Selected Topics in Optimization

Some slides borrowed from
http://www.stat.cmu.edu/~ryantibs/convexopt/
Overview

• Optimization problems are almost everywhere in statistics and machine learning.

\[
\min_x f(x)
\]

Idea/model

Optimization problem: inference model \( x \)
Example

• In a regression model, we want the model to minimize deviation from the dependent variable.
• In a classification model, we want the model to minimize classification error.
• In a generative model, we want to maximize the likelihood to produce the observed data.
• ...

Consider unconstrained, smooth convex optimization

$$\min_x f(x)$$

i.e., $f$ is convex and differentiable with $\text{dom}(f) = \mathbb{R}^n$. Denote the optimal criterion value by $f^* = \min_x f(x)$, and a solution by $x^*$

**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 \( \frac{1}{t} I \)

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

Choose next point \( y = x^+ \) to minimize quadratic approximation:

\[ x^+ = x - t \nabla f(x) \]
Blue point is $x$, red point is

$$x^+ = \arg\min_y f(x) + \nabla f(x)^T (y - x) + \frac{1}{2t} ||y - x||^2_2$$
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, gradient descent after 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 $\Delta x = -\nabla f(x)$
Backtracking picks up roughly the right step size (12 outer steps, 40 steps total):

Here $\alpha = \beta = 0.5$
Practicalities

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

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

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

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
Stochastic gradient descent

Consider minimizing a sum of functions

$$\min_x \sum_{i=1}^{m} f_i(x)$$

As $\nabla \sum_{i=1}^{m} f_i(x) = \sum_{i=1}^{m} \nabla f_i(x)$, gradient descent would repeat:

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

In comparison, stochastic gradient descent or SGD (or incremental gradient descent) repeats:

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

where $i_k \in \{1, \ldots m\}$ is some chosen index at iteration $k$
Two rules for choosing index $i_k$ at iteration $k$:

- **Cyclic rule**: choose $i_k = 1, 2, \ldots, m, 1, 2, \ldots, m, \ldots$
- **Randomized rule**: choose $i_k \in \{1, \ldots, m\}$ uniformly at random

Randomized rule is more common in practice

What’s the difference between stochastic and usual (called batch) methods? Computationally, $m$ stochastic steps $\approx$ one batch step. But what about progress?

- **Cyclic rule, $m$ steps**: $x^{(k+m)} = x^{(k)} - t \sum_{i=1}^{m} \nabla f_i(x^{(k+i-1)})$
- **Batch method, one step**: $x^{(k+1)} = x^{(k)} - t \sum_{i=1}^{m} \nabla f_i(x^{(k)})$
- **Difference in direction** is $\sum_{i=1}^{m} [\nabla f_i(x^{(k+i-1)}) - \nabla f_i(x^{(k)})]$

So SGD should converge if each $\nabla f_i(x)$ doesn’t vary wildly with $x$

Rule of thumb: SGD thrives far from optimum, struggles close to optimum ... (we’ll revisit in just a few lectures)
References and further reading

- L. Vandenberghe, Lecture notes for EE 236C, UCLA, Spring 2011-2012
Convex sets and functions

Convex set: $C \subseteq \mathbb{R}^n$ such that

$$x, y \in C \implies tx + (1 - t)y \in C \text{ for all } 0 \leq t \leq 1$$

Convex function: $f : \mathbb{R}^n \to \mathbb{R}$ such that $\text{dom}(f) \subseteq \mathbb{R}^n$ convex, and

$$f(tx + (1 - t)y) \leq tf(x) + (1 - t)f(y) \text{ for } 0 \leq t \leq 1$$

and all $x, y \in \text{dom}(f)$
Convex optimization problems

Optimization problem:

\[
\min_{x \in D} f(x)
\]

subject to \( g_i(x) \leq 0, \ i = 1, \ldots m \)

\( h_j(x) = 0, \ j = 1, \ldots r \)

Here \( D = \text{dom}(f) \cap \bigcap_{i=1}^{m} \text{dom}(g_i) \cap \bigcap_{j=1}^{p} \text{dom}(h_j) \), common domain of all the functions

This is a convex optimization problem provided the functions \( f \) and \( g_i, i = 1, \ldots m \) are convex, and \( h_j, j = 1, \ldots p \) are affine:

\[
h_j(x) = a_j^T x + b_j, \quad j = 1, \ldots p
\]
Local minima are global minima

For convex optimization problems, local minima are global minima.

Formally, if $x$ is feasible—$x \in D$, and satisfies all constraints—and minimizes $f$ in a local neighborhood,

$$f(x) \leq f(y) \text{ for all feasible } y, \quad \|x - y\|_2 \leq \rho,$$

then

$$f(x) \leq f(y) \text{ for all feasible } y$$

This is a very useful fact and will save us a lot of trouble!
Nonconvex Problem

• Convex problem: convex objective function, convex constraints, convex domain

• Non-convex problem: not all above conditions are met.

• Usually find approximations or local optimum.
Summary

• GD/SGD: both simple implementation
  • SGD: fewer iterations of the whole dataset, fast especially when data size is large; more able to get over local optimums for non-convex problems.
  • GD: less tricky stepsize tuning.

• Second-order methods (e.g. Newton methods, L-BFGS):
  • Simple stepsize tuning; closer to optimum for non-convex problems.
  • More memory cost.