[VA] SRC #012d - Then and Now: Area

Threaded Mode | Linear Mode

8th January, 2023, 00:46 **Post: #1**

Valentin Albillo
Senior Member

Posts: 970
Joined: Feb 2015
Warning Level: 0%

[VA] SRC #012d - Then and Now: Area

Hi, all,

Welcome to the 4th part of my ongoing ***SRC #012 - Then and Now***, where I'm showing that advanced vintage *HP* calcs which were great problem-solvers back **THEN** in the 80's are **NOW** still perfectly capable of solving recent, non-trivial problems intended to be tackled using modern *PC*s, never mind ancient calcs.

In the next weeks I'm proposing **six** *increasingly harder* such problems for you to try and solve using your vintage *HP* calcs while abiding by the *mandatory rules* summarized here:

> You **must** use **VINTAGE HP CALCS** (physical/virtual,) coding in either **RPN** (inc. *mcode*), **RPL** (variants existing at the time, inc. *SysRPL*) or **HP-71B** languages (inc. *BASIC, FORTH, Assembler*), so **NO** *XCAS, MATHEMATICA, MAPLE, EXCEL, C/C++/C#, PYTHON, LUA,* etc., **NO LENGTHY MATH** SESSIONS and **NO CODE PANELS**.
>
> On the plus side, you may use any official/popular modules, pacs or libraries available at the time such as the **Math Pac**, **HP-IL** and **JPC ROM**s for the **HP-71B**, the **Advantage Module**, **PPC ROM** and *EM* for the **HP-41**, and *libraries* for the ***RPL*** models, etc..

Once *P1 (**Probability**), P2 (**Root**) and P3 (**Sum**)* are over, now's the turn for ***Problem 4***, which deals with ***area*** and it's ever-so-slightly harder than the previous ones. Also, lest you'd think it looks like a textbook exercise, rest assured there's some nice surprises to deal with (if you've got what it takes, that is ...)

Problem 4: Area

> The thermodynamic efficiency of the combined cycle for a hypothetical gas turbine using *ethane* as fuel is related to the area of the region ***R*** of the *X-Y* plane defined by the expression
>
> $$\frac{1}{e^{\left((x-d)^3-y\right)^2}} > \frac{y^2}{M} + \frac{1}{e^{\sin y}}$$
>
> where *radians* are used, ***M*** is *30.070* (ethane's Molar mass in g/mol, see Properties,) and ***d*** *is 1.598* (Molar density in mol/dm^3). Write a program to compute this area.

Here the focus is on computing the ***area*** of ***R***, so you might begin by locating ***R*** via estimating *(2 decimal digits at most, ±d.dd)* the coordinates of some rectangle that fully encloses it, either by writing code to sweep the *X-Y* plane or simply by using a graphing calculator or online function plotter, your choice.

Once obtained, your program (which should take *no inputs*,) can refine and use them to compute and output ***R***'s area accurate to *10-12 correct digits* (give or take a few *ulp*,) and the faster the running time the better.

If I see interest, in a week or so I'll post my own *original solution* for the **HP-71B**, which is a *very short* program that does the job. In the meantime, let's see your very own clever solutions **AND** *remember the above rules*. this is *strictly for vintage HP calcs*, Ok ?

Also, please be kind and avoid *spoiling* it for other interested people by immediately posting your solution and/or results. *Wait* instead until next **Wednesday 9:00 pm GMT+1** so that other people will have a chance. In the meantime you can mull it over so that you eventually post the *one* (1) message featuring your fully refined solution instead of a *myriad* posts refining it little by little. Got it ? :)

V.

PM WWW FIND EDIT QUOTE REPORT

10th January, 2023, 12:33 **Post: #2**

J-F Garnier
Senior Member

Posts: 819
Joined: Dec 2013

RE: [VA] SRC #012d - Then and Now: Area

I will respect the wish of Valentin to postpone any results before next Wednesday, so I will not even say if I'm working on a solution, have a result, or have no idea at all.
However, the proposed equation looks strange from a dimensional point of view:

$$\frac{1}{e^{\left((x-d)^3-y\right)^2}} > \frac{y^2}{M} + \frac{1}{e^{\sin y}}$$

Since y appears in sin(y), it should be dimensionless, but this is not compatible with the term y^2/M that must be dimensionless too.
It could be an empirical formula coming from a fit of experimental data, still I would be curious to know its origin.

J-F

EMAIL PM WWW FIND QUOTE REPORT

10th January, 2023, 13:22 **Post: #3**

Valentin Albillo
Senior Member

Posts: 970
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012d - Then and Now: Area

.
Hi, **J-F**,

> **J-F Garnier Wrote:** (10th January, 2023 12:33)
>
> However, the proposed equation looks strange from a dimensional point of view:
>
> $$\frac{1}{e^{\left((x-d)^3-y\right)^2}} > \frac{y^2}{M} + \frac{1}{e^{\sin y}}$$
>
> Since y appears in sin(y), it should be dimensionless, but this is not compatible with the term y^2/M that must be dimensionless too.
> It could be an empirical formula coming from a fit of experimental data, still I would be curious to know its origin.

All will be clear in due time, for now please **ignore** any and all dimensions, consider only pure ***dimensionless*** **numerical values** throughout.

Thanks for your interest and eagerly awaiting for your results. If you would include some of your sleuthing, so much the better ... :)
V.

PM WWW FIND EDIT QUOTE REPORT

10th January, 2023, 14:16 **Post: #4**

PeterP
Member

Posts: 172
Joined: Jul 2015

RE: [VA] SRC #012d - Then and Now: Area

You were braver than me, JF, as the scraps of remnants of a physicist in my also tripped up over „radians" mixed with mols etc.

I will go one step further now than JF and say that I have no much of an idea at all. (And admit to being a bit sad about not able to follow and learn from the iterative approaches and discussions that have happened in past SSMCs and prior challenges in the series, given the immense delight and learning I derive from following along, feeling inspired, learning an idea, gleaning a hint. I am more of a collaborative learner than only reveling in others Feynman-like genius results, even though I do enjoy that also. But, as we are told, „rules must be obeyed, the winner takes it all", collaboration is for wimps…)

PM FIND QUOTE REPORT

11th January, 2023, 04:55 **Post: #5**

Valentin Albillo
Senior Member

Posts: 970
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012d - Then and Now: Area

.
Hi, **Martin**,

Martin Hepperle Wrote: (10th January, 2023 15:01)

A HP 85 probably does not qualify, even if it is a officially called "Calculator".

Nevertheless, it can produce just by its brute computing force a result with a rather long BASIC program (but, hey, with graphics output). The area of interest, I must admit, was found by trial and error, though.
The result is about 2.07662029 but needs at least N=32 samples in each:

Thanks for your interest in this ***Problem 4*** but please try to abide by the rules I stated in my OP, namely:

1) This is strictly intended for vintage HP calcs, the HP-85 isn't one. Post code only for the allowed models I listed.

2) You posted your solution 1.5 days before the date and time I specified. Please comply as others did.

3) I specifically requested that CODE panels should not be used, yet you did.

4) Your code doesn't produce the result you gave, and further you didn't include in your post any output from your program, i.e. neither the graphics nor the actual numerical results.

See if you can correct some or all of those 4 points and in the future please do your best to abide by the rules, they aren't optionsl and I stated them repeatedly and clearly enough.

Regards.
V.

PM WWW FIND EDIT QUOTE REPORT

11th January, 2023, 14:46 (This post was last modified: 11th January, 2023 14:47 by Martin Hepperle.) **Post: #6**

Martin Hepperle

Senior Member

Posts: 367
Joined: May 2014

RE: [VA] SRC #012d - Then and Now: Area

.. previous post removed due to violation of the rules and because there is surely a more elegant solution.

PM FIND QUOTE REPORT

11th January, 2023, 15:59 **Post: #7**

Valentin Albillo

Senior Member

Posts: 970
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012d - Then and Now: Area

.
Hi, **Martin**,

Martin Hepperle Wrote: (11th January, 2023 14:46)

.. previous post removed due to violation of the rules and because there is surely a more elegant solution.

Thanks, **Martin**, much appreciated. You can still very easily convert your existing program to run on the **HP-71B** or even implement a more-"elegant", more efficient and accurate solution, perhaps along the lines of the approach used to produce the result you posted.

Regards.
V.

PM WWW FIND EDIT QUOTE REPORT

12th January, 2023, 00:17 (This post was last modified: 12th January, 2023 00:21 by Albert Chan.) **Post: #8**

Albert Chan

Senior Member

Posts: 2,148
Joined: Jul 2018

RE: [VA] SRC #012d - Then and Now: Area

exp(-((x-d)^3-y)^2) > (R = y^2/M + exp(-sin(y)))   // Given: M=30.07, d=1.596

Let s = sqrt(-log(R)) ≥ 0

((x-d)^3-y)^2 < s^2
y-s < (x-d)^3 < y+s

x is real if s is real --> R ≤ 1 --> 0 ≤ y ≤ 2.82740261413

Height, $f(y) = x_2 - x_1 = \sqrt[3]{y+s} - \sqrt[3]{y-s}$

f slope is infinite when y = s --> R = exp(-y^2) --> y = 0 or 0.831971149978

Let a = 0.831971149978, a+b = a+B/2 = 2.82740261413
Infinite slopes at y = 0 and a, moved to z = 0

Area = $\int_0^{a+b} f(y)\ dy = \int_0^{1/2} \left[\left(f(a\,z) + f(a - a\,z) \right) \cdot a \ + \ f(a + B\,z) \cdot B \right] dz$

Let g(z) = RHS integrand, and substitute z = $x^3/2$, to make z=0 infinite slope, down to 0.
INTEGRAL built-in u-transform should turn curve to bell-shaped, easy to integrate.   (*)

Area = $\int_0^{1/2} g(z)\ dz = \int_0^1 g\left(\frac{x^3}{2}\right) \cdot \left(\frac{3}{2} x^2\ dx\right)$

```
10 DESTROY ALL @ M=30.07 @ A=.831971149978 @ B=2.82740261413-A @ P=1E-6
20 T=1/3 @ DEF FND(Y,S)=(Y+S)^T-SGN(Y-S)*ABS(Y-S)^T
30 DEF FNF(Y)=FND(Y,SQR(-LN(Y*Y/M+EXP(-SIN(Y)))))
40 B=B*2 @ DEF FNG(Z)=(FNF(A*Z)+FNF(A-A*Z))*A+FNF(A+B*Z)*B
50 SETTIME 0 @ DISP INTEGRAL(0,1,P,FNG(.5*IVAR^3)*IVAR^2)*1.5,TIME

>run
 2.07662636748     .49     ! @200× --> HP71B = 98 sec
>p=1e-9 @ run 50
 2.07662636775     .92     ! @200× --> HP71B = 184 sec
```

(*) at y = a+b, f slope is infinite too.
Above code assumed u-transform able to fix this edge.
If not, we move all edges to z=0, with this revised g(z).

```
>40 DEF FNG(Z)=(FNF(A*Z)+FNF(A-A*Z))*A+(FNF(A+B*Z)+FNF(A+B-B*Z))*B
>run
 2.07662636769     .6      ! @200× --> HP71B = 120 sec
>p=1e-9 @ run 50
 2.07662636775     1.2     ! @200× --> HP71B = 240 sec
```

EMAIL PM FIND QUOTE REPORT

12th January, 2023, 02:00 **Post: #9**

Fernando del Rey
Junior Member
Posts: 20
Joined: Dec 2013

RE: [VA] SRC #012d - Then and Now: Area

I've been working on this Problem 4 since Valentin posted this interesting challenge, but I have not yet reached a solution accurate to 10-12 correct digits as requested. Still, while not abiding to Valentín's instruction to post only one message featuring the fully refined solution, I am going to share what I have done so far, as I have some doubts that I'll be able to complete the work within the next few of days.

Problem 4 asks us to calculate the area in the *(x,y)* plane where the criteria (1) below is met:

1/e^(((x-d)^3-y)^2) > y^2/M+1/e^(sin(y)) (1)

To simplify the equation a little bit (and also the code to follow), let's define:

x'=x-d

Then we can represent equation (1) as:

1/e^((x'^3-y)^2) > y^2/M+1/e^(sin(y))

Let's then name:

1/e^((x'^3-y)^2) = f(x',y)

and:

y^2/M+1/e^(sin(y)) = g(y)

We know that *f(x',y)* will always be in the range from 0 to 1 as the denominator of the division is positive.

0<f(x',y)≤1

We also know that *g(y)* will be in the range from y2/M+e^-1 to ∞, therefore in the range from e^-1 to ∞.

$e^{(-1)}<g(y)<\infty$

In the *(f,g)* plane, we can represent the area that meets the three criteria:

$0<f\leq 1$
$g>e^{(-1)}$
$f>g$

I wish I knew how to include an image I have drawn inside this post, but I don't know how to do it. The area in the *(f,g)* plane meeting the three criteria listed above is a triangle defined by the points A=(1,1), B=(e^-1,e^-1) and C=(1,e^-1).

From point A we can now derive:

$1>g(y)$

$1>y^2/M+1/e$

$M(1-1/e)>y^2$

$|y|<\sqrt{(M(1-1/e))}=4.3598$

$-4.3598 < y < 4.3598$

And from point B we can derive:

$f(x',y)>e-1$

$1/e^{((x'^3-y)^2)} >1/e \rightarrow (x'^3-y)^2<1$

$|x'^3-y|<1$

And taking the extremes of *y* calculated earlier, we get:

$-1-4.3598<x'^3<1+4,3598$

$\sqrt[3]{(-5.3598)}<x'<\sqrt[3]{5.3598}$

$-1.75 < x' < 1.75$

To allow for some margin, we will use the following rectangle area where the original Problem 4 criteria is met:

$-1.76 < x' < 1.76$
$-4.36 < y < 4.36$

Now, to obtain an approximate value of the area where the criteria defined by equation (1) is met, we can use the following program for the HP-71B:

```
10 X1=-1.76 @ X2=1.76 @ Y1=-4.36 @ Y2=4.36 @ INPUT "N? ";N
20 D1=(X2-X1)/N @ D2=(Y2-Y1)/N @ D=D1*D2 @ S=0 @ T0=TIME
30 FOR Y=Y1+D2/2 TO Y2-D2/2 STEP D2
40 G=Y*Y/30.07+EXP(-SIN(Y))
50 FOR X=X1+D1/2 TO X2-D1/2 STEP D1
60 IF EXP(-(X*X*X-Y)*(X*X*X-Y))>G THEN S=S+D
70 NEXT X
80 NEXT Y
90 DISP "Area=";S;TIME-T0
```

This program divides the rectangle bounded by the limits of x' and y as defined above into NxN small rectangles and scans all small rectangles progressing row by row. If the criteria of equation (1) is met at the middle point of a given rectangle, its area is added to the summation S. If the criteria is not met, nothing is added to S.

At the end, the value S gives a rough approximation of the area where criteria (1) is met. Trying with different values of N, I get the followings times are for Emu71/Win running on my PC (which is about 970 times faster than the real 71B):

<u>Grid size; Area; Time (s)</u>
10x10; 2.455552; .02
20X20; 2.148608; .04
50x50; 2.11177472; .24
100x100; 2.09028864; .94
200x200; 2.07647616; 3.72
500x500; 2.0788703232; 23.61
1000x1000; 2.0765989376; 97.96

2000X2000; 2.07705168; 378.71

So, we seem to be converging to a value of around **2.077**, but the convergence is too slow to make this brute-force method practical. We need to apply a smarter method, which could take advantage of the INTEGRAL function provided by the 71B Math ROM. But I need to work on it, and I'm sure others will post correct solutions before I have time to do so.

By adding DISP X;Y at the end of line 60 you can get a rough idea of the shape and position of the area meeting criteria (1), by looking at the pairs of values being displayed as the program runs.

60 IF EXP(-(X*X*X-Y)*(X*X*X-Y))>G THEN S=S+D @ DISP X;Y

If you make the grid accurate enough, say 500x500, you can see a small area at the upper left side of the rectangle where criteria (1) is met, and then a much larger area in the middle to lower half of the rectangle.

f you make the grid 2000x2000, the small area at the upper left corner can be seen with better definition.

With a grid of 100x100 you don't get to see the small area in the upper left part, you will only see the large area starting towards the middle of the rectangle. It would be interesting to plot these areas, but I can't do it with a bare-bones 71B or with Emu71/Win.

It's very possible that there be more than 2 different areas where the criteria (1) is met, but I have been able to "see" only those two so far.

My proposed way forward would be to express the equation bounding the area where criteria (1) is met:

1/e^((x'^3-y)^2) = y^2/M+1/e^(sin(y))

Transforming it to isolate *x'* as a function of *y*, and then using the INTEGRAL function of the Math ROM to do the calculation. But I sill need to figure out what the boundary values for the integration would be, and how to cater for the fact that there seems to be separate areas (at least two) where criteria (1) is met. So still some work to do!

EMAIL PM FIND QUOTE REPORT

12th January, 2023, 09:31 **Post: #10**

Werner
Senior Member

Posts: 777
Joined: Dec 2013

RE: [VA] SRC #012d - Then and Now: Area

All calculators used (42S, DM42 and Free42) have

- 30.07 in variable "M"
- 1.598 in variable "d"
- RAD mode.

1. define the rectangle

Let's first determine the y-values of the rectangle.

It's clear that EXP(-((x-d)^3-y)^2) <= 1, and it is equal to 1 for all y=(x-d)^3
so y^2/M + EXP(-SIN(y)) = 1 is a border: anything < 1 may belong to the area, > 1 does not.
that turns out to be the case for y=0 and y=2.8274.., found with the 42S solver program.
So our area is between these two y values.

```
00 { 27-Byte Prgm }
01▸LBL "VA4Y"
02 MVAR "Y"
03 RCL "Y"
04 ENTER
05 X^2
06 RCL÷ "M"
07 X<>Y
08 SIN
09 +/-
10 E^X
11 +
12 1
13 -
14 END
```

Incidentally (and accidentally) I also found a tiny region -4.0851.. <= y <= -4.0492.. where the original inequality will hold (eg. y=-4.05 and x=0.004). There are no other regions since
EXP(-SIN(y)) >= 1/e, abs(y) < SQRT((1 - 1/e)*M) = 4.3598.

If we rewrite the orginal formula as an equality, we can isolate x for a given y:

(1) x = d + CBRT(y +/- SQRT(-LN(y^2/M + EXP(-SIN(y)))))

When the result of LN is positive, we're outside of the area and there are no real solutions for x.
Within the area, there are two results for x (that coincide at the edges), describing the shape.
A bit of trial and error gives the following rectangle boundaries, just for the purpose of graphing the area (I used Y=-0.1 to have the shape come clear off the bottom edge).

```
Main Tiny
X0: 0.97 -0.001
X1: 3.05 0.0045
Y0: -0.1 -4.049
Y1: 2.85 -4.086
```

2. graph the shape

I graphed the main shape on my DM42 with the following drawing routine, with GrMod=2
(I enlarged the X range to X0=0.58 and X1=3.44 to have the X and Y scale be the same)
Set the values with VARMENU "VA4D", EXIT the menu and do XEQ "VA4D".
(the program can be made a lot more efficient, but that was not the scope here).

VA4.bmp (Size: 12.31 KB / Downloads: 20)

```
00 { 167-Byte Prgm }
01▸LBL "VA4D"
02 MVAR "X0"
03 MVAR "X1"
04 MVAR "Y0"
05 MVAR "Y1"
06 MVAR "GrMod"
07 CLLCD
08 RCL "X1"
09 RCL- "X0"
10 RCL "ResX"
11 LSTO "X"
12 DSE ST X
13 ÷
14 LSTO "Sx"
15 RCL "Y0"
16 RCL- "Y1"
17 RCL "ResY"
18 DSE ST X
19 ÷
20 LSTO "Sy"
21▸LBL 10
22 RCL "Sx"
23 RCL× "X"
24 LASTX
25 -
26 RCL+ "X0"
27 LSTO "Xc"
28 RCL "ResY"
29 LSTO "Y"
30▸LBL 11
31 RCL "Sy"
32 RCL× "Y"
33 LASTX
34 -
35 RCL+ "Y1"
36 ENTER
37 ENTER
38 X^2
39 RCL÷ "M"
40 X<>Y
41 SIN
42 +/-
43 E^X
44 +
45 RCL "Xc"
46 RCL- "d"
47 3
48 Y^X
49 R^
50 -
```

```
51 X^2
52 +/-
53 E^X
54 X≤Y?
55 GTO 00
56 RCL "Y"
57 RCL "X"
58 PIXEL
59▸LBL 00
60 DSE "Y"
61 GTO 11
62 DSE "X"
63 GTO 10
64 END
```

3. calculate area

Well then. All that is needed is to integrate the shape over the Y-range. That can be done with a standard 42S integration routine, integrating dx = x2-x1, with x2 the positive root of (1):

```
c := SQRT(-LN(y^2/M + EXP(-SIN(y))));
dx := CBRT(y + c) - CBRT(y - c);
Area := integral(y=0 to 2.8274,dx);
```

Here I use LLIM=0 and ULIM=2.82740261413 for Main area
LLIM=-4.08514674764 and ULIM=-4.092122644 for Tiny area
the accurate limits are calculated with the solver program VA4Y above
The integration is done in Free42.

```
00 { 55-Byte Prgm }
01▸LBL "VA4"
02 MVAR "Y"
03 RCL "Y"
04 ENTER
05 X^2
06 RCL÷ "M"
07 X<>Y
08 SIN
09 +/-
10 E^X
11 +
12 LN
13 X>0?
14 CLX
15 X=0?
16 RTN
17 +/-
18 SQRT
19 ENTER
20 RCL+ "Y"
21 XEQ 13
22 X<>Y
23 RCL- "Y"
24 XEQ 13
25 +
26 RTN
27▸LBL 13 @ Cube Root
28 SIGN
29 LASTX
30 ABS
31 3
32 1/X
33 Y^X
34 ×
35 END
```

with ACC=1E-5 we get

2.07663 for the Main area
7.19762E-5 for the Tiny area

This accuracy needs 16383 function evaluations (for the Main area). That would take quite some time on a real 42S. The DM42 on USB takes about 94s.
Since the constant M is only given to 4 sig. digits, probably the Tiny area can be considered negligible. If not, I took the calculation of the integral a bit further, only on Free42 this time:

```
ACC #evals Main Tiny

1E-5 16383 2.07662797558 7.19761929874-5
1E-7 131071 2.07662603505 7.19761930422E-5
1E-9 524287 2.07662630959 =
1E-11 = =
1E-15 = =
```

So the sum of both areas is **2.07669828578**

And now I'm going to try and understand Albert's contribution ;-)

Cheers, Werner

EMAIL PM FIND QUOTE REPORT

12th January, 2023, 11:34 **Post: #11**

J-F Garnier
Senior Member

Posts: 819
Joined: Dec 2013

RE: [VA] SRC #012d - Then and Now: Area

(written before reading other's contributions)

So here is my solution.
First, my initial observations (LHS is Left-Hand Side, RHS is Right-Hand Side of the equation):
LHS and RHS are >0
LHS is <=1
RHS must be <=1 for a corresponding x value to exist in the LHS

Then, I noticed that the d value is irrelevant, it just translates the whole graph in the X-Y plane without changing its area, so I ignored d and replace (x-d) by x.

Then I used the HP-71B to find out the limits of y:
ymin is 0, it gives RHS=1, and any y<0 gives RHS>1 with no solution in the LHS.
y>sqrt(m) gives RHS>1 for sure, but is not the limit for RHS>1.
ymax can be obtained by solving RHS=1 for y>2, and is about 2.8.

Then, I noticed that x can be quite easily expressed as a function of y, since x occurs only once in the equation.
So I can calculate the area by integrating the length of the segment between the 2 points solutions of x for a given y, i.e. the two points where a horizontal line crosses the area contour, between y=0 and y=ymax.

Here is where the practical issues start. This integral takes *a lot of time* to compute, even with moderate accuracy.
Relating this aspect to the places where the curve enclosing the area is horizontal, I tried the split the calculation in different zones. With this approach I succeeded to compute the integral in a reasonable time, at least on the 71B, even if not particularly fast.

Below is the program and results for the HP-71B and HP-75C:

```
10 ! SRC12D
20 ! HP-71B version
30 ! For HP-75 version, change IVAR by dummy Y var
40 OPTION ANGLE RADIANS
50 M=30.07
60 !
70 DEF FNR(Y)=Y*Y/M+EXP(-SIN(Y)) ! RHS
80 DEF FNR1(Y)=Y*Y/M+EXP(-SIN(Y))-1 ! RHS-1
90 DEF FNL(X3,Y)=EXP(-(X3-Y)^2) ! LHS
100 DEF FNE(Y)=FNL(0,Y)-FNR(Y) ! LHS-RHS for X=0
110 !
120 A=FNROOT(2,3,FNR1(FVAR))
130 DISP "RHS=1 for Ymax=";A
150 C=FNROOT(.5,2,FNE(FVAR))
160 DISP "LHS=RHS for X= 0 and Y=";C
170 !
180 E=10^(-12) @ B=.01
190 DISP "Ix IBOUND"
200 T1=TIME
210 I0=INTEGRAL(0,.001,E*1000,FNF(IVAR)) @ DISP I0;IBOUND
220 I1=INTEGRAL(.001,C-B,E,FNF(IVAR)) @ DISP I1;IBOUND
230 I2=INTEGRAL(C-B,C+B,E*10,FNF(IVAR)) @ DISP I2;IBOUND
240 I3=INTEGRAL(C+B,A,E,FNF(IVAR)) @ DISP I3;IBOUND
250 S=I0+I2+I1+I3
260 T1=TIME-T1 @ DISP "AREA=";S;"in";T1
270 !
```

```
280 DEF FNF(Y)
290 X=SQR(-LOG(FNR(Y)))
300 X1=ABS(Y-X)^(1/3) @ IF Y<X THEN X1=-X1
310 X2=(Y+X)^(1/3)
320 FNF=X2-X1
330 END DEF
```

HP-71B results (real machine):

```
>RUN
RHS=1 for Ymax= 2.82740261413
LHS=RHS for X= 0 and Y= .831971149978
Ix IBOUND
5.42070674749E-4 5.42070677958E-13
1.23815989579 1.23815999491E-12
.023696731798 2.36967317996E-13
.814227669493 8.14227500886E-13
```
AREA= 2.07662636775 in 5558 s (about 1h32min)

HP-75 results (real machine):

```
RHS=1 for Ymax= 2.82740261413
LHS=RHS for X= 0 and Y= .831971149978
Ix IBOUND
5.4207067472E-4 9.99986161748E-14
1.23815989579 8.0765563033E-13
.023696731798 1.9746071305E-13
.814227669488 -1.89760803456E-13
```
AREA= 2.07662636775 in 33225 s (about 9h13min)

It is interesting to note how inefficient the HP-75 is for this problem, despite its faster speed. Especially, the last negative IBOUND value indicates that convergence was not detected with the maximum 32768 samples.
The HP-75 Math ROM integration code is based on the same Romberg algorithm than the 71B, but differs in the way the error is estimated (IBOUND) and the convergence is detected (see here for details), and it makes a huge difference for this problem.
However, it confirms the result obtained on the 71B. Good.

What about the accuracy?
I have no reference to compare with, I can only trust the HP integral algorithm and the IBOUND estimations for each zone.
I would expect 10 correct digits, maybe 11.

J-F

EMAIL PM WWW FIND QUOTE REPORT

12th January, 2023, 12:26 (This post was last modified: 12th January, 2023 12:40 by J-F Garnier.) **Post: #12**

J-F Garnier
Senior Member

Posts: 819
Joined: Dec 2013

RE: [VA] SRC #012d - Then and Now: Area

> **Fernando del Rey Wrote:** (12th January, 2023 02:00)
> If you make the grid accurate enough, say 500x500, you can see a small area at the upper left side of the rectangle where criteria (1) is met, and then a much larger area in the middle to lower half of the rectangle.

> **Werner Wrote:** (12th January, 2023 09:31)
> Incidentally (and accidentally) I also found a tiny region -4.0851.. <= y <= -4.0492.. where the original inequality will hold (eg. y=-4.05 and x=0.004).

Congratulations, Fernando and Werner, I missed it!
This was the "nice surprise" Valentin alluded to.
I vaguely suspected something like that, but I missed it although it would have been easy for me to see it by scanning the RHS from -sqr(m) to sqr(m):

```
>FOR Y=-5.5 to 5.5 STEP 0.5 @ FIX 1 @ DISP Y; @ STD @ DISP FNR(Y) @ NEXT Y
-5.5 1.49982769946
-5.0 1.21469841054
-4.5 1.04966788525
-4.0 1.00125597171 so close to 1 !!!
-3.5 1.11151914811
-3.0 1.45086446604
-2.5 2.02718534492
-2.0 2.61560067448
-1.5 2.78630642506
-1.0 2.35303256133
-0.5 1.62346023059
0.0 1
```

```
0.5 .627452895252
1.0 .464331687261
1.5 .443627546686
2.0 .535830072581
2.5 .757499135913
3.0 1.16768672176
3.5 1.8275622105
4.0 2.66354177734
4.5 3.3313121353
5.0 3.44028193128
5.5 3.03092654911
```

I will correct my calculation, but it's too late, I failed the test :-(

J-F

EMAIL PM WWW FIND QUOTE REPORT

12th January, 2023, 17:06 **Post: #13**

Albert Chan
Senior Member
Posts: 2,148
Joined: Jul 2018

RE: [VA] SRC #012d - Then and Now: Area

> **Werner Wrote:** (12th January, 2023 09:31)
> Incidentally (and accidentally) I also found a tiny region -4.0851.. <= y <= -4.0492.. where the original inequality will hold (eg. y=-4.05 and x=0.004).

Most likely, the region was fitted with inequality expression, not the other way around.
I would not worry about this, especially since OP suggested locate region by eye.

If we plot this, say, y = -5 .. 5, above "dot" area does not even show.

> **J-F Garnier Wrote:** (12th January, 2023 11:34)
> What about the accuracy?
> I have no reference to compare with, I can only trust the HP integral algorithm and the IBOUND estimations for each zone.

If integrand behave like a polynomial, IBOUND likely over-estimates true error.
IBOUND was based from relative error of current result vs previous, not true result vs current.

For polynomial-like integrand, we can get by with bigger eps (= less function evaluations)
That's the reason my code, with eps=10^-6, results in area of 10+ accuracy.

> **Albert Chan Wrote:** (12th January, 2023 00:17)
> Area = $\displaystyle\int_0^{1/2} g(z)\ dz = \int_0^1 g\left(\frac{x^3}{2}\right)\cdot\left(\frac{3}{2}x^2\ dx\right)$

I expected at z ≈ 0, g(z) behaves like c1 + c2*cbrt(z)
To simplify, say g(z) = z^(1/3), and we substitute with z=x^3

∫(z^(1/3) dz) = ∫((x^3)^(1/3) * (3*x^2 dx)) = 3*∫(x^3 dx)

RHS turned to plain polynomial, easy to integrate with Romberg.

EMAIL PM FIND QUOTE REPORT

12th January, 2023, 22:37 (This post was last modified: 12th January, 2023 22:38 by J-F Garnier.) **Post: #14**

J-F Garnier
Senior Member
Posts: 819
Joined: Dec 2013

RE: [VA] SRC #012d - Then and Now: Area

Update of my initial solution to include the "tiny" area at y= -4.05 .. -4.08 :

Program line changes or additions:

```
241 ! RHS<1 FOR Y=-4.05 !
242 F=FNROOT(-4,-4.05,FNR1(FVAR))
243 G=FNROOT(-4.05,-4.1,FNR1(FVAR))
244 I4=INTEGRAL(G,F,E*10000,FNF(IVAR)) @ DISP I4;IBOUND
250 S=I0+I2+I1+I3
260 T1=TIME-T1 @ DISP "AREA=";S+I4;"in";T1
265 DISP "(MAIN=";S;" TINY=";I4;")"
310 X2=ABS(Y+X)^(1/3) @ IF Y<-X THEN X2=-X2
```

New result (HP-71B):

```
RHS=1 for Ymax= 2.82740261413
LHS=RHS for X= 0 and Y= .831971149978
Ix IBOUND
5.42070674749E-4 5.42070677958E-13
1.23815989579 1.23815999491E-12
.023696731798 2.36967317996E-13
.814227669493 8.14227500886E-13
7.1976193049E-5 7.19937671652E-13
```

AREA= 2.07669834394 in 5594 s (about 1h33min)

```
(MAIN= 2.07662636775 TINY= 7.1976193049E-5 )
```

J-F

EMAIL PM WWW FIND QUOTE REPORT

13th January, 2023, 01:45 (This post was last modified: 13th January, 2023 20:42 by Albert Chan.) **Post: #15**

Albert Chan
Senior Member

Posts: 2,148
Joined: Jul 2018

RE: [VA] SRC #012d - Then and Now: Area

> **J-F Garnier Wrote:** (12th January, 2023 22:37)
> (MAIN= 2.07662636775 TINY= 7.1976193049E-5)

I still think the tiny area is not OP were asking ...
Anyway, we can confirm TINY area, assuming the dot look like an ellipse.
Or, we think of half ellipse area, with minor axis = FNF(center)

```
>f, g                                    ! (ymax, ymin), from JFG code
 -4.04921226445      -4.08514674762
>c = (f+g)/2 @ a = (f-g)/2 @ b = fnf(c)
>s = pi/2 * a * b                        ! half ellipse area
>s
 7.19760843094E-5
```

Almost 6 digits accuracy. Not bad for 1 function evaluation!

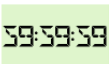
2 more function evaluations, we gain another digit accuracy.
Simpson's weight, for 5 points = [1, 4, 2, 4, 1] / 12

There is no need to evaluate odd points. FNF(y) matched ellipse numbers.
S1=0, we might as well go for Boole's rule : S2 + (S2-S1)/(16-1) = 16/15 * S2

```
>h = sqrt(0.75) * b
>s + ((fnf(c-a/2)-h) + (fnf(c+a/2)-h)) * 4/12 * 16/15 * (2*a)
7.19761691227E-5
```

EMAIL PM FIND QUOTE REPORT

13th January, 2023, 08:45 **Post: #16**



Werner
Senior Member

Posts: 777
Joined: Dec 2013

RE: [VA] SRC #012d - Then and Now: Area

> **Albert Chan Wrote:** (13th January, 2023 01:45)
> I still think the tiny area is not OP were asking ...

For once, Albert, I dare to disagree.
We were asked to calculate the area defined by the inequality.
If the tiny speck was to be omitted, the problem should have had an extra condition, like y>=0 or so, especially since 10-12 digits of accuracy were requested.
For the physical problem of the engine's efficiency, the tiny area does not matter, indeed.

Cheers, Werner

EMAIL PM FIND QUOTE REPORT

19th January, 2023, 19:08 **Post: #17**

Valentin Albillo
Senior Member

Posts: 970
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012d - Then and Now: Area

Hi, all,

Well, almost two weeks have elapsed since I posted my *OP*, which has already passed the *~1,400 views* mark (alas, once again when the difficulty increases the number of posts and/or views decreases,) and we've got a number of solutions and comments *(though no RPL ones, I wonder why ...)*, namely by ***PeterP***, ***Martin Hepperle***, ***Werner***, ***J-F Garnier***, ***Fernando del Rey***, and ***Albert Chan***. Thank you very much to all of you for your interest and valuable contributions.

Now, this is my detailed ***sleuthing*** *process* and resulting *original* ***solution***, plus *additional* ***comments****:*

> **Note**: though a number of graphics are featured in this solution, as suggested in the problem's statement, rest assured that **none of them are needed**, all intermediate and final results can be automatically computed without relying on graphics at all. The ultimate reason behind using them is that I feel they greatly help interested readers to better understand the steps to the solution, and they make for a more enjoyable presentation as compared to a wall of dry, terse math expressions alone.

My sleuthing process

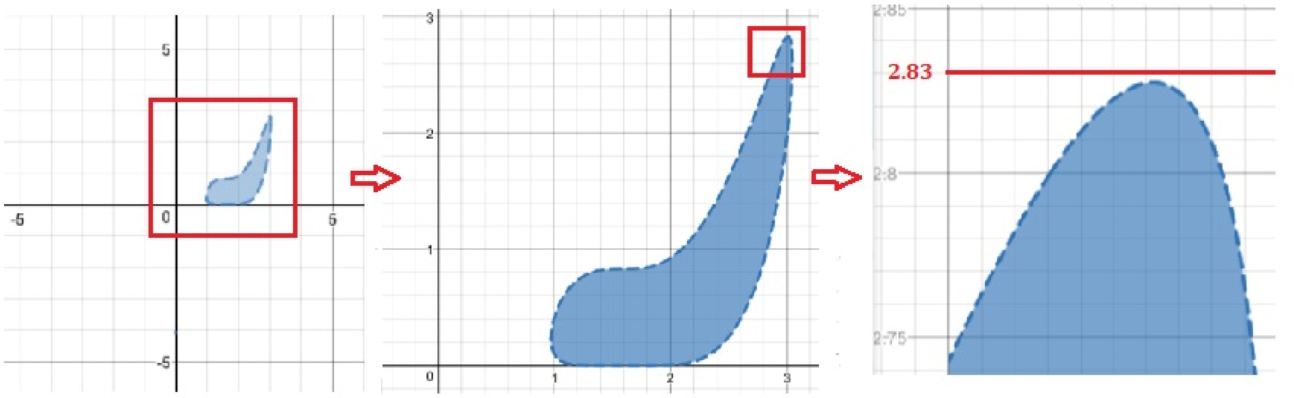
First of all, we rearrange the equation into this equivalent, simpler form:

$$e^{-\left((x-d)^3-y\right)^2} > \frac{y^2}{M} + e^{-\sin y}$$

Now, as suggested in my *OP*, we use a simple function plotter to get a visual idea of the region's shape and determine a rough *(d.dd)* approximate value for its ***ymin*** and ***ymax*** limits:

> **Note**: There's no need to get now the **xmin** and **xmax** limits because we'll compute them as a function of the **y** values, as will be seen below.

$$e^{-\left((x-d)^3-y\right)^2} > \frac{y^2}{M} + e^{-\sin y} \qquad d = 1.598 \qquad M = 30.07$$



and at first sight it seems that the region is a single island between ***ymin = 0*** and ***ymax ~ 2.83***, but there's the nagging question: *How can we be sure that there's only just <u>one</u> island ?* Perhaps it might be the case that the region consists of *several* disconnected subregions. How can we know for sure ?

One obvious way would be to obtain surefire limits for the valid ranges for ***x*** and ***y*** and carefully zoom and pan over the corresponding enclosing rectangle or else write scanning code to try and detect additional subregions, but this would be extremely time-consuming if the hypothetical subregions (*"islands"*) happen to be very small (or worse, *nonexistent*,) which might well be the case as we can't see them outright.

At any rate, even if our zooming and/or scanning *did* detect some small islands, we still wouldn't be sure that other *even smaller* ones didn't exist, so a more analytical approach is required and as the variable ***x*** appears just once, we isolate it, obtaining:

$$x_1 = 1.598 + \left(y + \sqrt{-\ln\left(\frac{y^2}{30.07} + e^{-\sin y}\right)}\right)^{\frac{1}{3}}$$

$$x_2 = 1.598 + \left(y - \sqrt{-\ln\left(\frac{y^2}{30.07} + e^{-\sin y}\right)}\right)^{\frac{1}{3}}$$

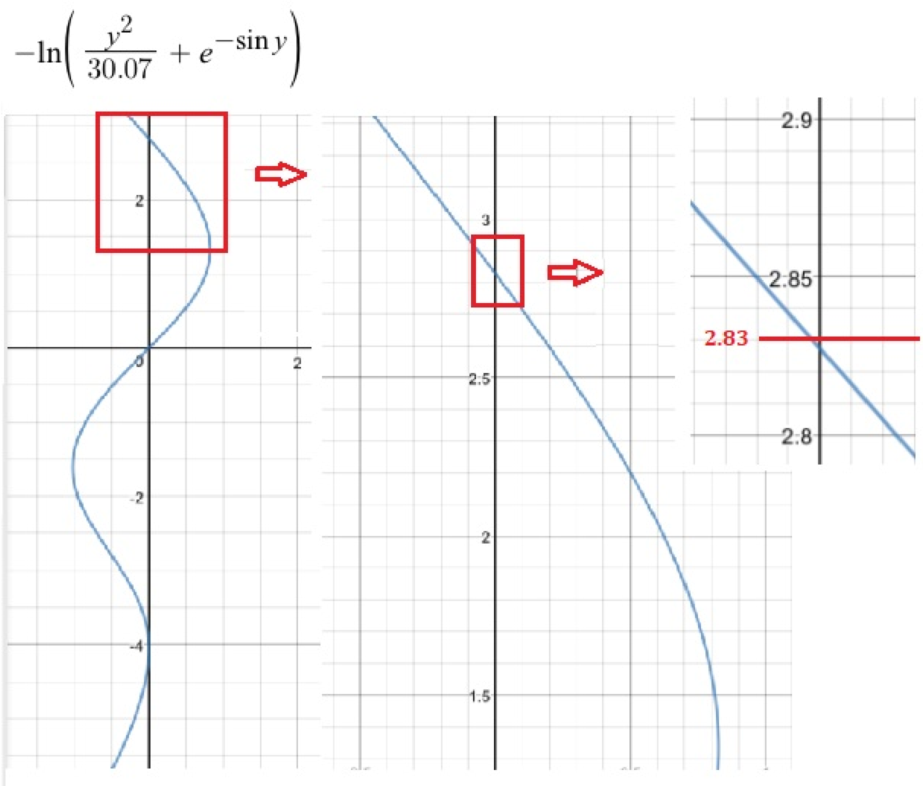
and it's clear that ***x*** will be real only if the expression under the square root is **≥ *0***. Two cases:

- If **> *0***, we'll have two distinct values, x_1 and x_2, and their *difference* will be the *width* of the island for a given value of ***y***.
- If **= *0***, then x_1 = x_2 and the *width* of the island will be ***0*** (i.e. the extreme *top* and *bottom* of the shape), so the

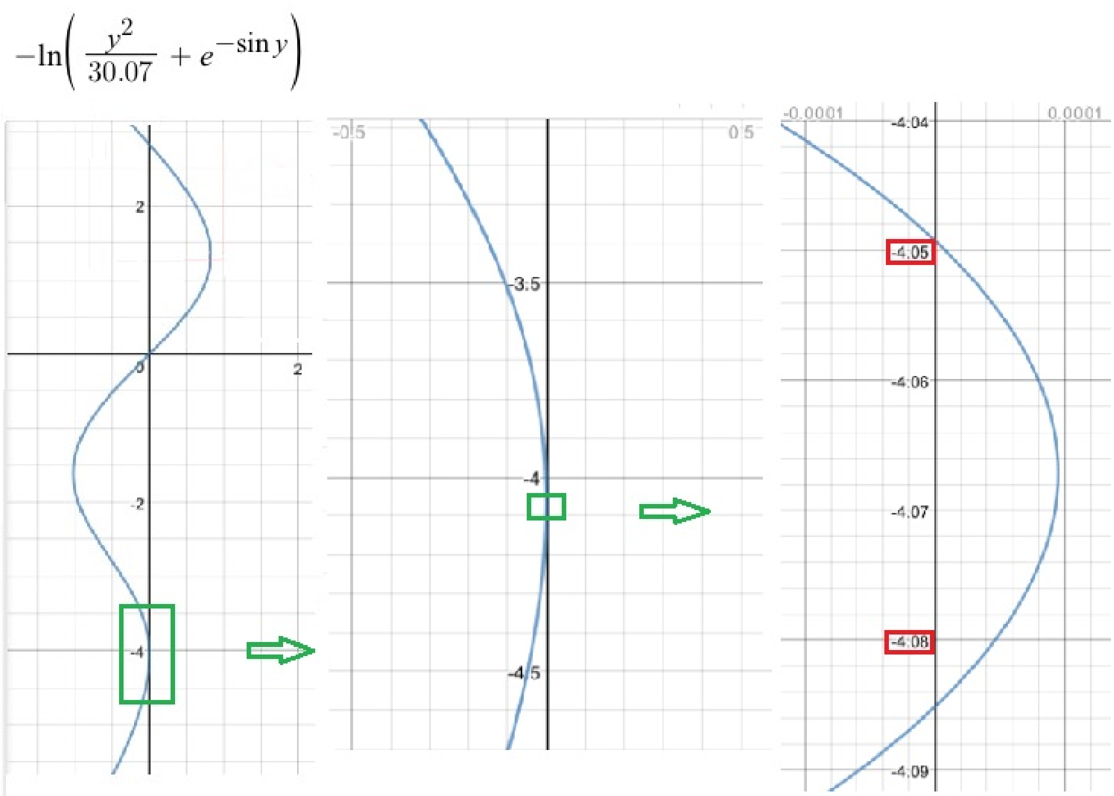
corresponding values of ***y*** are the limits ***ymin*** and ***ymax*** for each existing island and we'll compute them by simply finding all the roots of the equation:

$$-\ln\left(\frac{y^2}{30.07}+e^{-\sin y}\right)=0$$

Graphing the expression will give us the *number* of islands and a little zooming will provide initial approximations *(d.dd)* to the roots, i.e. the ***ymin***, ***ymax*** limits for *each* island:

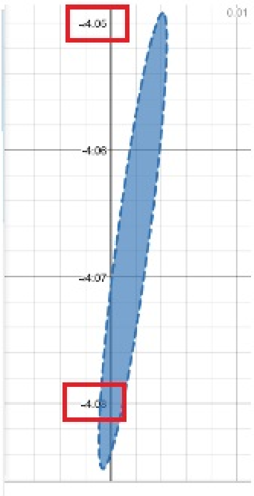


so we see that there's definitely one big island with ***ymin*** = ***0*** and ***ymax*** ~ ***2.83***, and possibly a much smaller island near ***y*** ~ ***-4***. Zooming a little more we have:



and indeed there's a *very small* island with ***ymin*** ~ ***-4.08*** and ***ymax*** ~ ***-4.05*** and no more islands are possible as the expression goes increasingly negative for ***y > ~ 2.83*** and ***y < ~ -4.08***. Now that we know the small island's location (the ***x*** can be obtained from the ***y*** using the formula above), we can visualize it:

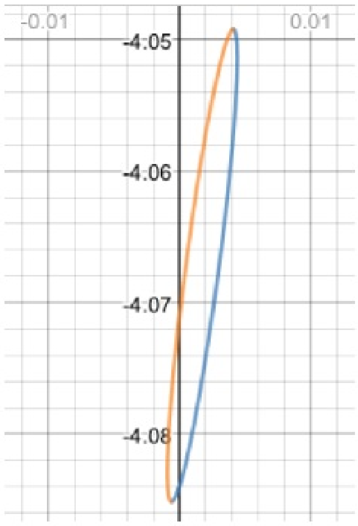
$$e^{-\left((x-1.598)^3-y\right)^2} > \frac{y^2}{30.07} + e^{-\sin y}$$



It's worth noting that the formulas above for $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$, if plotted independently, allows us to see how the perimeter of each island is composed of two separate *"branches"* which are joined seamlessly at their extremes:

$$x = 1.598 + \left(y + \sqrt{-\ln\left(\frac{y^2}{30.07} + e^{-\sin y}\right)}\right)^{\frac{1}{3}}$$

$$x = 1.598 + \left(y - \sqrt{-\ln\left(\frac{y^2}{30.07} + e^{-\sin y}\right)}\right)^{\frac{1}{3}}$$



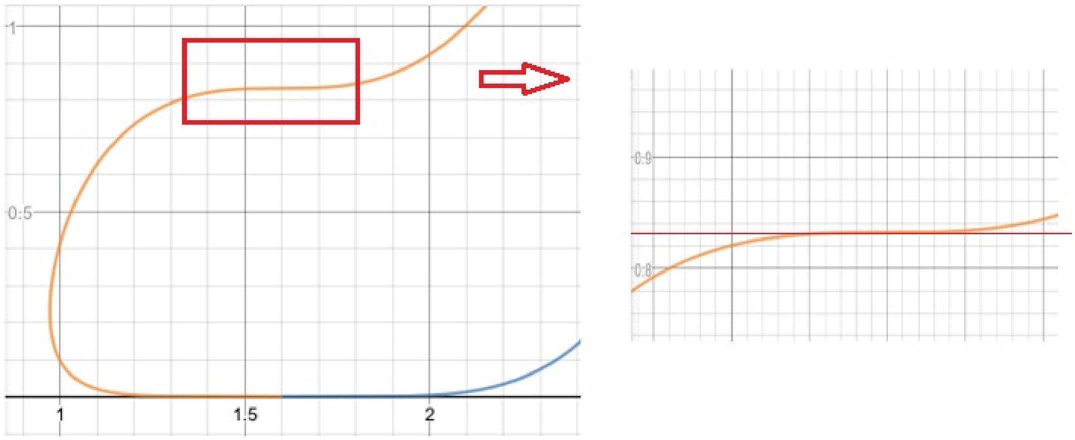
Now, knowing the ***y***-range [***ymin***, ***ymax***] for an island, its area is given by the expression:

$$Area = \int_{y\,\min}^{y\,\max} Width(y).dy = \int_{y\,\min}^{y\,\max} \left(x_1(y) - x_2(y)\right).dy$$

and the total area is the sum of the areas of both islands. The small island's integral is quite amenable to numerical integration save for the fact that *it might lose 2-3 digits to cancellation*, but as the resulting area is then added to the much larger one corresponding to the big island, those digits don't affect the sum to *12-digit* accuracy.

The integral for the big island, however, has ***two*** *singularities*, one at ***y*** = ***0*** (which doesn't cause any problems as this is the ***ymin*** extreme and the *Width* function being integrated is never evaluated there by **`INTEGRAL`**,) and another *inside* the range [***ymin***, ***ymax***], which causes **`INTEGRAL`** to be at least *400% slower* and even so it can only produce a value accurate to at most *7-8 digits*, not *12*.

The singularities can be recognized as the ***y*** values where the graph's tangent is *horizontal*, i.e., the graph goes parallel to the ***X*** axis. The ***y*** value of the inside singularity can be obtained by solving the equation **`SQR(-LN(Y*Y/30.07+EXP(-SIN(Y)))) = Y`** (which gives ***ysng*** ~ ***0.83***, as can be seen in the zoomed graphic below:



and we can then *split* the single problematic integral into two parts, the singularities being now at the extremes of the ranges of integration where they do no harm, like this:

$$Area = \int_{ymin}^{ysng} Width(y).dy + \int_{ysng}^{ymax} Width(y).dy$$

My original solution

My ***original solution*** is this little ***5-line***, *12-statement, 244-byte* **HP-71B** program (it can be made faster by using an additional line of code, but I like it this way):

```
1  DESTROY ALL @ DIM U @ H=1/3 @ D=FNROOT(1,1,SQR(FNG(FVAR))-FVAR)
2  DISP "Area:";FNI(0,D)+FNI(D,FNY(2.83))+FNI(FNY(-4.08),FNY(-4.05))

3  DEF FNG(Y)=-LN(Y*Y/30.07+EXP(-SIN(Y))) @ DEF FNY(Y)=FNROOT(Y,Y,FNG(FVAR))
4  DEF FNI(A,B)=INTEGRAL(A,B,1/10^10,FNW(IVAR)) @ DEF FNR(X)=SGN(X)*ABS(X)^H
5  DEF FNW(Y) @ U=SQR(FNG(Y)) @ FNW=FNR(Y+U)-FNR(Y-U)
```

Note: Notice that the parameter ***d*** = ***1.598*** does not appear in the solution's code, as the way it's featured in the inequality means that varying it only *translates* the whole region ***R*** along the ***X*** axis, which obviously *doesn't* alter ***R***'s area at all. Also, **FNROOT/FVAR** and **INTEGRAL/IVAR** are keywords from the **Math ROM**).

Line 1 performs some initialization and also computes ***ysng*** (`DIM U` is necessary to create variable ***U*** here and not inside `DEF FNW`, where we would fall prey to a well-known system bug).

Line 2 computes and displays the *total area* by evaluating and adding together the two integrals for the big island and the integral for the small island, while refining to full accuracy on the fly the *d.dd* approximations we got for the various ***ymin***, ***ymax***.

Line 3 defines `FNG` for the expression under the square root as a function of ***y***, as well as `FNY`, which refines to full accuracy the passed *d.dd* ***y*** approximation.

Line 4 defines `FNI`, which returns the integral of the *Width* function between specified limits to full accuracy, as well as `FNR`, which returns the real *cubic root* (with the correct sign) of the real argument passed to it.

Line 5 defines `FNW`, which returns the *Width* of an island at a given ***y*** value.

Let's run it:

```
>STD @ RADIANS @ RUN

            Area: 2.07669834394
```

The *20-digit* value is ***2.07669834394*** *76059651...*, so we've got a full ***12*** *correct digits*.

Note: all *20-digit* results given here were obtained using my original solution converted to *RPN* and run on ***Free42 Decimal***. See ***Additional comments*** below.

We can obtain additional results if ***immediately*** after running the *unmodified* program above we execute these statements directly from the command prompt:

```
>RES-IVALUE ►  2.07662636775     (big island's area   :   2.07662636775 45636287...)
>IVALUE     ►  7.19761930517E-5  (little island's area:   7.19761930423 36384588...E-5)
>D          ►   .831971149978    (big island's ysng   :   0.831971149979 07679799...)
>FNY(2.83)  ►  2.82740261413     (big island's ymax   :   2.82740261412 95600918...)
```

```
>FNY(-4.08) ► -4.08514674778      (little island's ymin:  -4.08514674763 83097474...)
>FNY(-4.05) ► -4.04921226429      (little island's ymax:  -4.04921226439 69504765...)
```

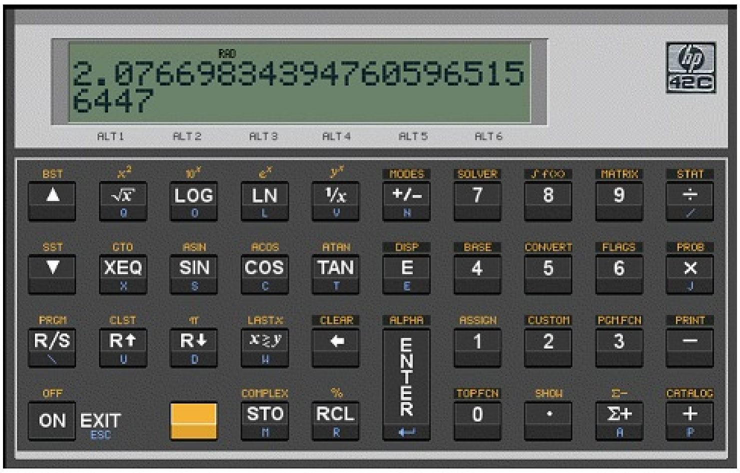
For timing, execute instead:

```
>SETTIME 0 @ CALL @ TIME
```

Additional comments

• It's quite trivial to convert my original *BASIC* solution for the **HP-71B** to **HP-42S**/***Free42***/**DM42**'s *RPN* code, matter of fact it can be done *on the fly* while typing the *RPN* instructions on the calculator/simulator, without ever needing to previously write down the *RPN* code on paper or a notepad file. I did it precisely that way, thanks to the utmost simplicity of classic *4-level RPN* code.

This allows for greatly enhanced accuracy if using ***Free42 Decimal***/***DM42***, as seen in this image, where the total ***Area*** has been computed and displayed to ***25*** *correct digits*. A bit of the resulting *RPN* code is shown as well but I'm not including the complete conversion to *RPN* here, that's left as an easy exercise for the reader. 🙂



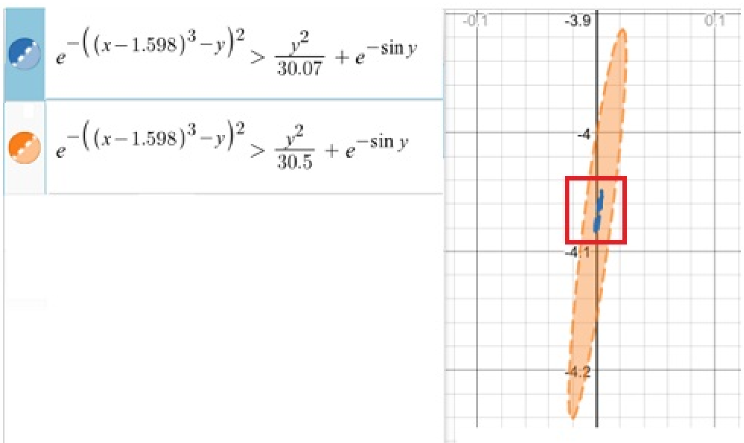

```
00 {57-Byte Prgm}
01▶LBL "FINT"
02 MVAR "Y"
03 RCL "Y"
04 X↑2
05 30.07
06 ÷
07 RCL "Y"
08 SIN
09 +/-
10 E↑X
11 +
12 LN
13 +/-
14 SQRT
15 STO "U"
16 RCL+ "Y"
17 XEQ "CURT"
18 RCL "U"
19 RCL- "Y"
20 XEQ "CURT"
21 +
22 END
```

• As some of you may have guessed (some did *not*), the problem's statement (*gas turbine's efficiency, ethane fuel, etc.)* makes absolutely *no sense* from a *"dimensional point of view"* and it's but a *made up* story I concocted in the spirit of those the great late ***Martin Gardner*** used to create as a pleasant wrapper around many of his puzzles and mathematical recreations, to make them all the more interesting (e.g., see *"The Magic Numbers of Dr [I.J.] Matrix"*).

For this particular problem, I *first* carefully selected the two parameters *(**d** = **1.598*** and **M** = ***30.07****)* to achieve the desired effect, and only afterwards did I look for *real-life* subject matters which would include those values. I quickly found that some *ethane* properties actually did, and the problem was thus stated in terms of ethane, further deciding on its use as fuel for some hypothetical gas turbine. Should I have found first that certain properties of *herrings* featured those numbers, the problem's description would've been built around herrings instead, possibly red ones.

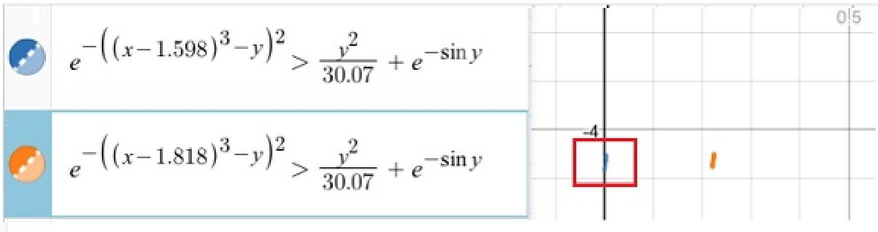
And what's the *"desired effect"* ? Well, to make the little island as inconspicuous as possible *without overdoing it*. To that effect, each parameter controls a particular aspect of the inconspicuousness, namely:

- **The parameter *M*** controls the *size* of the region ***R***, including the area of the main island and the number and areas of any existing additional islands, if any. Small variations of ***M*** result in a *very* noticeable effect, see for instance the *very* large increase in the size of the small island when changing from the problem's ***M*** = ***30.07*** to ***M*** = ***30.5***:



You may want to verify that by decreasing ***M*** the area of the small island tends to ***0***, and for ***M*** < ~***30.06480779965*** it *disappears* altogether so there's only the main island left. On the other hand, if we keep increasing ***M*** the small *"microscopic"* island grows to *"macroscopic"* size, becoming quite conspicuous, and as ***M*** tends to *infinity* further islands of various sizes do appear.

- **The parameter *d*** controls the *position* of the region ***R*** along the ***X*** axis, and the problem's value ***d* = *1.598*** places the small island just *under the **Y** axis*, which will tend to greatly obscure and hide it if the user plots the ***Y*** axis as usual, even if it would be otherwise quite visible, as seen in this image where ***d* = *1.818*** clearly shows the small island while the problem's ***d* = *1.598*** almost completely hides it under the ***Y*** axis, though its size is the same.



Well, that's all for now, I hope you enjoyed it. *Awesome-but-slightly-more-difficult* ***Problem 5*** will be posted next *February*.

> ***P.S.**: If you want to discuss other approaches, show any mathematical analysis (math "**lectures**"), use any tools (**Mathematica**, etc.), languages (**Python**, etc.), calculators (**HP Prime**, etc.) or computers (**SHARP** Pocket computers, laptops, etc.) please post your comments and/or code to this parallel thread kindly created by **EdS2**.*

V.

Forum software: MyBB, © 2002-2023 MyBB Group.