Today, want to finish up Gauss-Newton.

To do that, we need Newton's method.

After that, we'll take a look at gradient descent.

We will come back to Newton's method.

Recall: Newton's method in 1D:

Have C^1 function $f : \mathbb{R} \to \mathbb{R}$

want to solve $\boxed{f(x) = 0}$

This is a rootfinding problem.

How to do it if we can't find an analytical solution?

Use an iterative scheme: construct a sequence $\{x_n\}_{n=0}^{\infty}$

such that $x_n \to x^*$ as $n \to \infty$ and $f(x^*) = 0$.

There are many different iterative schemes — you will see a bunch of them in this class.

Newton's method for 1D rootfinding is:

$$x_{n+1} = x_n - f(x_n) / f'(x_n).$$

Whether this works is very sensitive to the choice of x_0!

How to derive? Fix some iterate x_n, let $x^* = x_n + \delta x$.

Then:

$$0 = f(x^*) = f(x_n + \delta x) = f(x_n) + f'(x_n)\delta x + O(\delta x^2).$$

Hence:

$$\delta x = -f(x_n) / f'(x_n) + O(\delta x^2).$$

We approximate δx with just $-f(x_n)/f'(x_n)$.

Call this approximation δx_n. We then define the sequence: $\{x_n\}_{n=0}^{\infty}$ by choosing x_0 and letting:

$$x_{n+1} = x_n - f(x_n)/f'(x_n). \quad \text{for } n \geq 0.$$

<u>Note</u>: the Taylor series we used to compute δx may <u>not be valid</u>! So this is not always justified.

There is a convergence theory for Newton's method which tells us:

1) when this converges
2) how fast it converges

Rootfinding and 1D Newton is a topic for <u>numerical analysis</u>.

We are interested in multidimensional problems, solving nonlinear systems of equations, and minimization problems.

<u>Note</u>: we can use Newton's method to minimize 1D functions, too. Apply it to:

$$g(x) = 0, \quad \text{where } g(x) = f'(x).$$

Get:

$$x_{n+1} = x_n - g(x_n)/g'(x_n) = x_n - f'(x_n)/f''(x_n).$$

Let's look at the multidimensional versions of these problems:

vector!

1) solve $F(\underline{x}) = 0$, $\quad F: \mathbb{R}^N \to \mathbb{R}^M$ (nonlinear system of equations)

2) solve $\nabla f(\underline{x}) = 0$, $\quad f: \mathbb{R}^N \to \mathbb{R}$ (necessary conds for optimality)

First, the nonlinear system:

$$0 = F(\underline{x}_n + \delta \underline{x}) = F(\underline{x}_n) + DF(\underline{x}_n)\delta \underline{x} + O(\|\delta \underline{x}\|^2)$$

Remember that $DF(\underline{x}_n)$ is the Jacobian of F evaluated at $\underline{x}_n$:

$$DF(\underline{x}_n) = \begin{bmatrix} \left.\frac{\partial F_1}{\partial x_1}\right|_{\underline{x}_n} & \cdots & \left.\frac{\partial F_1}{\partial x_N}\right|_{\underline{x}_n} \\ \vdots & \ddots & \vdots \\ \left.\frac{\partial F_M}{\partial x_1}\right|_{\underline{x}_n} & \cdots & \left.\frac{\partial F_M}{\partial x_N}\right|_{\underline{x}_n} \end{bmatrix} \in \mathbb{R}^{M \times N}$$

Not invertible if $M \neq N$! So, solve normal equations instead:

$$-F(\underline{x}_n) = DF(\underline{x}_n)\delta \underline{x} \Rightarrow -DF(\underline{x}_n)^T F(\underline{x}_n) = DF(\underline{x}_n)^T DF(\underline{x}_n)\delta \underline{x}$$

$$\Rightarrow \delta \underline{x} = -\left(DF(\underline{x}_n)^T DF(\underline{x}_n)\right)^{-1} DF(\underline{x}_n)^T F(\underline{x}_n)$$

$$\Rightarrow \delta \underline{x} = -DF(\underline{x}_n)^{\dagger} F(\underline{x}_n).$$

Get the iteration:

$$\underline{x}_{n+1} = \underline{x}_n - DF(\underline{x}_n)^{\dagger} F(\underline{x}_n).$$

Note: if $M = N$ and $DF(\underline{x}_n)$ is nonsingular, then:

$$\begin{aligned} DF(\underline{x}_n)^{\dagger} &= \left(DF(\underline{x}_n)^T DF(\underline{x}_n)\right)^{-1} DF(\underline{x}_n)^T \\ &= DF(\underline{x}_n)^{-1} DF(\underline{x}_n)^{-T} DF(\underline{x}_n)^T \\ &= DF(\underline{x}_n)^{-1}! \end{aligned}$$

Next, consider the multivariable minimization problem: (4)

[solve $\nabla f(\underline{x}) = 0$, $f : \mathbb{R}^N \to \mathbb{R}$

Same procedure:

$$\underline{0} = \nabla f(\underbrace{\underline{x}_n + \delta\underline{x}}_{\underline{x}^*}) = \nabla f(\underline{x}_n) + \nabla^2 f(\underline{x}_n)\,\delta\underline{x} + O(\|\delta\underline{x}\|^2).$$

So; get the iteration:

$$\underline{x}_{n+1} = \underline{x}_n - \nabla^2 f(\underline{x}_n)^{-1} \nabla f(\underline{x}_n).$$

Reasonable to expect to be able to invert the Hessian.

$$\nabla^2 f(\underline{x}_n) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_N \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_N^2} \end{bmatrix}.$$

OK: back to Gauss-Newton:

Nonlinear least squares problem:

$$\underset{\underline{c} \in \mathbb{R}^N}{\text{minimize}} \ \|\underline{y} - \underline{f}(\underline{c})\|_2^2 ,$$

where $\underline{y} \in \mathbb{R}^M$, $\underline{f} : \mathbb{R}^N \to \mathbb{R}^M$, N = # params, M = # obs.

Define $F(\underline{c})$ to be our cost function:

$$F(\underline{c}) = \|\underline{y} - \underline{f}(\underline{c})\|_2^2 .$$

In the previous class, we looked at computing the gradient of the cost function directly, "summation style" (or "in component form"). Let's do it another way. First, we have:

$$\underline{f}(\underline{c} + \delta\underline{c}) = \underline{f}(\underline{c}) + D\underline{f}(\underline{c})^T \delta\underline{c} + O(\|\delta\underline{c}\|^2).$$

Then:

$$
\begin{aligned}
F(\underline{c} + \delta\underline{c}) &= \|\underline{y} - \underline{f}(\underline{c} + \delta\underline{c})\|_2^2 \\
&= (\underline{y} - \underline{f}(\underline{c} + \delta\underline{c}))^T (\underline{y} - \underline{f}(\underline{c} + \delta\underline{c})) \\
&= (\underline{y} - \underline{f}(\underline{c}) - D\underline{f}(\underline{c})^T \delta\underline{c} + O(\|\delta\underline{c}\|_2^2))^T \\
&\qquad (\underline{y} - \underline{f}(\underline{c}) - D\underline{f}(\underline{c})^T \delta\underline{c} + O(\|\delta\underline{c}\|_2^2)) \\
&= \underline{y}^T\underline{y} - 2\underline{y}^T \underline{f}(\underline{c}) + \underline{f}(\underline{c})^T \underline{f}(\underline{c}) \\
&\qquad - 2\underline{y}^T D\underline{f}(\underline{c})^T \delta\underline{c} + 2\underline{f}(\underline{c})^T D\underline{f}(\underline{c})^T \delta\underline{c} \\
&\qquad + O(\|\delta\underline{c}\|_2^2) \\
&= F(\underline{c}) - 2(\underline{y} - \underline{f}(\underline{c}))^T D\underline{f}(\underline{c})^T \delta\underline{c} + O(\|\delta\underline{c}\|_2^2).
\end{aligned}
$$

Compare this with:

$$F(\underline{c} + \delta\underline{c}) = F(\underline{c}) + DF(\underline{c})\,\delta\underline{c} + O(\|\delta\underline{c}\|_2^2).$$

Hence:

$$DF(\underline{c}) = -2(\underline{y} - \underline{f}(\underline{c}))^T D\underline{f}(\underline{c})^T$$

$$\Rightarrow \nabla F(\underline{c}) = DF(\underline{c})^T = -2 D\underline{f}(\underline{c})^T (\underline{y} - \underline{f}(\underline{c})).$$

Let's try to solve the nonlinear system:

$$\nabla F(\underline{c}^*) = \underline{0}$$

using Newton's method. We computed $DF(\underline{c})$, still need $\nabla^2 F(\underline{c})$...

$$\nabla^2 F(\underline{c}) = \nabla \nabla F(\underline{c})$$
$$= -2 \nabla \left[D\underline{f}(\underline{c})^T (\underline{y} - \underline{f}(\underline{c})) \right] \ldots$$

How do we evaluate this? Remember: $DF^T = \nabla F \ldots$

$$D \left[D\underline{f}(\underline{c})^T (\underline{y} - \underline{f}(\underline{c})) \right] = \text{``} D^2 \underline{f}(\underline{c})^T (\underline{y} - \underline{f}(\underline{c})) \text{''}$$
$$- D\underline{f}(\underline{c})^T D\underline{f}(\underline{c})$$

Jacobian

What the hell is this?

Well, it should be a matrix, so $D^2 \underline{f}(\underline{c})$ is a 3D array of numbers...

But note: if we are close to optimum, $\underline{y} - \underline{f}(\underline{c})$ should be small, so we ignore this term, and approximate $D^2 F(\underline{c})$ with:

$$D^2 F(\underline{c}) \approx 2 D\underline{f}(\underline{c})^T D\underline{f}(\underline{c}).$$

Altogether, this gives Gauss-Newton;

$$\underline{c}_{n+1} = \underline{c}_n + \left(D\underline{f}(\underline{c}_n)^T D\underline{f}(\underline{c}_n) \right)^{-1} D\underline{f}(\underline{c}_n)^T (\underline{y} - \underline{f}(\underline{c}_n)).$$

So Gauss-Newton is almost — but not quite — a Newton's method.

Next time: start on unconstrained optim. more broadly.

Start w/ Gradient descent.