1.3 Limits of Accuracy

Goals of "Numerical Analysis": (1) Efficiently (i.e. quickly) (e.g. linear/quadratic converge) compute an answer (eg: solve: $x^2+x+1=0$); (2) Find answer within a specified level of accuracy.

Recall: $|x_a - r| = \boxed{\text{Error}}$ AND $\dfrac{|x_c - r|}{|r|} = \boxed{\text{relative error}}$

x_a: Approx. sol.
r: True solution

Definition: Assume that f is a function with root r, so that: $f(r)=0$, and x_a is an approximate solution of $f(x)=0$ (so $r \approx x_a$). For the root-finding problem, the $\boxed{\text{backward error}}$ of the approximation x_a is $\boxed{|f(x_a)|}$ and the $\boxed{\text{forward error}}$ is $|r - x_a|$.

Q: why "backward"/"forward"?

Process of root finding:

Equation $\longrightarrow$ Algorithm (Bisection, FPI) $\longrightarrow$ Solution

"Backward" error refers to error in equation/problem.
$|f(x_a)|$

"Forward" error refers to error post-algorithm.
$|x_a - r|$

$\boxed{\text{Ex.}}$ Let $f(x) = \sin x - x$; Find The $\boxed{FE}$ (forward error) & $\boxed{BE}$ (backward error) for $\boxed{x_c = .001}$.

$$FE = |r - x_a| = 10^{-3} \quad (r=0)$$

$$BE = |f(x_a)| = |s(.001) - (.001)| \approx 1.6667 * \underline{10}^{-10}$$

Note That The magnitudes of FE & BE can vary dramatically, as The previous example illustrates.

Recall That in Computational Mathematics, "stopping criteria" provide a set of conditions That elicit a Termination To a computational process/algorithm - generally speaking, There are $\boxed{\text{Three}}$ Types of stopping conditions (relevant To our course).

$\boxed{1}$ $\boxed{FE}$: Make $|x_a - r|$ as small as possible: e.g.

$|x_a - r| < \text{Tolerance}$ (say: $TOL = 10^{-4}$ OR $TOL = \text{Mach}_\varepsilon$).

Note: If r is not known directly, as is often The case, we commonly check: $|x_{i+1} - x_i| < TOL$ for $\boxed{\text{iterative algorithms}}$

$\boxed{2}$ $\boxed{BE}$: Make $|f(x_a)|$ small.

$\boxed{3}$ Impose limit on computation Time / # of computations.

e.g. Halt algorithm after 1 minute, etc.

:) One common difficulty for root-finding algorithms relates To The inflation of BE for $\boxed{\text{multiple roots}}$ of a function.

Definition

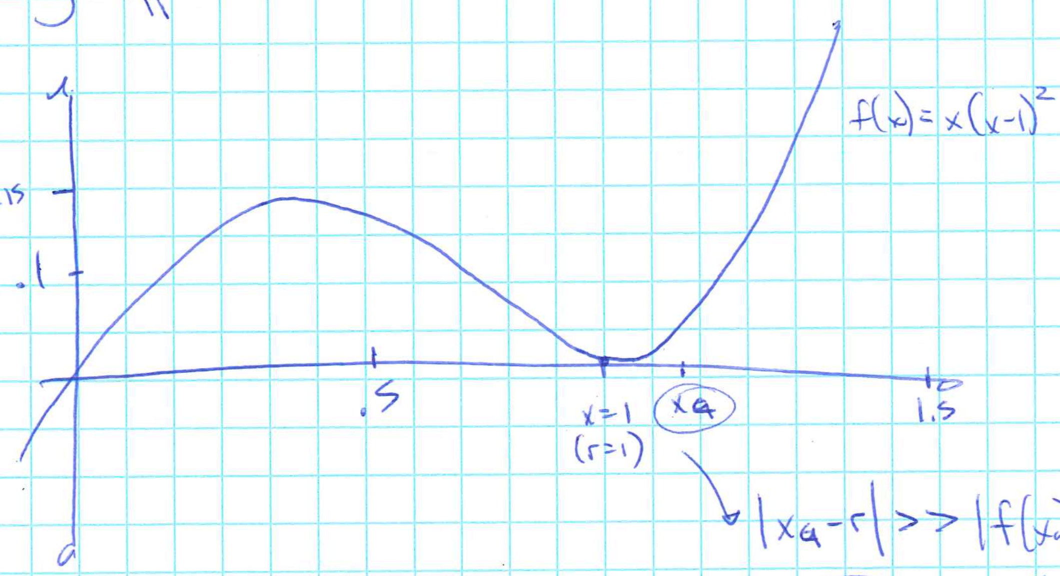
Assume that r is a <u>root</u> of a differentiable function f, so $f(r)=0$. Then, if $f(r)=0$, $f'(r)=0$, $f''(r)=0$, ..., and $f^{(m-1)}(r)=0$, <u>But</u> <u>$f^{(m)}(r) \neq 0$</u>, we say That <u>The root</u> r (of f) <u>has multiplicity m.</u> If The multiplicity of a <u>root is 1</u>, we call This root a $\boxed{\text{simple root.}}$

Ex. Let $f(x) = x(x-1)^2$. $f(x)$ has a root of <u>multiplicity 2</u> @ $x=1$, and a root of <u>multiplicity 1</u> @ $x=0$.

<u>Check</u>: $f(0)=0$, $f'(0)=0$, But $f''(0) \neq 0$. ✓

$f(1)=0$, $f'(1)=0$, $f''(1)=0$, But $f'''(1) \neq 0$. ✓

∘) <u>Note</u>: Because The graph of a function <u>relatively flat</u> near a multiple root, a great disparity exists between BE & FE for nearly approximate solutions.



$|x_a - r| >> |f(x_a)|$

(*) FE much larger Than BE

A classic example of a function with simple roots that are hard to find numerically is the so-called $\boxed{\text{Wilkinson Polynomial}}$:

$$\boxed{W(x) = (x-1)(x-2)\cdots(x-20)}$$ ($w(x)$ has simple roots for $x = 1, \ldots, 20$).

Using CAS: findzero of $W(x)$ with $x_0 = 16$ (actual result).

$\longrightarrow$ CAS output: 16.014680... only accurate to one decimal place!

Big Idea: Large error inflation (FE) for finding roots due to small relative errors in storing coeffs of $w(x)$ (BE).

So, for $w(x)$, (small BE) $\rightarrow$ (large FE)!

Analysis of the Sensitivity of Root-Finding

- A problem is called $\boxed{\text{sensitive}}$ if small errors in the input (this case: the equation) yield large errors in the output or solution.
- To better understand the phenomenon of "sensitivity", we establish a formula for predicting how far a root "moves" when the equation is altered.

Set-up: Assume the goal is to find the root r, where $\boxed{f(r) = 0}$.

Also suppose that a "small" change: $\boxed{\varepsilon g(x)}$ is made to the equation.

Let Δr denote the corresponding change in r.

Thus, $f(r) = 0 \longrightarrow \boxed{f(r+\Delta r) + \varepsilon g(r+\Delta r) = 0}$

(Note: $r+\Delta r$ satisfies the "perturbed" equation: $f(x) + \varepsilon g(x) = 0$).

We use the first degree Taylor expansions of $f(x)$ & $g(x)$ around the center: r.

Whence, $f(r+\Delta r) \approx f(r) + (\Delta r) f'(r) + \underline{\mathcal{O}(\Delta r^2)}$

& $g(r+\Delta r) \approx g(r) + (\Delta r) g'(r) + \underline{\mathcal{O}(\Delta r^2)}$

($\mathcal{O}$: "Big O" Notation).

So, $f(r+\Delta r) + \varepsilon g(r+\Delta r) =$

$$f(r) + (\Delta r) f'(r) + \varepsilon g(r) + \varepsilon (\Delta r) g'(r) + \underbrace{\mathcal{O}(\Delta r^2)}_{(\approx 0)} = 0$$

(Note for Δr small, $(\Delta r)^2 \approx 0$)

Now we solve for Δr:

$$(\Delta r)\left(f'(r) + g'(r)\right) \approx -\overset{=0}{f(r)} - \varepsilon g(r) = -\varepsilon g(r)$$

Consequently, $\Delta r \approx \dfrac{-\varepsilon g(r)}{f'(r) + \varepsilon g'(r)} \approx -\varepsilon\left(\dfrac{g(r)}{f'(r)}\right)$

we assume: $\underline{\varepsilon \ll f'(r)}$ & $\underline{f'(r) \neq 0}$.

In $\underline{\text{summary}}$, if r is a root of $f(x)$ & $r+\Delta r$ is a root of $f(x) + \varepsilon g(x)$, then:

$$\boxed{\Delta r \approx -\varepsilon \cdot \left(\frac{g(r)}{f'(r)}\right)}$$ where $\varepsilon \ll f'(r)$, $f'(r) \neq 0$.

sensitivity formula for roots.

Ex. Estimate the <u>largest root</u> of $P(x) = (x-1)(x-2)\cdots(x-6) - \underbrace{10^{-6}}_{\varepsilon}\underbrace{x^7}_{g(x)}$

<u>Note:</u> Without the $\varepsilon g(x)$ term, $r=6$ is the largest root of $P(x)$.

So, we now check <u>how much the root: $r=6$ is affected,</u> <u>when we "shift" the original equation with the "extra" $\varepsilon g(x)$ term?</u>

Using <u>the Sensitivity Formula</u>, we can readily approximate this quantity:

$$\Delta r \approx -\varepsilon\left(\frac{g(r)}{f'(r)}\right) \qquad \left(g(x) = x^7\right)$$

$$= -\varepsilon\left(\frac{6^7}{5!}\right) = \boxed{-2332.8\,\varepsilon}$$

This shows that <u>"input" errors</u> $\approx \varepsilon$ are <u>magnified 2000-fold</u>, roughly, in <u>output errors</u>!

<u>Check:</u> $r + \Delta r = 6 - 2332.8\,\varepsilon = \underline{6.0023328}$ $\quad (\varepsilon = 10^{-6})$

CAS: Solve: $P(x) = 0 \longrightarrow \underline{6.0023268}$ ✓

The previous example showed us how errors <u>propagate</u> in the root-finding problem; an error in the <u>sixth digit of the problem</u> caused an error in the <u>third digit</u> of the answer.

This error magnification is formalized as follows:

Def.

$$\text{Error magnification factor} = \frac{\text{relative FE}}{\text{relative BE}}$$

Error Magnification Factor $= \dfrac{\text{relative FE}}{\text{relative BE}} = \left| \dfrac{\Delta r / r}{\epsilon g(r)/g(r)} \right|$

$$= \left| \frac{-\cancel{\epsilon} g(r)/r f'(r)}{\cancel{\epsilon}} \right| = \boxed{\frac{|g(r)|}{|r f'(r)|}}$$

Ex. EMF for the previous problem:

$$\left| \frac{g(r)}{r f'(r)} \right| = \frac{6^7}{6 \cdot 5!} = \boxed{389} \approx 3.89 * \underline{10^2}.$$

Now, since $\epsilon = 10^{-6}$, the emf $\approx 10^2$ shows that we expect to lose $\underline{2}$ of the $\underline{6}$ decimal places of precision, as the previous example demonstrated.

Ex. Use the sensitivity formula to investigate the effect of changes in the x^{15} term of $W(x)$ for $r = 16$.

Define the perturbed function: $W_\epsilon(x) = W(x) + \epsilon \cdot g(x)$

with $g(x) = -\underbrace{1{,}672{,}280{,}820}_{\text{"}\epsilon\text{"}} x^{15}$. Note: $W'(16) = 15!\,4!$ (Why?)

Then $\underbrace{\Delta r}_{\text{change in root}} \approx \dfrac{16^{15}(1{,}672{,}280{,}820)\,\epsilon}{15!\,4!} \approx 6.1432 * 10^{13} \epsilon$

$(\epsilon = 10^{-16})$

emf $= \dfrac{|g(r)|}{|r f'(r)|} = \dfrac{16^{15} \cdot 1{,}672{,}280{,}820}{15!\,4!\,16} \approx 3.84\,10^{12}$ 12 digits of 16 lost!

Conditioning & Stability

The condition number: Measure of error magnification.

Stability: Magnification of small input errors due to algorithm
↳ inherent in problem

Condition number: Maximum error magnification over all input changes.

Ill-conditioned: Large condition #

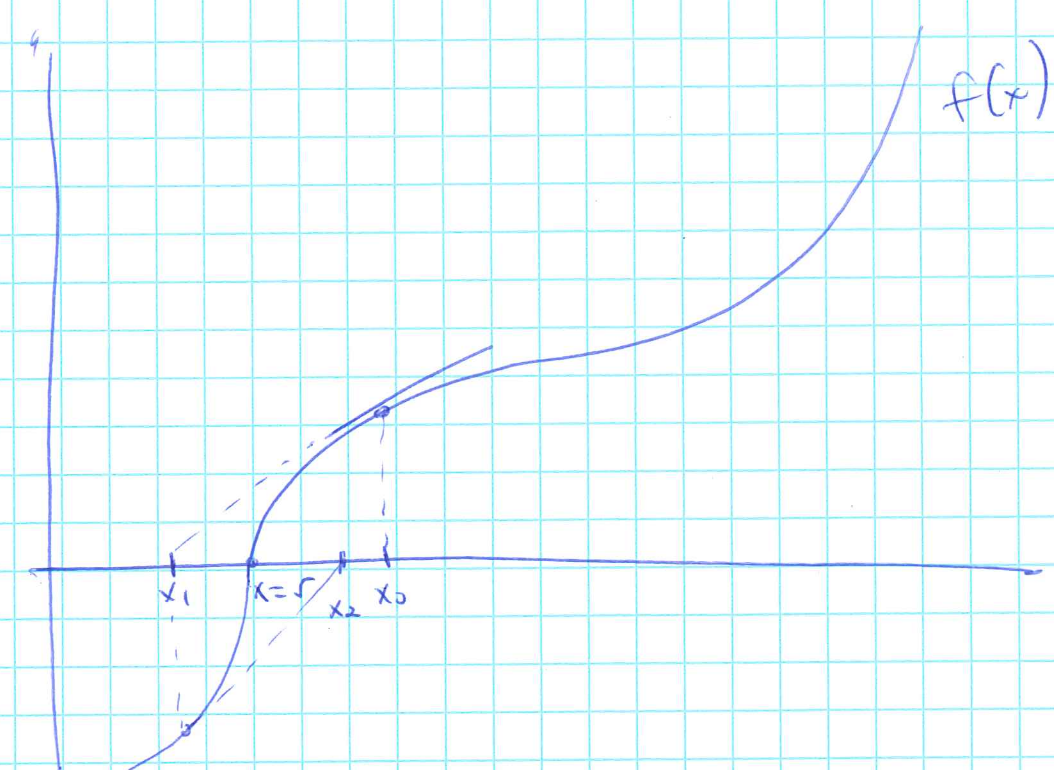
Well-conditioned: condition # ≈ 1.

1.4 Newton's Method

Newton's Method is an iterative, root-finding algorithm that generally converges faster than Bisection & FPI.

To find a root of $f(x) = 0$, a starting guess (x_0) is given, and the tangent line to the function @ x_0 is drawn; the intersection (x-intercept) of this tangent line is assigned the value x_1; the function is iterated likewise.

Newton's Method



Pseudo-code for Newton's Method

x_0 = initial guess

$$\boxed{x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}} \quad \text{for } i = 0, 1, 2, \ldots$$

Derivation of NM's formula:

Given an initial guess x_0, we solve for the x-intercept of the tangent line passing through $(x_0, f(x_0))$.

Using point-slope form of a line:

$$y - y_0 = m(x - x_0) \rightarrow 0 - f(x_0) = f'(x_0)(x - x_0)$$

$$\rightarrow x - x_0 = -\frac{f(x_0)}{f'(x_0)}$$

$$\rightarrow \boxed{x = x_0 - \frac{f(x_0)}{f'(x_0)}}$$

In general, then, $\boxed{x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}}$

Ex. With $x_0 = -.7$, use NM's to approximate a root of $\boxed{x^3 + x - 1 = 0}$.

$$\boxed{f'(x) = 3x^2 + 1}$$

So, $x_{i+1} = x_i - \dfrac{f(x_i)}{f'(x_i)}$

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \approx \boxed{.1271}$$

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} \approx \boxed{.9577}$$

$$\vdots$$

$$\boxed{x_7 = .68232780}$$ ← correct to eight decimal places!

<u>Note:</u> As we see, the convergence of NM's is ~~qualitatively faster~~ than Bisection & FPI; in general NM's is $\boxed{\text{quadratically convergent.}}$

$\boxed{\text{Def.}}$ Let e_i denote the error after step i of an iterative method. The iteration is $\boxed{\text{quadratically convergent}}$ if:

$$\boxed{\lim_{i \to \infty} \frac{e_{i+1}}{e_i^2} = M < \infty}$$

$\boxed{\text{Theorem}}$: Let f be <u>twice continuously differentiable</u> & $f(r) = 0$. If $\boxed{f'(r) \neq 0}$, then NM is <u>locally</u> & <u>quadratically</u> convergent to r.

$\boxed{\text{Proof}}$ To prove local convergence, note that NM is a particular form of FPI where $\boxed{g(x) = x - \dfrac{f(x)}{f'(x)}}$

Then, $g'(x) = \dfrac{1 - f'(x)^2 - f(x)f''(x)}{f'(x)^2} = \dfrac{f(x)f''(x)}{f'(x)^2}$

$$|g'(r)| = \left|\frac{f(r)f''(r)}{f'(r)^2}\right| = 0 < 1, \text{ so NM is locally convergent.}$$

Now we prove quadratic convergence.

Using Taylor Series

$$f(r) = f(x_i) + (r - x_i)f'(x_i) + \frac{(r-x_i)^2}{2}f''(c_i) \quad (c_i \in (x_i, r))$$

$$\shortparallel$$

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

$$\rightarrow \frac{-f(x_i)}{f'(x_i)} = r - x_i + \frac{(r-x_i)^2}{2}\frac{f''(c_i)}{f'(x_i)}$$

$$\rightarrow \underbrace{x_i - \frac{f(x_i)}{f'(x_i)}} - r = \frac{(r-x_i)^2}{2}\frac{f''(c_i)}{f'(x_i)} \quad \left((r-x_i)^2 = e_i^2\right)$$

$$\rightarrow x_{i+1} - r = e_i^2 \cdot \frac{f''(c_i)}{2f'(x_i)}$$

$$\rightarrow e_{i+1} = e_i^2\left|\frac{f''(c_i)}{2f'(x_i)}\right|$$

Note: $\boxed{\lim_{i\to\infty} c_i \to r}$

$$\rightarrow \lim_{i\to\infty}\frac{e_{i+1}}{e_i^2} = \left|\frac{f''(r)}{2f'(r)}\right| = M < \infty \quad (f'(r) \neq 0)$$

In this fashion, the error can be expressed iteratively as:

$$e_{i+1} \approx M e_i^2 \quad \left(M = \left|\frac{f''(r)}{2f'(r)}\right|\right).$$

Note: NM's does not always converge quadratically!

The algorithm is sensitive to the choice of x_0; in addition, NM's converges only linearly at multiple roots!

for multiple roots (m):

$$\boxed{\lim_{i\to\infty} \frac{e_{i+1}}{e_i} = S = \frac{m-1}{m}}$$

Ex. $\boxed{f(x) = x^2}$; let $\boxed{x_0 = 1.}$

$$\to x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \to x_1 = 1 - \frac{f(1)}{f'(1)} = .5 \ldots$$

$x_2 = .25$

$x_3 = .125$

$\vdots$

$$\left(\frac{e_i}{e_{i-1}} = \frac{1}{2}\right)$$

Note that NM's will, in fact, converge to the multiple root: $r = 0$, but $e_{i+1} = \frac{e_i}{2}$, so the convergence is only linear in this case.

Ex. Find the multiplicity of the root $r = 0$ for $f(x) = \sin x + x^2 \cos x - x^2 - x$, and estimate the number of steps required of NM's to converge to r within six decimal places.

$f(0) = 0.$

$f'(x) = \cos x + 2x\cos x - x^2 \to f'(0) = 0$ ✓

$f''(x) = -\sin x + 2\cos x - 4x\sin x - x^2\cos x - 2 \quad f''(0) = 0.$ ✓

However, $f'''(0) \neq 0$. So $r = 0$ has $\boxed{\text{multiplicity } 3.}$ ($m = 3$)

Thus, by the previous result (above), $\boxed{e_{i+1} \approx \frac{2}{3} e_i}$

$2 \; (m-1)$, $3 \; (m)$

So, we solve:

$$\left(\frac{2}{3}\right)^n < .5 \times 10^{-6} \to n \approx 35.78 \to \boxed{n = 36}$$

approximately 36 steps!

Thm: If f is $(m+1)$-times differentiable on $[a,b]$, which contains root r with multiplicity $m>1$, then Modified Newton's Method:

Note

$$\boxed{x_{i+1} = x_i - m \cdot \frac{f(x_i)}{f'(x_i)}}$$

← The catch: we must know the mult. of r (m) previously.

converges locally & quadratically to r.

Ex. We revisit the previous example, using Modified Newton's Method.

$f(x) = \sin x + x^2 \cos x - x^2 - x$; $x_0 = 1$

$\vdots$

$x_4 = .00000006072272$

$\boxed{x_5 = -.00000000633250}$ → 8 digits of accuracy!

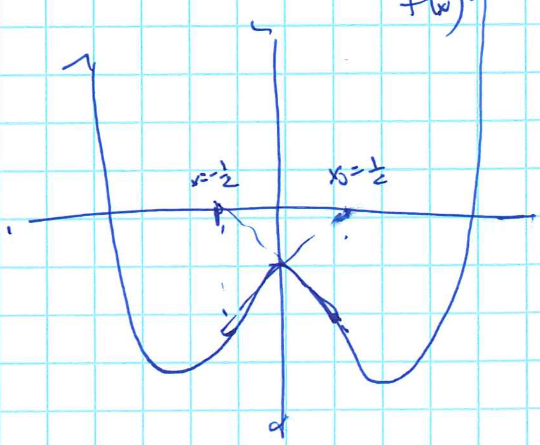
Also, note that using a table of values, we can observe quadratic convergence of modified NM's, since accuracy essentially doubles w/ each iteration.

Q: When can NM's fail? ① when $f'(r) = 0$ ② with poor choice of x_0 ③ For function with large $|f'(x)|$.

Ex. Apply NM's: $f(x) = 4x^4 - 6x^2 - 11/4$; $x_0 = -\frac{1}{2}$

$$x_{i+1} = x_i - \frac{4x_i^4 - 6x_i^2 - \frac{11}{4}}{16x_i^3 - 12x_i}$$

Picture →



$x_1 = -\frac{1}{2}$
$x_2 = \frac{1}{2}$
$x_3 = -\frac{1}{2}$
$x_4 = \frac{1}{2}$
$\vdots$

} infinite oscillations!