A final note on solving non-linear systems

Homotopy (morphing) in root-finding

Often hard to find a good enough starting point $x^{(0)}$ for Newton or quasi-Newton because the "basin of attraction" of the root is very small.

How to come up with a good $x^{(0)}$?

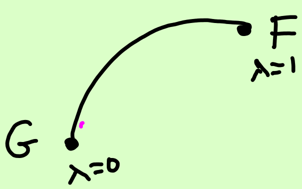
A powerful idea is to consider a family of systems $H: \mathbb{R}^n \times [0,1] \to \mathbb{R}^n$
$H(x,\lambda)$

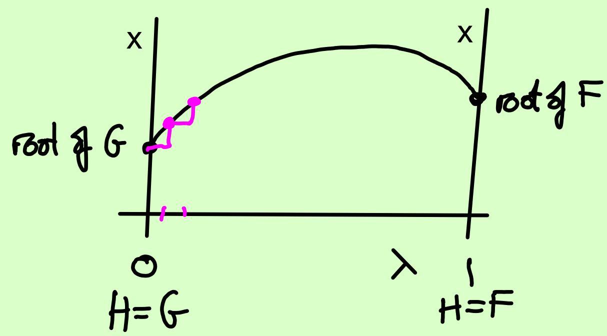
where $H(x,0) = G(x)$, $H(x,1) = F(x)$

some other "nicer, easier" function

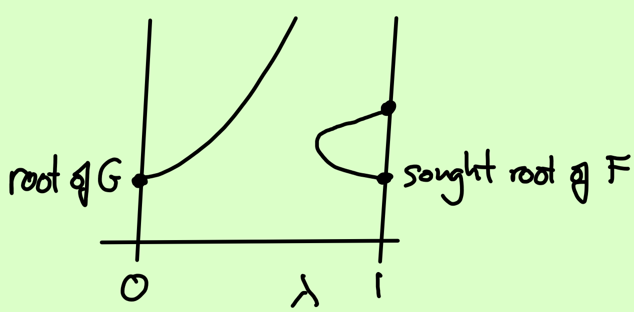
the function whose root we want

Then what we do is start with an easily obtained root of G, and then gradually "morph" G into F,
incrementing the parameter λ in $H(x,\lambda)$,
tracking the root as we go, using the root on the previous value of λ
as the starting x(0) for the current value of λ ,
until finally at $\lambda = 1$ we have a root of F.





Choosing a suitable homotopy is not necessarily easy.
A homotopy can easily fail, such as when the root of G and the sought root of F are not connected by a curve, maybe like this:



A lazy choice of homotopy, like G(x) = x - r (really easy to find a root (it's r!)) and $H(x,\lambda) = (1-\lambda)G(x) + \lambda F(x)$
is almost certain to fail.

Knowledge about the particular problem at hand will be useful in constructing a successful homotopy, such as in the toy suspension bridge system of Project Option 3. For example, a good approach might be to let G be a version of F with different, and particularly simplifying, parameter values, and let the homotopy be a linear ramp from the simplifying parameter values to the values you're actually interested in.

Ch. 9 Optimization

Given a scalar-valued function $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$

find a minimizer of f on D, that is find $x^* \in D$ s.t. $f(x^*) \leq f(x)\ \forall x \in D$.

or find a local minimizer, that is find $x^* \in D$ s.t $f(x^*) \leq f(x)\ \forall x \in$ some n'hood of x^*.

If you want to maximize some function g, then define f = -g and minimize f.

Simplest case is n=1: $f : \mathbb{R} \to \mathbb{R}$ Ex:

minimize negative goodness of toast

scalar variable x

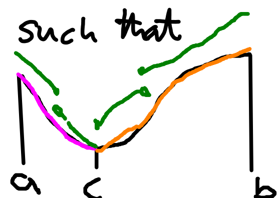
Ex: finding the "best" Bezier approximation to a quarter-circle in Homework 6 Q7.

This problem is reminiscent of 1D root-finding, but a little trickier.

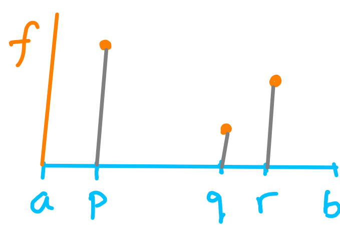
Recall the bisection method for root-finding, where we had a "bracket" [a,b] of the sought root: sign(f(a)) != sign(f(b)) guaranteeing a root in [a,b] if f is continuous, by IVT theorem. Idea was to shrink the bracket repeatedly until narrower than our error tolerance.

Can we do something similar for minimization?

Let's say that f is "unimodal" on [a,b] if $\exists\, c \in (a,b)$ such that
f is strictly decreasing on [a,c]
f is strictly increasing on [c,b].
Then c is the unique minimizer of f on [a,b].



As analog of the "bracket" in root-finding, for a function f, let's define a "vee" as a triple of points (p,q,r) such that p,q,r in [a,b] and f(p) > f(q) and f(r) > f(q).

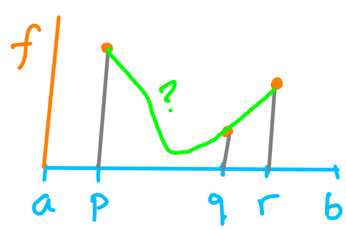


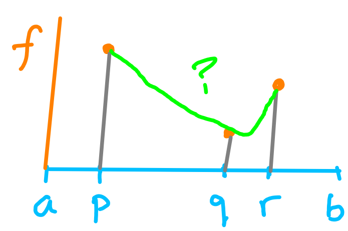
Then if f is unimodal on [a,b], we are guaranteed that

$c \in (p, r)$.

(Why?)

ni

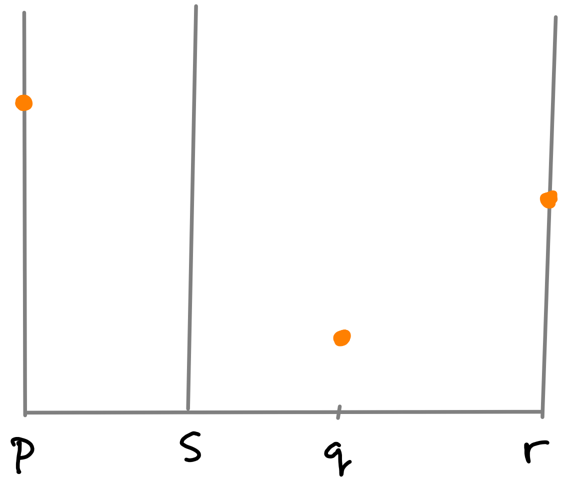




The idea, analogously to bisection, is to repeatedly shrink the "vee", such that the width "|r-p|" of the vee goes to zero.

How to do that?

Let's pick a point $s \in (p, r)$, $s \neq q$



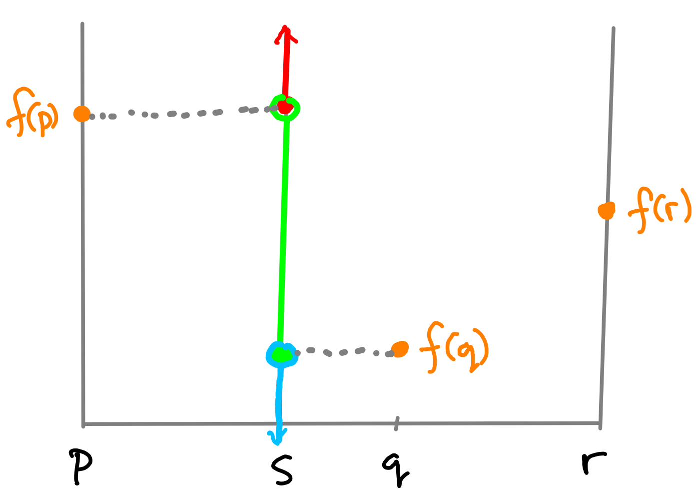
Your ideas on what to pick for s?

Idea 1: (p+r)/2

Idea 2: midpoint of one half, alternating

Idea 3: midpoint of larger subinterval

Regardless of the precise choice of s, suppose its between p and r,
and consider the possible outcomes for f(s):



If $f(s) \in$,

wait! Can't happen. Contradicts unimodality.

If $f(s) \in$

a new smaller vee is (s, q, r)

If $f(s) \in$

a new smaller vee is (p, s, q).

If $f(s) =$ $= f(q)$

a new smaller vee is (s, t, q)

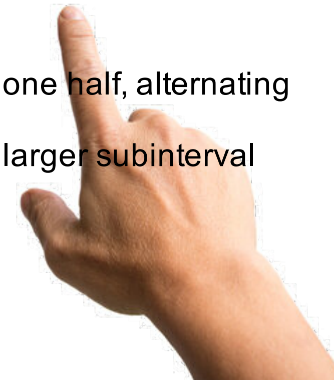
where t is anywhere between s & q.

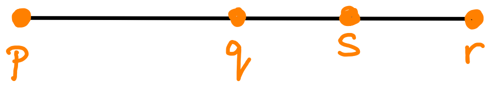
Let's find out what the guaranteed vee-width reduction is for your various ideas on choosing s.

Idea 1: (p+r)/2

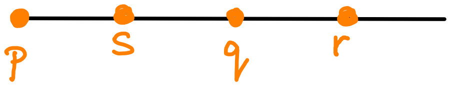
Idea 2: midpoint of one half, alternating

Idea 3: midpoint of larger subinterval





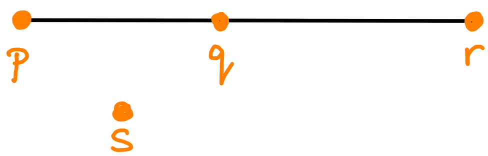
The new vee is
either p,q,s reducing by 3/4
or q,s,r reducing by 1/2

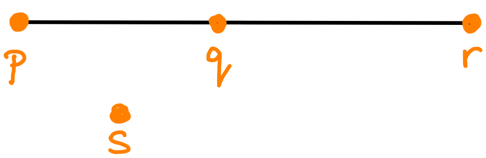


The new vee will be
either p,s,q reducing by 2/3
or s,q,r reducing by 2/3

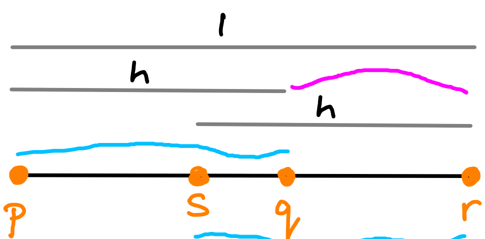
Best possible outcome would be to alternate 1/2 followed by a 2/3.
Average per-step reduction is sqrt(1/2 . 2/3) = sqrt(1/3) ~ .577
which is quite good - almost as good as bisection for root finding.
Worst case is sqrt(2/3 . 3/4) = sqrt(1/2) ~ .7

best guarantee





Is there a choice for s where we are assured a specific reduction on each step?
Maybe if we find a way to preserve the geometry,
regardless of which of the 2 new vees we are forced to choose?



Is it possible to choose h such that the 2 possible vee-widths are the same?

We would need $\frac{h}{1} = \frac{1-h}{h} \rightarrow h^2 + h - 1 = 0$

$$h = \frac{-1 \pm \sqrt{1^2 - (-4)}}{2} = \frac{-1+\sqrt{5}}{2} = \text{golden mean / golden ratio} = .618\ldots$$

This is called "golden mean search" or "golden section search".

Start by setting q = (1-h)p + hr, that is q is a fraction h of the way from p to r.

With this scheme we are assured a width reduction of ~.618 at each step.

(Almost as good as bisection in root-finding.)

A caution about precision

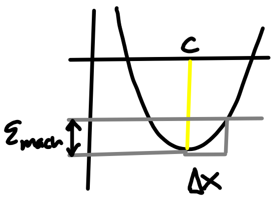
We cannot demand nearly as high precision in optimization as in root-finding.

Recall in root-finding it's ok to set "tol" $\sim 10\,\varepsilon_{mach} \sim 10^{-15}$.

$$\frac{a(x-c)^2+b-b}{b} = \frac{a}{b}(x-c)^2$$

$\overset{\text{set}}{\equiv} \varepsilon_{mach}$: $\Delta x = \sqrt{\frac{b}{a}}\sqrt{\varepsilon_{mach}}$

But for minimization, typically functions have quadratic minima,
and near c, f changes hardly at all as x changes.



In fact, if $f(x) \approx a(x-c)^2 + b$,

then if $\dfrac{f(x)-f(c)}{f(c)} = \varepsilon_{mach}$, $\left|\dfrac{\Delta x}{c}\right| = \sqrt{\dfrac{b}{ac^2}}\sqrt{\varepsilon_{mach}}$

or absolute $|\Delta x| = \sqrt{\dfrac{b}{a}}\sqrt{\varepsilon_{mach}}$.

So our limit of precision in finding the minimum is

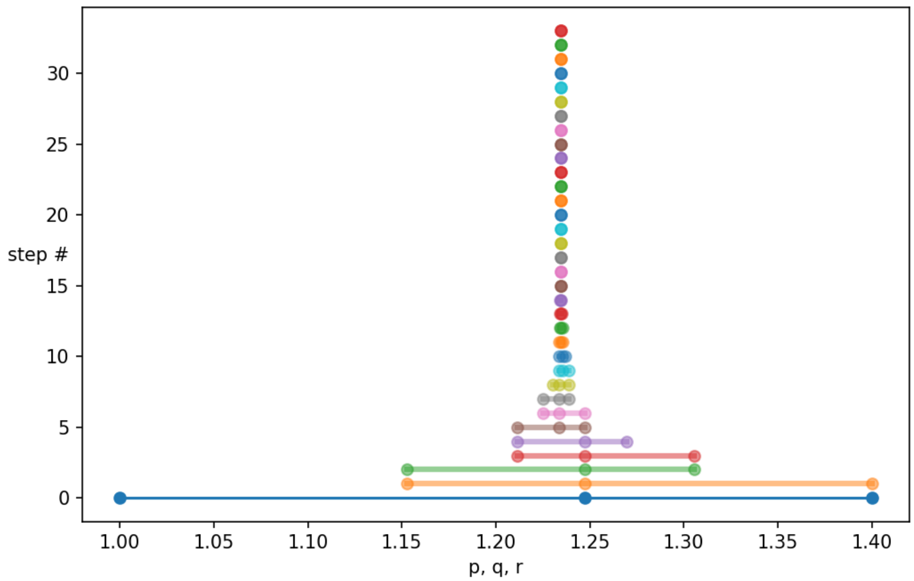
$$\sim \sqrt{\varepsilon_{mach}} \sim 10^{-8}, \text{ HUGE compared to } \varepsilon_{mach}.$$

Don't ask for more!

A quick implementation of golden section search ...

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib notebook
amp = 100

def golden(f,p,r,tol):

    h = (-1 + np.sqrt(5))/2
    q = (1-h)*p + h*r

    # verify that p,q,r is a vee
    assert( p<r and f(p)>f(q) and f(r)>f(q) )

    count = 0
    plt.plot((p,q,r),count*np.ones(3),'o-')
    while r-p > tol:
        s = p + r - q
        fs,fq = f(s),f(q)
        if s<q:
            if fs<fq:
                q,r = s,q
            elif fs>fq:
                p = s
            else: #fs==fq
                p,r = s,q
                q = (1-h)*p + h*r
        else: #s>q:
            if fs<fq:
                p,q = q,s
            elif fs>fq:
                r = s
            else: #fs==fq
                p,r = q,s
                q = (1-h)*p + h*r
        count += 1
        plt.plot([p,q,r], count*np.ones(3),'o-',color=f'C{count}',lw = 3,alpha=0.5)
    return p,r


def myf(x): return (x-1.23456789)**2 + 5   # an example function with a quadratic minimum

p,r = 1,1.4
x = np.linspace(p,r,200)
plt.figure(figsize=(8,5))
golden(myf,1,1.4,1.e-8)
```

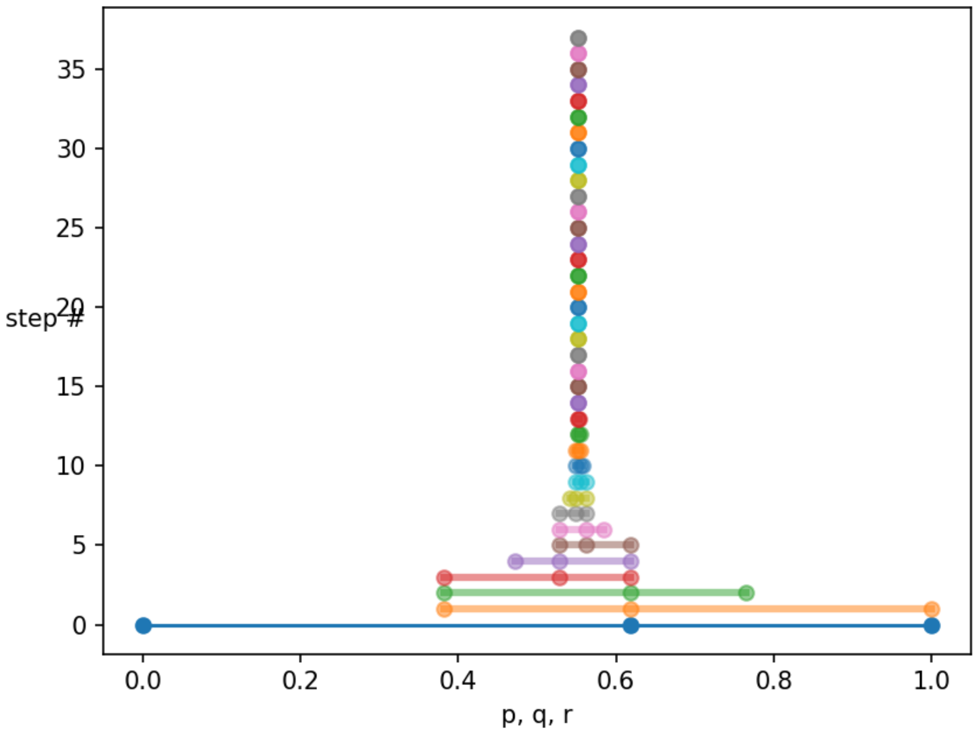
[space for results]



Now let's do Homework 6 Q7 (Bezier approx to quarter-circle) properly!

!

```
def bezier(P,t):  # columns of P are P0, P1, P2, P3
    P0,P1,P2,P3 = P.T
    s = 1 - t
    x =  s**3*P0[0] + 3*s**2*t*P1[0] + 3*s*t**2*P2[0] + t**3*P3[0]
    y =  s**3*P0[1] + 3*s**2*t*P1[1] + 3*s*t**2*P2[1] + t**3*P3[1]
    return x,y

def deviation_from_circle(q):
    P = np.array([[1,1,q,0],
                  [0,q,1,1]]) # columns of P are P0, P1, P2, P3
    t = np.linspace(0,1,500)
    x,y = bezier(P,t)
    return np.abs(x**2 + y**2 - 1).max()  # maximum deviation from circle

golden(deviation_from_circle,0,1,2e-8)
```




```
(0.5519149498173181, 0.5519149696417678)
```

Faster methods for smoother f

So far, we've depended only on f being unimodal. No smoothness (or even continuity) required.

We can do better (faster) if f has some smoothness.

For example, recall that Newton's method converges quadratically to a root of g if $g \in C^2$.

Now we are seeking a minimizer of f. If $f \in C^3$ then $g \equiv f' \in C^2$ and a local minimizer of f is a root of g to which Newton will converge quadratically:

$$x^{(k+1)} = x^{(k)} - \frac{f'(x^{(k)})}{f''(x^{(k)})}.$$

Another idea that doesn't depend on quite so much smoothness is **successive quadratic interpolation**.

```
Next class, we'll explore optimization with respect to a vector variable ( n > 1 ),
(which is a huge area).
```