A CLASSROOM NOTE: NEWTON'S METHOD DOUBLES DIGIT IMPROVEMENT

Richard D. Neidinger
Department of Mathematics
Davidson College
Box 7002
Davidson, North Carolina 28035
rineidinger@davidson.edu

The speed of convergence of Newton's root-finding method is dramatically demonstrated in examples where the number of accurate digits doubles with each iteration. But what is meant by this common description [1] of the formal concept called quadratic convergence? If the approximation and exact value round to the same digit, we call it an accurate digit; but where do you start counting? We could start at the first nonzero, counting accurate significant digits. One of my students counted accurate digits from the decimal point. Both are wrong; our digit counts may not double. The improvement, the number of newly accurate digits in each approximation, will roughly double.

Newton's method approximates a root r where $f(r)=0$ by a sequence of approximations $x_{k+1}=x_k-f(x_k)/f'(x_k)$ with absolute error $e_k=|r-x_k|$. To understand the issue of accurate digits (in bold), Table 1 lists results for Newton's method on the example $f(x)=300x-(1-(1+x)^{-360})$, where the zero of this function is the monthly interest rate in a present value problem over 30 years.

$x_1=.001$	$e_1=4.312\times10^{-5}$	
$x_2=.\mathbf{001}044821810919$	$e_2=1.706\times10^{-6}$	$e_2/e_1^{\,2}=918$
$x_3=.\mathbf{001043}117844288$	$e_3=2.444\times10^{-9}$	$e_3/e_2^{\,2}=839$
$x_4=.\mathbf{001043115400}624$	$e_4=5.028\times10^{-15}$	$e_4/e_3^{\,2}=842$
$x_5=.\mathbf{001043115400619}$		

Table 1

Assuming quadratic convergence, it is surprising that the magnitude of the error is not doubled on each step, expecting 10^{-5} to square to around 10^{-10}, which is the expectation that accurate digits to the right of the decimal point will double. But *quadratic convergence* means that $\lim_{k\to\infty}\frac{e_{k+1}}{e_k^{\,2}}=C$ for some

constant, as is verified in the third column. To double the number of accurate digits (to the right of the decimal point) would require $C \approx 1$. The example does show doubling in the number of correct significant digits, 3 to 6 to 12, but this is also not true in general. If we let $t = 1 + x$, and use Newton's method on $f(t) = 300(t-1) - (1 - t^{-360})$ with $t_1 = 1.001$, then we get exactly the approximations $t_n = 1 + x_n$. However, the significant digits now go from 6 to 9 to 15, and do not double. The magnitude of r controls the number of significant digits, but quadratic convergence is independent of r. At the end of this note, we show how to define $f(x)$ to give quadratic convergence for any C and r, affecting the number of accurate and significant digits. First, we show that the improvement in accurate digits will roughly double, for any function and initial value where quadratic convergence occurs.

The correct principle, independent of C and r, considers the ratio of errors:

$$\frac{e_{k+2}}{e_{k+1}} \approx C\, e_{k+1} \approx \left(\frac{e_{k+1}}{e_k} \right)^2 \qquad (1)$$

for any iteration that is close to the quadratic convergence limit. If $e_{k+1} / e_k \approx 10^{-n}$, then $x_{k+1} \approx r \pm 10^{-n} e_k$ and thus has about n more digits of accuracy than $x_k \approx r \pm e_k$. We need two approximations to see when and how the convergence has begun. If the first iteration increases accuracy by one digit, then $e_2 / e_1 \approx 10^{-1}$, so we expect $e_3 \approx 10^{-2} e_2$ and $e_4 \approx 10^{-4} e_3$, showing that the number of digits that improve between iterations doubles. Shifting the root and all approximations, say by 600, shows that this is not about significant digits.

Of course, we are limited by the crude concept of number of correct digits and the wall of finite machine precision. If we define the *digits of improvement* on step k to be $-\log(e_{k+1} / e_k)$, then the log base ten of (1) shows that digits of improvement do double on each step of quadratic convergence. In practice, we count the number of digits that agree between iterations, when both are rounded to that digit, and subtract for improvement, resulting in a range possibilities. In Table 1, $-\log(e_{k+1} / e_k)$ is about 1.4, 2.8, 5.7 for k = 1, 2, 3, resulting in 1, 3 and 6 improved digits. Improving digits like this will quickly hit the maximum 16 digits as limited by floating point representation on most machines, and counting digits 14 through 16 might be suspect due to roundoff effects. We use extended precision in *Mathematica* to compute results below.

We now show that Newton's method on $f(x)=(x-r)+C(x-r)^2$ will display quadratic convergence with limiting ratio C and root r. Digits of improvement will double, but not necessarily significant digits relative to r or digits to the right of the decimal point adjusted by C. Most numerical analysis texts prove that Newton's method converges quadratically under appropriate hypotheses, including an initial approximation close enough to a simple root ($f(r)=0$ but $f'(r)\neq 0$), as well as two continuous derivatives [2]. The value of C follows from Taylor's theorem about x_k :

$$0=f(r)=f(x_k)+f'(x_k)(r-x_k)+\frac{f''(\xi_k)}{2}(r-x_k)^2$$

for some ξ_k between r and x_k. Dividing by $f'(x_k)$ and moving all but the last term to the left,

$$-r+x_k-\frac{f(x_k)}{f'(x_k)}=\frac{f''(\xi_k)}{2f'(x_k)}(r-x_k)^2.$$

Using Newton's method iteration $x_{k+1}=x_k-\frac{f(x_k)}{f'(x_k)}$ and $e_{k+1}=\left|-r+x_{k+1}\right|$,

$$\lim_{k\to\infty}\frac{e_{k+1}}{(e_k)^2}=\lim_{k\to\infty}\left|\frac{f''(\xi_k)}{2f'(x_k)}\right|=\frac{|f''(r)|}{2|f'(r)|}=C,$$

assuming x_k converges to the root (a significant complication in the proof) and continuous derivatives. Any smooth function with this ratio of derivatives at the root r should display the desired convergence characteristics. We simply chose $f(r)=0$, $f'(r)=1$, and $f''(r)=2C$ in the Taylor polynomial about r.

For $r=600.001$ and $C=1000$ in $f(x)=(x-r)+C(x-r)^2$, Newton's method iteration starting with $x_1=600.0015$ results in

$$x_2=600.001125000000000000000000000000000000$$
$$x_3=600.001012500000000000000000000000000000$$
$$x_4=600.001000152439024390243902439024390243902439024$$
$$x_5=600.001000000023230573665078314909939710$$
$$x_6=600.001000000000000005396595277354293100$$
$$x_7=600.001000000000000000000000000000000003$$

Clearly, steps do not double correct significant digits and do not double correct digits to the right of the decimal point, but each step does double the number of digits that improve.

REFERENCES

1. Ward Cheney and David Kincaid, *Numerical Mathematics and Computing*, 6th Edition, p. 93, Thomson Brooks/Cole, Belmont, CA, ISBN: 978-0-495-11475-8 (2008).
2. Anne Greenbaum and Timothy P. Chartier, *Numerical Methods: Design, Analysis, and Computer Implementation of Algorithms*, p. 86, Princeton University Press, Princeton, NJ, ISBN: 978-0-691-15122-9 (2012).