HP Forum Archive 16

[Return to Index | Top of Index]

HP-15C Mini-Challenge: Impossibly Short !?

*Message #1 Posted by **Valentin Albillo** on 24 Aug 2006, 1:54 p.m.*

Hi all !

Once my (always too-short) summer vacations are over, I'm returning to "active status" on this forum, might as well say I certainly missed it (the converse probably isn't true ... ;-)

During my vacation period I've prepared a number of Mini-Challenges (tm) for various HP models as well as a full-fledged S&SMC, which I'll be posting for your alleged enjoyment these next weeks. For the time being, and just for you to flex your HP muscles in advance of what's to come, I'll leave you with this ultra-small, yet enticing

---

HP-15C Mini-Challenge

"Write an HP-15C program to find a root of $\mathbf{x^3 + 3x - A = 0}$ for *any* given numeric value of A"

Your routine must accept the value of **A** in the display (which can be any numerical value legal for the HP-15C) and return to the display the computed root **x**, like this:

```
(value of A) [GSB A] -> (value of X)
```

You should strive for absolute minimum program size, then speed.

Of course this mini-challenge would be dead easy for the HP-15C were it not for that accursed minimum-size consideration. Within a few days I'll post my original solution which fully meets the above requirements (and reproduces the below examples) and is just *9 steps* long.

Some examples, for you to test your very own solution:

```
A = 1          :    1,               GSB A  ->  0.3221853546
A = -10        :    10, CHS,         GSB A  ->  -1.698885490
A = 1.23456E12 :    1.23456, EEX, 12, GSB A  ->  10727.63686
```

That's all, you have at least the whole weekend to try and deliver, which, as I'm asking for only 9 steps or less worth of RPN code, it should be pretty affordable for nearly everyone here. Heck, it amounts to just *one* step per 10 hours ! :-)

Best regards from V.

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #2 Posted by **Karl Schneider** on 24 Aug 2006, 3:17 p.m.,*
in response to message #1 by Valentin Albillo

Valentin --

This solution isn't as compact as yours, but it offers some mathematical insight and a quick direct solution.

Mathematical references such as the Chemical Rubber Company (CRC) handbook illustrate two forms of a solution to the general cubic equation. The non-trigonometric form:

$x^3 + ax + b = 0$

is the *resolvent* form of the general cubic equation

$y^3 + py^2 + qy + r = 0$

which is obtained by substituting (x - p/3) in place of y.

(The trigonometric-substitution form of the solution appears to be the basis of the answer to this challenge.)

For a = 3 and b = A, the real-valued solution from the linear term will be given by

$[-A/2 + \text{sqrt}(A^2/4 + 1)]^{1/3} - [A/2 + \text{sqrt}(A^2/4 + 1)]^{1/3}$

And here's a 23-line program for that:

```
LBL A
CHS
2
/
STO 0
x²
1
+
sqrt(x)
STO 1
RCL 0
-
3
1/x
yˣ
RCL 0
RCL 1
+
3
1/x
yˣ
-
RTN
```

Paul Dale's straightforward (but much slower) routine utilizing SOLVE should work no matter what initial guesses are used, due to the first-order term.

Regards,

-- KS

Edited: 24 Aug 2006, 11:43 p.m.

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #3 Posted by **Crawl** on 24 Aug 2006, 8:11 p.m.,*
in response to message #1 by Valentin Albillo

Too easy!

```
2       001-            2
/       002-            10
sinh^-1 003-    43,22,23
3       004-            03
/       005-            10
sinh    006-    42,22,23
2       007-            2
*       008-            20
RTN     009-    43      32
```

I don't see it getting much shorter.

The idea behind this solution is the trigonometric identity

cos(3*t)= 4*cos(t)^3 - 3*cos(t)

With that said, any details could be figured out, but I'll go through them.

It turns out that the coefficents as given turn the cosines into hyperbolic sines. To save a little trouble, we'll use that bit of hindsight now and instead use the identity

sinh(t)^3 = sinh(3*t)/4 - 3*sinh(t)/4

or

4* sinh(t)^3 + 3 * sinh(t) - sinh(3*t) = 0

Set x = y * sinh(t) and we get

4 * y^3 * x^3 + 3 * y * x - sinh(3*t) = 0

Divide by 4 * y^3...

x^3 + (3/(4y^2)) * x - sinh(3*t) / (4y^3) = 0

Solve for y to make the x term have a coefficent of 3. (Note that this step is where you could run into problems with the cosine form. y would need to be imaginary in that case)

That would make y = 1/2 and

x^3 + 3 * x - 2*sinh(3*t) = 0

Now compare this to the original equation. It's identical in form! So, the constant terms should be equal, too, meaning

2*sinh(3*t) = A.

Solve for A in terms of t, and go back to the original substitution of x = y * sinh(t) to get the final answer

Edited: 24 Aug 2006, 8:34 p.m.

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #4 Posted by **Paul Dale** on 24 Aug 2006, 8:38 p.m.,*
in response to message #3 by Crawl

Quote:

```
2       001-            2
/       002-            10
sinh^-1 003-    43,22,23
3       004-            03
/       005-            10
sinh    006-    42,22,23
2       007-            2
*       008-            20
RTN     009-    43      32
```

One step better is to leave off the final RTN. So the target is now better than 8.

- Pauli

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #5 Posted by **Crawl** on 24 Aug 2006, 8:44 p.m.,*
in response to message #3 by Crawl

Opps, I made a mistake in my explanation (though the program is good).

The substition was x * y = sinh(t), not x = y * sinh(t).

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #6 Posted by **Artur-Brazil** on 24 Aug 2006, 10:44 p.m.,*
in response to message #3 by Crawl

I have to admit: many guys here should be working at NASA! They are genious! Artur from Brazil!

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #7 Posted by **Namir** on 25 Aug 2006, 1:01 a.m.,*
in response to message #3 by Crawl

I had a strong hunch that the solution for the Challenge uses some special transformation to solve for the roots of the particular form of the cubic equation.

Congrats on offering a clever solution!!!

Namir

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #8 Posted by **Paul Dale** on 24 Aug 2006, 8:34 p.m.,*
in response to message #1 by Valentin Albillo

Eleven steps not nine but it is a start:

```
01 LBL A
02 STO 0
03 SOLVE 0
04 RTN


05 LBL 0
```

```
06 ENTER
07 x^2
08 3
09 +
10 *
11 RCL- 0
```

To try to get better than this, I'm going to resort to some algebra and the cubic equation solution.

Oops, looks like I've been beaten to it.

- Pauli

Edited: 24 Aug 2006, 9:50 p.m.

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #9 Posted by **Karl Schneider** on 25 Aug 2006, 1:03 a.m.,*
in response to message #8 by Paul Dale

> Quote:
>
> ```
> 01 LBL A
> 02 STO 0
> 03 SOLVE 0
> 04 RTN
> 05 LBL 0
> 06 ENTER
> 07 x^2
> 08 3
> 09 +
> 10 *
> 11 RCL- 0
> ```

This is an answer that exploits several capabilities of the HP-15C -- namely, SOLVE and arithmetic with RCL -- that are not both shared with any models preceding the Pioneer-series HP-32S and HP-42S.

Note: the ENTER is unnecessary, because the SOLVE function loads the stack fed to the function being solved with the x-register value.

-- KS

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #10 Posted by **Paul Dale** on 25 Aug 2006, 1:23 a.m.,*
in response to message #9 by Karl Schneider

> Quote:
>
> Note: the ENTER is unnecessary, because the SOLVE function loads the stack fed to the function being solved with the x-register value.

Damn, I should have remembered that!

So this approach is down to ten steps with no algebra and little thought required :-)

- Pauli

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #11 Posted by **Mike T.** on 26 Aug 2006, 7:03 p.m.,*
in response to message #9 by Karl Schneider

Just out of curiosity what does RCL-0 do..?

Thanks

Mike T.

STO/RCL arithmetic (Re: 15C short challenge)

*Message #12 Posted by **Karl Schneider** on 26 Aug 2006, 8:03 p.m.,*
in response to message #11 by Mike T.

> Quote:
>
> Just out of curiosity what does RCL-0 do?

Subtracts the value recalled from storage register 0 from the value retrieved from the x-register, and places the result in the x-register. It is equivalent to

```
RCL 0
-
```

except that takes one fewer instruction and doesn't push the stack.

Storage-register arithmetic using STO was available on the HP-41C/CV, HP-11C, HP-34C and perhaps other models released prior to the HP-15C (in 1982), but the latter introduced storage-register arithmetic using RCL.

Interestingly, no other RPN- or RPL-based calculator from HP would have RCL arithmetic until the HP-32S and HP-42S in 1988. The HP-41CX (1983) didn't have it, and neither did the HP-28C (1987) or the HP-28S (1987).

Storage-register arithmetic is a more-valuable feature. For example, "STO- 0" is equivalent to

```
RCL 0
x<>y
-
STO 0
CLx
LASTx
```

except that the top level of the stack might be different.

(NOTE: These summaries ignore resultant LASTx contents, as addressed by Thomas Okken's post below...)

-- KS

Edited: 26 Aug 2006, 8:53 p.m. after one or more responses were posted

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #13 Posted by **Thomas Okken** on 26 Aug 2006, 8:19 p.m.,*
in response to message #12 by Karl Schneider

Regarding *Just out of curiosity what does RCL-0 do?*

> Quote:
>
> Subtracts the value recalled from storage register 0 from the value retrieved from the x-register, and places the result in the x-register. It is equivalent to
>
> RCL 0 -
>
> except that takes one fewer instruction and doesn't push the stack.

Also, **"RCL- 00"** puts the previous value of X in LASTX, while **"RCL 00 -"** obviously causes the contents of register 00 to get copied into LASTX.
At least, that's how it is on the HP-42S! I haven't tried this on my 15C yet.
On the 42S, I generally use RCL arithmetic whenever I can because it makes programs more compact, and faster, and it behaves more like you typically want re: what ends up in LASTX, and not losing the contents of T.

- Thomas

Re: STO/RCL arithmetic (Re: 15C short challenge)

*Message #14 Posted by **Valentin Albillo** on 28 Aug 2006, 8:20 a.m.,*
in response to message #12 by Karl Schneider

Hi, Karl:

Karl posted:

"Storage-register arithmetic is a more-valuable feature."

> I beg to differ. In my experience, I find myself using RCL arithmetic much more frequently than STO arithmetic.
>
> It's the case that you'll frequently store values once and recall them many times, specially within loops. Every time you can use RCL arithmetic within a loop, you're saving running time, because in the Voyager series decoding and executing instructions as program steps takes appreciable time regardless of the particular operation, even an NOP would. Thus RCL+ 01 is faster than RCL 01, +, and if this is executed within a loop, the effect is noticeable.
>
> Also, it's much more elegant and efficient to use the 4-level stack for arithmetic operations, instead of doing it via STO arithmetic, and this is greatly helped by RCL arithmetic, as stack register T isn't lost. Many programs using STO arithmetic can be profitably rewritten to use RCL arithmetic instead.
>
> Program listings are also shorter as well (though RAM consumption is the same), which is nice. I, for once, always missed RCL arithmetic in the HP-41C/CV/CX, and if being forced to choose between having just STO arithmetic or just RCL arithmetic, I'd certainly go for the latter. Fortunately, there's no need for us proud HP-15C users.

Best regards from V.

Re: STO/RCL arithmetic (Re: 15C short challenge)

*Message #15 Posted by **Klaus** on 28 Aug 2006, 8:33 a.m.,*
in response to message #12 by Karl Schneider

Hi Karl,

> Quote:

> Interestingly, no other RPN- or RPL-based calculator from HP would have RCL arithmetic until the HP-32S and HP-42S in 1988. The HP-41CX (1983) didn't have it, and neither did the HP-28C (1987) or the HP-28S (1987).

Actually, the HP-45 already had RCL arithmetic. I don't know why the HP-55 didn't have it anymore.

Greeting, Klaus

Re: STO/RCL arithmetic (Re: 15C short challenge)

*Message #16 Posted by **Valentin Albillo** on 28 Aug 2006, 8:41 a.m.,*
in response to message #15 by Klaus

Hi, Klaus:

Klaus posted:

> " I don't know why the HP-55 didn't have it anymore."
>
> > As stated in my reply to Karl, the usual culprit for RCL arithmetic not being implemented is that, for older programmable models, such as the HP-55, it needs a *lot* of additional unique keycodes (RCL+0, RCL+1, ..., RCL+ .9, RCL* 0, RCL*1, ..., RCL* .9, say), and usually there are just 128 or 256 unique keycodes available, so RCL arithmetic has to go.
> >
> > There's also the amount of ROM space required, which the HP-55 probably used up completely, what with its (rudimentary) programmability and timer functions. But RCL arithmetic probably requires very little ROM space to implement, so this might not be the decisive point. 128/256 keycodes maximum is.

Best regards from V.

Re: STO/RCL arithmetic (Re: 15C short challenge)

*Message #17 Posted by **Thomas Okken** on 28 Aug 2006, 2:12 p.m.,*
in response to message #16 by Valentin Albillo

> Quote:
>
> As stated in my reply to Karl, the usual culprit for RCL arithmetic not being implemented is that, for older programmable models, such as the HP-55, it needs a *lot* of additional unique keycodes (RCL+0, RCL+1, ..., RCL+ .9, RCL* 0, RCL*1, ..., RCL* .9, say), and usually there are just 128 or 256 unique keycodes available, so RCL arithmetic has to go.

Actually, the HP-55 has unmerged keycodes, so RCL arithmetic could have been implemented without running out of opcodes. Maybe they just ran out of ROM space -- the 55 has a pretty impressive instruction set, and those unit conversions must also have been pretty expensive in terms of ROM usage.

- Thomas

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #18 Posted by **Artur-Brazil** on 24 Aug 2006, 10:47 p.m.,*
in response to message #1 by Valentin Albillo

Vallentin

Have you ever thought about publishing a book with your problems and all solutions provided to them, begining for the most simples and direct ones to the most worked out and applying the more mathematical skills? Best regards and thank you very much! Artur

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #19 Posted by **Valentin Albillo** on 28 Aug 2006, 8:33 a.m.,*
in response to message #18 by Artur-Brazil

Hi, Artur:

Artur posted:

"Have you ever thought about publishing a book with your problems and all solutions provided to them, begining for the most simples and direct ones to the most worked out and applying the more mathematical skills?"

> First of all, thanks for your interest in my mini-challenges. I've never thought about publishing them in any form, because I fully know I don't have the time to do it, though I would certainly welcome if someone did. Also, I don't keep copies myself, so I would need to look in the archives to find them all, which would increase the time and effort required even further.
>
> Some time ago, a frequent contributor to this forum gathered together a number of my mini-challenges and compiled them all into a single PDF file, including all posted answers to each of them, in order. It might well be the case that it is still available on-line somewhere, on in the archives.

Best regards from V.

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #20 Posted by **Bill (Smithville, NJ)** on 28 Aug 2006, 9:45 a.m.,*
in response to message #19 by Valentin Albillo

Hi Valentin & Artur,

The file is still available. Just checked and it downloaded fine.

It goes through the first 11 SSMC's. Haven't had time to update it with the later Challenges. And I haven't got around to putting all the mini-challenges into a PDF yet. Someday, I hope.... :)

Link to original message:

SSMC in PDF Form

Enjoy.

Bill

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #21 Posted by **Valentin Albillo** on 29 Aug 2006, 7:53 a.m.,*
in response to message #20 by Bill (Smithville, NJ)

Hi, Bill:

Bill posted:

"The file is still available. Just checked and it downloaded fine."

> Thank you very much. I couldn't find the original post and though I remembered that "Bill" was the kind person who compiled them all into a PDF, I wasn't absolutely sure about the exact "Bill" among the usual contributors named so.

"It goes through the first 11 SSMC's. Haven't had time to update it with the later Challenges. And I haven't got around to putting all the mini-challenges into a PDF yet. Someday, I hope.... :)"

That would be great, indeed, your present and potentially future work is **much** appreciated.

Best regards from V.

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #22 Posted by **Giancarlo (Italy)** on 25 Aug 2006, 3:40 a.m.,*
in response to message #1 by Valentin Albillo

Hi Valentin!
Just a few words to welcome you back in the Forum!

> Quote:
>
> might as well say I certainly missed it (the converse probably isn't true ... ;-

...well, not so sure about that!
Warmest regards.

Thank you very much, Giancarlo ! :-) [NT]

*Message #23 Posted by **Valentin Albillo** on 29 Aug 2006, 4:06 a.m.,*
in response to message #22 by Giancarlo (Italy)

Best regards from V.

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #24 Posted by **Kiyoshi Akima** on 25 Aug 2006, 2:33 p.m.,*
in response to message #1 by Valentin Albillo

Well, a bunch of people beat me to it, but here's what I had.

I first tried brute force. In RPL on the 48GX:

```
1 0 3 4 ROLL NEG 4 ->ARRY PROOT
```

for nine steps to get all three roots, real or complex.

Brute force in RPN on the 15C:

```
01 LBL B
02 *
03 3
04 +
05 *
06 RCL- 0
07 RTN
08 LBL A
09 STO 0
10 SOLVE B
```

Almost identical to Paul's program, but not quite down to nine steps. It will need two RTNs at the end if its not the last program in memory. And it doesn't work for complex values of A, which the RPL program handles with ease. (I assume *"any* given numeric value of A." to include the complex.) It can also be ridiculously slow for large values of A.

Then, with the trigonometric identity. In RPL:

```
<< 2 / ASINH 3 / SINH DUP + >>
```

which in RPN is identical to Crawl's with [DUP +] replacing [2 *]. Complex values of A handled with ease.

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #25 Posted by **Crawl** on 25 Aug 2006, 4:37 p.m.,*
in response to message #24 by Kiyoshi Akima

By the way, my original program should work with complex values of A just fine on the 15C... provided, of course, that the 15c is in complex mode.

But why stop there? Using RPL, this program will solve the equation for A (and hence X) as a square MATRIX! In fact, the matrix can even have complex entries. It also encompasses the original solution; just have A be a 1x1 matrix whose entry was the original A.

<< 2. / 'ASINH(X)' DIAGMAP ->NUM 3. / 'SINH(X)' DIAGMAP ->NUM 2. * >>

The 15C can of course also handle matrices, but I'm not sure how to write a program to solve that equation in matrix form on the 15C (frankly, I don't really know much about the 15C's matrix mode, or its capabilities).

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #26 Posted by **Chris Dean** on 26 Aug 2006, 6:23 p.m.,*
in response to message #25 by Crawl

```
Here is a 10 liner for a HP-12C. It however runs really really slowly but does eventually reach the required solution. It uses the formula


     x =  A
         ---
         x^2+3


that is direclty obtainable from the original equation.


01 STO 0
02 2
03 y^x
04 3
05 +
06 1/x
07 RCL 0
08 X
09 g PSE
10 g GTO 02


Enter A onto the stack and the R/S. The numbers shown will gradually converge to the solution. For a second number type g GTO 00 before entering another number onto the stack.


A better solution that coverges quicker is obtained by using the Newton-Raphson iteration method to obtain the formula


   x = 2x^3+A
```

```
       ------
       3x^2+3


A miserable 18 line program on the HP-12C produces the required results with reasonable speed.


01 STO 0
02 STO 1
03 3
04 y^x
05 2
06 x
07 RCL 0
08 +
09 RCL 1
10 2
11 y^x
12 3
13 x
14 3
15 +
16 /
17 g PSE
18 g GTO 02


Again enter A onto the stack and then R/S. The numbers shown will converge reasonably quickly to the solution. For a second number type g GTO 00 before entering a number onto the stack.


Regards


Chris Dean
```

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #27 Posted by **Crawl** on 28 Aug 2006, 2:48 a.m.,*
in response to message #26 by Chris Dean

The first program doesn't actually converge for the example case of A = 1.23456e12.

It just oscillates back and forth between 4.1152e11 and 7.29046656e-12.

The question is, is that due to some sort of numerical error, or something else?

If we'd suppose your method would go back and forth between two values, that would be equivalent to

```
x = A/(  (A/(x^2 + 3))^2 + 3)
```

Or

3 x^5 - A x^4 + 18 x^3 - 6A X^2 + (27 + a^2) X - 9 = 0

That's a 5th degree equation, so it could obviously have two additional solutions beyond what the original cubic had.

Plugging in A and solving that equation on the 49G+ with PROOTS shows that the original cubic's example solution of 10727.63686 is still a root. However, 4.1152e11 is also a root to that quintic!

What about 7.29046656e-12? Not according to the 49G+ (though obviously numerical errors could be made even there. 27 might as well not even exist in the 27+a^2 term, for instance). Instead, the 49+ claims a different root that's very close to zero (5.9e-24). The difference between those values is probably due to some numerical error, and not a theoretical issue.

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #28 Posted by **Crawl** on 28 Aug 2006, 5:58 p.m.,*
in response to message #27 by Crawl

I made a mistake. The final term in the quintic, -9, should have been -9A. With the correct equation, the 49G+ does correctly give 7.29e-12 as the second added root.

Alternatively, you can divide the quintic by the original cubic, giving

3x^2 - A x + 9 = 0

That quadractic's two roots are the ones the method alternates between.

We note that the quadratic only has real solutions if a^2 > 108.

Indeed, even with a only equal to 11, the 12c's program converges to oscillating between 1.232408 and 2.434258, rather than converging to the true solution 1.781618.

On the other hand, with a equal to 10, it (very, very slowly) converges to the correct solution of 1.6988854.

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #29 Posted by **Crawl** on 29 Aug 2006, 8:51 a.m.,*
in response to message #28 by Crawl

Is anyone interested in solutions for non-HP calculators? ;)

I discovered that the Casio fx-115ms has an undocumented ability to do some simple "programming". It works fairly well for things like summing infinite series, or really any iterated process that converges fairly quickly (another example would be calculating Pi with Vieta's formula).

Here's an improvement on the

x = A / (x^2 + 3)

method for that calculator.

The improvement is that it takes two subsequent iterates and averages them. This suppresses the oscillations.

Here's how to do it.

Store A (from the equation) in A (the memory register).

Store A in X, too.

Enter (remembering to use "Alpha" rather than "recall" when a variable is the first entry on a line)

```
A/(X^2 + 3)->Y
A/(Y^2 + 3)->B
(B+Y)/2->X
```

Then tap up to the A/(X^2+3)->Y line, and hit shift-copy.

Now hitting "equals" repeatedly iterates.

What's the improvement?

After keying all that in, hitting "equals" 12 times converges to the correct value for A = 10! And that took forever on the 12C, even with the calculator automating the loops.

And for A = 11, it not only CONVERGES unlike the original case, it also does so with only 15 further "equals" presses!

What about for A = 1.23456e12? Nope, it still diverges. HOWEVER, you can replace the third line, where we had been taking the arithmetic average with

```
Sqrt(B*Y)->X
```

the geometric average.

You still would never, ever, ever want to run that on the real calculator, but the method in principle works. I ran that method on Qbasic and it took over an hour, but it seems to have eventually converged to the right value (within 9 digits).

The reason I posted this for the Casio is that it's so very simple on that calculator (only 3 lines, really).

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #30 Posted by **Chris Dean** on 29 Aug 2006, 1:38 p.m.,*
in response to message #29 by Crawl

```
Thanks for your interest in the algorithm I proposed. Do you have
any idea why the iterations do not converge on the 12C.


Is it due to rounding errors? The Newtons method does not suffer
from this condition even though it contains cubed and squared terms.


I even tried adding a 'guess' for the 1.23456e12 case and it still
oscillated.


Regards


Chris Dean
```

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #31 Posted by **Crawl** on 29 Aug 2006, 7:45 p.m.,*
in response to message #30 by Chris Dean

> Quote:
>
> ```
> Thanks for your interest in the algorithm I proposed. Do you have
> any idea why the iterations do not converge on the 12C.
>
>
> Is it due to rounding errors? The Newtons method does not suffer
> from this condition even though it contains cubed and squared terms.
> ```

```
I even tried adding a 'guess' for the 1.23456e12 case and it still
oscillated.

Regards

Chris Dean
```

It's not due to rounding errors; it's more fundamental. It would happen on any computer or calculator.

To see what's going on, near the solution y(x) = x, we can approximate the function (in this case, A/(x^2 + 3)) by a straight line.

Let y(x) = x have solution xo. Then the straight line will be

y = m (x-xo) + xo.

with m being the slope where y(x) intersects y = x. The form of the straight line was to make y(xo) = xo.

Now, the method is basically putting y back into x (ie., compositing the function with itself). If we'd do that with the straight line, we'd get

```
m((m (x-xo) + xo) - xo) + xo
```

=

$m^2(x-xo) + xo$

If you'd put y into into x n times rather than just once or twice, you'd get

m^n (x - xo) + xo

For this to converge, it would have to take on a stable value as n tends to infinity. For that to happen, you can see that m has to have an absolute value less than 1. Not only that will that make the infinite composition converge, but you can see that m^n -> 0 will get rid of the term that involves x, meaning that it's also independent of whatever the starting "guess" (the original x) is.

On the other hand, if y(x) does not exactly equal x, but is very, very close to it (even if you put in a close "guess"), but if m is greater than 1, then the m^n * (x-xo) term will tend to infinity -- that is, the infinite compositions will diverge -- even if (x - xo) is very small but not zero.

For a real function, not a straight line, it might not diverge to infinity (because the y =mx + b approximation won't be valid everywhere), but at least the m^nx term will push the compositions away from y(x) = x, and towards something else -- like oscillations. We'd except OSCILLATIONS to be related to a slope of -1 (rather than 1), because $(-1)^n$ really does oscillate as n increases.

So, what about y = A / (x^2 + 3) ? The derivative is

$-2ax/(3+x^2)^2$

=

$-2yx/((3+x^2)$

When y = x this is

$-2x^2/((3+x^2)$

Setting this equal to -1 and rearrainging gives

$x^2 - 3 = 0$

Or x = the squareroot of 3.

So, if x is greater than the squareroot of 3, it's impossible for the method to converge -- the slope will push the composition away.

Earlier, I mentioned that for the method to oscillate, the oscillating values would have to satisfy

3x^2 - A x + 9 = 0

but also that this only has real valued solutions if a^2 > 108.

If a = sqrt(108) = 6 sqrt(3), we can solve that quadratic with the quadratic formula.

The solution is ... also the squareroot of 3!

What that means is, as soon as the infinite compositions CAN start to oscillate rather than converge, they WILL start to oscillate!

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #32 Posted by **Chris Dean** on 30 Aug 2006, 2:54 a.m.,*
in response to message #31 by Crawl

Thanks for the analysis.

Chris

Re: HP-15C Mini-Challenge: Impossibly Short !?

*Message #33 Posted by **Kiyoshi Akima** on 28 Aug 2006, 1:50 p.m.,*
in response to message #24 by Kiyoshi Akima

With a little more tweaking to minimize the number of program steps (though not necessary program size), it comes down to seven steps on the 15C, including the LBL and RTN. The RTN isn't necessary if this is the last program in memory, so a six-stepper is possible.

```
001-42,21,11 LBL A
002-45,10, 2 RCL/ 2
003-43,22,23 ASINH
004-45,10, 3 RCL/ 3
005-42,22,23 SINH
006-45,20, 2 RCL* 2
007- 43   32 RTN
```

The program requires the constant 2 in R2 and the constant 3 in R3, but as we're striving for "absolute minimum program size," data register usage doesn't count, does it?.

Re: HP-15C Mini-Challenge: My Original Solution & Comments

*Message #34 Posted by **Valentin Albillo** on 28 Aug 2006, 6:40 p.m.,*
in response to message #1 by Valentin Albillo

Hi all,

As always, I'm delighted with the extreme ingenuity displayed by the many excellent contributions posted, thanks a lot for your continued interest and appreciation. My original solution is the following *9-step* routine which fulfills all given requirements:

```
01 LBL A
02  2
03  /
04 ASINH
05  3
06  /
07 SINH
08  2
09  *
```

which needs no RTN instruction (as end-of-program-memory performs and implied RTN), but *does* need the label, else the routine wouldn't perform as stated in the given examples (GSB A, ...). Also, there's no need to include a SF 8 instruction to set complex mode, because the only way the routine can receive a complex A parameter as argument in the X-register is for the calculator to *already* be in complex mode at entry.

This simple routine makes use of the fact that cubic equations of the given type, which are usually solved in closed form by a complicated radical-based formula including cube roots and square roots, can also be computed using *trigonometric functions*, either of the usual *circular* variety (SIN, COS, ASIN, ACOS, ...) or the *hyperbolic* variety (SINH, COSH, ASINH, ACOSH, ...) depending on the argument.

Usually one or the other kind will be used in order to keep all computations within the domain of real numbers, but it isn't actually necessary if your model allows for *complex arguments*, because circular and hyperbolic trigonometrics are intimately related, the ones becoming the others when the arguments change from real to complex, and vice versa. Thus, sticking with just circular or just hyperbolic for all possible cases will do, provided you can deal with complex arguments.

In this case, the root **x** is given by the short expression:

```
x = 2 * Sinh(ArcSinh(A/2)/3)
```

which is valid for *any* given numeric value of A which is legal for the HP-15C, from -9.999999999E99 to +9.999999999E99 in the real domain, and also for complex values of A. That's not the case when using radical-based formulas, which might overflow/underflow, can make it difficult to select the adequate value among the *three* possible values for the *two* required cube roots if dealing with complex arguments, and may require storing and recalling complex values to and from registers, which is somewhat cumbersome in the HP-15C.

That's also not the case for SOLVE-based routines, which can't deal with complex values of the A parameter, may take a long time to converge, and may fail to converge altogether. Besides, this trigonometric approach is *much faster* than either the radical- or the SOLVE-based ones.

Let's try my original routine for a complex parameter, **A = 1 + 2i**:

```
1, ENTER, 2, I, GSB A -> x = 0.4716050820 + 0.6060784194i
```

Let's check the result:

```
ENTER, ENTER, ENTER, *, 3, +, * ->  x = 1.000000000 + 2.000000000i
```

and you many see that it holds. This trigonometric approach can be extended to more general cubic equations, i.e.:

$$x^3 + p*x + q = 0$$

where we have:

```
x = 2*Sqrt(-p/3)*Cos(ArcCos(3*q/(2*p)*Sqrt(-3/p))/3)
```

where the trigonometrics must be able to accept arbitrary values as arguments (i.e., greater than 1 for ArcCos, or even complex), or else you may reformulate it in terms of hyperbolics and keep everything real and in range. As an example, for the equation:

$$x^3 - 6*x - 2 = 0$$

the above trigonometric formula quickly gives **x** = 2.601679132. This approach can also be extended to certain equations of higher degrees: even though the general 5th-degree equation (aka *quintic*) does require *elliptic* trigonometric functions, there are some families of 5th-degree equations which are amenable to this circular or hyperbolic trigonometric treatment, i.e. the quintic:

$16*x^5 - 180*x^3 + 405*x - 136 = 0$

has five real roots given by the trigonometric expression:

```
x = 3* Sin[1/5*(ArcASin(136/243) + 0*Pi, +-2*Pi, +-4*Pi)]
```

as seen in my article ***HP-12C Tried & Tricky Trigonometrics***, freely available on-line from my calculator web page.

Certainly, you'll agree with me that cubic equations having very simple closed-form solutions in terms of **hyperbolic functions** (instead of the ungainly Cardano formulas featured everywhere, in particular for cubic equations having just one real root, where all arguments are real), is quite *elegant* and highly *unexpected*, and rarely taught anywhere.

Next time that someone asks you what hyperbolic functions are useful for, you can confidently reply: *"to solve cubic equations easily and quickly"* and let him/her utterly perplexed. :-)

Best regards from V.

[Return to Index | Top of Index]

Go back to the main exhibit hall