Long Live The HP-35 !

Valentín Albillo (HPCC #1075)

I was too young when the **HP-35** was released back in 1972, just *35 years ago* (February 1st) for that *revolutionary* event to affect me in any way at the time. However, though more than 3 years would have to pass by till I saw the very first **HP** calculator (an **HP-21**, becoming instantly enamoured both of the calculator proper and its awesome **RPN** paradigm), I was nevertheless very keen to get an electronic calculator to satisfy my innate number-crunching lust. I had a slide rule and tables, but my numerically-oriented mathematical interests made mandatory for me to be able to *quickly* and *accurately* compute *transcendental functions* in order to be able to numerically solve equations involving them.

That I couldn't do with what I had, so I spent a real fortune (at 16, it was), namely the equivalent of 1974's USD 150, to get my very first electronic calculator, an 8-digit, 4-function **Sears** LED model. It was a tremendous improvement over the slide rule but still *far less* than I needed having no trascendentals and, worse, not even square root. But since there was no pocket calculator in sight that would do those, at any price, I had to make do with it and simply went on to develop "*programs*" for it, that is, very specific *sequences of keystrokes*, as short as possible, that given some *input* value would automatically, by simply executing them *repeatedly*, produce some desired *output*, such as square roots, cube roots, or even roots of polynomial equations.

I didn't know these sequences I was concocting were essentially *programs*, but they worked well enough in the **Sears** so I reckoned that if it had transcendental functions, the types of equations I could deal with would expand *enormously*, because although I had also developed keystroke sequences to compute transcendental functions in the 4-function **Sears**, they were long and tiresome and having them as part of an iterative method certainly squared the bother.

A dream come true

Then the **HP-35** came and it marked the dawn of affordable technical computation. For a modern reader it may seem utterly primitive in all respects but back then it was really *state-of-the-art*, tremendously expanding what you could do and how accurately you could it, while at the same time vastly reducing the effort to do it.

And of course, a keen user could go with it *much farther* than its awesome (for the time) repertoire of functions would seem to allow. To demonstrate this assertion and to let you indulge in a fond remembrance of the number-crunching strugglings of that time, I've ported the spirit of my first **Sears'** "programs" to the **HP-35**, for you to see how *truly complex* computations could be done with it at the time, which would let many a knowledgeable engineer and teacher with their jaws dropping in amazement. Put yourself in a 1972 frame of mind and bear with me.

Three sample applications:

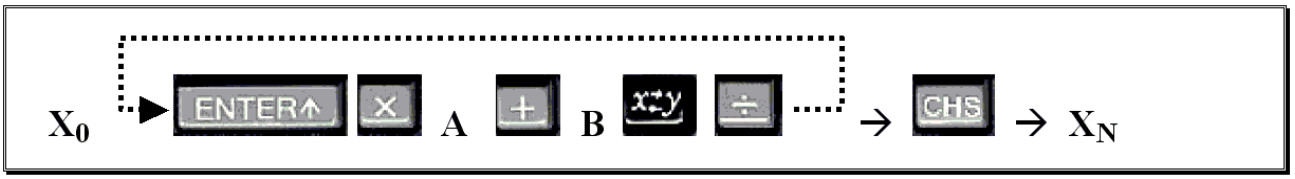
1. Let's suppose that we must solve the *cubic* equation $x^3 + Ax + B = 0$ for given **A**, **B**, and assume we don't remember the exact formulae but need to find a root with high (for 1972) accuracy, say to 6 or 7 decimal places.

Perhaps we might know about **Newton**'s iterative method, but although fast converging, it requires the computation of *both* the function's value and its *derivative* at every iteration and this does require a certain attention to detail which can be tiresome and prone to error. We would be better off if we could use some very simple, "programmatic procedure" which would produce the result reasonably fast while being as automatic and short as possible so that we can indulge in *very fast*, mindless key-pressing without having to think much about every step being performed. This way, with a little practice, we can get our results fast and with *a minimum of intellectual effort*, the results just appearing almost automatically.

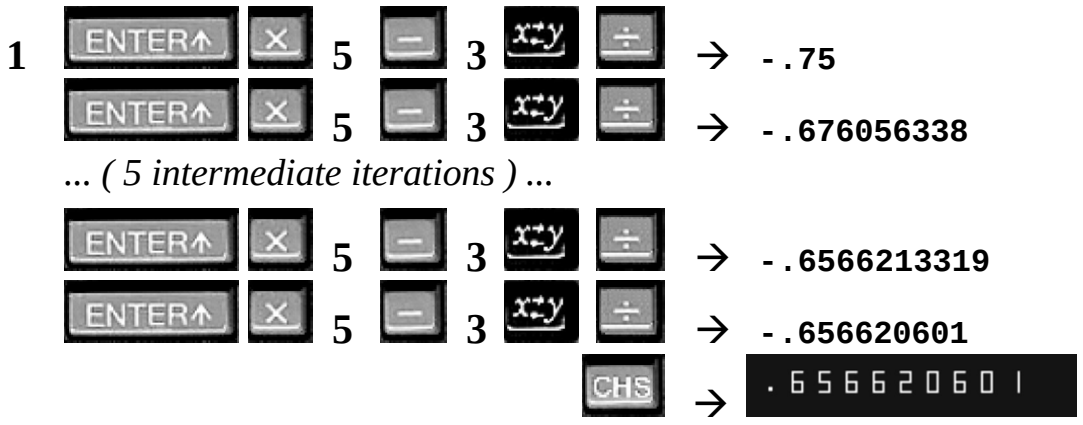
For this kind of cubic equation, we can formally proceed as follows:

1. the original cubic equation is: $x^3 + Ax + B = 0$
2. dividing by x and arranging we get: $x^2 + A = -B/x$
3. and "isolating" the <u>*second*</u> x we get: $x = -B/(x^2 + A)$

so our blind iterative procedure ("program" !) for the **HP-35** is then:



where X_0 is some initial guess and we repeat the above sequence until consecutive X_i are equal to within the desired precision, then **CHS** gets us the final result X_N. For instance, let's find a small (<1) positive root of $x^3 - 5x + 3 = 0$:

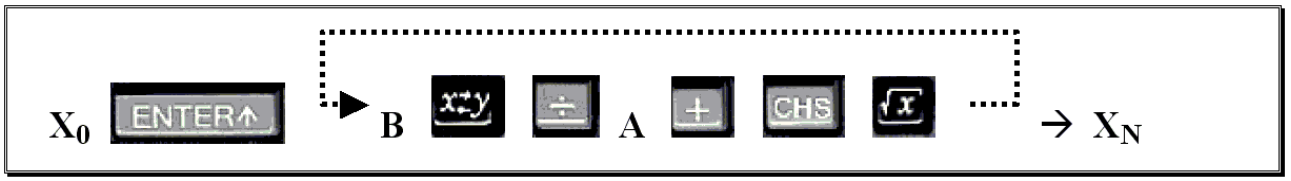


As the root is <u>***x = .6566204310***</u> we've got about *7 decimal places* by *routinely* performing just nine iterations of the above **HP-35** "program".

If the above program fails to converge, you can always try to *rearrange* the equation as an iterative procedure in a number of different ways, for example you could alternatively rearrange the computation as follows:

1. the original cubic equation is: $x^3 + Ax + B = 0$
2. dividing by x and arranging we get: $x^2 = -B/x - A$
3. and "isolating" the *first* x we get: $x = Sqrt(-B/x - A)$

and thus our new **HP-35** program is now:



where we need to do an initial **ENTER** but no final **CHS.** Let's find a *second*, larger positive root of this very same equation, $x^3 - 5x + 3 = 0$:

2 ENTER↑

3 x⇄y ÷ **5** − CHS √x → **1.870828693**

3 x⇄y ÷ **5** − CHS √x → **1.842941276**

... (4 intermediate iterations) ...

3 x⇄y ÷ **5** − CHS √x → **1.834250513**

3 x⇄y ÷ **5** − CHS √x → **1.834244966**

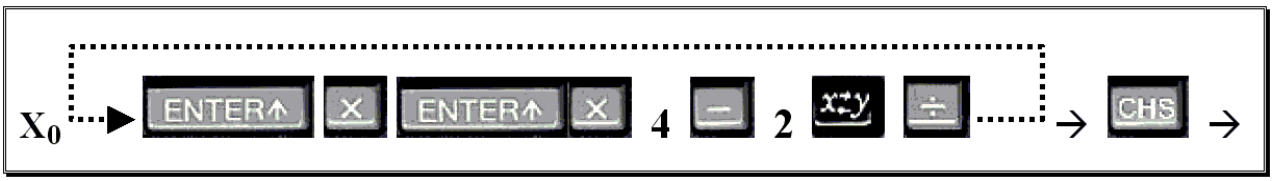
As the root is ***x = 1.834243184*** we've got again about *7 decimal places*, this time requiring just *eight* "mindless" iterations of the above **HP-35** sequence.

Of course, should you remember the (complicated) explicit formula which gives the roots, *you can easily compute them all in one go* thanks to the powerful **RPN** implementation available for the first time in pocket format in the **HP-35**, like this:

```
2 ENTER 5 ENTER 3 ÷ SQRT * 3 ENTER * 2 ÷ 5 CHS ÷ 3 ENTER
5 ÷ SQRT * ARC COS 3 ÷ STO COS * →   1.834243186   { 1st root }
                Rdown RCL 120 + COS * →   -2.490863614   { 2nd root }
                Rdown RCL 240 + COS * →   .6566204306   { 3rd root }
```

which gets you *all 3 real roots* in no time, using *no parentheses at all*, and with a *minimum* number of keystrokes needed. Several years would still have to pass by before the competition could even approach this kind of functionality.

In cases where *no* explicit formulas are available, the iterative approach really shines, such as when finding a root of $x^5 - 4x + 2 = 0$, where the procedure:

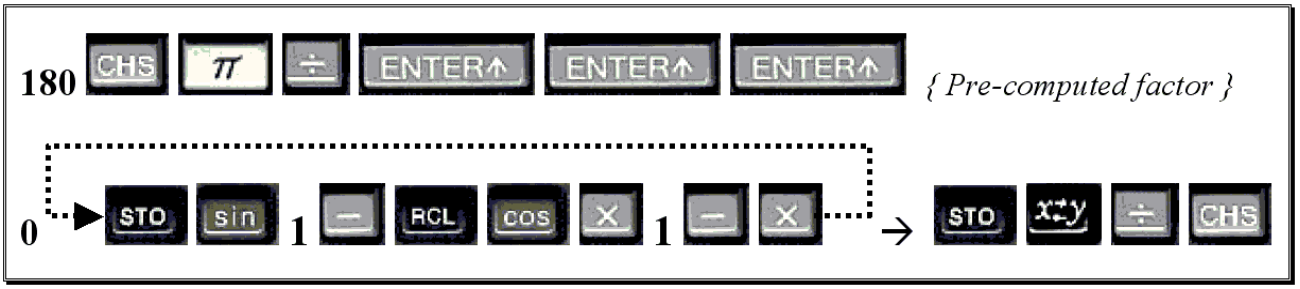


produces ***x = .5084995116***, correct to about *8 decimal places*, starting from the initial $\mathbf{X_0 = 1}$, in as few as 7 *purely mechanical* iterations, no thinking involved.

2. Let's now attack a more complicated problem, which will yield just as easily:

> ***"To find the minimum distance between the curve y = Sin(x) and the grid point (1,1), where x is expressed in radians".***

The following **HP-35** program will first produce the ***x*** coordinate of the point over the curve which is nearest to the point **(1,1)**, and then we'll compute the ***y*** coordinate as well as the requested *minimum distance*:



After ten "no-thinking" iterations, it produces ***x = 1.061756154*** (which is correct to about *6 places*), which we then proceed to use as follows:

RCL sin → **.87321**2671 *{ the Y-coordinate of the nearest point, correct to 5 places }*

1 − ENTER↑ × x⇄y 1 − ENTER↑ × + √x → **.1410278318**

which is the minimum distance, correct to about *9 places* !. So the computed answer is that **(1.06175, .87321)** is the point of ***y = Sin(x)*** closest to the grid point **(1,1)**, being at a minimum distance ***d* = 0.141027831**.

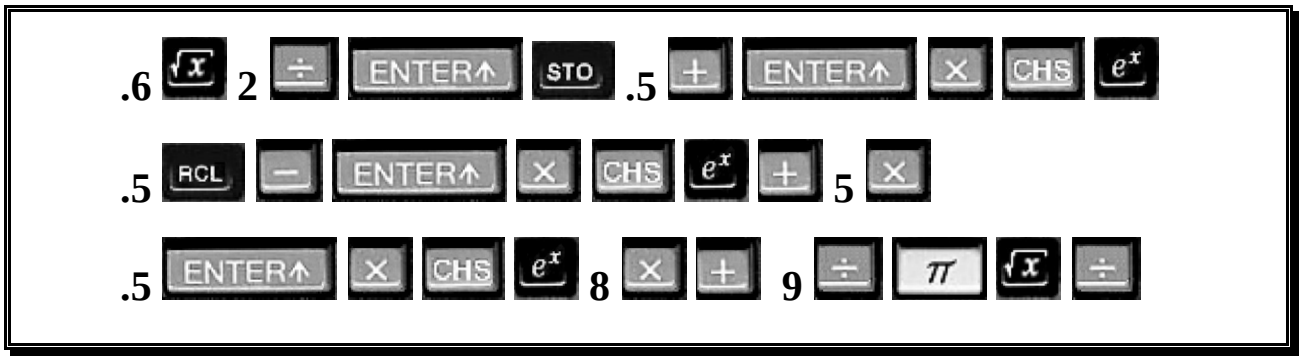
Notice that even though the **HP-35** computes trigonometrics *only in sexagesimal degrees*, all necessary conversions to radians where made on the fly. The conversion factor was continually kept on the top level of the **RPN** stack where it was *automatically replicated* each time it was used. How's that for sheer performance *and* elegance ? And all of this 35 years ago !!

3. Just as we've seen that root-finding is perfectly workable with the **HP-35**, *numerical integration* holds no terrors either. For example, say we're asked to find a numerical value for the following well-known probability-related integral:

$$\mathbf{f(1) = \frac{2}{\sqrt{\pi}} \int_0^1 e^{-x^2} \cdot dx}$$

which can't be expressed by elementary functions. These days we would take a *built-in* numerical integration facility for granted in nearly any modern calculator and many vintage ones, such as the **HP-34C**, **HP-15C**, **HP-41C** + Advantage ROM, or **HP-71B** + Math ROM, say, and failing that, we would be able to program our own integration procedure. But back in *those* days it was amazing enough that you could compute the exponential function e^x itself in a pocket calculator, let alone some transcendental integral expression of it.

What to do about it ? As we'll see, having the basic elementary functions at hand, plus the utmost convenience of the classic 4-level **RPN** stack, allows us to evaluate this integral *quickly* and *accurately* in as few as 39 keystrokes, like this:



which produces the result ***f(1) = .8426900188*** accurate to about 5 decimal places in no time (I timed myself at just 40 seconds), by simply using a 5th -order, 3-point **Gaussian** quadrature applied to the whole [0,1] integration interval. No *pocket* electronic device of the time (not to mention slide rules) could provide such accurate results with such speed and convenience.

Final remarks

At a time when most personal technical calculations where limited in speed and accuracy to what slide rules or basic electronic calculators could offer, the **HP-35**, with its fast, accurate transcendental functions plus the awesome elegance and power of its classic **RPN** stack, was a truly *"out of the blue"* cornerstone that revolutionized the world of portable computation changing it forever, a big step for both **HP** and humanity as a whole indeed. **Long live the HP-35 !**