Gradient Descent

Ryan Tibshirani
Convex Optimization 10-725

Last time: canonical convex programs

- Linear program (LP): takes the form

$$\begin{aligned} \min_x \quad & c^T x \\ \text{subject to} \quad & Dx \le d \\ & Ax = b \end{aligned}$$

- Quadratic program (QP): like LP, but with quadratic criterion
- Semidefinite program (SDP): like LP, but with matrices
- Conic program: the most general form of all

Gradient descent

Consider unconstrained, smooth convex optimization

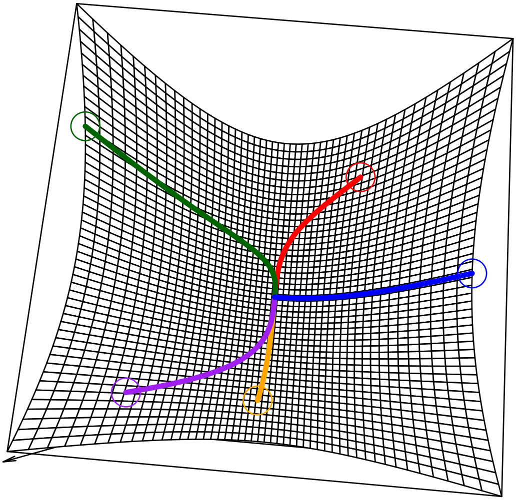
$$\min_x \ f(x)$$

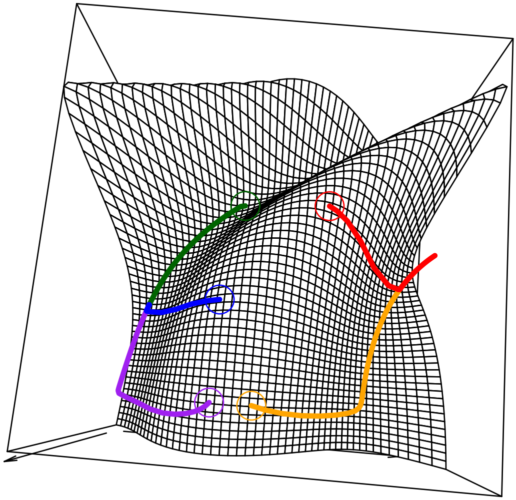
That is, f is convex and differentiable with $\mathrm{dom}(f) = \mathbb{R}^n$. Denote optimal criterion value by $f^\star = \min_x \ f(x)$, and a solution by $x^\star$

Gradient descent: choose initial point $x^{(0)} \in \mathbb{R}^n$, repeat:

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

Stop at some point





Gradient descent interpretation

At each iteration, consider the expansion

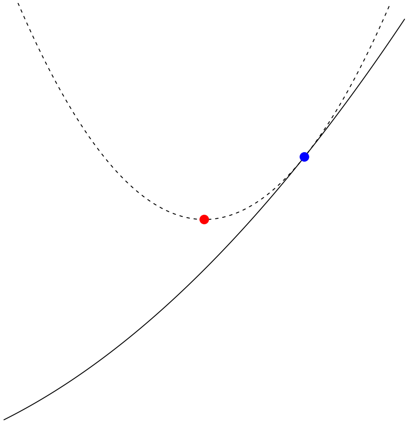
$$f(y) \approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2t} \|y - x\|_2^2$$

Quadratic approximation, replacing usual Hessian $\nabla^2 f(x)$ by $\frac{1}{t} I$

$f(x) + \nabla f(x)^T (y - x)$	linear approximation to f
$\frac{1}{2t} \|y - x\|_2^2$	proximity term to x, with weight $1/(2t)$

Choose next point $y = x^+$ to minimize quadratic approximation:

$$x^+ = x - t \nabla f(x)$$



Blue point is x, red point is

$$x^+ = \underset{y}{\mathrm{argmin}} \; f(x) + \nabla f(x)^T (y-x) + \frac{1}{2t}\|y-x\|_2^2$$

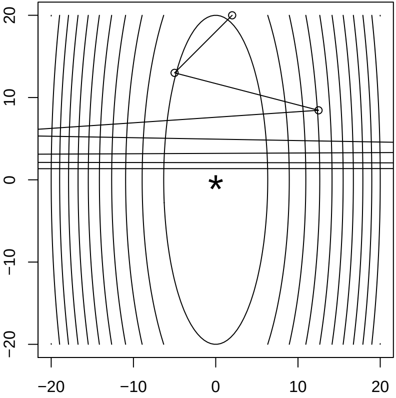
Outline

Today:

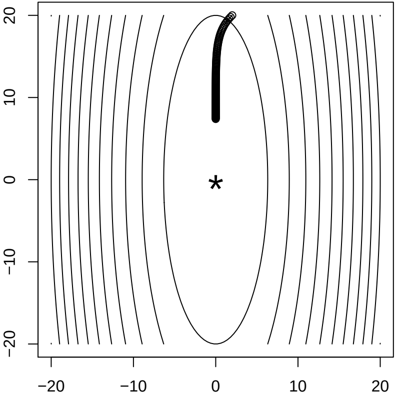
- How to choose step sizes
- Convergence analysis
- Nonconvex functions
- Gradient boosting

Fixed step size

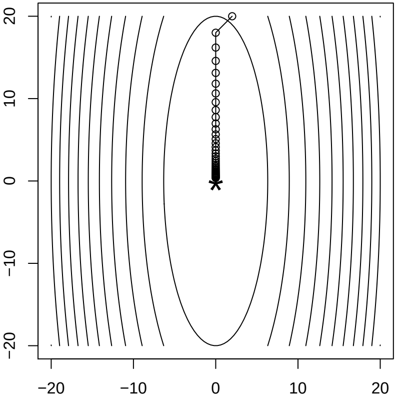
Simply take $t_k = t$ for all $k = 1, 2, 3, \ldots$, can diverge if t is too big. Consider $f(x) = (10x_1^2 + x_2^2)/2$, gradient descent after 8 steps:



Can be slow if t is too small. Same example, gradient descent after 100 steps:



Converges nicely when t is "just right". Same example, 40 steps:



Convergence analysis later will give us a precise idea of "just right"

Backtracking line search

One way to adaptively choose the step size is to use backtracking line search:

- First fix parameters $0 < \beta < 1$ and $0 < \alpha \leq 1/2$
- At each iteration, start with $t = t_{\text{init}}$, and while

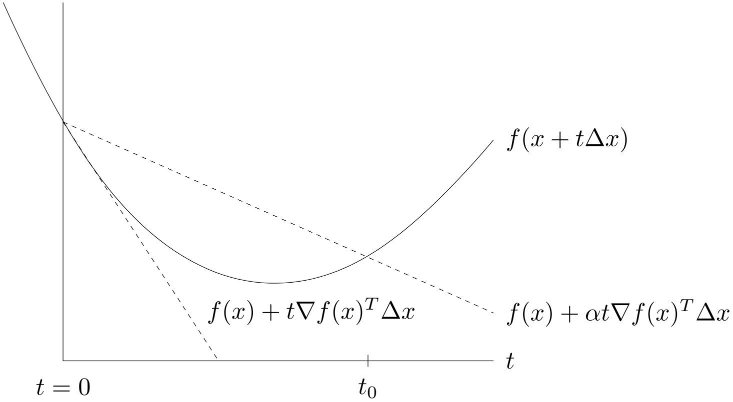
$$f(x - t\nabla f(x)) > f(x) - \alpha t \|\nabla f(x)\|_2^2$$

  shrink $t = \beta t$. Else perform gradient descent update

$$x^+ = x - t\nabla f(x)$$

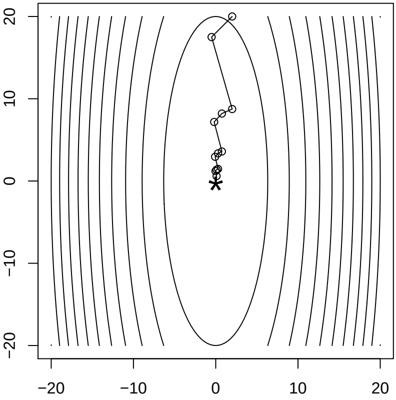
Simple and tends to work well in practice (further simplification: just take $\alpha = 1/2$)

Backtracking interpretation



For us $\Delta x = -\nabla f(x)$

Setting $\alpha = \beta = 0.5$, backtracking picks up roughly the right step size (12 outer steps, 40 steps total),



Exact line search

We could also choose step to do the best we can along direction of negative gradient, called exact line search:

$$t = \underset{s \geq 0}{\operatorname{argmin}} \; f(x - s\nabla f(x))$$

Usually not possible to do this minimization exactly

Approximations to exact line search are typically not as efficient as backtracking, and it's typically not worth it

Convergence analysis

Assume that f convex and differentiable, with $\mathrm{dom}(f) = \mathbb{R}^n$, and additionally that ∇f is Lipschitz continuous with constant $L > 0$,

$$\|\nabla f(x) - \nabla f(y)\|_2 \le L\|x - y\|_2 \quad \text{for any } x, y$$

(Or when twice differentiable: $\nabla^2 f(x) \preceq LI$)

Theorem: Gradient descent with fixed step size $t \le 1/L$ satisfies

$$f(x^{(k)}) - f^\star \le \frac{\|x^{(0)} - x^\star\|_2^2}{2tk}$$

and same result holds for backtracking, with t replaced by β/L

We say gradient descent has convergence rate $O(1/k)$. That is, it finds ϵ-suboptimal point in $O(1/\epsilon)$ iterations

Analysis for strong convexity

Reminder: strong convexity of f means $f(x) - \frac{m}{2}\|x\|_2^2$ is convex for some $m > 0$ (when twice differentiable: $\nabla^2 f(x) \succeq mI$)

Assuming Lipschitz gradient as before, and also strong convexity:

Theorem: Gradient descent with fixed step size $t \le 2/(m+L)$ or with backtracking line search search satisfies

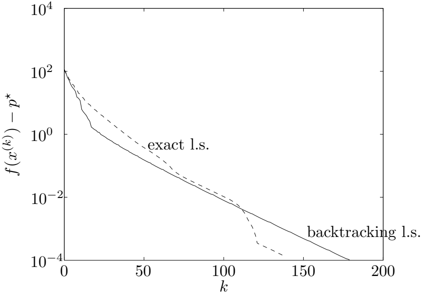
$$f(x^{(k)}) - f^\star \le \gamma^k \frac{L}{2}\|x^{(0)} - x^\star\|_2^2$$

where $0 < \gamma < 1$

Rate under strong convexity is $O(\gamma^k)$, exponentially fast! That is, it finds ϵ-suboptimal point in $O(\log(1/\epsilon))$ iterations

Called linear convergence

Objective versus iteration curve looks linear on semi-log plot



(From B & V page 487)

Important note: $\gamma = O(1 - m/L)$. Thus we can write convergence rate as

$$O\left(\frac{L}{m}\log(1/\epsilon)\right)$$

Higher condition number $L/m \Rightarrow$ slower rate. This is not only true of in theory ... very apparent in practice too

A look at the conditions

A look at the conditions for a simple problem, $f(\beta) = \frac{1}{2}\|y - X\beta\|_2^2$

Lipschitz continuity of ∇f:

- Recall this means $\nabla^2 f(x) \preceq LI$
- As $\nabla^2 f(\beta) = X^T X$, we have $L = \lambda_{\max}(X^T X)$

Strong convexity of f:

- Recall this means $\nabla^2 f(x) \succeq mI$
- As $\nabla^2 f(\beta) = X^T X$, we have $m = \lambda_{\min}(X^T X)$
- If X is wide (X is $n \times p$ with $p > n$), then $\lambda_{\min}(X^T X) = 0$, and f can't be strongly convex
- Even if $\sigma_{\min}(X) > 0$, can have a very large condition number $L/m = \lambda_{\max}(X^T X)/\lambda_{\min}(X^T X)$

Practicalities

Stopping rule: stop when $\|\nabla f(x)\|_2$ is small

- Recall $\nabla f(x^\star) = 0$ at solution $x^\star$
- If f is strongly convex with parameter m, then

$$\|\nabla f(x)\|_2 \leq \sqrt{2m\epsilon} \implies f(x) - f^\star \leq \epsilon$$

Pros and cons of gradient descent:

- Pro: simple idea, and each iteration is cheap (usually)
- Pro: fast for well-conditioned, strongly convex problems
- Con: can often be slow, because many interesting problems aren't strongly convex or well-conditioned
- Con: can't handle nondifferentiable functions

Can we do better?

Gradient descent has $O(1/\epsilon)$ convergence rate over problem class of convex, differentiable functions with Lipschitz gradients

First-order method: iterative method, which updates $x^{(k)}$ in

$$x^{(0)} + \mathrm{span}\{\nabla f(x^{(0)}), \nabla f(x^{(1)}), \ldots \nabla f(x^{(k-1)})\}$$

Theorem (Nesterov): For any $k \leq (n-1)/2$ and any starting point $x^{(0)}$, there is a function f in the problem class such that any first-order method satisfies

$$f(x^{(k)}) - f^\star \geq \frac{3L\|x^{(0)} - x^\star\|_2^2}{32(k+1)^2}$$

Can attain rate $O(1/k^2)$, or $O(1/\sqrt{\epsilon})$? Answer: yes (we'll see)!

Analysis for nonconvex case

Assume f is differentiable with Lipschitz gradient, now nonconvex. Asking for optimality is too much. Let's settle for a ϵ-substationary point x, which means $\|\nabla f(x)\|_2 \le \epsilon$

Theorem: Gradient descent with fixed step size $t \le 1/L$ satisfies

$$\min_{i=0,\dots,k} \|\nabla f(x^{(i)})\|_2 \le \sqrt{\frac{2(f(x^{(0)}) - f^\star)}{t(k+1)}}$$

Thus gradient descent has rate $O(1/\sqrt{k})$, or $O(1/\epsilon^2)$, even in the nonconvex case for finding stationary points

This rate cannot be improved (over class of differentiable functions with Lipschitz gradients) by any deterministic algorithm[1]

[1] Carmon et al. (2017), "Lower bounds for finding stationary points I"

Gradient boosting

The Annals of Statistics
2001, Vol. 29, No. 5, 1189–1232

1999 REITZ LECTURE

GREEDY FUNCTION APPROXIMATION: A GRADIENT BOOSTING MACHINE[1]

BY JEROME H. FRIEDMAN

Stanford University

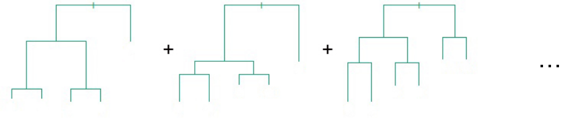
Function estimation/approximation is viewed from the perspective of numerical optimization in function space, rather than parameter space. A connection is made between stagewise additive expansions and steepest-descent minimization. A general gradient descent "boosting" paradigm is developed for additive expansions based on any fitting criterion. Specific algorithms are presented for least-squares, least absolute deviation, and Huber-M loss functions for regression, and multiclass logistic likelihood for classification. Special enhancements are derived for the particular case where the individual additive components are regression trees, and tools for interpreting such "TreeBoost" models are presented. Gradient boosting of regression trees produces competitive, highly robust, interpretable procedures for both regression and classification, especially appropriate for mining less than clean data. Connections between this approach and the boosting methods of Freund and Shapire and Friedman, Hastie and Tibshirani are discussed.

Given responses $y_i \in \mathbb{R}$ and features $x_i \in \mathbb{R}^p$, $i = 1, \ldots, n$

Want to construct a flexible (nonlinear) model for response based on features. Weighted sum of trees:

$$u_i = \sum_{j=1}^{m} \beta_j \cdot T_j(x_i), \quad i = 1, \ldots, n$$

Each tree T_j inputs x_i, outputs predicted response. Typically trees are pretty short



Pick a loss function L to reflect setting. For continuous responses, e.g., could take $L(y_i, u_i) = (y_i - u_i)^2$

Want to solve

$$\min_{\beta} \sum_{i=1}^{n} L\Big(y_i, \sum_{j=1}^{M} \beta_j \cdot T_j(x_i)\Big)$$

Indexes all trees of a fixed size (e.g., depth $= 5$), so M is huge. Space is simply too big to optimize

Gradient boosting: basically a version of gradient descent that is forced to work with trees

First think of optimization as $\min_u \; f(u)$, over predicted values u, subject to u coming from trees

Start with initial model, a single tree $u^{(0)} = T_0$. Repeat:

- Compute negative gradient d at latest prediction $u^{(k-1)}$,

$$d_i = -\left[\frac{\partial L(y_i, u_i)}{\partial u_i}\right]\Bigg|_{u_i = u_i^{(k-1)}}, \quad i = 1, \ldots, n$$

- Find a tree T_k that is close to a, i.e., according to

$$\min_{\text{trees } T} \sum_{i=1}^{n} (d_i - T(x_i))^2$$

  Not hard to (approximately) solve for a single tree

- Compute step size α_k, and update our prediction:

$$u^{(k)} = u^{(k-1)} + \alpha_k \cdot T_k$$

  Note: predictions are weighted sums of trees, as desired

References and further reading

- S. Boyd and L. Vandenberghe (2004), "Convex optimization", Chapter 9
- T. Hastie, R. Tibshirani and J. Friedman (2009), "The elements of statistical learning", Chapters 10 and 16
- Y. Nesterov (1998), "Introductory lectures on convex optimization: a basic course", Chapter 2
- L. Vandenberghe, Lecture notes for EE 236C, UCLA, Spring 2011-2012