

Exam 3 next week

HW12

4 ch 1 \rightarrow (hopefully) today

Goals.

10 opt methods

nD opt methods

\sim nonlinear lsq.

Newton's Method

1. ...

Reuse the Taylor approximation idea, but for optimization.

$$x = x_k \rightarrow f(x+h) \approx f(x) + f'(x)h + f''(x)\frac{h^2}{2} =: \hat{f}(h)$$

$$0 = \hat{f}'(h) = f'(x) + f''(x)h$$

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

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

Newton for solving

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

\rightarrow locally quadr. conv. because eqn. to solve - y Newton

Demo: Newton's Method in 1D [cleared]

Steepest Descent/Gradient Descent

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

Direction of steepest desc. $-\nabla f$

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

$\alpha \mapsto f(x_k + \alpha s_k) \leftarrow$ min. that
"line search"

Empirically \therefore linear conv.

Steepest Descent: Convergence

Consider quadratic model problem:

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

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

$$e_{k+1} = \|x_{k+1} - x^*\| = \rho e_k$$

Steepest Descent: Convergence

Consider quadratic model problem:

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

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

Define error $\mathbf{e}_k = \mathbf{x}_k - \mathbf{x}^*$. Then can show:

$$\|\mathbf{e}_{k+1}\|_A = \sqrt{\mathbf{e}_{k+1}^T A \mathbf{e}_{k+1}} = \frac{\sigma_{\max}(A) - \sigma_{\min}(A)}{\sigma_{\max}(A) + \sigma_{\min}(A)} \|\mathbf{e}_k\|_A$$

→ confirms linear convergence.

Convergence constant related to conditioning:

$$\frac{\sigma_{\max}(A) - \sigma_{\min}(A)}{\sigma_{\max}(A) + \sigma_{\min}(A)} = \frac{\kappa(A) - 1}{\kappa(A) + 1}$$

Hacking Steepest Descent for Better Convergence

Extrapolation methods:



Heavy ball method:



[Demo: Steepest Descent](#) [\[cleared\]](#) (Part 2)

Hacking Steepest Descent for Better Convergence

Extrapolation methods:

Look back a step, maintain '*momentum*'.

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \nabla f(\mathbf{x}_k) + \beta_k (\mathbf{x}_k - \mathbf{x}_{k-1})$$

Heavy ball method:

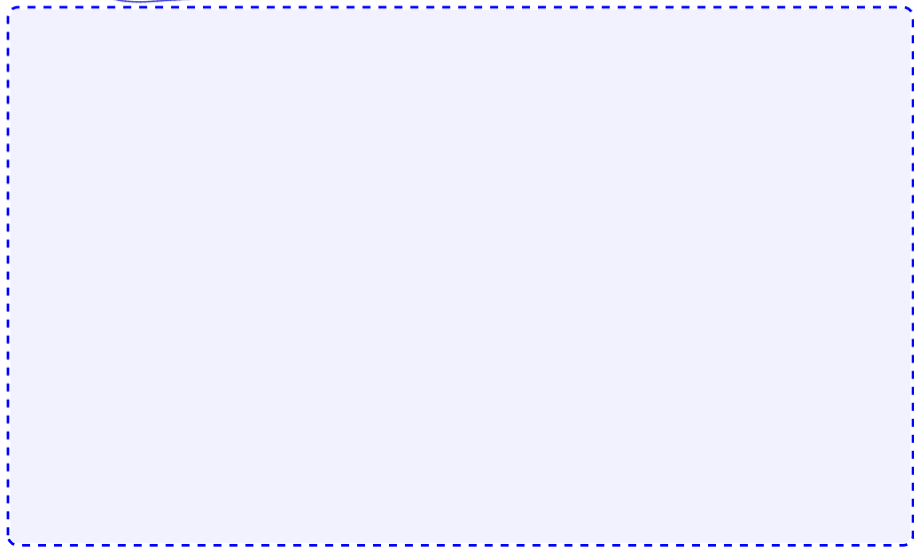
constant $\alpha_k = \alpha$ and $\beta_k = \beta$. Gives:

$$\|\mathbf{e}_{k+1}\|_A = \frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \|\mathbf{e}_k\|_A$$

[Demo: Steepest Descent \[cleared\]](#) (Part 2)

Optimization in Machine Learning

What is *stochastic gradient descent* (SGD)?



Optimization in Machine Learning

What is *stochastic gradient descent (SGD)*?

Common in ML: Objective functions of the form

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

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

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \frac{1}{n} \sum_{i=1}^n \nabla f_i(\mathbf{x}_k).$$

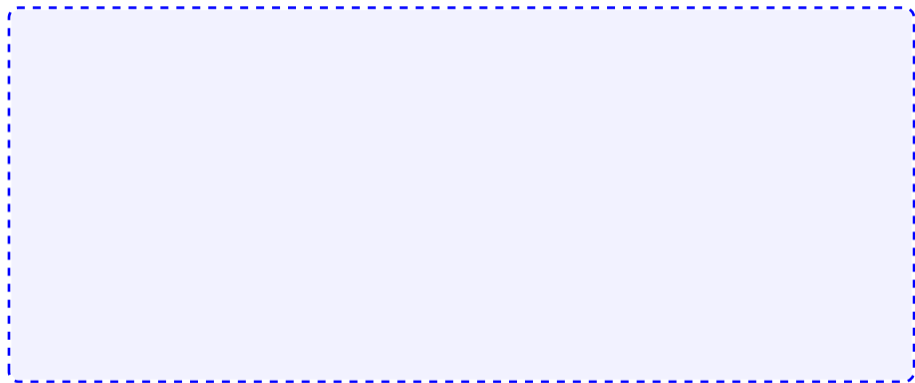
Stochastic GD uses one (or few, “*minibatch*”) observation at a time:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \nabla f_{\phi(k)}(\mathbf{x}_k).$$

ADAM optimizer: GD with exp. moving avgs. of ∇ and its square.

Conjugate Gradient Methods

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 *the space spanned* by the last two step directions?

$$(\alpha_k, \beta_k) = \operatorname{argmin}_{\alpha_k, \beta_k} \left[f\left(\mathbf{x}_k - \alpha_k \nabla f(\mathbf{x}_k) + \beta_k (\mathbf{x}_k - \mathbf{x}_{k-1})\right) \right]$$

- ▶ 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)

Demo: Conjugate Gradient Method [cleared]

orth: $\mathbf{x}^T \mathbf{y} = 0$

A -orth: $\mathbf{x}^T \mathbf{A} \mathbf{y} = 0$

A spd

$$A = Q D Q^T$$

↑
diag > 0

$$\tilde{x}^T A \tilde{y} = (Q^T \tilde{x}) D (Q^T \tilde{y})$$

Nelder-Mead Method

Idea:

simplex gymnastics

[Demo: Nelder-Mead Method](#) [\[cleared\]](#)

Newton's method (n D)

$$f: \mathbb{R}^n \rightarrow \mathbb{R} \quad \nabla$$

What does Newton's method look like in n dimensions?

↳ Newton in 1D:
solve $f' = 0$

↳ in n D:
use N. to solve $\nabla f = 0$

$$f(\vec{x} + \vec{s}) \approx f(\vec{x}) + \nabla f(\vec{x})^T \vec{s} + \frac{1}{2} \vec{s}^T H_f(\vec{x}) \vec{s}$$

$$\hat{f}(\vec{s})$$

$$0 = \nabla \hat{f}(\vec{s}) \rightarrow$$

$$H_f(\vec{x}) \vec{s} = -\nabla f(\vec{x})$$

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

$$\nabla \hat{f}(\vec{s}) = \nabla f(\vec{x})^T + \vec{s}^T H_f(\vec{x}) \quad \frac{\partial \hat{f}}{\partial s_i} =$$

Newton's method (n 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.

$$x_{u+1} = x_u - \alpha_k B_k^{-1} \nabla f(\vec{x})$$

$\alpha_k \rightarrow$ line search parameter

$$s_k = \vec{x}_{u+1} - \vec{x}_u$$
$$y_k = \nabla f(\vec{x}_{u+1}) - \nabla f(\vec{x}_u)$$

$B_{k+1} s_k = y_k \leftarrow$ 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?

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

