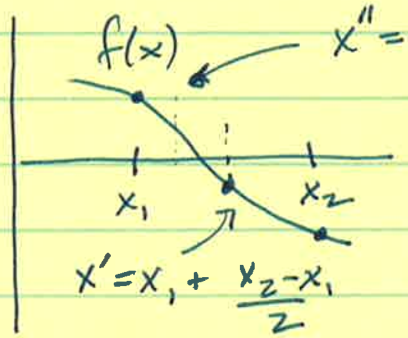
Nonlinear Equations

python nonlinear

① Root Finding

for example: binary search method



$x'' = x_1 + \frac{x' - x_1}{2}$

& so on

② Finding Minimums

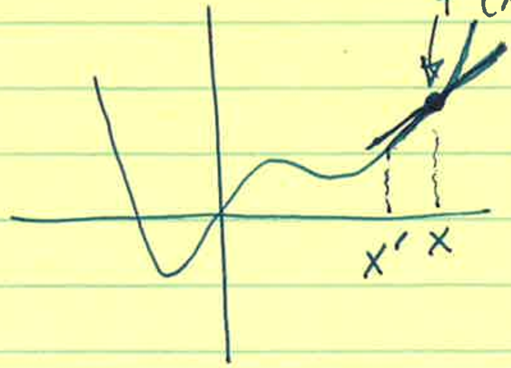
Could proceed with a binary search

or

Gauss-Newton method & gradient descent

$$x' = x - \underbrace{\gamma}_{\text{some constant}} f'(x)$$

$f'(x)$ is positive ~~negative~~



One might approximate

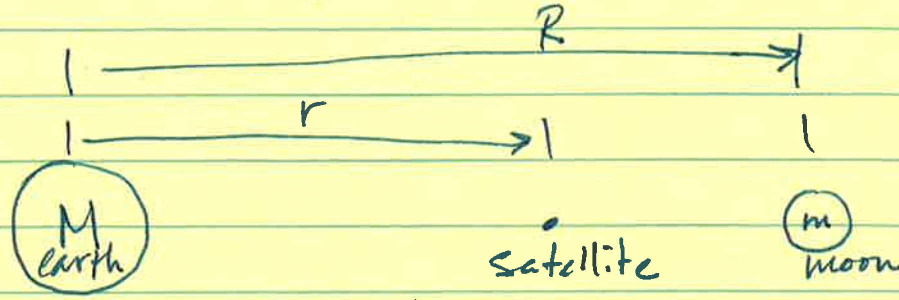
$$f'(x) \cong \frac{f(x_2) - f(x_1)}{x_2 - x_1}$$

You may find a local minimum instead of the global minimum

Solution to Nonlinear Equations

python
nonlinear
-a

Exercise 6.16 Lagrange point



- satellite stays between earth & moon in orbit

$$\frac{GM}{r^2} - \frac{Gm}{(R-r)^2} = \omega^2 r$$

$$\frac{GM}{r^2} - \frac{Gm}{(R-r)^2} - \omega^2 r = 0 \qquad \text{find } r$$

Ex6.16.py

Have class find earth sun lagrange point for solar array

Exercise 6.18 Optimum temperature of light bulb

intensity of radiation of body at temperature T

$$I(\lambda) = 2\pi A h c^2 \, \lambda^{-5} / (e^{hc/\lambda k_B T} - 1)$$

python
nonlinear
-b

λ wavelength
A area of filament
h Planck's constant
c speed of light
k_B Boltzmann constant

But most radiation is infrared or ultraviolet & not visible

Visible wavelengths are between 390nm (λ_1) & 750nm (λ_2)

So visible energy is $\int_{390nm}^{750nm} I(\lambda)\,d\lambda$

& total energy is $\int_0^\infty I(\lambda)\,d\lambda$

So efficiency = $\eta = \dfrac{\int_{390nm}^{750nm} I(\lambda)\,d\lambda}{\int_0^\infty I(\lambda)\,d\lambda}$

At what temperature is η maximized?
with $x = hc/\lambda k_B T$

$$\eta = \frac{\int_{hc/\lambda_2 k_B T}^{hc/\lambda_1 k_B T} \frac{x^3}{e^x - 1}\,dx}{\int_0^\infty \frac{x^3}{e^x - 1}\,dx}$$

python nonlinear-c

but denominator can be evaluated

$$\eta = \frac{15}{\pi^4} \int_{hc/\lambda_2 k_B T}^{hc/\lambda_1 k_B T} \frac{x^3}{e^x - 1}\, dx$$

Goal: maximize eta with respect to T

Show Ex6.18.py

optimum T = 6640K   problem: Tungsten melts at $\underline{3695\,K}$

note: sensitivity to starting point

Example: Multivariable minimization

$$f(x,y) = \left(4 - 2.1x^2 + \frac{x^4}{3}\right)x^2 + xy + (4y^2 - 4)y^2$$

treat 2 variables as components of 1 vector

i.e. x[0] = x

x[1] = y

show scipy_optimize_sixhump.py

note: with fmin_bfgs the minimum you find depends on starting point

with basinhopping a number of random starting points are tried