then
                     print("leaf_count_result<=2*leaf_count_optimal");
                 fi;
                 return "A"," ";
             else
                 if debug then
                     print("leaf_count_result>2*leaf_count_optimal");
                 fi;
                 return "B",cat("Leaf count of result is larger than twice the leaf count of opt
                                   convert(leaf_count_result,string),"$ vs. $2(",
                                   convert(leaf_count_optimal,string),")=",convert(2*leaf_count
             fi;
         fi;
     else #ExpnType(result) > ExpnType(optimal)
         if debug then
             print("ExpnType(result) > ExpnType(optimal)");
         fi;
         return "C",cat("Result contains higher order function than in optimal. Order ",
                        convert(ExpnType_result,string)," vs. order ",
                        convert(ExpnType_optimal,string),".");
     fi;

end proc:

#
# is_contains_complex(result)
# takes expressions and returns true if it contains "I" else false
#
#Nasser 032417
is_contains_complex:= proc(expression)
  return (has(expression,I));
end proc:
```

```
# The following summarizes the type number assigned an expression
# based on the functions it involves
# 1 = rational function
# 2 = algebraic function
# 3 = elementary function
# 4 = special function
# 5 = hyperpergeometric function
# 6 = appell function
# 7 = rootsum function
# 8 = integrate function
# 9 = unknown function

ExpnType := proc(expn)
  if type(expn,'atomic') then
    1
  elif type(expn,'list') then
    apply(max,map(ExpnType,expn))
  elif type(expn,'sqrt') then
    if type(op(1,expn),'rational') then
       1
    else
       max(2,ExpnType(op(1,expn)))
    end if
  elif type(expn,'`^`') then
    if type(op(2,expn),'integer') then
      ExpnType(op(1,expn))
    elif type(op(2,expn),'rational') then
      if type(op(1,expn),'rational') then
         1
      else
         max(2,ExpnType(op(1,expn)))
      end if
    else
         max(3,ExpnType(op(1,expn)),ExpnType(op(2,expn)))
    end if
  elif type(expn,'`+`') or type(expn,'`*`') then
    max(ExpnType(op(1,expn)),max(ExpnType(rest(expn))))
  elif ElementaryFunctionQ(op(0,expn)) then
    max(3,ExpnType(op(1,expn)))
  elif SpecialFunctionQ(op(0,expn)) then
    max(4,apply(max,map(ExpnType,[op(expn)])))
  elif HypergeometricFunctionQ(op(0,expn)) then
    max(5,apply(max,map(ExpnType,[op(expn)])))
  elif AppellFunctionQ(op(0,expn)) then
    max(6,apply(max,map(ExpnType,[op(expn)])))
  elif op(0,expn)='int' then
    max(8,apply(max,map(ExpnType,[op(expn)]))) else
```

```
  9
  end if
end proc:


ElementaryFunctionQ := proc(func)
  member(func,[
        exp,log,ln,
        sin,cos,tan,cot,sec,csc,
        arcsin,arccos,arctan,arccot,arcsec,arccsc,
        sinh,cosh,tanh,coth,sech,csch,
        arcsinh,arccosh,arctanh,arccoth,arcsech,arccsch])
end proc:

SpecialFunctionQ := proc(func)
  member(func,[
        erf,erfc,erfi,
        FresnelS,FresnelC,
        Ei,Ei,Li,Si,Ci,Shi,Chi,
        GAMMA,lnGAMMA,Psi,Zeta,polylog,dilog,LambertW,
        EllipticF,EllipticE,EllipticPi])
end proc:

HypergeometricFunctionQ := proc(func)
  member(func,[Hypergeometric1F1,hypergeom,HypergeometricPFQ])
end proc:

AppellFunctionQ := proc(func)
  member(func,[AppellF1])
end proc:

# u is a sum or product.  rest(u) returns all but the
# first term or factor of u.
rest := proc(u) local v;
  if nops(u)=2 then
     op(2,u)
  else
     apply(op(0,u),op(2..nops(u),u))
  end if
end proc:

#leafcount(u) returns the number of nodes in u.
#Nasser 3/23/17  Replaced by build-in leafCount from package in Maple
leafcount := proc(u)
   MmaTranslator[Mma][LeafCount](u);
end proc:
```

Sympy grading function

```
#Dec 24, 2019. Nasser M. Abbasi:
#             Port of original Maple grading function by
#             Albert Rich to use with Sympy/Python
#Dec 27, 2019  Nasser. Added `RootSum`. See problem 177, Timofeev file
#             added 'exp_polar'
from sympy import *

def leaf_count(expr):
   #sympy do not have leaf count function. This is approximation
   return round(1.7*count_ops(expr))

def is_sqrt(expr):
   if isinstance(expr,Pow):
      if expr.args[1] == Rational(1,2):
         return True
      else:
         return False
   else:
      return False

def is_elementary_function(func):
   return func in [exp,log,ln,sin,cos,tan,cot,sec,csc,
         asin,acos,atan,acot,asec,acsc,sinh,cosh,tanh,coth,sech,csch,
         asinh,acosh,atanh,acoth,asech,acsch
      ]

def is_special_function(func):
   return func in [ erf,erfc,erfi,
          fresnels,fresnelc,Ei,Ei,Li,Si,Ci,Shi,Chi,
          gamma,loggamma,digamma,zeta,polylog,LambertW,
          elliptic_f,elliptic_e,elliptic_pi,exp_polar
       ]

def is_hypergeometric_function(func):
   return func in [hyper]

def is_appell_function(func):
   return func in [appellf1]

def is_atom(expn):
   try:
      if expn.isAtom or isinstance(expn,int) or isinstance(expn,float):
        return True
      else:
        return False
```

```
    except AttributeError as error:
        return False

def expnType(expn):
    debug=False
    if debug:
        print("expn=",expn,"type(expn)=",type(expn))

    if is_atom(expn):
       return 1
    elif isinstance(expn,list):
        return max(map(expnType, expn))   #apply(max,map(ExpnType,expn))
    elif  is_sqrt(expn):
        if isinstance(expn.args[0],Rational): #type(op(1,expn),'rational')
            return 1
        else:
            return max(2,expnType(expn.args[0]))  #max(2,ExpnType(op(1,expn)))
    elif isinstance(expn,Pow):   #type(expn,'`^`')
        if isinstance(expn.args[1],Integer):  #type(op(2,expn),'integer')
            return expnType(expn.args[0])   #ExpnType(op(1,expn))
        elif isinstance(expn.args[1],Rational):  #type(op(2,expn),'rational')
            if isinstance(expn.args[0],Rational): #type(op(1,expn),'rational')
                return 1
            else:
                return max(2,expnType(expn.args[0]))  #max(2,ExpnType(op(1,expn)))
        else:
            return max(3,expnType(expn.args[0]),expnType(expn.args[1])) #max(3,ExpnType(op(1,expn)),ExpnTy
    elif isinstance(expn,Add) or isinstance(expn,Mul): #type(expn,'`+`') or type(expn,'`*`')
        m1 = expnType(expn.args[0])
        m2 = expnType(list(expn.args[1:]))
        return max(m1,m2)  #max(ExpnType(op(1,expn)),max(ExpnType(rest(expn))))
    elif is_elementary_function(expn.func):  #ElementaryFunctionQ(op(0,expn))
        return max(3,expnType(expn.args[0]))  #max(3,ExpnType(op(1,expn)))
    elif is_special_function(expn.func): #SpecialFunctionQ(op(0,expn))
        m1 = max(map(expnType, list(expn.args)))
        return max(4,m1)   #max(4,apply(max,map(ExpnType,[op(expn)])))
    elif is_hypergeometric_function(expn.func): #HypergeometricFunctionQ(op(0,expn))
        m1 = max(map(expnType, list(expn.args)))
        return max(5,m1)   #max(5,apply(max,map(ExpnType,[op(expn)])))
    elif is_appell_function(expn.func):
        m1 = max(map(expnType, list(expn.args)))
        return max(6,m1)   #max(5,apply(max,map(ExpnType,[op(expn)])))
    elif isinstance(expn,RootSum):
        m1 = max(map(expnType, list(expn.args))) #Apply[Max,Append[Map[ExpnType,Apply[List,expn]],7]],
        return max(7,m1)
    elif str(expn).find("Integral") != -1:
```

```
        m1 = max(map(expnType, list(expn.args)))
        return max(8,m1)    #max(5,apply(max,map(ExpnType,[op(expn)])))
    else:
        return 9

#main function
def grade_antiderivative(result,optimal):

    #print ("Enter grade_antiderivative for sagemath")
    #print("Enter grade_antiderivative, result=",result," optimal=",optimal)

    leaf_count_result  = leaf_count(result)
    leaf_count_optimal = leaf_count(optimal)

    #print("leaf_count_result=",leaf_count_result)
    #print("leaf_count_optimal=",leaf_count_optimal)

    expnType_result  = expnType(result)
    expnType_optimal = expnType(optimal)

    if str(result).find("Integral") != -1:
        grade = "F"
        grade_annotation =""
    else:
        if expnType_result <= expnType_optimal:
            if result.has(I):
                if optimal.has(I): #both result and optimal complex
                    if leaf_count_result <= 2*leaf_count_optimal:
                        grade = "A"
                        grade_annotation =""
                    else:
                        grade = "B"
                        grade_annotation ="Both result and optimal contain complex but leaf count of result is large
                else: #result contains complex but optimal is not
                    grade = "C"
                    grade_annotation ="Result contains complex when optimal does not."
            else: # result do not contain complex, this assumes optimal do not as well
                if leaf_count_result <= 2*leaf_count_optimal:
                    grade = "A"
                    grade_annotation =""
                else:
                    grade = "B"
                    grade_annotation ="Leaf count of result is larger than twice the leaf count of optimal. "+str(le
        else:
            grade = "C"
            grade_annotation ="Result contains higher order function than in optimal. Order "+str(ExpnType_
```

```
    #print("Before returning. grade=",grade, " grade_annotation=",grade_annotation)

    return grade, grade_annotation
```

SageMath grading function

```
#Dec 24, 2019. Nasser: Ported original Maple grading function by
#              Albert Rich to use with Sagemath. This is used to
#              grade Fricas, Giac and Maxima results.
#Dec 24, 2019. Nasser: Added 'exp_integral_e' and 'sng', 'sin_integral'
#              'arctan2','floor','abs','log_integral'
#June 4, 2022  Made default grade_annotation "none" instead of "" due
#              issue later when reading the file.
#July 14, 2022. Added ellipticF. This is until they fix sagemath, then remove it.

from sage.all import *
from sage.symbolic.operators import add_vararg, mul_vararg

debug=False;

def tree_size(expr):
    r"""
    Return the tree size of this expression.
    """
    #print("Enter tree_size, expr is ",expr)

    if expr not in SR:
        # deal with lists, tuples, vectors
        return 1 + sum(tree_size(a) for a in expr)
    expr = SR(expr)
    x, aa = expr.operator(), expr.operands()
    if x is None:
        return 1
    else:
        return 1 + sum(tree_size(a) for a in aa)

def is_sqrt(expr):
    if expr.operator() == operator.pow:   #isinstance(expr,Pow):
        if expr.operands()[1]==1/2: #expr.args[1] == Rational(1,2):
            if debug: print ("expr is sqrt")
            return True
        else:
            return False
    else:
        return False
```

```
def is_elementary_function(func):
    #debug=False
    m = func.name() in ['exp','log','ln',
            'sin','cos','tan','cot','sec','csc',
            'arcsin','arccos','arctan','arccot','arcsec','arccsc',
            'sinh','cosh','tanh','coth','sech','csch',
            'arcsinh','arccosh','arctanh','arccoth','arcsech','arccsch','sgn',
        'arctan2','floor','abs'
        ]
    if debug:
        if m:
            print ("func ", func , " is elementary_function")
        else:
            print ("func ", func , " is NOT elementary_function")


    return m

def is_special_function(func):
    #debug=False
    if debug:
        print ("type(func)=", type(func))

    m= func.name() in ['erf','erfc','erfi','fresnel_sin','fresnel_cos','Ei',
          'Ei','Li','Si','sin_integral','Ci','cos_integral','Shi','sinh_integral'
          'Chi','cosh_integral','gamma','log_gamma','psi,zeta',
          'polylog','lambert_w','elliptic_f','elliptic_e','ellipticF',
          'elliptic_pi','exp_integral_e','log_integral']

    if debug:
        print ("m=",m)
        if m:
            print ("func ", func ," is special_function")
        else:
            print ("func ", func ," is NOT special_function")


    return m


def is_hypergeometric_function(func):
    return func.name() in ['hypergeometric','hypergeometric_M','hypergeometric_U']

def is_appell_function(func):
    return func.name() in ['hypergeometric']   #[appellf1] can't find this in sagemath
```

```
def is_atom(expn):

   #debug=False
   if debug:
       print ("Enter is_atom, expn=",expn)

   if not hasattr(expn, 'parent'):
      return False


   #thanks to answer at https://ask.sagemath.org/question/49179/what-is-sagemath-equivalent-to-atomic-t
   try:
      if expn.parent() is SR:
         return expn.operator() is None
      if expn.parent() in (ZZ, QQ, AA, QQbar):
         return expn in expn.parent() # Should always return True
      if hasattr(expn.parent(),"base_ring") and hasattr(expn.parent(),"gens"):
         return expn in expn.parent().base_ring() or expn in expn.parent().gens()

      return False

   except AttributeError as error:
      print("Exception,AttributeError in  is_atom")
      print ("cought exception" , type(error).__name__ )
      return False


def expnType(expn):

   if debug:
      print (">>>>>Enter expnType, expn=", expn)
      print (">>>>>is_atom(expn)=", is_atom(expn))

   if is_atom(expn):
      return 1
   elif type(expn)==list:   #isinstance(expn,list):
      return max(map(expnType, expn))   #apply(max,map(ExpnType,expn))
   elif  is_sqrt(expn):
      if  type(expn.operands()[0])==Rational:  #type(isinstance(expn.args[0],Rational):
         return 1
      else:
         return max(2,expnType(expn.operands()[0]))  #max(2,expnType(expn.args[0]))
   elif expn.operator() == operator.pow:   #isinstance(expn,Pow)
      if type(expn.operands()[1])==Integer:   #isinstance(expn.args[1],Integer)
         return expnType(expn.operands()[0])   #expnType(expn.args[0])
      elif type(expn.operands()[1])==Rational:   #isinstance(expn.args[1],Rational)
         if type(expn.operands()[0])==Rational:  #isinstance(expn.args[0],Rational)
```

```
                return 1
            else:
                return max(2,expnType(expn.operands()[0]))  #max(2,expnType(expn.args[0]))
        else:
            return max(3,expnType(expn.operands()[0]),expnType(expn.operands()[1])) #max(3,expnType(expn.
    elif expn.operator() == add_vararg or expn.operator() == mul_vararg: #isinstance(expn,Add) or isinst
        m1 = expnType(expn.operands()[0]) #expnType(expn.args[0])
        m2 = expnType(expn.operands()[1:]) #expnType(list(expn.args[1:]))
        return max(m1,m2)  #max(ExpnType(op(1,expn)),max(ExpnType(rest(expn))))
    elif is_elementary_function(expn.operator()):  #is_elementary_function(expn.func)
        return max(3,expnType(expn.operands()[0]))
    elif is_special_function(expn.operator()): #is_special_function(expn.func)
        m1 = max(map(expnType, expn.operands()))      #max(map(expnType, list(expn.args)))
        return max(4,m1)   #max(4,m1)
    elif is_hypergeometric_function(expn.operator()): #is_hypergeometric_function(expn.func)
        m1 = max(map(expnType, expn.operands()))       #max(map(expnType, list(expn.args)))
        return max(5,m1)   #max(5,m1)
    elif is_appell_function(expn.operator()):
        m1 = max(map(expnType, expn.operands()))       #max(map(expnType, list(expn.args)))
        return max(6,m1)   #max(6,m1)
    elif str(expn).find("Integral") != -1: #this will never happen, since it
                #is checked before calling the grading function that is passed.
                #but kept it here.
        m1 = max(map(expnType, expn.operands()))       #max(map(expnType, list(expn.args)))
        return max(8,m1)   #max(5,apply(max,map(ExpnType,[op(expn)])))
    else:
        return 9

#main function
def grade_antiderivative(result,optimal):


    if debug:
        print ("Enter grade_antiderivative for sagemath")
        print("Enter grade_antiderivative, result=",result)
        print("Enter grade_antiderivative, optimal=",optimal)
        print("type(anti)=",type(result))
        print("type(optimal)=",type(optimal))

    leaf_count_result  = tree_size(result) #leaf_count(result)
    leaf_count_optimal = tree_size(optimal) #leaf_count(optimal)

    #if debug: print ("leaf_count_result=", leaf_count_result, "leaf_count_optimal=",leaf_count_optimal)


    expnType_result  = expnType(result)
    expnType_optimal = expnType(optimal)
```

```
if debug: print ("expnType_result=", expnType_result, "expnType_optimal=",expnType_optimal)

if expnType_result <= expnType_optimal:
    if result.has(I):
        if optimal.has(I): #both result and optimal complex
            if leaf_count_result <= 2*leaf_count_optimal:
                grade = "A"
                grade_annotation ="none"
            else:
                grade = "B"
                grade_annotation ="Both result and optimal contain complex but leaf count of result is larger th
        else: #result contains complex but optimal is not
            grade = "C"
            grade_annotation ="Result contains complex when optimal does not."
    else: # result do not contain complex, this assumes optimal do not as well
        if leaf_count_result <= 2*leaf_count_optimal:
            grade = "A"
            grade_annotation ="none"
        else:
            grade = "B"
            grade_annotation ="Leaf count of result is larger than twice the leaf count of optimal. "+str(leaf_c
else:
    grade = "C"
    grade_annotation ="Result contains higher order function than in optimal. Order "+str(expnType_resul


print("Before returning. grade=",grade, " grade_annotation=",grade_annotation)

return grade, grade_annotation
```