10-701 Recitation 2: Optimization

Jay-Yoon Lee

01/29/2015

Many of the slides are recycled form Dougal Sutherland's recitation

Motivation

- Much of the time in ML/stats, we're finding the best model to fit our data (MLE, MAP, …)
  - MLE (Maximum Likelihood Estimator)

$$\hat{\theta} = \underset{\theta}{argmax} P(D|\theta)$$

Motivation

- Much of the time in ML/stats, we're finding the best model to fit our data (MLE, MAP, …)
  - MLE (Maximum Likelihood Estimator)

$$\hat{\theta} = \underset{\theta}{argmax} P(D|\theta)$$

  - MAP (Maximum A-Posteriori Estimator)

$$\hat{\theta} = \underset{\theta}{argmax} P(\theta|D)$$

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)}$$

Motivation

- General form of optimization:
  - Loss + Penalty

$$\underset{\text{models } M}{\arg\min} \sum_{i=1}^{n} \ell(x_i; M) + \text{penalty}(M)$$

- How we do that: optimization.
- When we can: convex optimization.

Analytic minima

- Set gradient respect Beta to zero and solve

$$J_\lambda(\beta) = \frac{1}{2}\|X\beta - y\|_2^2 + \frac{1}{2}\lambda\|\beta\|_2^2$$

Gradient descent

- Start at some point, follow the gradient towards (a) minimum

$$x \leftarrow x_0$$

while termination conditions don't hold **do**

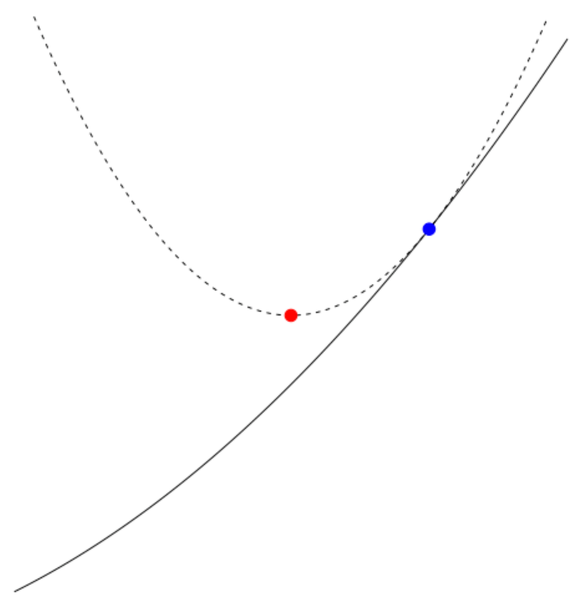
$$x \leftarrow x - \eta \nabla f(x)$$

end while

Gradient descent interpretation

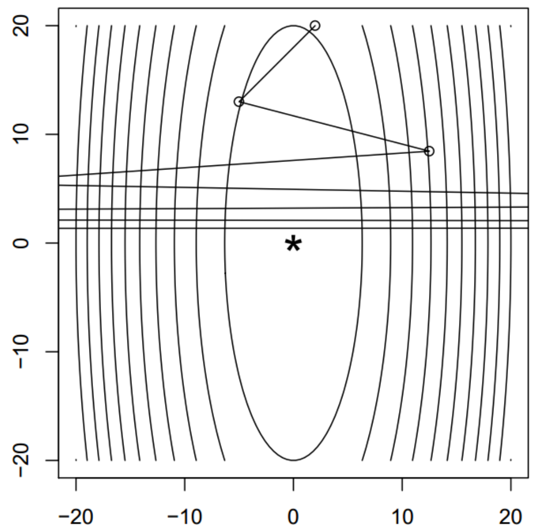
Approximate the function with a quadratic:

$$f(y) \approx \underbrace{f(x) + \nabla f(x)^T (y - x)}_{\text{linear approximation to } f} + \underbrace{\frac{1}{2\eta} \|y - x\|_2^2}_{\text{proximity to } x}$$

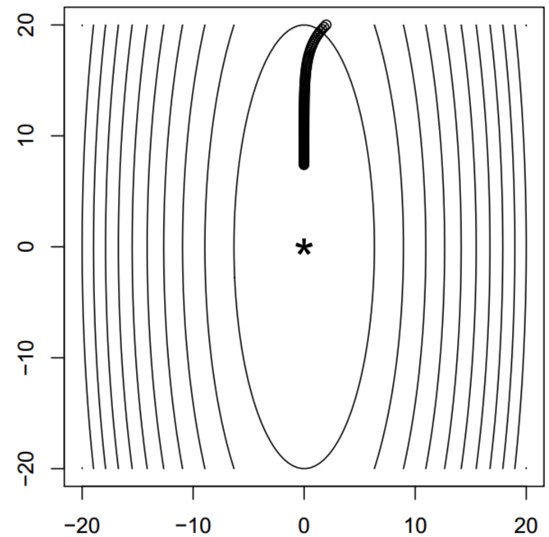


Choosing the step size

$\eta_t = t$, it is too big



too small η_t, after 100 iterations

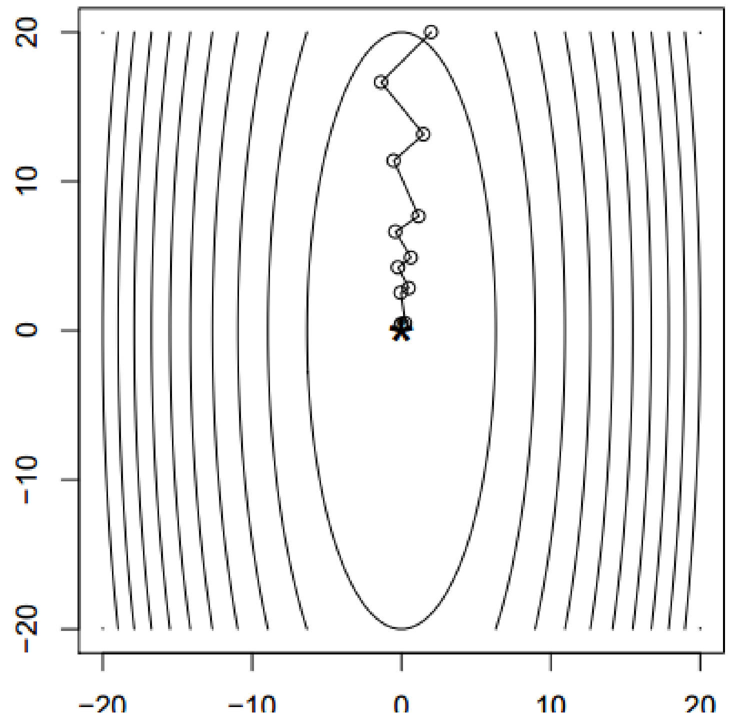


Backtracking

- Fix a backoff parameter $0 < \beta < 1$
- At each iteration:
  - Start with $\eta = 1$
  - While $f\left(x - \eta \nabla f(x)\right) > f(x) - \frac{\eta}{2} \|\nabla f(x)\|^2$
    - Back off $\eta = \beta \eta$

Backtracking line search

A typical choice $\beta = 0.8$, converged after 13 iterations:



How to terminate

- When change in iterates is small
  - When gradient is small
  - When change in function value is small
  - When backtracking step size gets too small
- Or after a fixed time/steps budget
- …

Stochastic gradient "descent"

- Usually we're minimizing the empirical loss:

$$\frac{1}{n}\sum_i \ell(x_i; M) \qquad \frac{1}{n}\sum_i \nabla_M \ell(x_i; M)$$

- We do this to approximate the expected loss:

$$\mathbb{E}_x\left[\ell(x; M)\right] \qquad \mathbb{E}_x\left[\nabla_M \ell(x_i; M)\right]$$

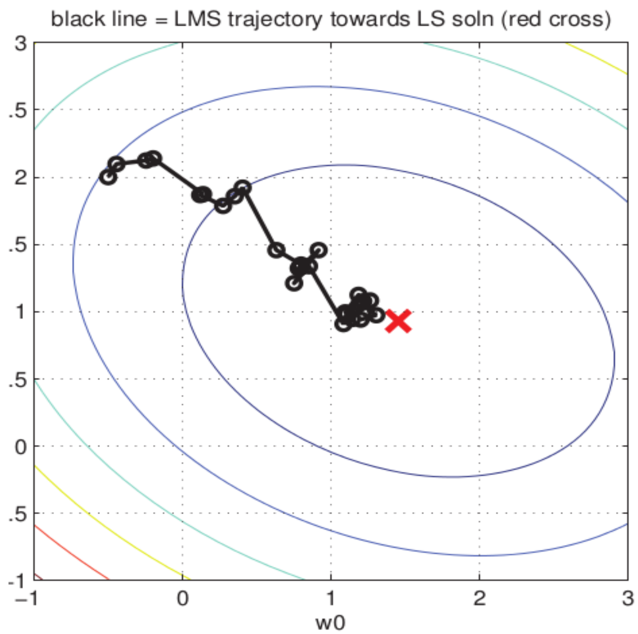
- But we can also use rougher, cheaper approx.:

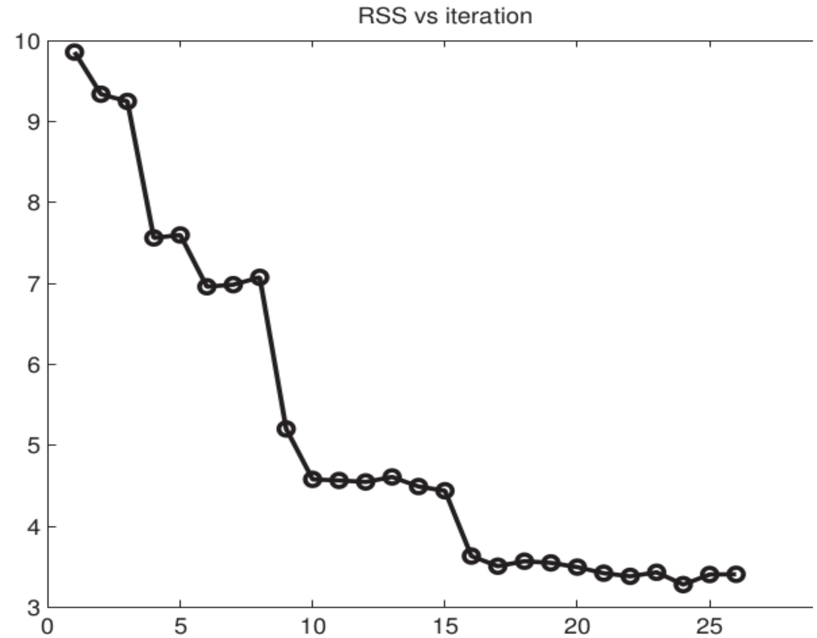
$$\ell(x_i; M) \qquad \nabla_M \ell(x_i; M)$$

SGD

- “Online” optimization
- Can do it based on a stream of samples
  - No need to remember old ones, then
- Iterations are **much** cheaper
- Requires more iterations
- One big problem: not a descent method!

SGD





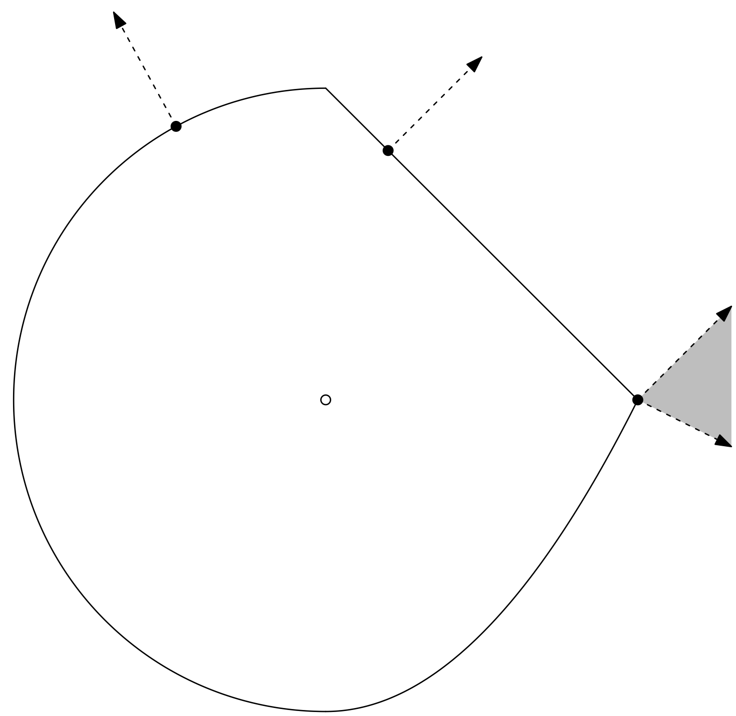
- Iterations are **much** cheaper
- Requires more iterations
- But, objective does not “Descent” Always

Mini-batch gradient

- Like SGD, but calculate gradients over a subset of training points instead of just one
- Can be a nice medium between full gradient descent and SGD
  - Not a descent method, but “closer” to one
  - Iterations more expensive than SGD
  - Converges faster than SGD

Subgradients

- When your optimization problem is convex but not differentiable



Lecture note on 10-725 Convex Optimization by Ryan Tibshirani

Subgradients

- When your optimization problem is convex but not differentiable

Lasso problem can be parametrized as

$$\min_x \frac{1}{2}\|y - Ax\|^2 + \lambda\|x\|_1$$

where $\lambda \geq 0$. Consider simplified problem with $A = I$:

$$\min_x \frac{1}{2}\|y - x\|^2 + \lambda\|x\|_1$$

Claim: solution of simple problem is $x^\star = S_\lambda(y)$, where S_λ is the **soft-thresholding operator:**

$$[S_\lambda(y)]_i = \begin{cases} y_i - \lambda & \text{if } y_i > \lambda \\ 0 & \text{if } -\lambda \leq y_i \leq \lambda \\ y_i + \lambda & \text{if } y_i < -\lambda \end{cases}$$

Lecture note on 10-725 Convex Optimization by Ryan Tibshirani

Subgradients

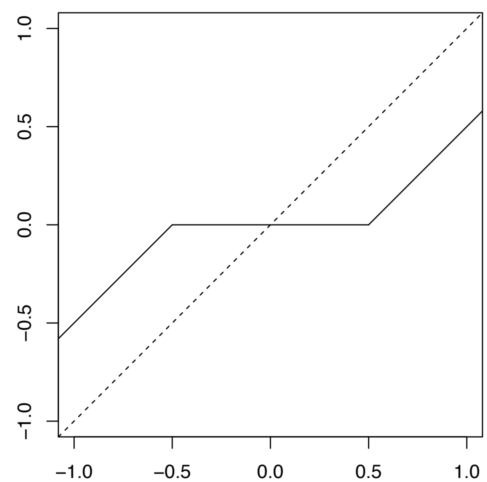
Why? Subgradients of $f(x) = \frac{1}{2}\|y - x\|^2 + \lambda\|x\|_1$ are

$$g = x - y + \lambda s,$$

where $s_i = \text{sign}(x_i)$ if $x_i \neq 0$ and $s_i \in [-1, 1]$ if $x_i = 0$

Now just plug in $x = S_\lambda(y)$ and check we can get $g = 0$

Soft-thresholding in one variable:



Subgradients

- When your optimization problem is convex but not differentiable
- Subgradient descent:
  - same algorithm, but use any subgradient instead of the gradient

$$x^{(k)} = x^{(k-1)} - t_k \cdot g^{(k-1)}, \quad k = 1, 2, 3, \ldots,$$

where $g^{(k-1)}$ is any subgradient of f at $x^{(k-1)}$

- This is slow.

Generalized gradient descent

- Objective is the sum of a convex, differentiable *g* and a convex *h*: $\min_x g(x) + h(x)$

$$x \leftarrow \mathsf{prox}_\eta \left(x - \eta \nabla g(x)\right)$$

$$\mathsf{prox}_\eta(x) = \arg\min_z \frac{1}{2\eta} \|x - z\|^2 + h(z)$$

- e.g. LASSO, projected gradient descent

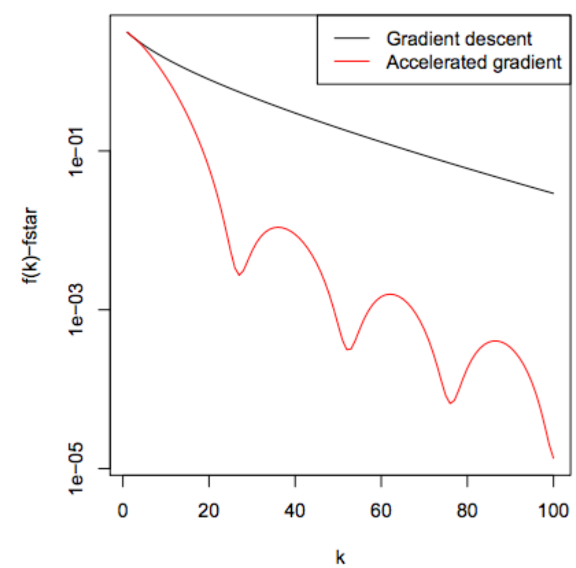
Accelerated gradient method

- At each step *k*:

$$y \leftarrow x^{(k-1)} + \frac{k-2}{k+1}\left(x^{(k-1)} - x^{(k-2)}\right)$$

$$x^{(k)} \leftarrow \text{prox}_{\eta_k}\left(y - \eta_k \nabla g(y)\right)$$

- *y* term carries "momentum"
- Provably better convergence
  - $O(1/k^2)$: optimal for first-order



Newton's method

- Gradient descent minimizes

$$f(y) \approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \frac{1}{\eta} I (y - x)$$

- Newton's method: quadratic approximation

$$f(y) \approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(x) (y - x)$$

- Takes v. few iterations for v. accurate answer
  - Iterations are very expensive
  - Diverges with bad initialization
- Damped Newton: line search, trust region

Sort-of second-order methods

- Quasi-Newton methods
  - Approximate Hessian from the gradient
  - BFGS, **L-BFGS**
- Truncated Newton
  - Partially optimize quadratic with conjugate gradient

Standard problem forms

- Linear programs (LPs)

$$\min c^T x \quad \text{subject to } Ax \le b, Ex = g$$

- Quadratic programs (QPs)

$$\min c^T x + \frac{1}{2} x^T H x \quad \text{subject to } Ax \le b, Ex = g$$

- Cone programs

$$\min c^T x \quad \text{subject to } Ax + b \in K, x \in L$$