12

Optimization

Let's try to weaken the requirement $f(\vec{x}) = \vec{0}$. $(f: \mathbb{R}^n \to \mathbb{R}^n)$

Idea: minimize $\|f(x)\|_2$

But: Is the norm really necessary?

Create a problem statement for "optimization".

$f: \mathbb{R}^n \longrightarrow \mathbb{R}$ not $\mathbb{R}^n$

called the "<u>objective function</u>"

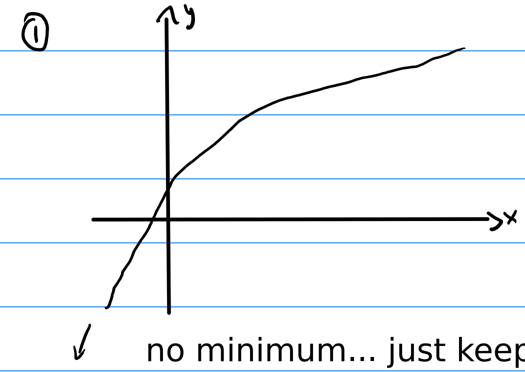
Find $\vec{x}$ so that $f(\vec{x})$ assumes the smallest possible value.

What if I'm interested in the largest possible value of a function g instead?

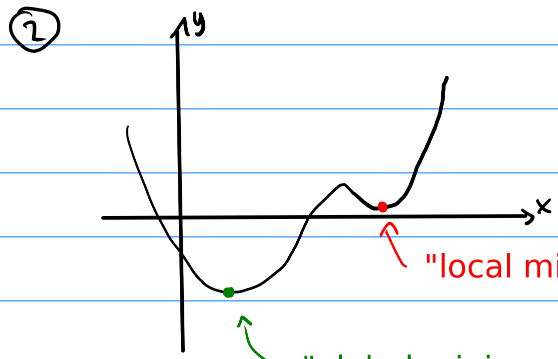
Consider $-g(x) = f(x)$

max of g = min of f

What could go wrong?



no minimum... just keeps going



How can we tell if we've got a (local) minimum in 1D? Remember calculus!

necessary condition: $f'(x) = 0$

sufficient condition: $f'(x) = 0$ and $f''(x) > 0$

And in n dimensions?

a vector -- the "gradient"

necessary condition: $\nabla f(x) = 0$

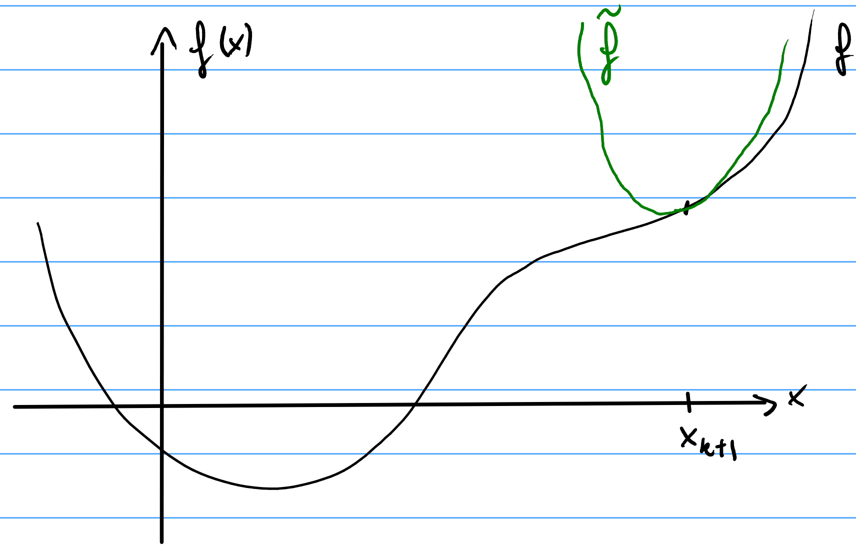
sufficient condition: $\nabla f(x) = 0$ and $H_f(x)$ positive definite

$$H_f(x) = \begin{pmatrix} \frac{\partial^2}{\partial x_1 \partial x_1} f & \cdots & \frac{\partial^2}{\partial x_1 \partial x_n} f \\ \vdots & & \vdots \\ \frac{\partial^2}{\partial x_n \partial x_1} f & \cdots & \frac{\partial^2}{\partial x_n \partial x_n} f \end{pmatrix}$$

Hessian matrix

Let's steal the idea from Newton's method for equation solving.

Build a simple version of f and minimize that. Let's try in 1D first.



Does a linear approximation (a line) help at all?

No, a linear function has no minimum.

(Other than maybe "at infinity". But that's not helpful.)

So: need at least a quadratic function.

from Taylor's theorem

$$\tilde{f}(x+h) = f(x) + f'(x)h + f''(x)\frac{h^2}{2} \approx f(x+h)$$

Now minimize that.

$$\frac{\partial}{\partial h}\tilde{f}(x+h) = 0 \quad \rightsquigarrow \quad \frac{\partial}{\partial h}\tilde{f}(x+h) = f'(x) + f''(x)h$$

$$\rightsquigarrow \quad -f'(x) = f''(x)h$$

$$\rightsquigarrow \quad h = -\frac{f'(x)}{f''(x)}$$

$$\rightsquigarrow \quad x_{k+1} = x_k + h = x_k - \frac{f'(x_k)}{f''(x_k)}$$

Does that look at all familiar?

Yes, that's just like doing solving f'(x)=0 with Newton's method.

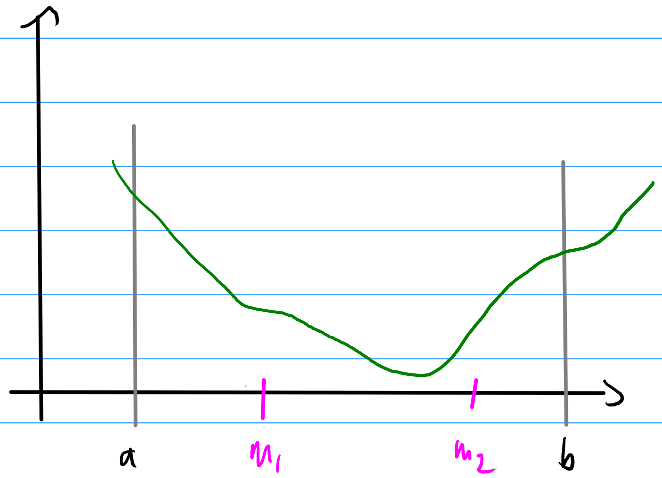
So this gets to be called Newton's method, too.

To be precise: Newton's method for optimization.

Demo: Newton's method in 1D

Golden Section Search

Let's try to create an analog to 'bisection', with a type of bracket.



Is one middle point in the bracket good enough?

No, no idea which half has the minimum. Need at least two.

Next: what condition are we going to maintain throughout?

In particular: Is "the minimum is in the bracket" feasible?

Consider $f(m_1) = f(m_2$. Then we don't have a lot of information.

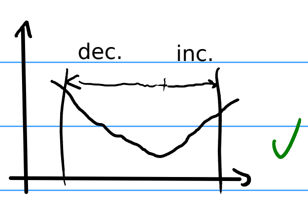
The minimum could be anywhere.

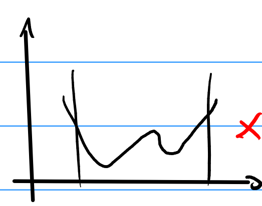
So we cannot promise that the minimum stays in the bracket.

=> Assume more, promise less.

What does it mean for f to be 'unimodal'?

f is decreasing up to a point x^* , then increasing.

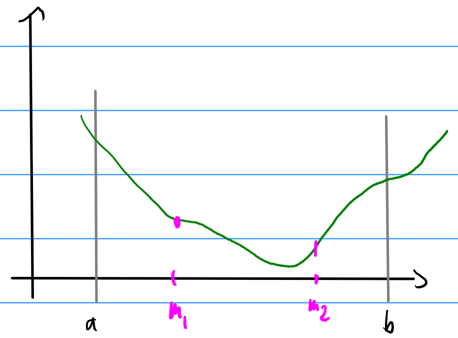




Reality check: Do we typically know that a function is unimodal in a bracket?

No. But we'll use the method as if we did.

So how do we maintain unimodality in each bracket?



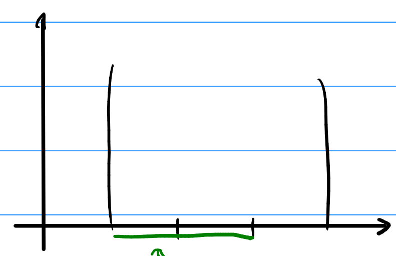
$f(m_1) > f(m_2)$ $\rightsquigarrow$ reduce to $[f(m_1), f(b)]$

$f(m_1) < f(m_2)$ $\rightsquigarrow$ reduce to $[f(a), f(m_2)]$

$f(m_1) = f(m_2)$ $\rightsquigarrow$ no info, pick one. (and maintain unimod.)

Where do we put the midpoints?

First idea: Thirds of [a,b].



zoomed-in bracket: cannot reuse $f(m_1)$

Better idea: Find points that make this possible.

$$m_2 = a + \overbrace{\left(\frac{\sqrt{5}-1}{2}\right)}^{.618}(b-a)$$

Demo: Proportions of the Golden Section

$$m_1 = a + \overbrace{\left(1 - \frac{\sqrt{5}-1}{2}\right)}^{.381}(b-a)$$

What's the convergence order of Golden Section Search?

Linear

Steepest Descent

What do we do in n dimensions?

Idea: Go in direction of steepest descent.

What does that mean mathematically?

$d = -\nabla f(x_k) \quad \rightsquigarrow \quad \vec{x}_{k+1} = \vec{x}_k + \alpha \vec{d}$

And how far do we go? (i.e. what is α?)

Good question. Use a 1D optimization method to find out!

Do an example: $f(x) = \frac{1}{2} x_0^2 + 2.5\, x_1^2$

$$\nabla f(x) = \begin{pmatrix} x_1 \\ 5x_2 \end{pmatrix}$$

Search direction: $d = -\begin{pmatrix} (x_k)_0 \\ 5(x_k)_1 \end{pmatrix}$

Demo: Steepest Descent

What's the convergence order in the example in the demo?

Linear

Can we do better by using information from the second derivative?

Of course. ;) -> Newton.

Newton's method in n dimensions

Step 1: Write down a quadratic approximation $\tilde{f}$ to f at x_k .

1D: $\tilde{f}(x+h) = f(x) + f'(x)\,h + f''(x)\,\frac{h^2}{2}$

nD: $\tilde{f}(\vec{x}+\vec{h}) = f(\vec{x}) + \vec{\nabla} f(\vec{x}) \cdot \vec{h} + \frac{1}{2} h^T H_f(x)\,\vec{h}$

Remember: $H_f(x) = \begin{pmatrix} \frac{\partial^2}{\partial x_1 \partial x_1} f & \cdots & \frac{\partial^2}{\partial x_1 \partial x_n} f \\ \vdots & & \vdots \\ \frac{\partial^2}{\partial x_n \partial x_1} f & \cdots & \frac{\partial^2}{\partial x_n \partial x_n} f \end{pmatrix}$

Step 2: Find minimum of $\tilde{f}$. To do so, take derivative and set to zero.

$0 \stackrel{!}{=} \nabla_h \tilde{f}(x+h) = \vec{\nabla} f(\vec{x}) + H_f(x)\,\vec{h}$

$\rightsquigarrow \quad H_f(x)\,\vec{h} = -\vec{\nabla} f(x)$

$\rightsquigarrow \quad \vec{h} = -H_f(\vec{x})\,\nabla \vec{f}(x)$

$\rightsquigarrow \quad \vec{x}_{k+1} = \vec{x}_k - H_f^{-1}(\vec{x}_k)\,\vec{\nabla} f(\vec{x}_k)$

Do an example: $f(x) = \frac{1}{2} x_0^2 + 2.5\, x_1^2$

$$\nabla f(x) = \begin{pmatrix} x_0 \\ 5x_1 \end{pmatrix}$$

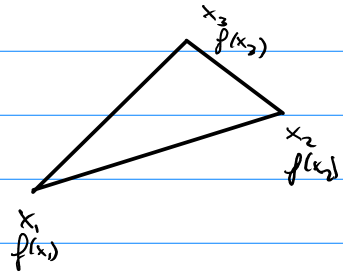
$$H_f(x) = \begin{pmatrix} 1 & 0 \\ 0 & 5 \end{pmatrix}$$

Demo: Newton's method in n dimensions

What if we don't even have one derivative, let alone two?!

Options:

- Nelder-Mead Method ("Amoeba method")



How many points in n dim?

$$f(x_1) \le f(x_2) \le f(x_3)$$

Demo: Nelder-Mead

- Secant updating methods (for example "BFGS")

Broyden
Fletcher
Goldfarb
Shanno

The "trust region" idea applies in optimization, too!

(see end of Nonlinear Equations chapter)

Constrained Optimization

Modify the problem statement of optimization to accommodate a constraint.

$$f: \mathbb{R}^n \to \mathbb{R}$$

Find x so that $f(x)$ assumes the smallest possible value...

...of all points where $g(x) = 0$.

$g: \mathbb{R}^n \to \mathbb{R}^m$ $\quad$ m: number of "constraints"

What does a solution/minimum x^* of this problem look like?

I.e. what are some necessary conditions on x^* ?

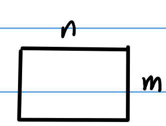
$g(x) = 0$ (obviously)

All descent directions at x^* must cause the constraints to be violated.

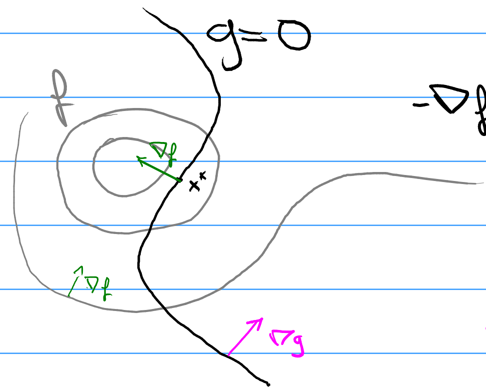
As math: $\quad g: \mathbb{R}^n \to \mathbb{R}^m$

$$-\nabla f(x^*) \in \text{rowspace } J_g(x^*)$$

J_g:



$$\Leftrightarrow \quad -\nabla f(x) = J_g^T(x^*)\,\lambda \quad \text{for some } \lambda$$



$$-\nabla f = \nabla g \cdot \lambda = \nabla g^T \lambda$$
$$= J_g^T \lambda \quad \text{for some } \lambda$$

$\nabla f \perp \{g=0\}$ at x^*!

Miracle: Reduce constrained to un-constrained optimization.

Define a new function of more unknowns: x and λ, $\lambda \in \mathbb{R}^m$

$$\mathcal{L}(x,\lambda) := f(x) + g(x)^T\lambda$$

What are the necessary conditions for an un-constrained minimum of $\mathcal{L}$? !

$$\nabla \mathcal{L} = \begin{pmatrix} \nabla_x \mathcal{L} \\ \nabla_\lambda \mathcal{L} \end{pmatrix} = \begin{pmatrix} \nabla f(x) + J_g(x)^T\lambda \\ g(x) \end{pmatrix} = 0$$

exactly the necessary conditions
for the constrained minimum of f!

Using Newton's method on $\mathcal{L}$ gets a new name:

"Sequential Quadratic Programming"

Can you do an example?

Minimize $(x-2)^4 + 2(y-1)^2$ subject to $x+4y=3$

Minimizing $(x-2)^4 + 2(y-1)^2$ while ignoring the constraint.

yields $x=2$, $x=1$. As expected, that minimum violates the constraint.

So, find Lagrangian:

$$\mathcal{L}(x,y,\lambda) = (x-2)^4 + 2(y-1)^2 + \lambda\big((x+4y)-3\big)$$

added another dimension, the Lagrange multiplier λ

rewritten to g(x)=0 form

Then use an unconstrained optimization method on this,

and the minimum (in x,y) should satisfy the constraint.

$$\nabla\mathcal{L}(x,y,\lambda) = \begin{pmatrix} 4(x-2)^3 + \lambda \\ 4(y-1) + 4\lambda \\ x+4y-3 \end{pmatrix}$$

$$H_{x,y,\lambda}(x,y,\lambda) = \begin{pmatrix} 12(x-2)^2 & 0 & 1 \\ 0 & 4 & 4 \\ 1 & 4 & 0 \end{pmatrix}$$

Demo: Sequential Quadratic Programming