Problem: Find a root of a smooth function

$$f(x) = 0. \qquad f: \mathbb{R} \to \mathbb{R}$$

Two methods (so far):
- Interval Bisection.
- Fixed-Point Iteration

Interval Bisection.:

Good:
- guaranteed convergence (other methods don't have such guarantees)
- rate of convergence independent of function (can determine # iterations required for a given abs error tolerance).
- only requires evaluate $f(x)$. (other algos might require derivative $f'(x)$).
- easy to implement.

Bad:
- converges slowly
- does not make use of much information about f, only 1 bit per iteration (sign)
- requires an initial bracket, which can be hard to obtain.

Fixed-Point Iteration

Good:
- can converge faster (which we observed).
- only requires an initial estimate

Bad:
- might not converge or if $|g'(x)| \approx 1$. then convergence is slow.
- requires a definition of $g(x)$ ~ harder to automate.

eg// $f(x) = x^3 - x - 1$ $\quad$ $g(x) = x^3 - 1$ diverges.

$g(x) = (x+1)^{1/3}$ works.

Convergence of Fixed-Point Iteration

Suppose $g(x)$ has a fixed-point $x^* = \underbrace{g(x^*)}_{**}$.
If $|g'(x^*)| < 1$, then there is an interval containing x^* such that if x_0 is in that interval, then fixed-point iteration $x_{k+1} = \underbrace{g(x_k)}_{*}$ converges to x^*.

Proof (sketch)

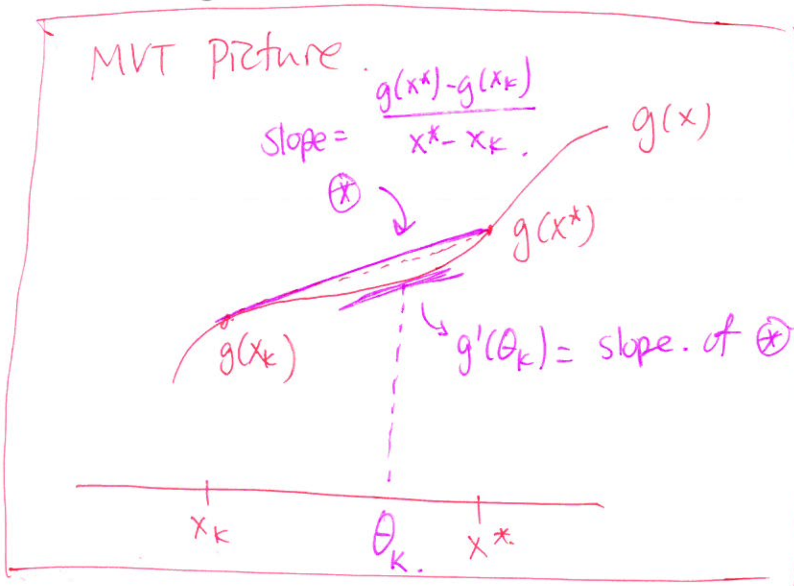
Suppose that x^* is a fixed-point of $g(x)$. Then,

$$e_{k+1} = \underbrace{x_{k+1}}_{\substack{\text{prediction} \\ \text{at } k+1 \text{ iter.}}} - \underbrace{x^*}_{\text{actual}} = \underbrace{g(x_k)}_{*} - \underbrace{g(x^*)}_{**}$$

Since $g(x)$ is smooth, we can apply the mean-value Thm:

There exists θ_k between x_k and x^* such that

$$\frac{\overbrace{g(x_k) - g(x^*)}^{= e_{k+1}}}{\underbrace{x_k - x^*}_{= e_k}} = g'(\theta_k)$$



$$\Rightarrow \quad \frac{e_{k+1}}{e_k} = g'(\theta_k)$$

If $|g'(x^*)| < 1$, then if we start x_0 close to x^* then there exists a constant $C \in \mathbb{R}$ such that

$$|g'(\theta_k)| \le C < 1.$$

which means that

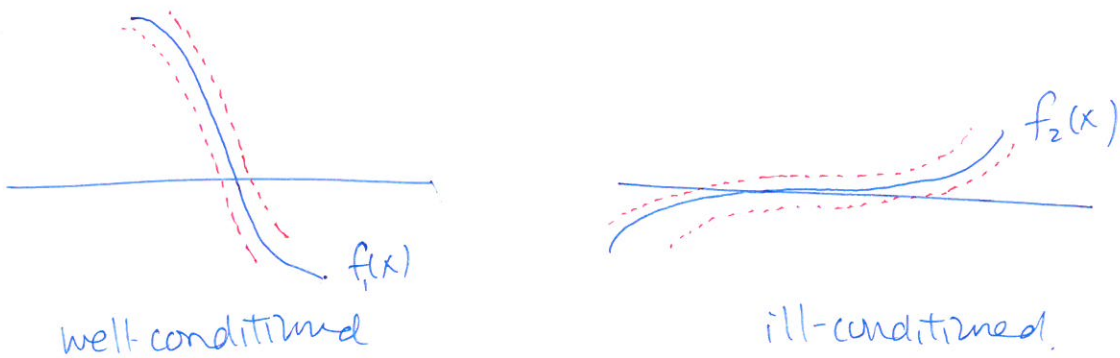
$$\lim_{k\to 0} \frac{|e_{k+1}|}{|e_k|} = \lim_{x\to 0} |g'(\theta_k)|. \le c.$$

So. $|e_{k+1}| \le c|e_k| \le \dots \le c^k |e_0|$.

Since $c<1$, $|e_k| \to 0$ as $k\to\infty$.

Sensitivity and Conditioning

Q: For which $f(x)$ is root-finding $(f(x)=0)$ well conditioned?



Sensitivity of root-finding problem is opposite to that of evaluating the function $f(x)$. $f^{-1}(0)$.

To measure sensitivity, we must use an absolute condition number because $f(x)=0$.

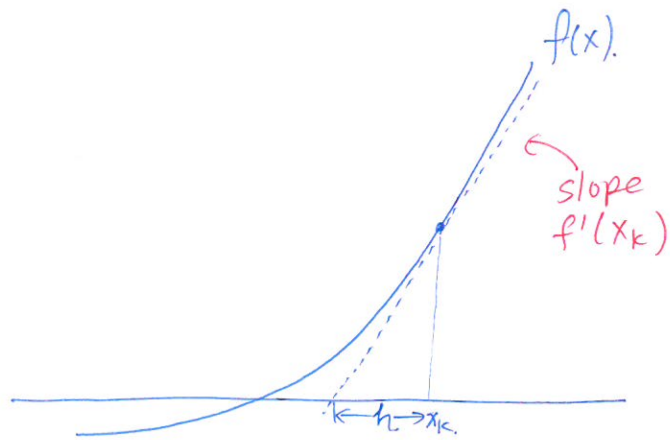
The absolute C.N. for evaluating a function f near x^* is $\left(\frac{\Delta y}{\Delta x}\approx\right)$ $|f'(x^*)|$.

$\Rightarrow$ $\boxed{\text{Root finding problem has absolute C.N. } \left|\frac{1}{f'(x^*)}\right|}$

$\Rightarrow$ If $f'(x^*)=0$ C.N. is infinite.

Newton's method

Idea: Approximate $f(x)$ with a linear function, and find a root of the linear function.



We use the Taylor series approximation of $f(x)$: $\underbrace{f(x_k+h) \approx f(x_k) + f'(x_k)\cdot h}_{\text{linear approx to } f(x)}$.

Set $0 = f(x_k) + f'(x_k)\cdot h$

$$h = -\frac{f(x_k)}{f'(x_k)}$$

$$\boxed{\text{So } x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}}$$

Example $f(x) = x - 0.2\sin(x) - 0.5$

k	x_k	$\|x_k - x_{k-1}\|$		e_k	
0	0.0 init guess	6.3×10^{-1}			
1	0.625	9.5×10^{-3}		9.5×10^{-3}	
2	0.6154745	6.3×10^{-6}	$\times 2$	6.3×10^{-6}	$\times 2$
3	0.6154682	2.8×10^{-12}	$\times 2$	2.7×10^{-12}	$\times 2$.
4	0.6154682				

Summary: e_k decreases by a square factor each step
$\Rightarrow$ quadratic conv.

Newton's method is equivalent to finding a fixed-point of $g(x) = x - \frac{f(x)}{f'(x)}$.

To analyze the convergence of $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$,

we look at
$$g'(x) = \cancel{1} - \frac{\cancel{(f'(x))^2} - f(x)\, f''(x)}{(f'(x))^2}$$
$$= \frac{f(x)\, f''(x)}{(f'(x))^2}.$$

So $|g'(x)| < 1$ if $\left| \frac{f(x) f''(x)}{(f'(x))^2} \right| < 1$.

So Newton method converges when
$$\left| \frac{f(x)\, f''(x)}{(f'(x))^2} \right| < 1$$

can't be too large

can't be too small.

To measure the rate of conv.

the rate of convergence r of the sequence e_k.

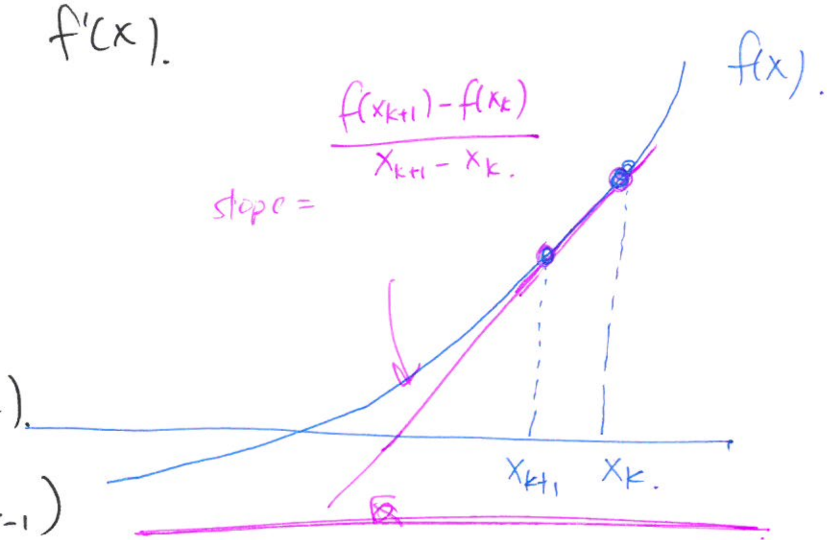
is the largest value r such that
$$\lim_{k \to \infty} \frac{|e_{k+1}|}{|e_k|^r} = c \neq 0.$$

Can show that for Newton's method $r=2$.

Secant Method

One drawback of Newton's method is that we need both the function $f(x)$ and its derivative $f'(x)$.

Idea: ~~Base~~ Use the finite difference approximation to the derivative on successive iterates (x_k).



$$f'(x) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$$

→ i.e. use finite difference to approximate the derivative $f'(x_k)$

Algo: Start with x_0, x_1.

then $$x_{k+1} = x_k - f(x_k) \cdot \frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}$$

Can show that conv. of secant method is superlinear. $(r = 1.618)$

Secant method is fast like Newton's method, requires no derivatives.

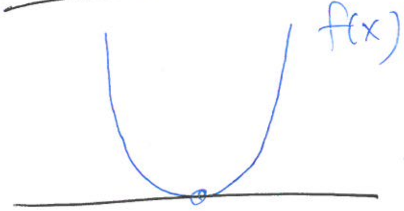
eg// $f(x) = x - \tan(x)$.

$f(3.5) = +3.13$
$f(4.5) = -0.137$ } should be a root in $[3.5, 4.5]$

$$f'(x) = 1 - \sec^2(x) = -\tan^2(x).$$

k	x_k
0	4.0
1	6.12
2	238
3	1978 ...

Multiple Roots



In this picture, $f'(x^*) = 0$.

The problem $f(x) = 0$ is ill-conditioned.

Def'n A solution x^* of $f(x) = 0$ is called a <u>root of multiplicity m</u> if we can factor $f(x)$ into $f(x) = (x - x^*)^m \cdot q(x)$.

where $q(x^*) \neq 0$.

Def'n If $m = 1$, x^* is called a <u>simple root</u>.

$\Rightarrow$ Newton's method will have a harder time finding multiple roots. (non-simple roots)

A Robust Algorithm

- Seen many algos, which to implement?
- Combine a reliable but slow method like bisection with a fast but not always reliable method like secant method.