Statistical Learning for Text Data Analytics
Unconstrained Optimization Techniques

Yangqiu Song

Hong Kong University of Science and Technology

yqsong@cse.ust.hk

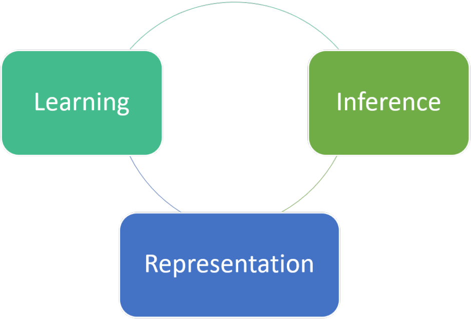
Spring 2018

∗Contents are based on materials created by Peter Richtérik, Mark Schmidt, Francis Bach, Tianbao Yang, Rong Jin, Shenghuo Zhu, and Qihang Lin

Reference Content

- Peter Richtérik and Mark Schmidt. ICML Tutorial on Modern Convex Optimization Methods for Large-scale Empirical Risk Minimization. `https://icml.cc/2015/tutorials/2015_ICML_ConvexOptimization_I.pdf`
- Francis Bach. NIPS 2016 Tutorial on Large-Scale Optimization: Beyond Stochastic Gradient Descent and Convexity. `http://www.di.ens.fr/~fbach/fbach_tutorial_vr_nips_2016.pdf` and `http://www.di.ens.fr/~fbach/ssra_tutorial_vr_nips_2016.pdf`
- Tianbao Yang, Qihang Lin, and Rong Jin. KDD Tutorial on Big Data Analytics: Optimization and Randomization. `http://homepage.cs.uiowa.edu/~tyng/kdd15tutorial.html`
- Tianbao Yang, Rong Jin and Shenghuo Zhu. SDM Tutorial on Stochastic Optimization for Big Data Analytics: Algorithms and Library. `http://homepage.divms.uiowa.edu/~tyng/tutorial.html`

Course Topics



- Representation: language models, word embeddings, topic models
- Learning: supervised learning, semi-supervised learning, sequence models, deep learning, optimization techniques
- Inference: constraint modeling, joint inference, search algorithms

NLP applications: tasks introduced in Lecture 1

Overview

1. Introduction
2. Background
3. Unconstrained Convex Optimization
    - Gradient Based Optimization
    - Stochastic Subgradient
    - Finite-Sum Methods
    - Non-Smooth Objectives
4. Optimization for Neural Networks

Make-up Session

We will have a make-up session on 08 Mar 2018 (Thu), 6:00pm (Rm2408)

Overview

1. Introduction

2. Background

3. Unconstrained Convex Optimization
   - Gradient Based Optimization
   - Stochastic Subgradient
   - Finite-Sum Methods
   - Non-Smooth Objectives

4. Optimization for Neural Networks

Log-Linear Models: Definitions

- We define a conditional log-linear model $P(Y|X)$ as:
  - $\mathcal{Y}$ is the set of events (for language modeling, $\mathcal{V}$)
  - $\mathcal{X}$ is the set of contexts (for n-gram language modeling, $\mathcal{V}^{n-1}$)
  - $\phi : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^d$ is a feature vector function
  - $\mathbf{w} \in \mathbb{R}^d$ are the model parameters

$$P_{\mathbf{w}}(Y = y|X = x) = \frac{\exp(\mathbf{w}^\top \phi(x, y))}{\sum_{y' \in \mathcal{Y}} \exp(\mathbf{w}^\top \phi(x, y'))}$$

* $P_{\mathbf{w}}(Y = y|X = x) \triangleq P(Y = y|X = x, \mathbf{w})$

Breaking It Down

$$P_{\mathbf{w}}(Y = y|X = x) = \frac{\exp(\mathbf{w}^\top \phi(x, y))}{\sum_{y' \in \mathcal{Y}} \exp(\mathbf{w}^\top \phi(x, y'))}$$

- linear score $\mathbf{w}^\top \phi(x, y)$
- nonnegative $\exp(\mathbf{w}^\top \phi(x, y))$
- normalizer $\sum_{y' \in \mathcal{Y}} \exp(\mathbf{w}^\top \phi(x, y')) \triangleq Z_{\mathbf{w}}(x)$
- "Log-linear" comes from the fact that:

$$\log P_{\mathbf{w}}(Y = y|X = x) = \mathbf{w}^\top \phi(x, y) - \underbrace{\log Z_{\mathbf{w}}(x)}_{constant\ in\ y}$$

- This is an instance of the family of generalized linear models

Log-Linear N-Gram Models

$$P_{\mathbf{w}}(Y = y|X = x) = \frac{\exp(\mathbf{w}^\top \phi(x, y))}{\sum_{y' \in \mathcal{Y}} \exp(\mathbf{w}^\top \phi(x, y'))}$$

- Consider an n-gram language model, where $X = \mathcal{V}^{n-1}$ and $Y = \mathcal{V}$. Let $\mathbf{h}_i = \{w_{i-1}, \ldots, w_{i-n+1}\}$, $x_i = w_i$:

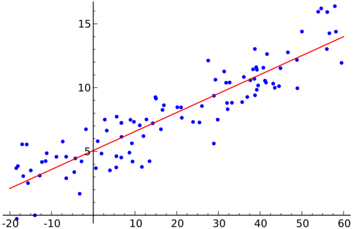
$$P_{\mathbf{w}}(\mathcal{W}) = \prod_{i=1}^{N} P(w_i|w_{i-1}, \ldots, w_{i-n+1})$$

$$\overset{\text{parameterization}}{=} \prod_{i=1}^{N} \frac{\exp\left(\mathbf{w}^\top \phi(\mathbf{h}_i, x_i)\right)}{Z_{\mathbf{w}}(\mathbf{h}_i)}$$

Other Learning Problems: Least Square Regression

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{N} \underbrace{\sum_{i=1}^{N}(y_i - \mathbf{w}^\top \mathbf{x}_i)^2}_{\text{Empirical Loss}} + \underbrace{\frac{\lambda}{2}||\mathbf{w}||_2^2}_{\text{Regularization}}$$

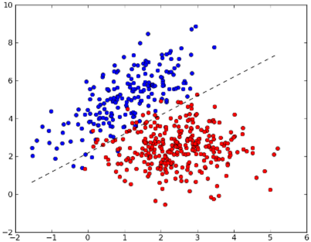
- $\mathbf{x}_i \in \mathbb{R}^d$: d-dimensional feature vector
- y_i: target variable
- $\mathbf{w} \in \mathbb{R}^d$: model parameters
- N: number of data points



Other Learning Problems: Classification

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{N}\sum_{i=1}^{N} \ell(y_i\mathbf{w}^\top\mathbf{x}_i) + \frac{\lambda}{2}||\mathbf{w}||_2^2$$

- $y_i \in \{+1, -1\}$
- Loss function $\ell(z)$, $z = y_i\mathbf{w}^\top\mathbf{x}_i$
  - SVMs: (squared) hinge loss $\ell(z) = \max(0, 1 - z)^p$ where $p = 1, 2$
  - Logistic regression: $\ell(z) = \log(1 + exp(-z))$

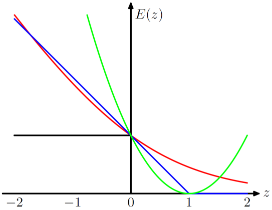


Other Learning Problems: Logistic Regression

- Consider the case where $Y \in \{+1, -1\}$

$$\begin{aligned} P_{\mathbf{w}}(Y = +1 | X = x) &= \frac{\exp(\mathbf{w}^\top \phi(x,+1))}{\exp(\mathbf{w}^\top \phi(x,+1)) + \exp(\mathbf{w}^\top \phi(x,-1))} \\ &= \frac{1}{1 + \exp(\mathbf{w}^\top(\phi(x,-1) - \phi(x,+1)))} \\ &= \sigma(\mathbf{w}^\top \phi(x,+1) - \phi(x,-1)) \\ &\overset{\text{notation change}}{=} \sigma(y\mathbf{w}^\top \mathbf{f}(x)) \end{aligned}$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ is logistic function



Other Learning Problems: Feature Selection

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{N}\sum_{i=1}^{N} \ell(y_i \mathbf{w}^\top \mathbf{x}_i) + \lambda ||\mathbf{w}||_1$$

- ℓ_1 regularization: $||\mathbf{w}||_1 = \sum_{i=1}^{d} |w_i|$
- λ controls sparsity level

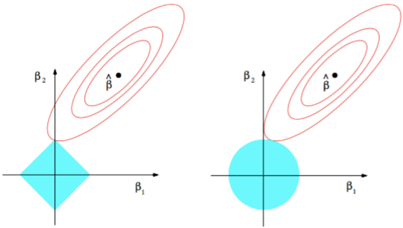


FIGURE 3.11. *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions $|\beta_1| + |\beta_2| \le t$ and $\beta_1^2 + \beta_2^2 \le t^2$, respectively, while the red ellipses are the contours of the least squares error function.*

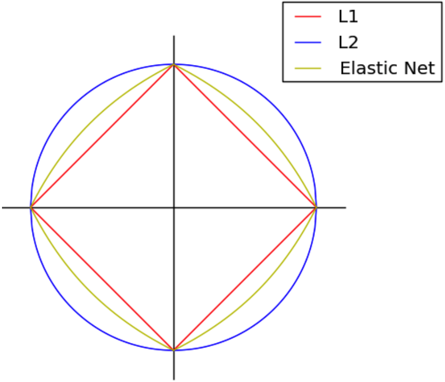
(Elements of Statistical Learning by Hastie, Tibshirani, and Friedman)

Other Learning Problems: Feature Selection

Feature Selection using Elastic Net

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{N}\sum_{i=1}^{N}\ell(y_i\mathbf{w}^\top\mathbf{x}_i) + \lambda(||\mathbf{w}||_1 + \gamma||\mathbf{w}||_2^2)$$

- Elastic net regularizer, more robust than ℓ_1 regularizer



Why (Unconstrained) Stochastic Optimization?

- Big data challenge
  - Google processes 5.13B queries/day (2013)
  - Twitter receives 340M tweets/day (2012)
  - Facebook has 2.5 PB of user data + 15 TB/day (4/2009) (1PB=1015bytes=1000terabytes)
  - eBay has 6.5 PB of user data + 50 TB/day (5/2009)

80% data is unstructured (IBM, 2010)

- Foundation of deep learning optimization

Why Learning from Big Data is Hard??

- Too many data points
  - Issue: can't afford go through data set many times
  - Solution: Stochastic Optimization
- High dimensional data
  - Issue: can't afford second order optimization (Newton's method)
  - Solution: first order method (i.e., gradient based method)

Overview

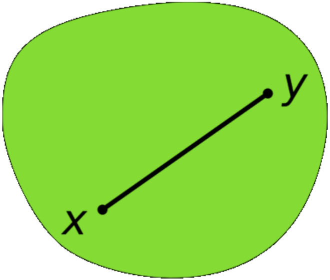
1. Introduction
2. Background
3. Unconstrained Convex Optimization
   - Gradient Based Optimization
   - Stochastic Subgradient
   - Finite-Sum Methods
   - Non-Smooth Objectives
4. Optimization for Neural Networks

Vector, Norm, Inner product, Dual Norm

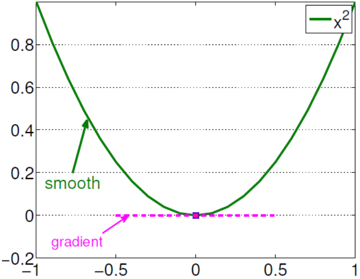
- Bold letters $\mathbf{x} \in \mathbb{R}^d$ (data vector), $\mathbf{w} \in \mathbb{R}^d$ (model parameter): d-dimensional vectors, y_i denotes response variable of ith data
- $\mathbf{w}, \mathbf{w}' \in \mathcal{X}$ finite dimensional variable, $\mathcal{X}$ a normed space
- Norm $\|\mathbf{w}\| : \mathbb{R}^d \to \mathbb{R}_+$, e.g.,
  - ℓ_1 norm: $\|\mathbf{w}\|_1 = \sum_i |w_i|$
  - ℓ_2 norm: $\|\mathbf{w}\|_2 = \sqrt{\sum_i w_i^2}$
  - ℓ_∞ norm: $\|\mathbf{w}\|_\infty = \max_i |w_i|$
- Inner product $\langle \mathbf{w}, \mathbf{w} \rangle = \mathbf{w}^\top \mathbf{w} = \sum_{i=1}^d x_i \cdot w_i$

Convex Optimization

$$\mathbf{w} = \arg\min_{\mathbf{w}\in\mathcal{X}} f(\mathbf{w})$$



(a) $\mathcal{X}$ is a convex domain

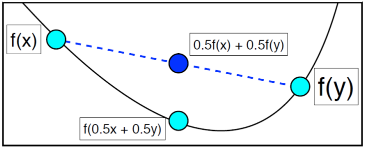


(b) $f(x)$ is a convex function

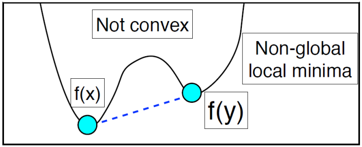
Convex Function: Three Characterizations (1)

A function f is convex if for all $\mathbf{w}_x$ and $\mathbf{w}_y$ we have

$$f(\alpha \mathbf{w}_x + (1-\alpha)\mathbf{w}_y) \leq \alpha f(\mathbf{w}_x) + (1-\alpha) f(\mathbf{w}_y), \forall \mathbf{w}_x, \mathbf{w}_y \in \mathcal{X}, \alpha \in [0,1]$$



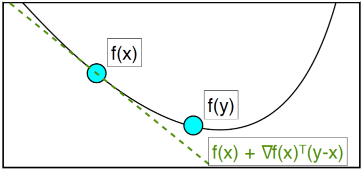
- Function is below linear interpolation between $\mathbf{w}_x$ and $\mathbf{w}_y$
- Implies that all local minima are global minima



Convex Function: Three Characterizations (2)

A function f is convex if for all $\mathbf{w}_x$ and $\mathbf{w}_y$ we have

$$f(\mathbf{w}_y) \geq f(\mathbf{w}_x) + \nabla f(\mathbf{w}_x)^\top(\mathbf{w}_y - \mathbf{w}_x)$$



- The function is globally above the tangent at $\mathbf{w}_x$ (first-order condition, differentiable f with convex domain)
- If $\nabla f(\mathbf{w}_x) = 0$, implies $\mathbf{w}_x$ is a global minimizer

Convex Function: Three Characterizations (3)

A twice-differentiable function f is convex if for all $\mathbf{w}$ we have

$$\nabla^2 f(\mathbf{w}) \succeq 0$$

- All eigenvalues of ‘Hessian” are non-negative
- The function is at or curved upwards in every direction
- This is usually the easiest way to show a function is convex

Convergence Measure

- Most optimization algorithms are iterative

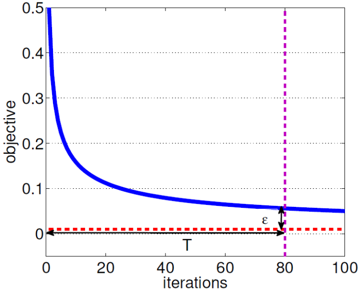
$$\mathbf{w}_{t+1} = \mathbf{w}_t - \Delta \mathbf{w}_t$$

- Convergence Rate: after T iterations, how good is the solution

$$f(\mathbf{w}_T) - \min_{\mathbf{w} \in \mathcal{X}} f(\mathbf{w}) \leq \epsilon(T)$$

- Iteration Complexity: the number of iterations $T(\epsilon)$ needed to have

$$f(\mathbf{w}_T) - \min_{\mathbf{w} \in \mathcal{X}} f(\mathbf{w}) \leq \epsilon \quad (\epsilon \ll 1)$$



- Total Runtime = Per-iteration Cost × Iteration Complexity

More on Convergence Measure

Convergence Rate: after T iterations, how good is the solution
Iteration Complexity: the number of iterations $T(\epsilon)$ needed to have

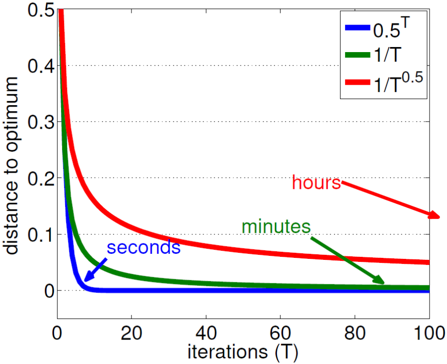
- Big $O(\cdot)$ notation: explicit dependence on T or ϵ

	Convergence Rate	Iteration Complexity
Linear	$O\left(\mu^T\right)$ $(\mu < 1)$	$O\left(\log\left(\frac{1}{\epsilon}\right)\right)$ $(\epsilon \ll 1)$
Sub-linear	$O\left(\frac{1}{T^\alpha}\right)$ $(\alpha > 0)$	$O\left(\log\left(\frac{1}{\epsilon^{1/\alpha}}\right)\right)$ $(\epsilon \ll 1)$

Why are we interested in Bounds?

Why Are We Interested in Bounds?

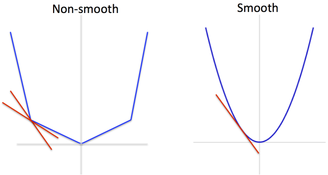
	Convergence Rate	Iteration Complexity
Linear	$O\left(\mu^{T}\right)$ $(\mu < 1)$	$O\left(\log\left(\frac{1}{\epsilon}\right)\right)$ $(\epsilon \ll 1)$
Sub-linear	$O\left(\frac{1}{T^{\alpha}}\right)$ $(\alpha > 0)$	$O\left(\log\left(\frac{1}{\epsilon^{1/\alpha}}\right)\right)$ $(\epsilon \ll 1)$



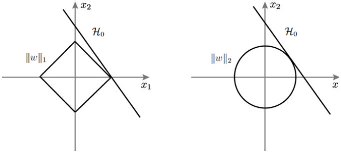
Theoretically, we consider $O\left(\mu^{T}\right) \prec O\left(\frac{1}{T^{2}}\right) \precsim O\left(\frac{1}{T}\right) \precsim O\left(\frac{1}{\sqrt{T}}\right)$

Factors that affect Iteration Complexity

- Property of function: e.g., smoothness of function



- Domain $\mathcal{X}$: size and geometry



- Size of problem: dimension and number of data points

Overview

1. Introduction

2. Background

3. Unconstrained Convex Optimization
   - Gradient Based Optimization
   - Stochastic Subgradient
   - Finite-Sum Methods
   - Non-Smooth Objectives

4. Optimization for Neural Networks

Why In Particular Learn About Convex Optimization?

- Among only efficiently-solvable continuous problems
- You can do a lot with convex models: least squares, lasso, generlized linear models, SVMs, CRFs, etc.
- Empirically effective non-convex methods are often based on methods with good properties for convex objectives (functions are locally convex around minimizers)
- Tools from convex analysis are being extended to non-convex

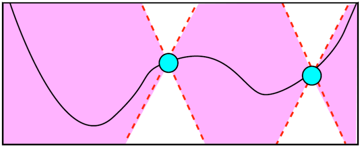
How Hard Is Real-valued Optimization?

How long to find an ϵ-optimal minimizer of a real-valued function?

$$\min_{\mathbf{w}\in\mathbb{R}^d} f(\mathbf{w})$$

- General function: impossible!
- We need to make some assumptions about the function:
  - Assume f is Lipschitz-continuous: (can not change too quickly)

$$|f(\mathbf{w}) - f(\mathbf{w}')| \leq L\|\mathbf{w} - \mathbf{w}'\|$$



- After T iterations, the error of any algorithm is $O(\frac{1}{T^{1/d}})$ (and grid-search is nearly optimal)
- Optimization is hard, but assumptions make a big difference (we went from impossible to very slow)

Motivation for Gradient Methods

- We can solve convex optimization problems in polynomial-time by interior-point methods
- But these solvers require $O(d^2)$ or worse cost per iteration
  - Infeasible for applications where d may be in the billions
- Large-scale problems have renewed interest gradient methods:

$$\mathbf{w}^{t+1} = \mathbf{w}^{t} - \eta^{t} \nabla f(\mathbf{w}^{t})$$

  - Only have $O(d)$ iteration cost!
  - But how many iterations are needed?

Logistic Regression with ℓ_2-Norm Regularization

- Let's consider logistic regression with 2-norm regularization

$$f(\mathbf{w}) = \sum_{i=1}^{N} \log(1 + \exp(-y_i(\mathbf{w}^\top \mathbf{x}_i))) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

- Objective f is convex
- First term is Lipschitz continuous, second term is not
- But we have

$$\mu I \preceq \nabla^2 f(\mathbf{w}) \preceq L I$$

  for some μ and L; I is a diagonal matrix
- We say that the gradient is Lipschitz-continuous
- We say that the function is strongly-convex

Properties of Lipschitz-Continuous Gradient

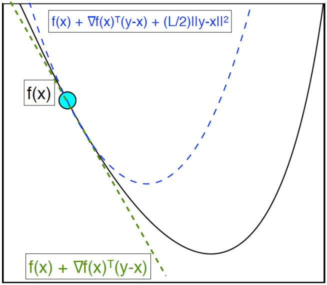
- From Taylor's theorem, for some $\mathbf{z}$ we have:

$$f(\mathbf{w}') = f(\mathbf{w}) + \nabla f(\mathbf{w})^\top(\mathbf{w}' - \mathbf{w}) + \frac{1}{2}(\mathbf{w}' - \mathbf{w})^\top \nabla^2 f(\mathbf{z})(\mathbf{w}' - \mathbf{w})$$

- Use that $\nabla^2 f(\mathbf{w}) \preceq LI$

$$f(\mathbf{w}') \leq f(\mathbf{w}) + \nabla f(\mathbf{w})^\top(\mathbf{w}' - \mathbf{w}) + \frac{L}{2}\|\mathbf{w}' - \mathbf{w}\|^2$$

- Global quadratic upper bound on function value



Properties of Lipschitz-Continuous Gradient

$$f(\mathbf{w}') \leq f(\mathbf{w}) + \nabla f(\mathbf{w})^\top (\mathbf{w}' - \mathbf{w}) + \frac{L}{2}\|\mathbf{w}' - \mathbf{w}\|^2$$

- Variant of gradient method if we set $\mathbf{w}^{t+1}$ to minimum $\mathbf{w}'$ value (right side gradient $(\mathbf{w}') = 0$):

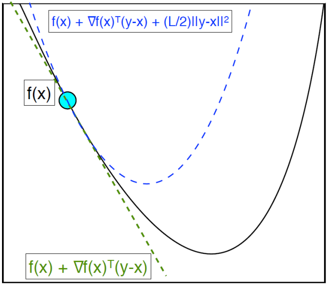
$$\begin{aligned} \mathbf{w}^{t+1} &= \arg\min_{\mathbf{w}'}\{f(\mathbf{w}^t) + \nabla f(\mathbf{w}^t)^\top (\mathbf{w}' - \mathbf{w}^t) + \frac{L}{2}\|\mathbf{w}' - \mathbf{w}^t\|^2\} \\ \Rightarrow \mathbf{w}^{t+1} &= \mathbf{w}^t - \frac{1}{L}\nabla f(\mathbf{w}^t) \end{aligned}$$

- Plugging this value in:

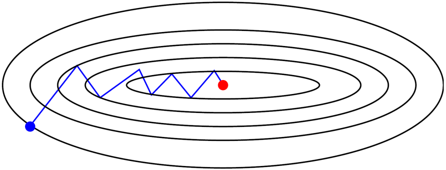
$$f(\mathbf{w}^{t+1}) \leq f(\mathbf{w}^t) - \frac{1}{2L}\|\nabla f(\mathbf{w}^t)\|^2$$

Properties of Lipschitz-Continuous Gradient

Guaranteed decrease of objective



$$f(\mathbf{w}^{t+1}) \leq f(\mathbf{w}^t) - \frac{1}{2L}\|\nabla f(\mathbf{w}^t)\|^2$$



Properties of Strong-Convexity

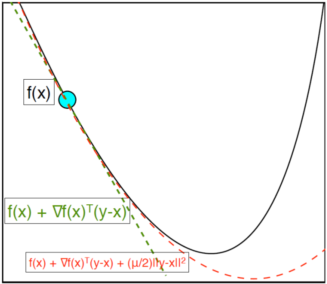
- From Taylor's theorem, for some $\mathbf{z}$ we have:

$$f(\mathbf{w}') = f(\mathbf{w}) + \nabla f(\mathbf{w})^\top(\mathbf{w}' - \mathbf{w}) + \frac{1}{2}(\mathbf{w}' - \mathbf{w})^\top \nabla^2 f(\mathbf{z})(\mathbf{w}' - \mathbf{w})$$

- Use that $\mu I \preceq \nabla^2 f(\mathbf{w})$

$$f(\mathbf{w}) + \nabla f(\mathbf{w})^\top(\mathbf{w}' - \mathbf{w}) + \frac{\mu}{2}\|\mathbf{w}' - \mathbf{w}\|^2 \leq f(\mathbf{w}')$$

- Global quadratic lower bound on function value



Properties of Strong-Convexity

$$f(\mathbf{w}) + \nabla f(\mathbf{w})^\top (\mathbf{w}' - \mathbf{w}) + \frac{\mu}{2}\|\mathbf{w}' - \mathbf{w}\|^2 \leq f(\mathbf{w}')$$

- Minimizing both sides in terms of $\mathbf{w}'$ gives

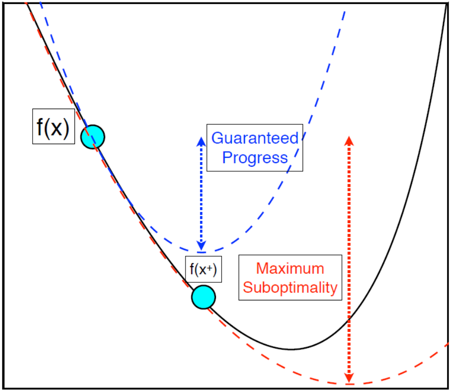
$$f(\mathbf{w}^*) \geq f(\mathbf{w}^t) - \frac{1}{2\mu}\|\nabla f(\mathbf{w}^t)\|^2$$

- Upper bound on how far we are from the solution

Linear Convergence of Gradient Descent

- We have bounds on $\mathbf{w}^{t+1}$ and $\mathbf{w}^*$:

$$f(\mathbf{w}^{t+1}) \leq f(\mathbf{w}^t) - \frac{1}{2L}\|\nabla f(\mathbf{w}^t)\|^2 \qquad f(\mathbf{w}^*) \geq f(\mathbf{w}^t) - \frac{1}{2\mu}\|\nabla f(\mathbf{w}^t)\|^2$$



Linear Convergence of Gradient Descent

- We have bounds on $\mathbf{w}^{t+1}$ and $\mathbf{w}^*$:

$$f(\mathbf{w}^{t+1}) \leq f(\mathbf{w}^t) - \frac{1}{2L}\|\nabla f(\mathbf{w}^t)\|^2 \qquad f(\mathbf{w}^*) \geq f(\mathbf{w}^t) - \frac{1}{2\mu}\|\nabla f(\mathbf{w}^t)\|^2$$

- Combine them we have:

$$f(\mathbf{w}^{t+1}) - f(\mathbf{w}^*) \leq \left(1 - \frac{\mu}{L}\right)[f(\mathbf{w}^t) - f(\mathbf{w}^*)]$$

- This gives a linear convergence rate:

$$f(\mathbf{w}^t) - f(\mathbf{w}^*) \leq \left(1 - \frac{\mu}{L}\right)^t [f(\mathbf{w}^0) - f(\mathbf{w}^*)]$$

- Each iteration multiplies the error by a fixed amount (very fast if μ/L is not too close to zero)

Maximum Likelihood Logistic Regression

- What about maximum-likelihood logistic regression?

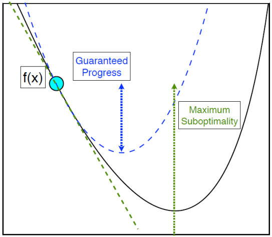
$$f(\mathbf{w}) = \sum_{i=1}^{N} \log(1 + \exp(-y_i(\mathbf{w}^\top \mathbf{x}_i)))$$

- We now only have

$$0 \preceq \nabla^2 f(\mathbf{w}) \preceq LI$$

- Convexity only gives a linear upper bound on $f(\mathbf{w}^*)$:

$$f(\mathbf{w}^*) \geq f(\mathbf{w}) + \nabla f(\mathbf{w})^\top (\mathbf{w}^* - \mathbf{w})$$



Maximum Likelihood Logistic Regression

- What about maximum-likelihood logistic regression?

$$f(\mathbf{w}) = \sum_{i=1}^{N} \log(1 + \exp(-y_i(\mathbf{w}^\top \mathbf{x}_i)))$$

- We now only have

$$0 \preceq \nabla^2 f(\mathbf{w}) \preceq LI$$

- Convexity only gives a linear upper bound on $f(\mathbf{w}^*)$:

$$f(\mathbf{w}^*) \geq f(\mathbf{w}) + \nabla f(\mathbf{w})^\top (\mathbf{w}^* - \mathbf{w})$$

- If $\mathbf{w}^*$ exists, we have the sublinear convergence rate:

$$f(\mathbf{w}^t) - f(\mathbf{w}^*) = O(1/t)$$

  (compare to slower $O(\frac{1}{T^{1/d}})$ for general Lipschitz functions)

  Proof: http://www.stat.cmu.edu/~ryantibs/convexopt-F13/scribes/lec6.pdf

- If f is convex, then $f + \lambda\|\mathbf{w}\|^2$ is strongly-convex.

Gradient Method: Practical Issues

- In practice, searching for step size (line-search) is usually much faster than $\eta = 1/L$ (and doesn't require knowledge of L)
- Basic Armijo backtracking line-search:
  1. Start with a large value of η
  2. Divide η in half until we satisfy (typically value is $\gamma = 0.0001$):

  $$f(\mathbf{w}^{t+1}) \leq f(\mathbf{w}^t) - \gamma\eta\|\nabla f(\mathbf{w}^t)\|^2$$

  - Practical methods may use Wolfe conditions (so η isn't too small), and/or use interpolation to propose trial step sizes.
  - https://en.wikipedia.org/wiki/Wolfe_conditions
- Also, check your derivative code:

$$\nabla f_i(\mathbf{w}) \approx \frac{f(\mathbf{w} + \delta \mathbf{e}_i) - f(\mathbf{w})}{\delta}$$

- For large-scale problems you can check a random direction **d**:

$$\nabla f(\mathbf{w})^\top \mathbf{d} \approx \frac{f(\mathbf{w} + \delta \mathbf{d}) - f(\mathbf{w})}{\delta}$$

Accelerated Gradient Method

- Is this the best algorithm under these assumptions?
- Nesterov's accelerated gradient method (Nesterov (1983)):

$$\mathbf{w}^{t+1} = \mathbf{v}^{t} - \eta^{t} \nabla f(\mathbf{v}^{t}), \quad \mathbf{v}^{t+1} = \mathbf{w}^{t} + \beta^{t}(\mathbf{w}^{t+1} - \mathbf{w}^{t})$$

Algorithm	Assumptions	Rate
Gradient	Convex	$O(\frac{1}{T})$
Nesterov	Convex	$O(\frac{1}{T^2})$
Gradient	Strongly-Convex	$O((1-\mu/L)^{\top})$
Nesterov	Strongly-Convex	$O((1-\sqrt{\mu/L})^{\top})$

- For logistic regression and many other losses, we can get linear convergence without strong-convexity (Luo and Tseng (1993))
- Nesterovs method is much more general than linear CG
- Linear CG is much faster than Nesterov for convex quadratics
- Nonlinear CG is typically faster than Nesterov but has no complexity guarantees and sometimes fails

Newton's Method

- The oldest differentiable optimization method is Newton's
- Modern form uses the update

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta\mathbf{d}$$

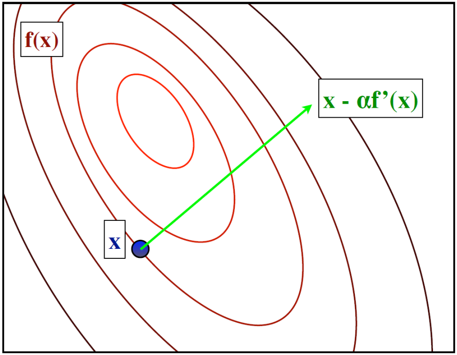
where $\mathbf{d}$ is a solution to the system

$$\nabla^2 f(\mathbf{w})\mathbf{d} = \nabla f(\mathbf{w}) \; (\nabla^2 f(\mathbf{w}) \succ 0)$$

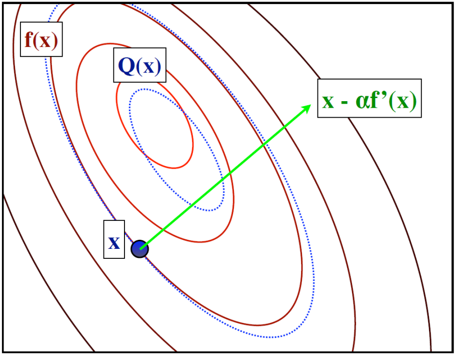
- Equivalent to minimizing the quadratic approximation:

$$f(\mathbf{w}') \approx f(\mathbf{w}) + \nabla f(\mathbf{w})^\top(\mathbf{w}' - \mathbf{w}) + \frac{1}{2\eta}\|\mathbf{w}' - \mathbf{w}\|^2_{\nabla^2 f(\mathbf{w})} \; (\|\mathbf{w}\|^2_{\mathbf{H}} = \mathbf{w}^\top \mathbf{H}\mathbf{w})$$

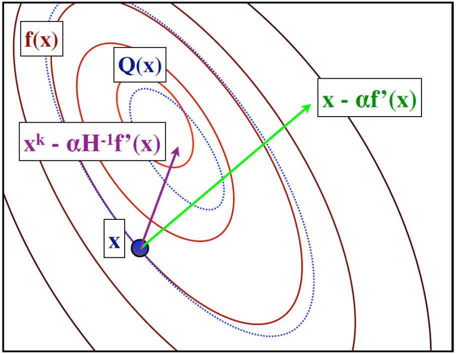
Newton's Method



Newton's Method



Newton’s Method



Convergence Rate of Newton's Method

- If $\nabla^2 f(\mathbf{w})$ is Lipschitz-continuous and $\nabla^2 f(\mathbf{w}) \succeq \mu$, then close to $\mathbf{w}^*$ Newton's method has local superlinear convergence:

  $$f(\mathbf{w}^{t+1} - \mathbf{w}^*) \leq \rho^t [f(\mathbf{w}^t - \mathbf{w}^*)]$$

  with $\lim_{t \to \infty} \rho^t = 0$
- Converges very fast, use it if you can!
- But requires solving $\nabla^2 f(\mathbf{w})\mathbf{d} = \nabla f(\mathbf{w})$

Newton's Method: Practical Issues

There are many practical variants of Newton's method:

- Modify the Hessian to be positive-definite.
- Only compute the Hessian every m iterations.
- Only use the diagonals of the Hessian.
- Quasi-Newton: Update a (diagonal plus low-rank) approximation of the Hessian (BFGS, L-BFGS).
- Hessian-free: Compute **d** inexactly using Hessian-vector products:

$$\nabla^2 f(\mathbf{w})\mathbf{d} \approx \lim_{\delta \to 0} \frac{\nabla f(\mathbf{w} + \delta \mathbf{d}) - \nabla f(\mathbf{w})}{\delta}$$

References I

Luo, Z.-Q. and Tseng, P. (1993). Error bounds and convergence analysis of feasible descent methods: A general approach. *Annals of Operations Research*, 46-47(1):157–178.

Nesterov, Y. (1983). A method of solving a convex programming problem with convergence rate o(1/k2). *Soviet Mathematics Doklady*, 27:372–376.