Exam 3 next week
HW12
Ch 1 -> (hopefully) today

Goals:
10 opt methods
no opt methods
~ nonlinear lsq.
Newton’s Method

Reuse the Taylor approximation idea, but for optimization.

\[ x = x_n \rightarrow f(x_n + h) \approx f(x_n) + f'(x_n)h + f''(x_n)\frac{h^2}{2} =: \hat{f}(h) \]

\[ f'(h) = f'(x_n) + f''(x_n)h \quad \Rightarrow h = -\frac{f'(x_n)}{f''(x_n)} \]

\[ x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)} \]

\[ x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \]

\[ \rightarrow \text{locally qudr rate conv. because eqn. to solve y Newton} \]

**Demo:** Newton’s Method in 1D [cleared]
Steepest Descent/Gradient Descent

Given a scalar function $f : \mathbb{R}^n \to \mathbb{R}$ at a point $x$, which way is down?

Direction of steepest desc. $-\nabla f$

$x_{k+1} = x_k + \alpha s_k$

$s_k = -\nabla f(x_k)$

$\alpha \to f(x_k + \alpha s_k) \leq \min \text{ that line search}$

Empirically: Linear conv.

Demo: Steepest Descent [cleared] (Part 1)
Consider quadratic model problem:

\[ f(x) = \frac{1}{2} x^T A x + c^T x \]

where \( A \) is SPD. (A good model of \( f \) near a minimum.)
Steepest Descent: Convergence

Consider quadratic model problem:

\[ f(x) = \frac{1}{2} x^T A x + c^T x \]

where \( A \) is SPD. (A good model of \( f \) near a minimum.)

Define error \( e_k = x_k - x^* \). Then can show:

\[
\|e_{k+1}\|_A = \sqrt{e_{k+1}^T A e_{k+1}} = \frac{\sigma_{\text{max}}(A) - \sigma_{\text{min}}(A)}{\sigma_{\text{max}}(A) + \sigma_{\text{min}}(A)} \|e_k\|_A
\]

\( \rightarrow \) confirms linear convergence.

Convergence constant related to conditioning:

\[
\frac{\sigma_{\text{max}}(A) - \sigma_{\text{min}}(A)}{\sigma_{\text{max}}(A) + \sigma_{\text{min}}(A)} = \frac{\kappa(A) - 1}{\kappa(A) + 1}.
\]
Hacking Steepest Descent for Better Convergence

Extrapolation methods:

Demo: Steepest Descent [cleared] (Part 2)
Hacking Steepest Descent for Better Convergence

Extrapolation methods:

Look back a step, maintain 'momentum'.

\[ x_{k+1} = x_k - \alpha_k \nabla f(x_k) + \beta_k (x_k - x_{k-1}) \]

Heavy ball method:

constant \( \alpha_k = \alpha \) and \( \beta_k = \beta \). Gives:

\[ \|e_{k+1}\|_A = \frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \|e_k\|_A \]

Demo: Steepest Descent [cleared] (Part 2)
What is *stochastic gradient descent (SGD)*?
Optimization in Machine Learning

What is stochastic gradient descent (SGD)?

Common in ML: Objective functions of the form

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x),$$

where each $f_i$ comes from an observation ("data point") in a (training) data set. Then "batch" (i.e. normal) gradient descent is

$$x_{k+1} = x_k - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(x_k).$$

Stochastic GD uses one (or few, "minibatch") observation at a time:

$$x_{k+1} = x_k - \alpha \nabla f_{\phi(k)}(x_k).$$

ADAM optimizer: GD with exp. moving avgs. of $\nabla$ and its square.
Can we optimize in *the space spanned* by the last two step directions?

**Demo:** Conjugate Gradient Method [cleared]
Conjugate Gradient Methods

Can we optimize in \textit{the space spanned} by the last two step directions?

\[ (\alpha_k, \beta_k) = \arg\min_{\alpha_k, \beta_k} \left[ f\left( x_k - \alpha_k \nabla f(x_k) + \beta_k (x_k - x_{k-1}) \right) \right] \]

- Will see in more detail later (for solving linear systems)
- Provably optimal first-order method for the quadratic model problem
- Turns out to be closely related to Lanczos (A-orthogonal search directions)

\textbf{Demo: Conjugate Gradient Method [cleared]}
\[ A \approx Q D Q^T \]
\[ \gamma \approx A \hat{y} = (Q^T x) D (Q^T y) \]
Nelder-Mead Method

Idea:

Demo: Nelder-Mead Method [cleared]
Newton’s method ($n$ D)

What does Newton’s method look like in $n$ dimensions?

\[ \frac{\partial f}{\partial s} = \nabla f(x) + \left( \frac{1}{2} \nabla^2 f(x) s \right) s = 0 \]

Use $N$ to solve $\nabla^2 f(x) s = -\nabla f(x)$
Newton’s method \((n \text{ D})\): Observations

**Drawbacks?**

- Need \(2\) derivatives
- Expensive: need Hessian solve
- Dependent on cond. of Hessian

**Demo:** Newton’s Method in \(n\) dimensions [cleared]
Quasi-Newton Methods

Secant/Broyden-type ideas carry over to optimization. How? Come up with a way to update to update the approximate Hessian.

\[ x_{n+1} = x_n - \alpha_k B_k^{-1} \nabla f(x_n) \]
\[ \alpha_k \text{ -- line search parameter} \]
\[ s_k = x_{n+1} - x_n \]
\[ y_k = \nabla f(x_{n+1}) - \nabla f(x_n) \]
\[ B_{k+1} s_k = y_k \quad \text{-- secant condition} \]

BFGS: Secant-type method, similar to Broyden:

\[ B_{k+1} = B_k + \frac{y_k y_k^T}{y_k^T s_k} - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} \]
Nonlinear Least Squares: Setup

What if the $f$ to be minimized is actually a 2-norm?

$$f(x) = \|r(x)\|_2, \quad r(x) = y - a(x)$$