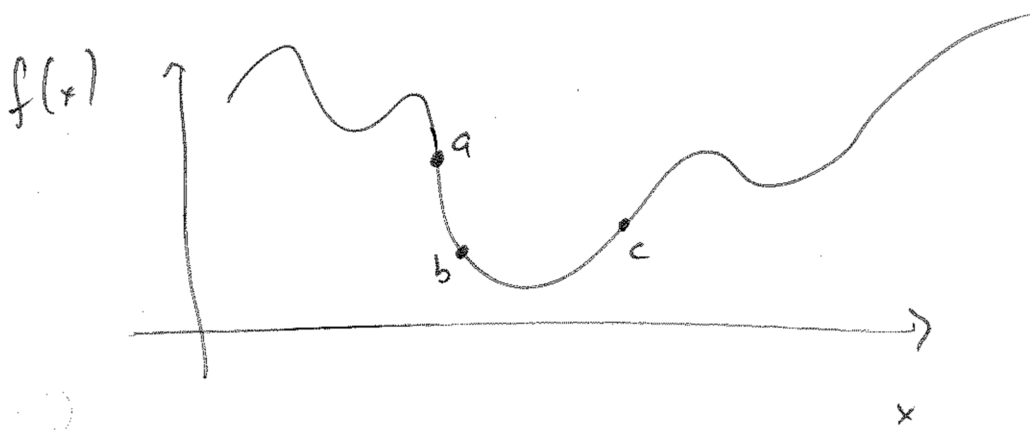
MINIMIZATION :

Another common problem in physics is the search for function minima, such as search for lowest energy stable state (eg protein folding problem).
We will address the simplest case of 1-D fctn minimization. Consider:



Given some arbitrary fctn, how to search?
Search for $\emptyset$'s of $f'(x)$? Problems...

- only tells you extrema
- may not know $f'(x)$
- non-analytic fctn

If you can bracket a minimum, a fool-proof method to locate the minimum is the <u>Golden Section Search</u>.

First, how many points to bracket a minimum? <u>3</u>.

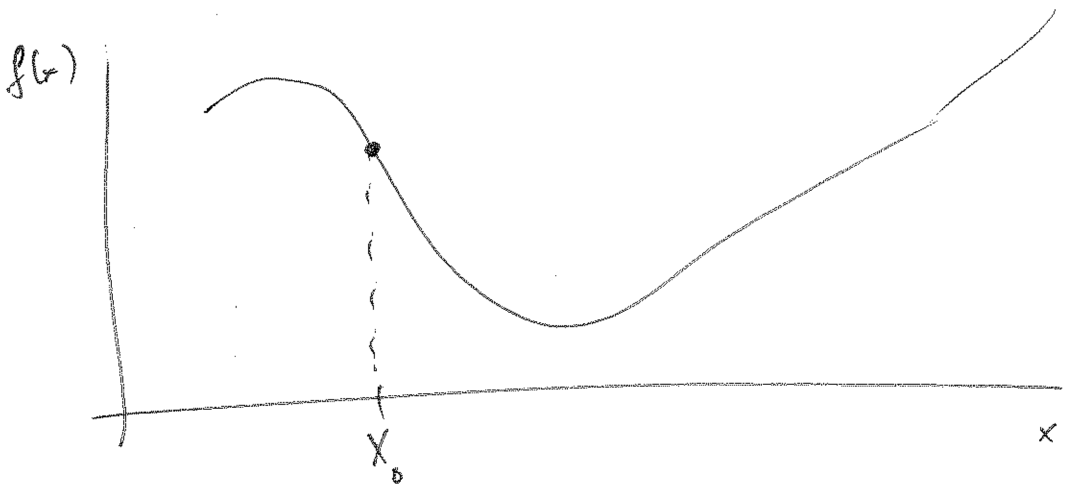
$$(a, b, c) \quad \& \quad \left.\begin{array}{l} f(b) < f(a) \\ f(b) < f(c) \end{array}\right\}$$ ~~guar~~ garauntees minimum on (a,b)

<u>NB</u>: not a global minimum!!

Minimum Bracket:

The golden section seach assumes we have ~~already~~ already bracketed to a triplet (a, b, c). How can we obtain this?

Here is one simple crude method:



~~Step~~ Start from an arbitrary point x_0. We want to step towards the minimum in steps Δ. We know we have (means downhill) bracket when $f(a) > f(b)$
$f(c) > f(b)$.

① Determine direction: If $f(x_0+\Delta) < f(x_0)$, step forward.
$f(x_0+\Delta) > f(x_0)$, step backward

② Pick candidate points $a = x_0$ $b = x_0+\Delta$ $c = x_0+2\Delta$

③ Check if $f(b) < f(a)$ & $f(b) < f(c)$.

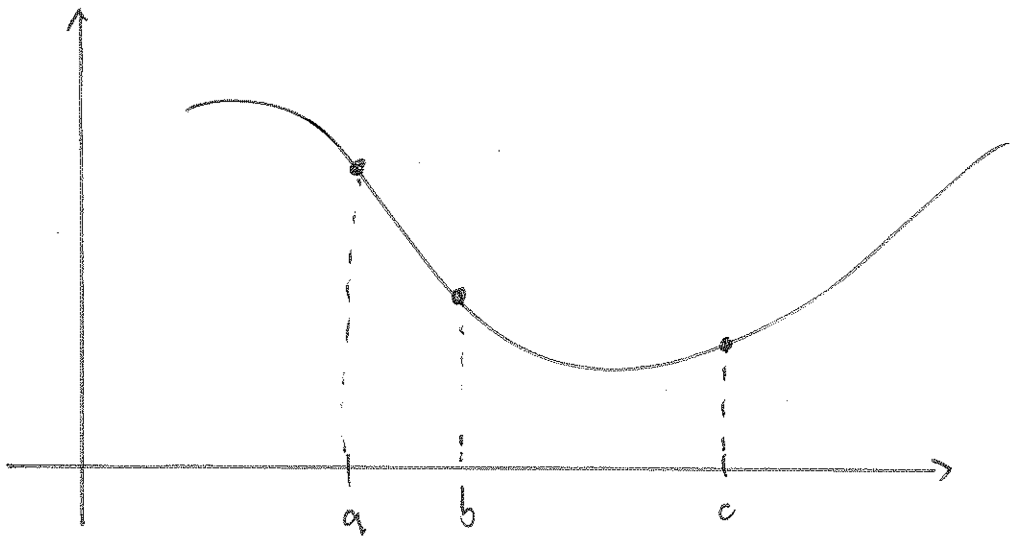
If yes, done!

If no, go to next step, $a = x_0+\Delta$
$b = x_0+2\Delta$
$c = x_0+3\Delta$ etc...

NB: If Δ too large, step past minimum!!

GOLDEN SECTION SEARCH:

Given points (a, b, c) that bracket a minimum, we can refine our estimate via a process very similar to the bisection method of root finding.



Given (a, b, c), we ~~otherwant~~ want to choose x (new point) that improves our estimate.

~~If b were midp~~

Consider a bisection approach:

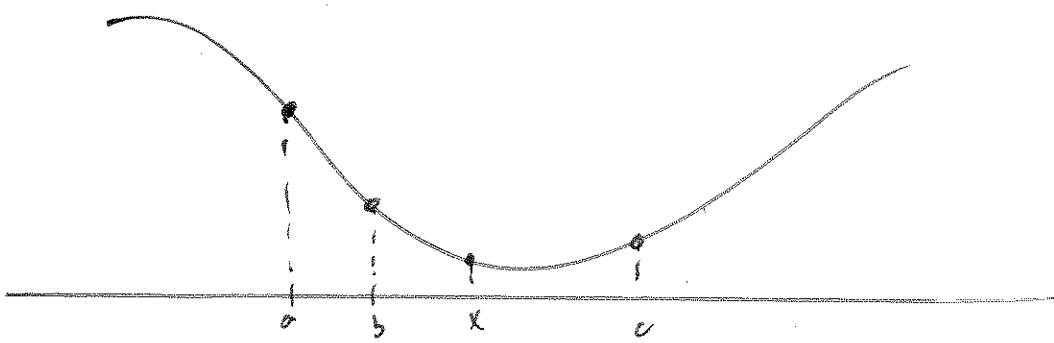
$$b = \frac{a+c}{2} \; ; \quad x = \frac{a+b}{2} \text{ or } \frac{b+c}{2} \; ; \text{ new interval either } (a\,b\,x) \text{ or } (b\,x\,c)$$

$\Rightarrow$ new interval is either $1/4$ or $3/4$ the size of ~~thee~~ the previous interval. This is non-optimal!

Hence we do not want to bisect --- we want the "golden section"

$\Rightarrow$ Choose x so that whichever interal we pick, it will have the same size!

$b-a \neq c-b$



We want x ~~sho~~ so that $x-a = c-b$

Clearly, put x in larger sub-interval. How to determine x?
If we assume (a, b, c) are chosen with the optimal ratio, define

$$\frac{b-a}{c-a} = g \qquad \frac{c-b}{c-a} = (1-g) \qquad g \equiv \text{"golden ratio"}$$

$$\& \quad x-a = c-b$$

If x chosen correctly;

$$\frac{x-b}{c-b} = g = \frac{x-b}{c-a}\underbrace{\boxed{\frac{c-a}{c-b}}}_{=\frac{1}{1-g}}$$

substitute $x = a+c-b$

$$= \frac{1}{1-g}\,\frac{a+c-b-b}{c-a}$$

$$= \frac{1}{1-g}\left[\underbrace{\frac{a-b}{c-a}}_{=-g} + \underbrace{\frac{c-b}{c-a}}_{=1-g}\right]$$

$$= 1 - \frac{g}{1-g}$$

$$\Rightarrow \frac{g}{1-g} = 1-g \quad \Rightarrow \quad g = (1-g)^2 \quad \Rightarrow \quad g^2 - 3g + 1 = 0$$

$$\Rightarrow \boxed{g = \frac{3-\sqrt{5}}{2} = 0.38197\ldots}$$

So now we can define our algorithm:

(1) Given (a, b, c) bracketing the minimum, choose the larger of A) $(b-a)$ or B) $(c-b)$

(2) In the larger interval, choose a new point X, either

A) $(b-a) > (c-b) \rightsquigarrow \quad x = \cancel{a+g(b-a)} = b - g(b-a)$

B) $(c-b) > (b-a) \rightsquigarrow \quad x = b + g(c-b)$

(3) Evaluate $f(x)$

A) If $f(x) < f(b)$: min on (a, x, b)

else min on (x, b, c)

B) If $f(x) < f(b)$: min on (b, x, c)

(a, b, x)

(4) Repeat until $|c-a| <$ tolerance;

return either b or midpoint of (C, a)

<u>NB</u>: even if (a,b,c) not originally optimal (golden) subsequent choices will be!

How small can we make tolerance?

As small as machine precision ϵ?

Consider expansion near root r

$$f(x) \approx f(r) + 0 + \frac{1}{2} f''(x)(x-r)^2 + \mathcal{O}(x-r)^3$$

$$\Rightarrow \quad f(x) - f(r) \approx \frac{1}{2} f''(r)\, \delta^2$$ ~~if $(x-r)$ = machine precision~~

Since we must compare $f(x)$ & $f(r)$, $f(x) - f(r) \geq \epsilon$

$$\Rightarrow \quad \epsilon \leq \frac{1}{2} f''(r)\, \delta^2$$

$$\Rightarrow \quad \delta \geq \sqrt{\frac{2\epsilon}{f''(r)}}$$

Since $f''(r)$ typically ≈ 1, it does not make sense for $\delta < \epsilon^{1/2}$. For a 64-bit double, we have 16-digits of ~~precis~~ precion, so $\frac{\epsilon}{f(x)} = 10^{-16}$

$$\Rightarrow \quad \underline{\delta \geq 10^{-8} f(x)}$$