Better Real Root-Finding

W. Kahan, Prof. Emeritus
Math. Dept., & E.E. & Computer Sci. Dept.
Univ. of Calif. @ Berkeley

For the Seminar on Scientific Computing & Numerical Linear Algebra
2 April 2014

Distilled from "Lecture Notes on Real Root-Finding" :
<www.eecs.berkeley.edu/~wkahan/Math128/RealRoots.pdf>
<www.eecs.berkeley.edu/~wkahan/Math128/SOLVEkey.pdf>
pp. 42-4 of <www.eecs.berkeley.edu/~wkahan/B0u1der.pdf>

Posted at <www.eecs.berkeley.edu/~wkahan/Math128/LecRlRtF.pdf>

Abstract:

Given a program to compute a real function f(x) of a real scalar x , and given one or two guesses at z , firmware like HP calculators' [SOLVE] key and MATLAB's `fzero` seek a root z of the equation " f(z) = 0 ". Old software like `fzero` requires initial guesses that straddle z ; [SOLVE] accepts guesses that need not straddle a root; and for three decades [SOLVE] has allowed searches for a root to stray temporarily outside the domain of f . Evidently the theory of real rootfinding has advanced beyond almost all real rootfinding software.

The theory will be surveyed.

A worthy project for a mathematically adept CS student is to make well-engineered rootfinding software more available.

For proofs and other details see

www.eecs.berkeley.edu/~wkahan/Math128/RealRoots.pdf
www.eecs.berkeley.edu/~wkahan/Math128/SOLVEkey.pdf
pp. 42-4 of www.eecs.berkeley.edu/~wkahan/B0u1der.pdf

Given a real equation “ $f(z) = 0$ ”, we solve it for its real ***root*** z ,
a ***zero*** of function f ,

typically by ***Iteration*** :

$$x_{n+1} := U(x_n) \longrightarrow z = U(z)\ ,\ \text{a}\ \textbf{\textit{fixed point}},\quad \text{as}\ n \longrightarrow \infty\ .$$

How should the Iterating Function U be constructed?

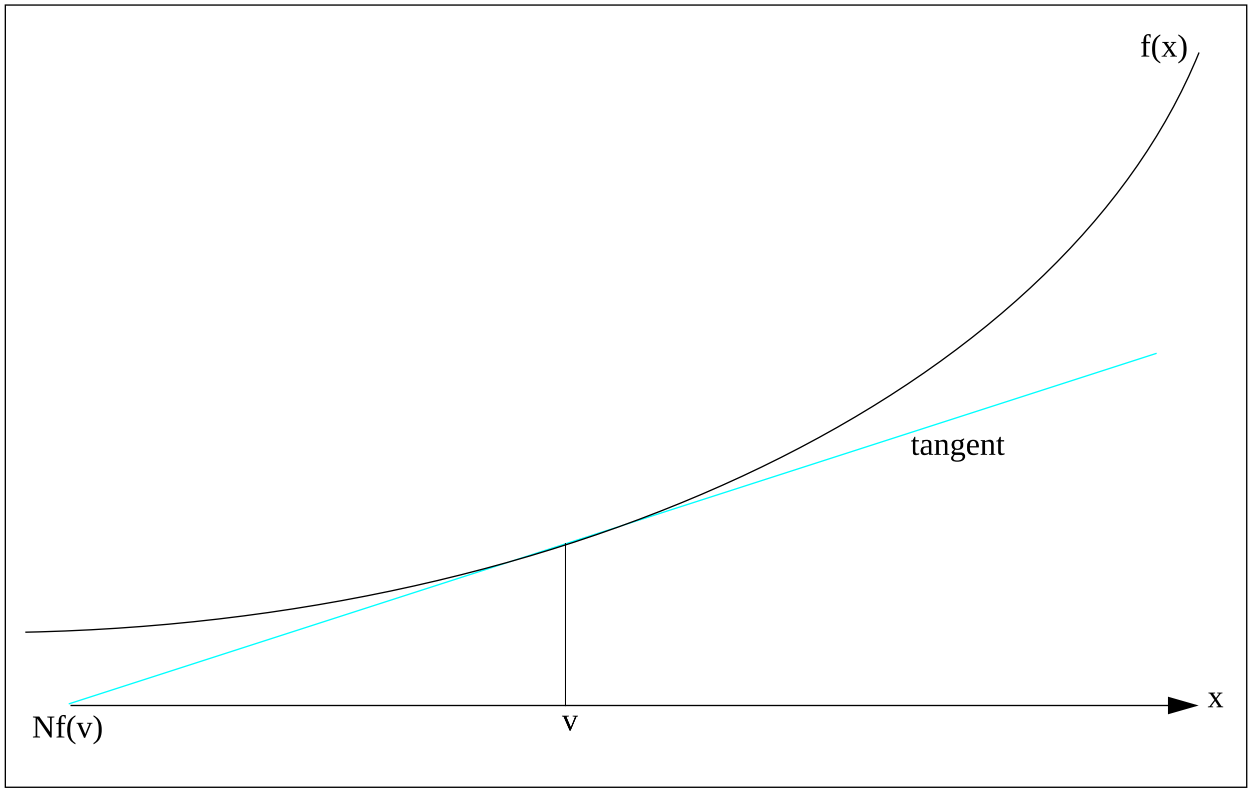
Iteration Functions Used Frequently:

Newton’s:

$$x_{n+1} := Nf(x_n) \quad \text{where} \qquad Nf(x) := x - f(x)/f'(x)\ .$$

Secant:

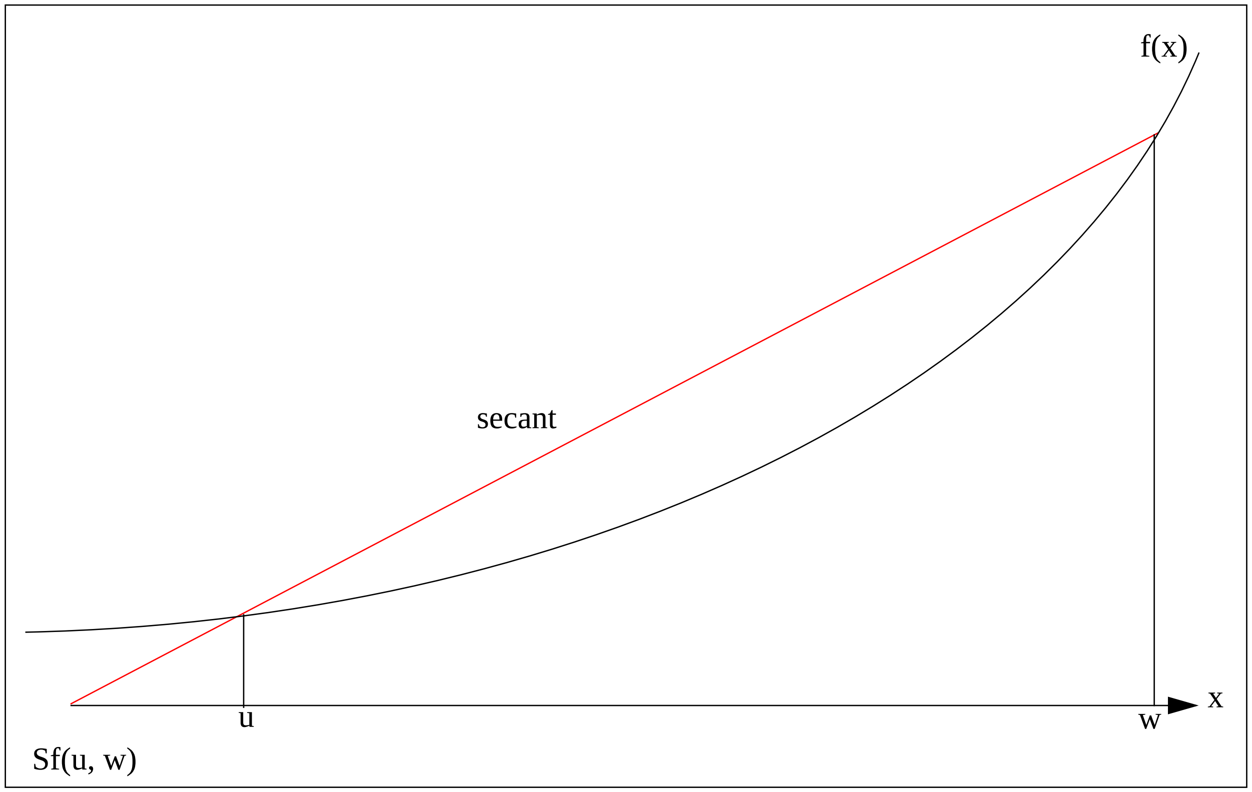
$$x_{n+1} := Sf(x_n, x_{n-1}) \quad \text{where} \quad Sf(x, y) := x - f(x)(x-y)/(\,f(x) - f(y)\,)\ .$$



Newton's iteration:

$$x_{n+1} := Nf(x_n) \quad \text{where} \qquad Nf(x) := x - f(x)/f'(x) \; .$$

Usually convergence to a simple zero is *Quadratic* (*Order* 2) if it occurs.



Secant iteration:

$$x_{n+1} := Sf(x_n, x_{n-1}) \quad \text{where} \quad Sf(x, y) := x - f(x)(x-y)/(f(x) - f(y)) .$$

Usually convergence to a simple zero is *Superlinear* (Order 1.618) if it occurs.

Six questions that arise before equation “ $f(z) = 0$ ” can be solved:

«1» Which equation?
Infinitely many, some far easier to solve than others, have the same root(s) z .

«2» What method?
Among infinitely many, which iterative method will work well with the chosen f ?

«3» Where should the search for a root begin?
A global theory of an iteration’s convergence may compensate for a poor guess at z .

«4» How fast can the iteration be expected to converge?
Convergence much slower than a local theory predicts is ominous.

«5» When should iteration be stopped?
Error-analysis helps here. And the possibility that no z exists may have to be faced.

«6» How will the computed root’s accuracy be assessed?
Error-analysis needed here is unlikely to be performed; few can do it.

And then these questions lead to many more …

More Questions: For most answers see §_ of …/RealRoots.pdf

Is some simple Combinatorial (Homeomorphically invariant) condition both Necessary and Sufficient for convergence of $x_{n+1} := U(x_n) \to z = U(z)$? (Yes; §5)

Is that condition relevant to the design of rootfinding software? (Very much so; §6)

Do other iterations $x_{n+1} := U(x_n)$ besides Newton's exist? (Not really; §3)

Must there be a neighborhood of z within which Newton's iteration converges if $f'(x)$ and $x - f(x)/f'(x)$ are both continuous? (Maybe Not**!** §7)

Do useful conditions less restrictive than Convexity suffice Globally for the convergence of both Newton's and the Secant iteration? (Yes; §8)

Why are these less restrictive conditions not Projective Invariants, as are Convexity and the convergence of Newton's and Secant iterations? (I don't know; §A3)

More Questions continue …

More Questions continue …

Is slow convergence to a multiple root worth accelerating? (Probably not; §7)

Can slow convergence from afar be accelerated with no risk of overshooting and thus losing the desired root? (In certain common cases, Yes; §10)

When should iteration be stopped? (***Not*** for the reasons usually cited; §6)

Which of Newton's and Secant iterations converges faster? (Depends; §7)

Which of Newton's and Secant iterations converges from a wider range of initial guesses at z ? (Secant, unless z has even multiplicity; §9)

Some of these many questions will be examined hereunder …

Do other iterations $x_{n+1} := U(x_n)$ besides Newton's exist? (Not really; §3)

Thesis 3.1: Newton's Iteration is Ubiquitous
Suppose that U is differentiable throughout some neighborhood Ω of a root z of the given equation " $f(z) = 0$ ". If the iteration $x_{n+1} := U(x_n)$ converges in Ω to z from every starting point x_0 in Ω , then this iteration must be Newton's iteration applied to some equation $g(z) = 0$ equivalent on Ω to the given equation; in other words,

$$U(x) = x - g(x)/g'(x) , \quad \text{and } g(x) \longrightarrow 0 \text{ in } \Omega \text{ only as } x \longrightarrow z .$$

Defense: $g(x) = \pm\exp(\int dx/(x - U(x)))$ with a "constant" of integration that may jump when x passes from one side of z to the other, reflecting the fact that U is unchanged when $g(x)$ is replaced by, say, $-3g(x)$ for all x on one side of z .

Consequently, much of what we learn about Newton's and the closely related Secant Iterations may be applicable to other iteration formulae.

Is some simple Combinatorial (Homeomorphically invariant) condition both
Necessary and Sufficient for convergence of $x_{n+1} := U(x_n) \to z = U(z)$? (Yes; §5)

Theorem 5.1: Sharkovsky's No-Swap Theorem [1964, 5]

Suppose U maps a closed interval Ω continuously into itself; then the iteration $x_{n+1} := U(x_n)$ converges to some fixed–point $z = U(z)$ from every x_0 in Ω if and only if four conditions, each of which implies all the others, hold throughout Ω :

No-Swap Condition: U exchanges no two distinct points of Ω ; in other words, if $U(U(x)) = x$ in Ω then $U(x) = x$ too. (Proof? Draw a picture!)

No Separation Condition: No x in Ω can lie strictly between $U(x)$ and $U(U(x))$; in other words, if $(x - U(x))(x - U(U(x))) \le 0$ then $U(x) = x$.

No Crossover Condition: If $U(x) \le y \le x \le U(y)$ in Ω then $U(x) = y = x = U(y)$.

One-Sided Condition: If $x_1 := U(x_0) \ne x_0$ in Ω , then all subsequent iterates $x_{n+1} := U(x_n)$ also differ from x_0 and lie on the same side of it as does x_1 .

Sharkovsky's theorem greatly simplifies many assessments of (non)convergence.

A direct application of Sharkovsky’s No-Swap Theorem:

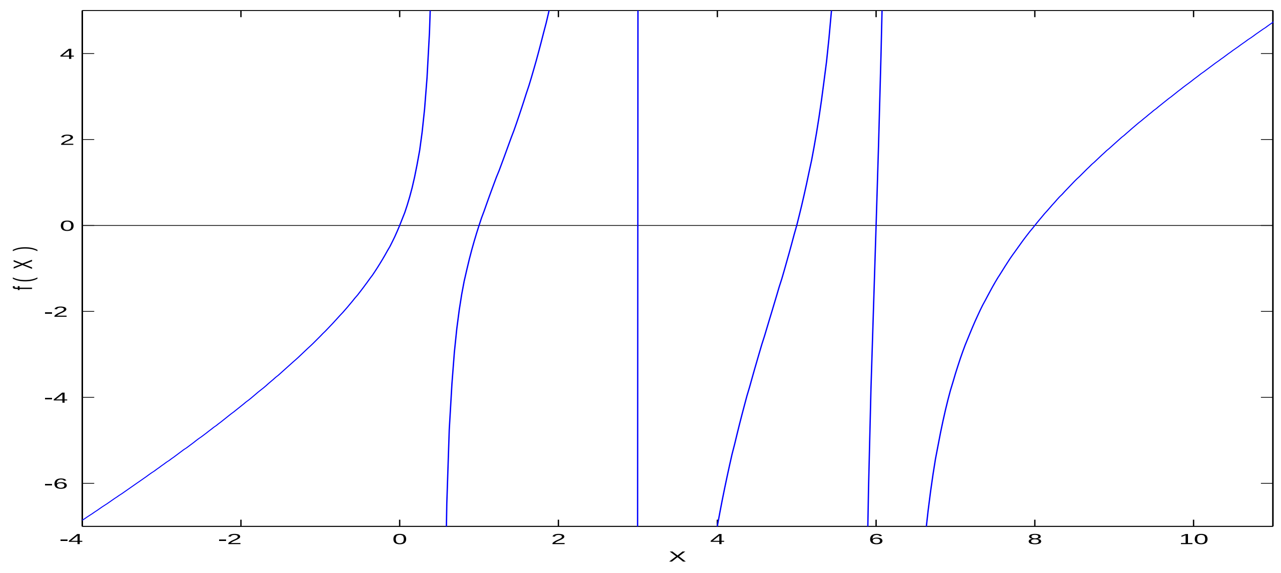
Example 5.4: Suppose f is a rational function with simple real interlacing zeros and poles, one of them a pole at ∞. One instance is

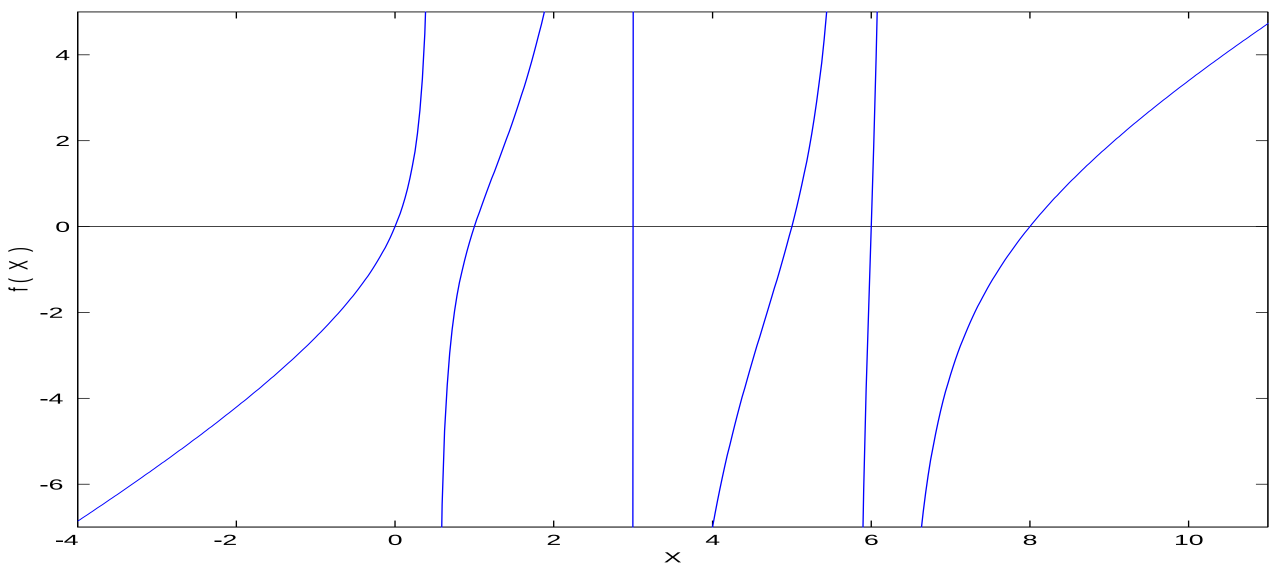
$f(x) := p(x)/p'(x)$ where $p(x)$ is a polynomial all of whose zeros are real.

Another instance is

$f(x) := \det(xI - A)/\det(xI - \hat{A}) = \prod_i (x - z_i)/\prod_j (x - \hat{o}_j)$ in which A is an hermitian matrix from which $\hat{A}$ is obtained by striking off A ‘s last row and column, and the I ’s are identity matrices. The zeros z_i are eigenvalues of A, and the poles $\hat{o}_j$ are the distinct eigenvalues of $\hat{A}$ not also eigenvalues of A.

The zeros z_i and poles $\hat{o}_j$ interlace, *i.e.*, $z_0 < \hat{o}_1 < z_1 < \hat{o}_2 < z_2 < \ldots < \hat{o}_K < z_K$.





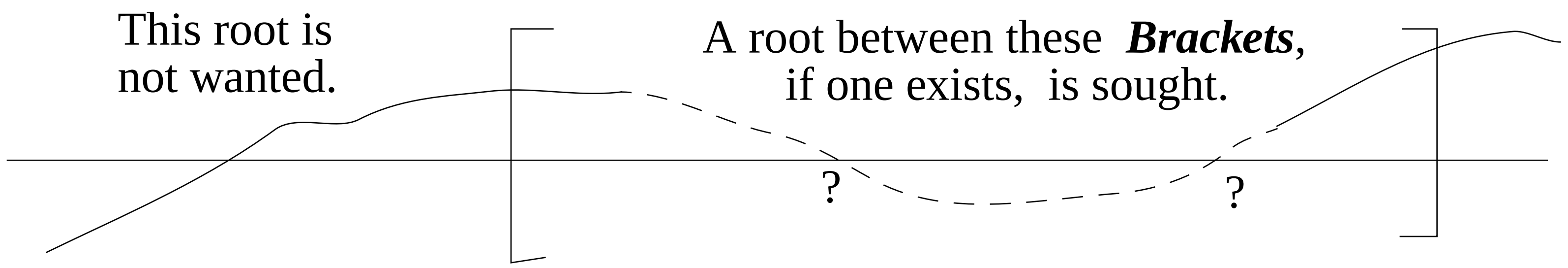
Like Y. Saad [1974], we shall try to compute the zeros by running Newton's iteration

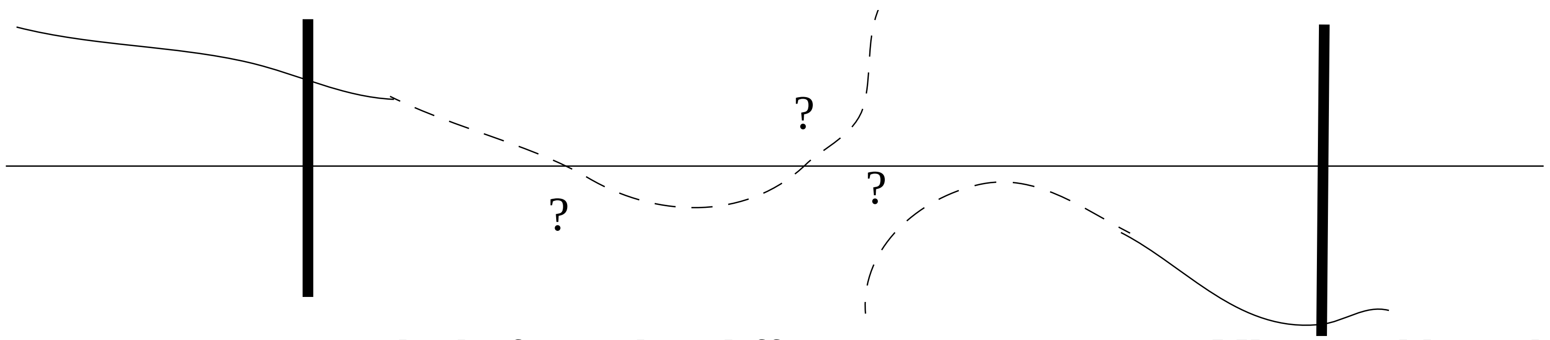
$x_{n+1} := x_n - f(x_n)/f'(x_n)$. Does it converge? If so, to what?

These are thorny questions, considering how spiky is the graph of f, and yet Newton's iteration can now be proved to converge to some zero z_i from every real starting value x_0 except a countable nowhere-dense set of starting values from which the iteration must converge accidentally (after finitely many steps) to a pole $\hat{o}_j$. The proof comes directly from Sharkovsky's No Swap Theorem after a little algebra.

One-Sided Condition: If $x_1 := U(x_0) \neq x_0$ in Ω, then all subsequent iterates $x_{n+1} := U(x_n)$ also differ from x_0 and lie on the same side of it as does x_1.

This condition's violation can be detected by a program, and thus motivates its measures designed to prevent a raw iteration's non-convergence … *Brackets* and *Straddles*.


This root is not wanted.
A root between these ***Brackets***, if one exists, is sought.
?
?



Two arguments x at which f(x) takes different signs must ***Straddle*** an odd number of points where f changes sign, and a rootfinder should surely find one of them.

Rootfinding software succeeds if it ensures that every iteration shrinks the width of any available Bracket or Straddle, and shrinks the widths ultimately to zero.

Do useful conditions less restrictive than Convexity suffice Globally for the convergence of both Newton's and the Secant iteration? (Yes; §8)

Corollary 8.3: A Weakened Convexity Condition

Suppose $f = g - h$ is a differentiable difference between two convex functions, one non–decreasing and the other non-increasing, throughout a closed interval Ω . Then Newton's iteration $x_{n+1} := x_n - f(x_n)/f'(x_n)$, started from any x_0 in Ω , either converges in Ω to the zero z of f or leaves Ω .

The iteration cannot meander in Ω endlessly.

An Early Application:

Rapid calculations of Interest Rates and *Internal Rates of Return* by Financial Calculators like the HP-12C, still being bought since 1982.

A mortgage's interest rate is the positive zero of a polynomial whose degree is the number of payments in the life of the mortgage. With the possibility of daily electronic funds transfers, the degree can be enormous.

Which of Newton's and Secant iterations converges from a wider range of initial guesses at z ? (Secant, unless z has even multiplicity; §9)

Theorem 9.2: Suppose $f'(x)$ and $Nf(x) := x - f(x)/f'(x)$ are continuous throughout a closed finite interval Ω strictly inside which f does not vanish without also reversing sign there. If Newton's iteration converges in Ω from every initial x_0 in Ω , then it converges to the sole zero z of f in Ω , and Secant iteration also converges in Ω to z from every two starting points x_0 and x_1 in Ω .

The converse need not be true; Secant can converge when Newton's doesn't.

Alas, the proof is very long.
It begins with *Projective Maps*.

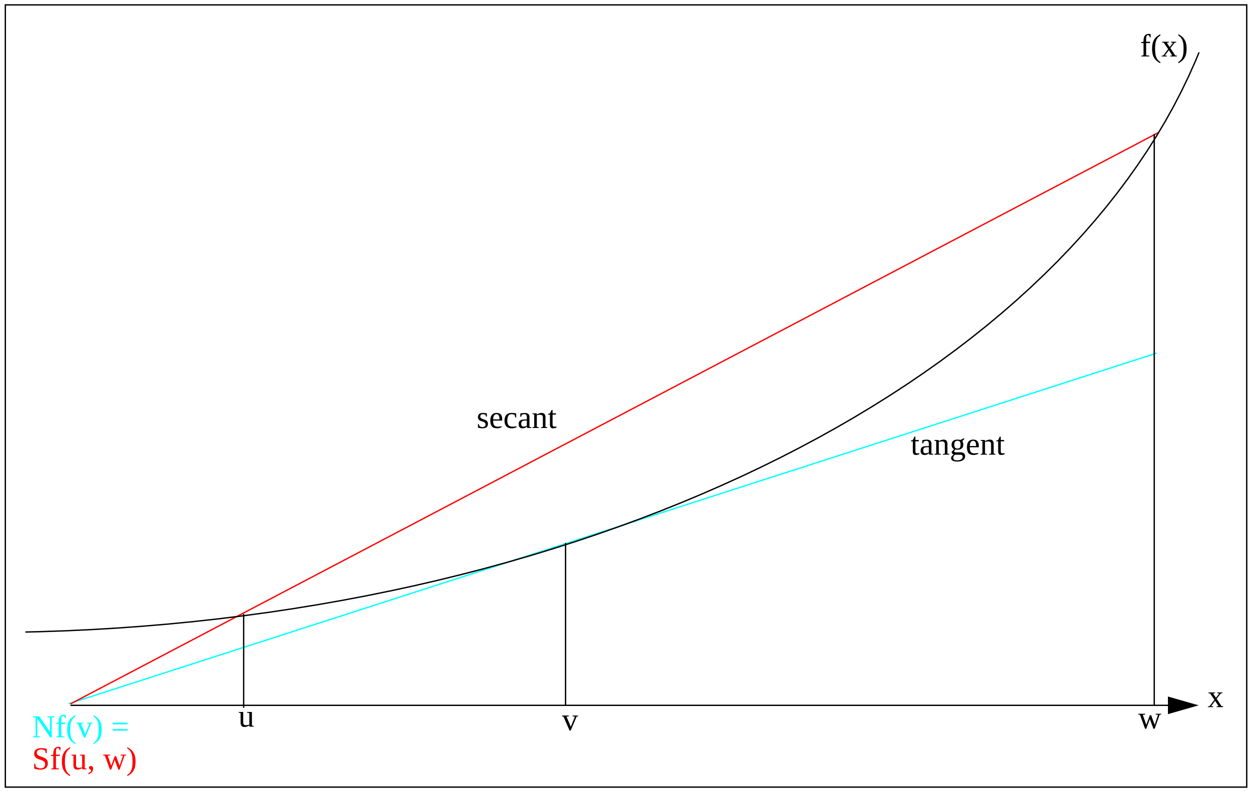
A Projective Map of the plane to itself must map every straight line to a straight line. Therefore tangents are mapped to tangents, and secants to secants.

The Projective Maps needed to prove Theorem 9.2 map the x-axis to itself, but may map a finite point on it to ∞ .

Lemma 9.1: An Intermediate Value

If $Sf(u, w) := u - f(u)\cdot(u - w)/(f(u) - f(w))$ does not lie between u and w, *i.e.* if $f(u)\cdot f(w) > 0$, and if $f'(x)$ is finite throughout $u \le x \le w$, then some v exists strictly between u and w satisfying either

$$Nf(v) := v - f(v)/f'(v) = Sf(u, w) \quad \text{or} \quad f(v) = f'(v) = 0 .$$



The proof begins with a projective map that takes $Sf(u, w)$ to ∞.

Is slow convergence to a multiple root z worth accelerating? (Probably not; §7)

Theorem 7.5: Suppose $|f'(x)|$ increases as x moves away from z through some neighborhood Ω on one side of a zero z of f . Then $0 < (Nf(x) - z)/(x-z) < 1$ and so Newton's iteration converges monotonically to z from every initial x_0 in Ω . Similarly $0 < (Sf(x,y) - z)/(x-z) < 1$ for all x and y in Ω and so Secant iteration converges monotonically to z from every initial x_0 and x_1 in Ω . But convergence can be arbitrarily slow if $f'(z) = f(z) = 0$.

Theorem 7.6: Under the convexity hypothesis of Theorem 7.5, the iterates x_n may converge to z arbitrarily slowly, though monotonically; but $f(x_n)$ tends monotonically to 0 at least so fast that

$$\sum_n (2^n f(x_n))^2 \le f(x_0)^2 (x_0 - z)/(x_0 - x_1) .$$

In other words, $f(x_n) \to 0$ faster than $O(1/2^n)$ as $n \to \infty$. (Often $O(e^{-n})$.)

There is a special but common case that can be accelerated modestly without overshoot.

Define

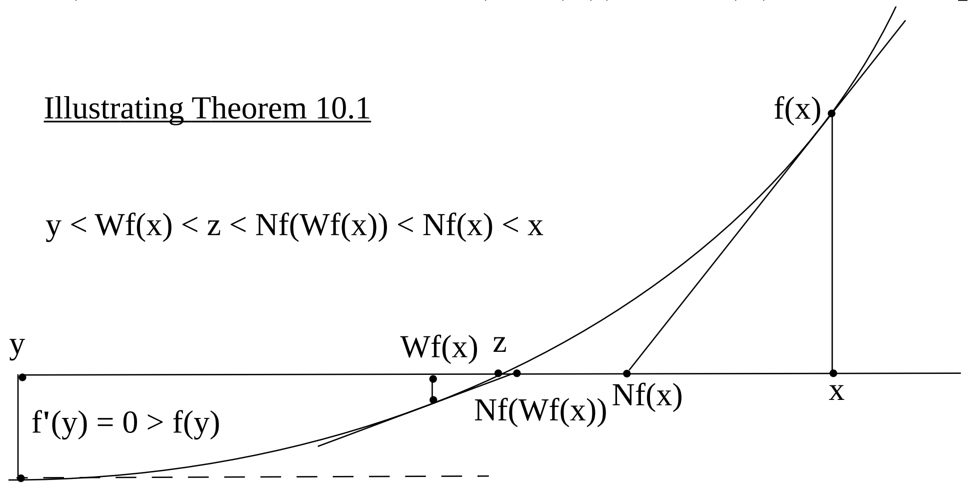
$Nf(x) := x - f(x)/f'(x)$ (Newton's iteration function) and

$Wf(x) := x - 2\cdot f(x)/f'(x)$ (Doubled-Newton's iteration function) .

This $Wf(x)$ can be iterated with no harm from overshoot in the following circumstances:

Theorem 10.1: Suppose that $f'(y) = 0 \ge f(y)$ at the left-hand end of a finite interval $y \le x \le x_0$ throughout which f'' is a positive nondecreasing function; also assume $f(x_0) > 0$. Then, in that interval, ...

1) The equation $f(z) = 0$ has just one root $z \ge y$, and $Wf(x) < Nf(x)$ when $x > z$.
2) $Nf(x) \ge z$ for all $x > y$, and then $Wf(x) > y$ unless $Wf(x) = y = z$.
3) If $x > z$ then $Nf(Wf(x)) \le Nf(x)$, with equality only when $f''' \equiv 0$.


Illustrating Theorem 10.1
y < Wf(x) < z < Nf(Wf(x)) < Nf(x) < x
f(x)
y
Wf(x)
z
Nf(Wf(x))
Nf(x)
x
f'(y) = 0 > f(y)

In other words, iterates of Wf approach z twice as fast as Nf does until Wf overshoots z , and then reverting to Nf loses less than one iteration.

Typical Behavior of Rootfinding Software

Three phases:

1. Flailing: Successive iterates may seem erratic if no Straddle has been found; or else they seem to be taking a long leisurely walk from a poor initial guess; or else they seem to be departing only reluctantly from that initial guess.

2. Converging: Differences between successive iterates are dwindling fast enough, or values of $|f(x)|$ at successive iterates are decaying fast enough, that this phase cannot last long. Higher order iterating formulas are worth their cost only if such extraordinarily high accuracy is sought as would prolong this phase.

3. Dithering: Roundoff, mostly in $f(x)$, makes iterates seem to behave erratically; or one side of a Straddle is doing most the moving as it shrinks intolerably slowly.

When will the iteration stop?

When should iteration be stopped? (***Not*** for the reasons usually cited; §6)

Stopping when consecutive iterates differ by less than a preassigned tolerance suggests that the iterates are in error by not much more than this tolerance. This suggestion can be utterly wrong if convergence was very slow, and also if convergence seemed fast but the graph of f(x) is too nearly L-shaped, as often occurs when f involves exponentials.

A good policy is to stop when |f(x)| falls below a tolerance not too much bigger than roundoff's contribution to f(x) as determined by an error-analysis. Requiring an error-analysis turns this policy into a *Counsel of Perfection* impractical for too many users of numerical software. And the connection between the tolerance upon |f(x)| and the error |x - z| in the root may entail a further error-analysis difficult even for experts.

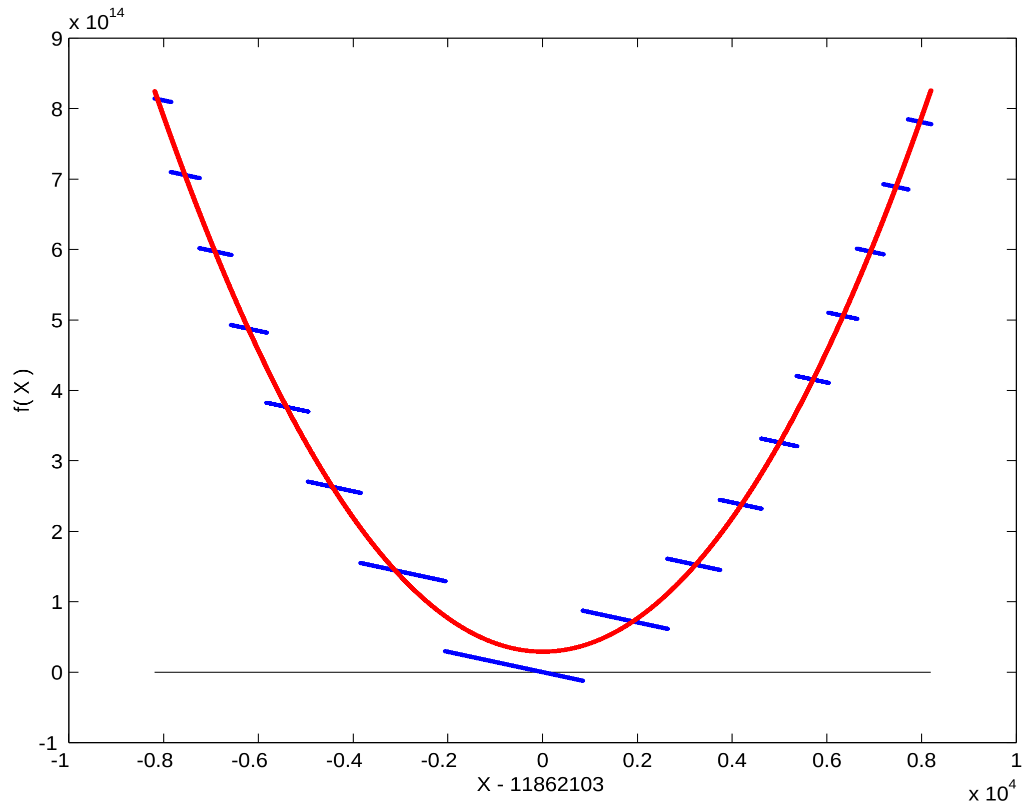
The iteration has to stop when no more floating-point numbers lie strictly inside its Bracket or Straddle. To reach this state without wasting too much time Dithering must challenge the rootfinding software designer's management of Brackets and Straddles in the face of the raggedness of roundoff.

The next simple example is a cubic polynomial

$$f(x) := ((x - 23722988)\cdot x + 16770435\cdot 2^{23})\cdot x + 9968105\cdot 2^{34}$$

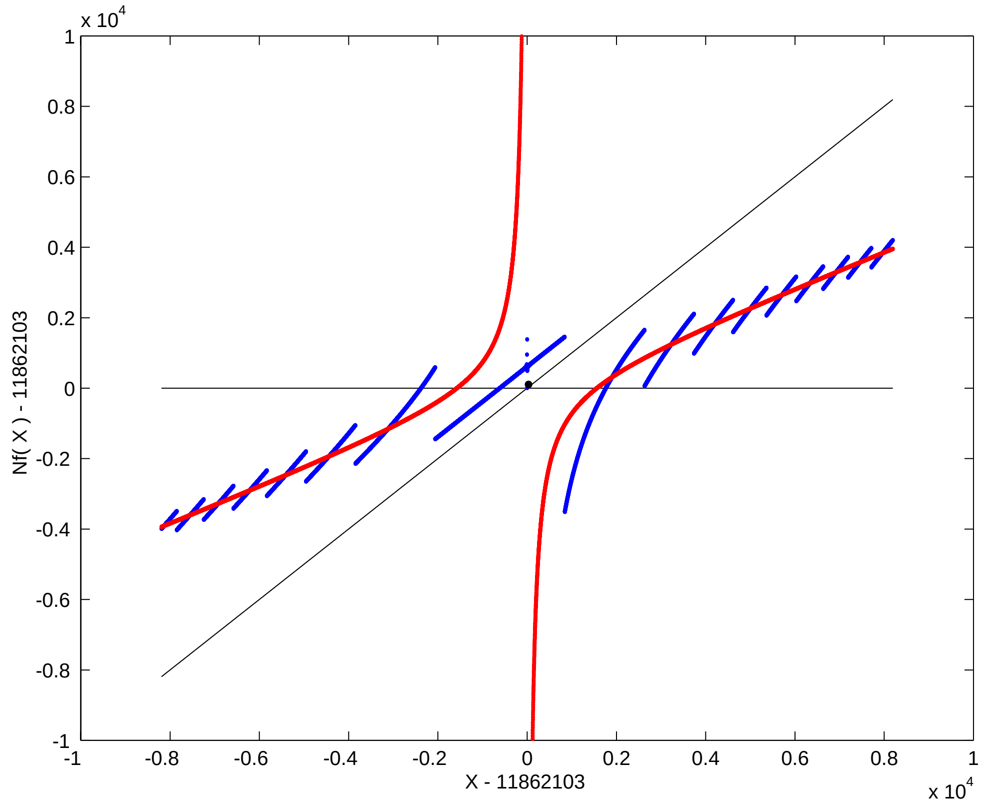
with coefficients and all arithmetic restricted to 24 sig.bit floating-point.

Example 6.4:



The jagged curve is cubic f(x) plotted at 16385 consecutive floating-point values x computed in 24 sig.bit floating-point. The smooth curve is f(x) uncontaminated by roundoff. Note the segments sloping the wrong way. Near x ≈ 11862945 are two places where the computed cubic reverses sign although the uncontaminated cubic's sole real zero z ≈ -1217.051909940… lies far away to the left of this picture.

The jagged graph below exhibits Nf(x) := x - f(x)/f'(x) for the cubic f(x) still computed in 24 sig.bit floating-point at the same 16385 consecutive floating-point values x . The nearly hyperbolic graph is Nf(x) uncontaminated by roundoff.



Starting from the far right, shrinking Straddles are easier to find and to manage with Newton’s than with Secant iterates though both are inhibited by Binary Chops.

When should iteration be stopped? (***Not*** for the reasons usually cited; §6)

A policy that aborts execution as soon as f(x) is NaN, because an iterate strayed outside the domain of f , can prematurely abort the search for a zero of f .

How can the search's trial-arguments x be restricted to the domain of f if its boundary is unknown? Is this boundary easier to find than a zero of f ?

Example:

$$\mathrm{shoe}(x) := (\tan(x) - \arcsin(x))/(x\cdot|x|^3) \quad \text{except} \quad \mathrm{shoe}(0) := +\infty .$$

We seek a root Z > 0 of the equation shoe(Z) = 0 if such a root exists. (We don't know.)
We know x = 0.5 lies in shoe's domain, but (pretend) we don't know its boundary.

Does your rootfinder find Z ? Or does it persuade you that Z probably does not exist ?

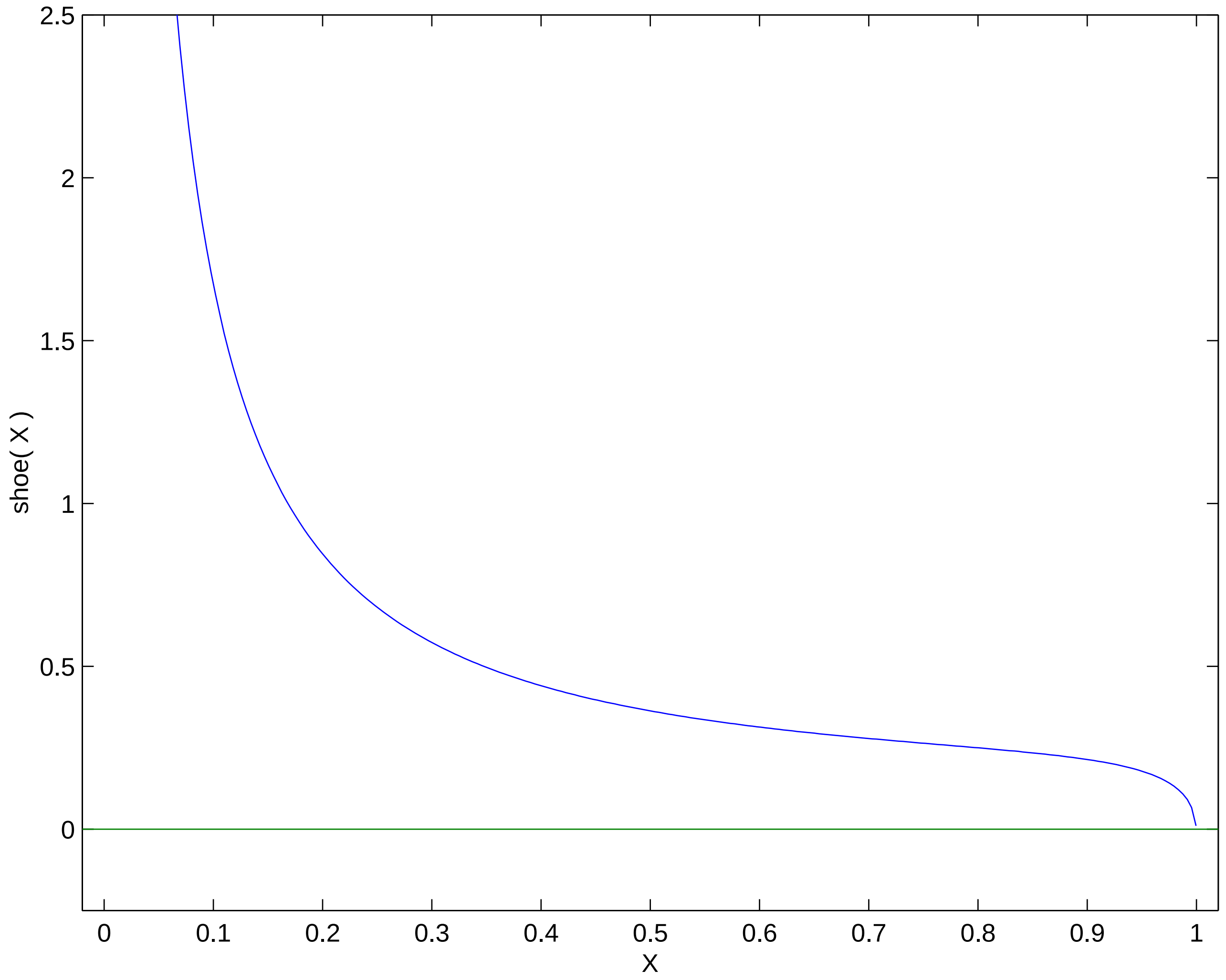
Try, say, each of 19 initial guesses x = 0.05, 0.1, 0.15, 0.2, …, 0.5, …, 0.9, 0.95 .

`fzero` in MATLAB 6.5 on a PC said it cannot find a root near any one of them.

`root` in MathCAD 3.11 on an old Mac diverged, or converged to a huge *complex* no.

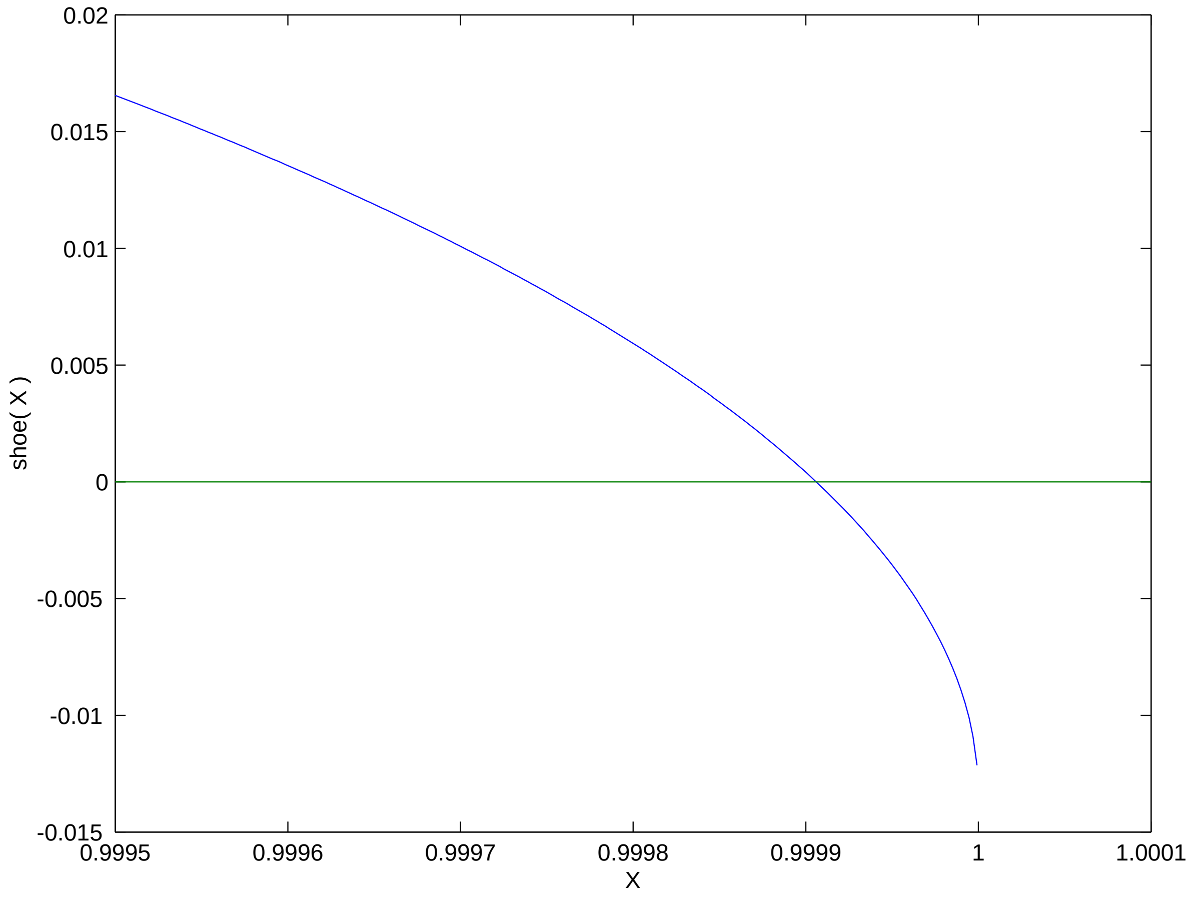
How did [SOLV] on HP-18C, 19C and 28C handheld calculators find what they didn't ?

shoe(x) := (tan(x) – arcsin(x))/(x·|x|^3)



If no positive Z in shoe(x) ’s domain satisfied shoe(Z) = 0 ,
then the SHOE would leak at its toe.

shoe(x) := (tan(x) – arcsin(x))/(x·|x|^3)



Notice the 1000-fold change in the scale of the x - axis.

The HP-28C found the root Z = 0.999906012413 from each of those 19 first guesses.

What did the calculator know/do that the computers didn't ? … **Defer Judgment !**

I think some Casio calculators too may know how to do it.

Conclusion:

A good real rootfinder is needed to be made widely available.

Its mathematical engineering should approach efficiency as high as current theory allows.

Its software engineering should offer ease of use and flexibility limited only by language.

To solve " $f(z) = 0$ " the rootfinder should accept a program that computes *either*
$f(x)$ or $f(x)/f'(x)$, for any numerical inputs x , perhaps an array of them.

The rootfinder must accept one or two initial guesses at z , and a Bracket if no Straddle.

If f is actually $f(x, \text{parameters})$, passing the parameters to f should be convenient.

Tolerances upon $|f(x)|$, $|\Delta x|$ and execution time should accept 0, 0 and ∞ , but
the rootfinder must always terminate with an indication of what has been found,
a zero, a sign reversal, a nonzero near minimum of $|f(x)|$, … .