HP Forum Archive 19

[Return to Index | Top of Index]

Lambert W (HP-12C+)

*Message #1 Posted by **Gerson W. Barbosa** on 7 Nov 2009, 11:50 p.m.*

I have found a way to compute the the Lambert W function without using the solver, but only in the range (-1/e, 1]. However, the method allows the second branch to be computed all through the valid range, as shown below:

```
--------------------


Lambert W (2nd branch), -1/e =< x < 0


01 CHS
02 1/x
03 STO 0
04 ENTER
05 ENTER
06 ENTER
07 STO 1
08 LN
09 *
10 RCL 1
11 x<>y
12 -
13 x=0
14 GTO 19
15 CLx
16 RCL 0
17 LST x
18 GTO 07
19 LST x
20 RCL 0
21 /
22 CHS
23 GTO 00
                         12C+            actual


-0.367879441 R/S ->  -1.000000000     -1.000030530
-0.367       R/S ->  -1.070791879     -1.070791887
```

```
-0.3         R/S ->  -1.781337023     -1.781337023
-0.000000001 R/S -> -23.897019580    -23.897019585
```

(actual results obtained with Egan Ford's LAMBERTWB function in his special functions library - See message #6 in this thread).

$W_{-1}(-1/x) = -(...x*\ln(x*\ln(x*\ln(x*\ln(x))))...)/x$ (yet to be proved)

The principal branch, at least in the range (-1/e, 1], can be computed with the following program:

```
Lambert W (principal branch), -1/e > x >= 1


24 STO 0
25 ENTER
26 ENTER
27 ENTER
28 STO 1
29 e^x
30 /
31 RCL 1
32 x<>y
33 -
34 x=0
35 GTO 40
36 CLx
37 RCL 0
38 LST x
39 GTO 28
40 LSX x
41 GTO 00
```

```
                                12C+           actual


 0.999999999 GTO 24 R/S ->  0.567143290     0.567143290
 0.5         GTO 24 R/S ->  0.351733711     0.351733711
 0.000000000 GTO 24 R/S ->  0.000000000     0.000000000
-0.3         GTO 24 R/S -> -0.489402227    -0.489402227
-0.367       GTO 24 R/S -> -0.932399183    -0.932399185
```

```
(occasionally the program will loop for certain arguments, as for 1, due to rounding errors - in this case stop the program, the answer should be in register 1)

--------------------
```

Except for arguments whose convergences are very slow, these should work on the 12C as well. Also, by filling the stack with the desired argument and iterating two keystrokes until convergence is reached, this should work even on a non-programmable RPN calculator. The method may not be so practical but is somewhat interesting.

Gerson.

Edited: 8 Nov 2009, 7:12 a.m.

Re: Lambert W (HP-12C+)

*Message #2 Posted by **Namir** on 8 Nov 2009, 8:23 a.m.,*
in response to message #1 by Gerson W. Barbosa

Gerson,

Thanks for the program and for the reference. I have often come across the Lambert function, but did not stop to read about it. Your program and the Wikipedia reference gave me the info I need to implement the Lambert function (if and when I feel like doing that). The approximations are interesting.

Namir

Re: Lambert W (HP-12C+)

*Message #3 Posted by **Gerson W. Barbosa** on 8 Nov 2009, 12:21 p.m.,*
in response to message #2 by Namir

Hello Namir,

Thanks for your interest!

> Quote:
>
> The approximations are interesting.

Actually these are supposed to be exact solutions (except for accumulated rounding errors, mainly at the extremes of the convergence intervals). I've just found this paper which sheds some light on the subject (Iteration, pages 18 and 19).

The programs have been rewritten:

```
------------
```

```
01 CHS               20 STO 0
02 1/x               21 STO 1
03 STO 0             22 4
04 STO 1             23 EEX
05 LN                24 CHS
06 RCL 0             25 1
07 *                 26 0
08 RCL 1             27 STO 2
09 x<>y              28 RCL 0
10 -                 29 RCL 1
11 x=0               30 e^x
12 GTO 15            31 /
13 LST x             32 RCL 1
14 GTO 04            33 x<>y
15 LST x             34 STO 1
16 RCL 0             35 -
17 /                 36 ENTER
18 CHS               37 *
19 GTO 00            38 SQRT
                     39 RCL 2
                     40 x<=y
                     41 GTO 28
                     42 RCL 1
                     43 GTO 00


                                              actual
-0.3678794411         R/S ->  -1.0000000000  (-1.0000197080)
 1.0000000000 GTO 20  R/S ->   0.5671432905  ( 0.5671432904)


-------------
```

Regards,

Gerson.

Edited: 8 Nov 2009, 1:54 p.m.

Re: Lambert W (HP-12C+)

*Message #4 Posted by **Crawl** on 10 Nov 2009, 12:22 p.m.,*
in response to message #3 by Gerson W. Barbosa

I was playing with that a little.

It seems like

y(n+1)=x/exp(y(n))

converges to W(x) up to about 2.7 (probably supposed to be e). If you use

y(n+1)=(x/exp(y(n))+y(n))/2

it converges much faster in the previous range (because the oscillations are suppressed)(though it doesn't necessarily speed the convergence for negative argument), and also extends the range substantially, to about 60 (probably supposed to be 3*exp(3)).

y(n+1)=(x/exp(y(n))+2*y(n))/3

converges all the way to 5*exp(5)~742.

It's probably possible to use this or a similar idea to get an arbitrarily wide range of convergence.

In fact

y(n+1) = (x/exp(y(n)) + (y(n))^2)/(1+y(n))

seems to converge over the entire range. You just need a good seed for y(0). Maybe x for x < 1, log(x) for x>1. Or just the natural log of x+1 for all x.

Edited: 10 Nov 2009, 12:26 p.m.

Re: Lambert W (HP-12C+)

*Message #5 Posted by **Gerson W. Barbosa** on 10 Nov 2009, 4:14 p.m.,*
in response to message #4 by Crawl

> Quote:
>
> In fact
>
> y(n+1) = (x/exp(y(n)) + (y(n))^2)/(1+y(n))
>
> seems to converge over the entire range. You just need a good seed for y(0). Maybe x for x < 1, log(x) for x>1. Or just the natural log of x+1 for all x.

That's great! A good estimate for x>1 might be ln-ln(ln(x)). Perhaps your first expression and another initial estimate should be used for x<1.01 so that only a few iterations are needed in that range.

Some examples (Excel 2007)

$$y_{n+1} = (x/\exp(y_n) + y_n^2)/(1+y_n)$$

```
x =     1.001

n       y_n

0       6.90925457110691
1       6.03581510741710
2       5.17828530065102
3       4.34105591640598
4       3.53072548897921
5       2.75791050589333
6       2.04091021461044
7       1.41252273893777
8       0.92807090671205
9       0.65196072145989
10      0.57301422876145
11      0.56752983893071
12      0.56750507994374
13      0.56750507944168
14      0.56750507944168

x =     2

0       1.05966010114161
1       0.88171184629193
2       0.85324579958077
3       0.85260581752487
4       0.85260550201380
5       0.85260550201373
6       0.85260550201373

x =     123456789012345

0       28.96730703740500
1       29.08342041664470
2       29.07698748392480
3       29.07696600979440
4       29.07696600955620
5       29.07696600955620

x =     2.E+100

0       225.50944807065200
1       225.53347447862600
2       225.53318919246000
```

```
3       225.53318915157900
4       225.53318915157900


------------------
Formulas


x =     2E+100


n       yn


0       =LN(D5)-LN(LN(D5))
1       =($D$5/EXP(D9)+D9^2)/(1+D9)
2       =($D$5/EXP(D10)+D10^2)/(1+D10)
------------------
```

Regards,

Gerson.

Edited: 10 Nov 2009, 4:38 p.m.

Re: Lambert W (HP-12C+)

*Message #6 Posted by **Crawl** on 10 Nov 2009, 9:00 p.m.,*
in response to message #5 by Gerson W. Barbosa

I notice y(n+1)=log(x/y(n)) converges to the second branch quickly if x is very small, but converges very, very slowly if y~-1. I thought a bit about speeding up the convergence, but am not quite there.

Incidentally, look at the end of this paper.

> Quote:
>
> Finally, in an eloge near the beginning of the two volumes of Lambert's mathematical papers (published in Zurich in 1946 and 1948) we find that Lambert had dreams of building a machine to do symbolic calculation (while Pascal's machine merely did arithmetic). Thus his wishes anticipated those of Lady Ada Lovelace by roughly one century, and the actuality of symbolic computation systems by roughly two.

I'd think Lambert would have to be impressed by the HP50g.

Re: Lambert W (HP-12C+)

*Message #7 Posted by **Crawl** on 10 Nov 2009, 11:11 p.m.,*
in response to message #6 by Crawl

Okay, I think I have a good improvement to

y(n+1)=log(x/y)

Try

y(n+1)=(log(x/y)+1)*y/(1+y)

And maybe y(0)=log(-x)

Small x test

x=-.000001

Converges to -16.6265089014 in only three iterations. Takes 10 iterations for the other method.

Medium x test

x=-0.18

Converges to -2.7127679118 in only four iterations. Takes about 27 iterations to converge with other method.

Large x test

x=-0.36787944

Converges to -1.00007980968 in 17 iterations. Using qbasic and 100,000 iterations with the other method, has only converged to -1.00007975...

Edited: 10 Nov 2009, 11:17 p.m.

Re: Lambert W (HP-12C+)

*Message #8 Posted by **Crawl** on 10 Nov 2009, 11:53 p.m.,*
in response to message #7 by Crawl

D'oh! That's just Newton's method to solve y-log(x/y)=0. Oh, well.

Re: Lambert W (HP-12C/C+)

*Message #9 Posted by **Gerson W. Barbosa** on 11 Nov 2009, 5:41 p.m.,*
in response to message #4 by Crawl

Quote:

---

y(n+1) = (x/exp(y(n)) + (y(n))^2)/(1+y(n))

The HP-12C program below uses your formula and y(0)=ln(x+2)-ln(ln(x+2)) for all x. It's very fast on the 12C+. Running times are acceptable on the original 12C.

```
Lambert W function (principal branch)


01 STO 0        16 1            31 +
02 2            17 0            32 x<>y
03 +            18 x<>y         33 /
04 LN           19 y^X          34 RCL 1
05 ENTER        20 STO 2        35 x<>y
06 LN           21 1            36 STO 1
07 -            22 RCL 1        37 -
08 STO 1        23 +            38 ENTER
09 LN           24 LST x        39 *
10 1            25 ENTER        40 SQRT
11 0            26 *            41 RCL 2
12 LN           27 RCL 0        42 x<=y
13 /            28 LST x        43 GTO 21
14 9            29 e^x          44 RCL 1
15 -            30 /            45 GTO 00
```

		12C+	actual	time (12C)
9.999999e99	R/S ->	224.8431063	224.8431064	09.5 s
99.999999990	R/S ->	3.385630139	3.385630140	13.9 s
2.000000000	R/S ->	0.852605502	0.852605502	13.7 s
1.000000000	R/S ->	0.567143290	0.567143290	16.4 s
0.5	R/S ->	0.351733711	0.351733711	15.9 s
0.000000000	R/S ->	5.765480-25	0.000000000	17.2 s
-0.3	R/S ->	-0.489402227	-0.489402227	22.4 s
-0.367	R/S ->	-0.932399184	-0.932399185	30.9 s
-0.000100000	R/S ->	-0.000100010	-0.000100010	17.8 s

Edited to include running times on the original HP-12C

Edited: 12 Nov 2009, 7:14 a.m.

Re: Lambert W (HP-12C/C+)

*Message #10 Posted by **Crawl** on 17 Nov 2009, 12:15 p.m.,*
in response to message #9 by Gerson W. Barbosa

I think this forum is the first place I've ever heard of the Lambert W. I've been reading up on it a bit. There's some interesting stuff about it.

Re: Lambert W (HP-12C/C+)

*Message #11 Posted by **Gerson W. Barbosa** on 17 Nov 2009, 2:29 p.m.,*
in response to message #10 by Crawl

> Quote:
>
> I think this forum is the first place I've ever heard of the Lambert W.

That's where I first heard of it too. I think it was Valentin Albillo who first mentioned it here.

I like its being capable of providing analytical solutions to some problems that would otherwise require a numerical solver, like the ridget hurler example in the HP-15C manual:

$h = 5000(1 - e^{-t}/20) - 200t$

The problem consists in solving the equation for t when $h = 0$. If Lambert W were used, the solution would be simply

$t = 20W(-5/4\ e^{-5/4}) + 25$

Or $t = 9.284255090$, when evaluated with the HP-12C program above.

Edit to correct a typo.

Edited: 21 Nov 2009, 9:25 a.m.

[Return to Index | Top of Index]

Go back to the main exhibit hall