

Today, want to finish up Gauss-Newton.

70

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^2 function $f: \mathbb{R} \rightarrow \mathbb{R}$
want to solve $\underline{f(x) = 0}$
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 $x_n \rightarrow x^*$ as $n \rightarrow \infty$ 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: $f(x_n) - f(x^*) = f$

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

Hence:

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

(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:

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

Note: the Taylor series we used to compute δx may not be valid! 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 numerical analysis.

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

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

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

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:

- 1) solve $\underbrace{F(\underline{x}) = 0}_{\text{vector!}}, \quad F: \mathbb{R}^N \rightarrow \mathbb{R}^M$ (nonlinear system of equations)
- 2) solve $\nabla f(\underline{x}) = 0, \quad f: \mathbb{R}^N \rightarrow \mathbb{R}$ (necessaryconds) (for optimality)

(3)

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 . $\underline{x}_n \approx \underline{x}$.

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

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

$$\begin{aligned} -F(\underline{x}_n) &= DF(\underline{x}_n)\delta\underline{x} \implies -DF(\underline{x}_n)^T F(\underline{x}_n) = DF(\underline{x}_n)^T DF(\underline{x}_n)\delta\underline{x} \\ &\implies \delta\underline{x} = -(DF(\underline{x}_n)^T DF(\underline{x}_n))^{-1} DF(\underline{x}_n)^T F(\underline{x}_n) \\ &\implies \delta\underline{x} = -DF(\underline{x}_n)^+ F(\underline{x}_n). \end{aligned}$$

Get the iteration:

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

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

$$\begin{aligned} DF(\underline{x}_n)^+ &= (DF(\underline{x}_n)^T DF(\underline{x}_n))^{-1} DF(\underline{x}_n)^T \\ &= DF(\underline{x}_n)^{-1} DF(\underline{x}_n)^T DF(\underline{x}_n)^T \\ &= DF(\underline{x}_n)^{-1}. \end{aligned}$$

Next, consider the multivariable minimization problem; (4)

$$\left\{ \text{solve } \nabla f(\underline{x}) = 0, \quad f: \mathbb{R}^N \rightarrow \mathbb{R} \right.$$

Same procedure:

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

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.

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

OK: back to Gauss-Newton:

Nonlinear least squares problem:

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

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

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 (5) 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:

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

Compare this with:

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

Hence:

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

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

(6)

Let's try to solve the nonlinear system:

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

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

$$\nabla^2 F(\underline{c}) = \nabla \nabla F(\underline{c})$$

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

How do we evaluate this? Remember: $D\underline{F} = \nabla F\dots$

$$D \left[D\underline{s}(\underline{c})^T (\underline{y} - \underline{s}(\underline{c})) \right] = "D^2 \underline{s}(\underline{c})^T (\underline{y} - \underline{s}(\underline{c}))"$$

Jacobian

$$- D\underline{s}(\underline{c})^T D\underline{s}(\underline{c})$$

What the hell is this?

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

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

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

Altogether, this gives Gauss-Newton:

(7)

$$\underline{c}_{\text{ntn}} = \underline{c}_n - (\underline{D}\underline{f}(\underline{c}_n)^T \underline{D}\underline{f}(\underline{c}_n))^{-1} \underline{D}\underline{f}(\underline{c}_n)^T (\underline{y} - \underline{f}(\underline{c}_n)).$$

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

Next time: start on unconstrained optim., more broadly.

Start w/ Gradient descent,