Stochastic Optimization for Big Data Analytics

Tianbao Yang[‡], Rong Jin[†], Shenghuo Zhu[‡]

Tutorial@SDM 2014
Philadelphia, Pennsylvania

[‡]NEC Laboratories America, [†]Michigan State University

April 26, 2014

The updates are available here

http://www.cse.msu.edu/~yangtia1/sdm14-tutorial.pdf

Thanks.

Some Claims

No

- This tutorial is not an exhaustive literature survey
- The algorithms are not necessary the best for small data
- The theories may not carry over to non-convex optimization

Yes

- start-of-the-art Stochastic Optimization for SVM, Logistic Regression, Least Square Regression, LASSO
- A Generic Distributed Library

Outline

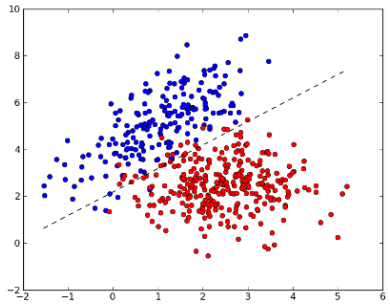
Outline

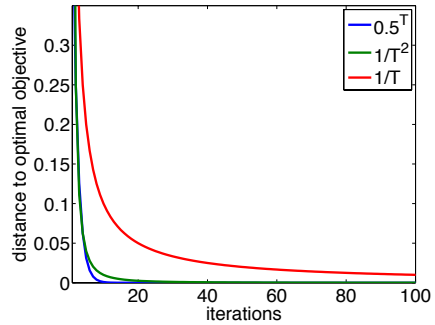
Introduction

Machine Learning problems and Stochastic Optimization

- Classification and Regression in different forms
- Motivation to employ STochastic OPtimization (STOP)
- Basic Convex Optimization Knowledge

Three Steps for Machine Learning and Pattern Recognition



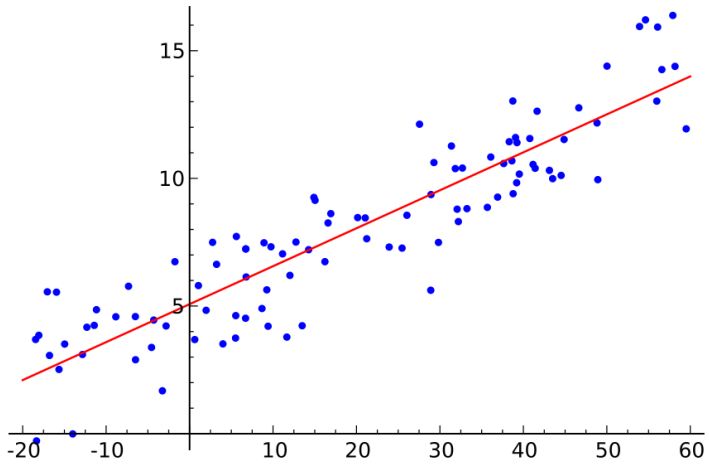


Data → Model → Optimization → Data

Learning as Optimization

Least Square Regression Problem:

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

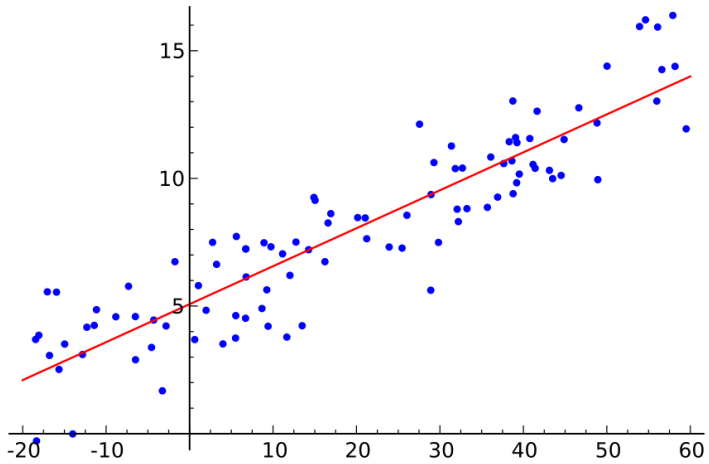


- $\mathbf{x}_i \in \mathbb{R}^d$: d-dimensional feature vector
- $y_i \in \mathbb{R}$: target variable
- $\mathbf{w} \in \mathbb{R}^d$: model parameters
- n: number of data points

Learning as Optimization

Least Square Regression Problem:

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

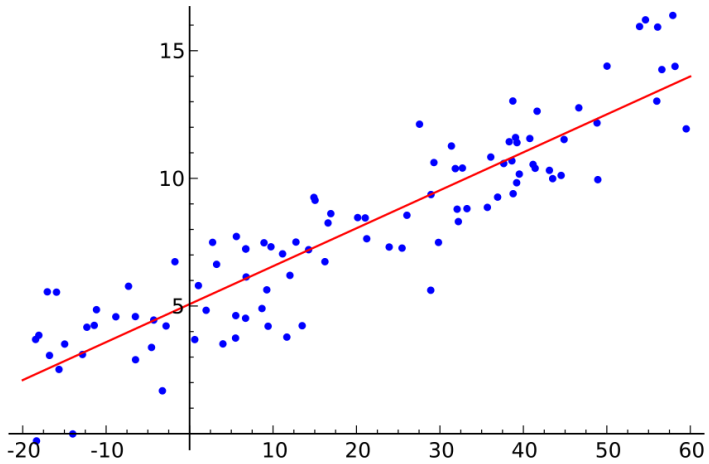


- $\mathbf{x}_i \in \mathbb{R}^d$: d-dimensional feature vector
- $y_i \in \mathbb{R}$: target variable
- $\mathbf{w} \in \mathbb{R}^d$: model parameters
- n: number of data points

Learning as Optimization

Least Square Regression Problem:

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

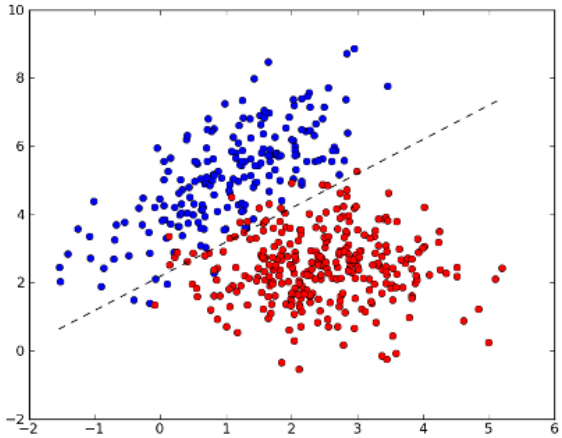


- $\mathbf{x}_i \in \mathbb{R}^d$: d-dimensional feature vector
- $y_i \in \mathbb{R}$: target variable
- $\mathbf{w} \in \mathbb{R}^d$: model parameters
- n: number of data points

Learning as Optimization

Classification Problems:

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

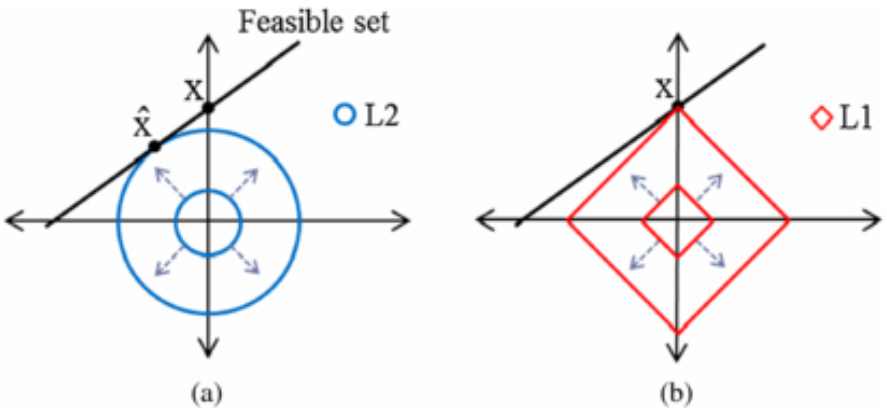


- $y_i \in \{+1, -1\}$: label
- Loss function $\ell(z)$: $z = y\mathbf{w}^\top \mathbf{x}$
  1. SVMs: (squared) hinge loss $\ell(z) = \max(0, 1-z)^p$, where $p = 1, 2$
  2. Logistic Regression: $\ell(z) = \log(1 + \exp(-z))$

Learning as Optimization

Feature Selection:

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

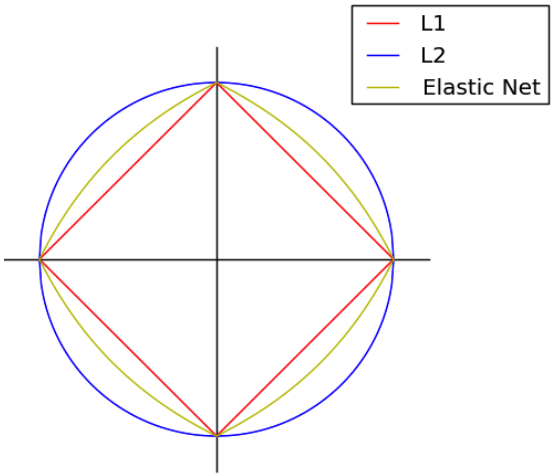


- ℓ_1 regularization $\|\mathbf{w}\|_1 = \sum_{i=1}^{d} |w_i|$
- λ controls sparsity level

Learning as Optimization

Feature Selection using Elastic Net:

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



- Elastic net regularizer, more robust than ℓ_1 regularizer

Learning as Optimization

Multi-class/Multi-task Learning:

$$\min_{\mathbf{W}} \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{W}\mathbf{x}_i, y_i) + \lambda R(\mathbf{W})$$

- $\mathbf{W} \in \mathbb{R}^{K \times d}$
- $R(\mathbf{W}) = \|\mathbf{W}\|_F^2 = \sum_{k=1}^{K} \sum_{j=1}^{d} W_{kj}^2$: Frobenius Norm
- $R(\mathbf{W}) = \|\mathbf{W}\|_* = \sum_i \sigma_i$: Nuclear Norm (sum of singular values)
- $R(\mathbf{W}) = \|\mathbf{W}\|_{1,\infty} = \sum_{j=1}^{d} \|W_{:j}\|_\infty$: $\ell_{1,\infty}$mixed norm

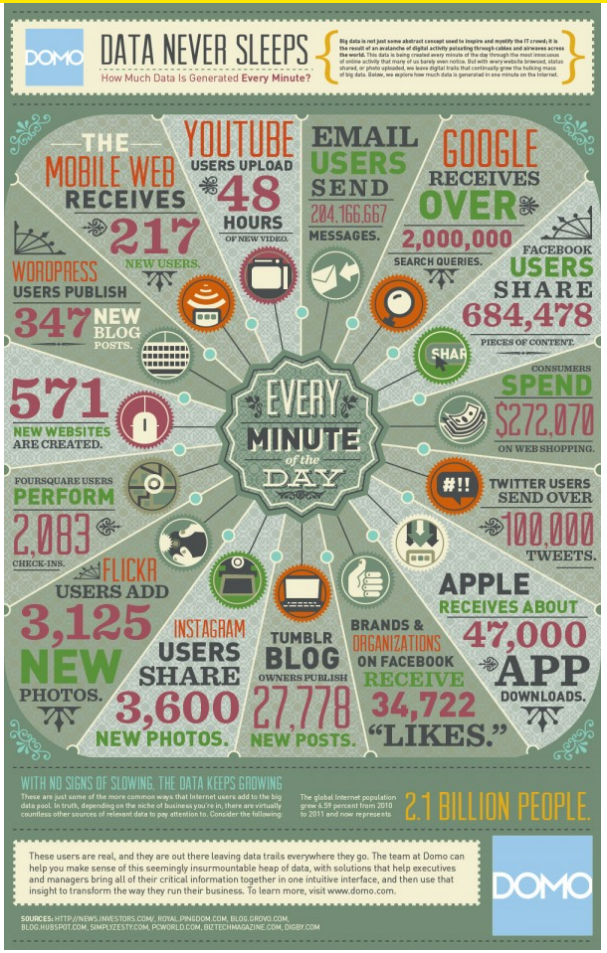
Extensions to Matrix Cases are possible

Big Data Challenge

Huge amount of data generated every day

- Facebook users upload 3 million photos
- Goolge receives 3 billion queries
- Youtube users upload over 1,700 hours video
- Global internet population is 2.1 billion people
- 247 billion emails sent

Data Analytics



http://www.visualnews.com/2012/06/19/how-much-data-created-every-minute/

Why Learning from Big Data is Hard?

$$\min_{\mathbf{w}\in\mathbb{R}^d} \underbrace{\frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{w}^\top \mathbf{x}_i, y_i) + \lambda R(\mathbf{w})}_{\text{empirical loss + regularizer}}$$

Too many data points

- Issue: can't afford go through data set many times
- Solution: Stochastic Optimization

Why Learning from Big Data is Hard?

$$\min_{\mathbf{w}\in\mathbb{R}^d} \underbrace{\frac{1}{n}\sum_{i=1}^{n}\ell(\mathbf{w}^\top\mathbf{x}_i, y_i) + \lambda R(\mathbf{w})}_{\text{empirical loss + regularizer}}$$

High dimensional data

- Issue: can't afford second order optimization (Newton's method)
- Solution: first order method (i.e, gradient based method)

Why Learning from Big Data is Hard?

$$\min_{\mathbf{w}\in\mathbb{R}^d} \underbrace{\frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{w}^\top \mathbf{x}_i, y_i) + \lambda R(\mathbf{w})}_{\text{empirical loss + regularizer}}$$

Data are distributed over many machines

- Issue: expensive (if not impossible) to move data
- Solution: Distributed Optimization

Stochastic Optimization

Stochastic Optimization:

$$\min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w}) = \mathrm{E}_\xi[f(\mathbf{w};\xi)]$$

- $f(\mathbf{w};\xi)$ is convex, $F(\mathbf{w})$ is convex
- ξ random variable

Methods:

1. Sample Average Approximation, $\xi_1, \ldots, \xi_n$

$$\min_{\mathbf{w}\in\mathcal{W}} \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{w};\xi_i)$$

2. Stochastic Approximation: $\nabla f(\mathbf{w};\xi)$ (stochastic gradient)

Stochastic Optimization

Stochastic Optimization:

$$\min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w}) = \mathrm{E}_\xi[f(\mathbf{w};\xi)]$$

- $f(\mathbf{w};\xi)$ is convex, $F(\mathbf{w})$ is convex
- ξ random variable

Methods:

1. Sample Average Approximation, $\xi_1, \ldots, \xi_n$

$$\min_{\mathbf{w}\in\mathcal{W}} \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{w};\xi_i)$$

2. Stochastic Approximation: $\nabla f(\mathbf{w};\xi)$ (stochastic gradient)

Machine Learning is Stochastic Optimization

Goal:

$$\min_{\mathbf{w}\in\mathcal{W}} \mathrm{E}_{\xi=(\mathbf{x},y)}[\mathrm{Loss}(\mathbf{w}^\top \mathbf{x}, y)]$$

Empirical Regularized Loss Minimization

$$\min_{\mathbf{w}\in\mathcal{W}} \underbrace{\frac{1}{n}\sum_{i=1}^{n}}_{\mathrm{E}_{\xi=i}} \Big[\mathrm{Loss}(\mathbf{w}^\top \mathbf{x}_i, y_i) + \lambda R(\mathbf{w})\Big]$$

Machine Learning is Stochastic Optimization

Goal:

$$\min_{\mathbf{w}\in\mathcal{W}} \mathrm{E}_{\xi=(\mathbf{x},y)}[\mathrm{Loss}(\mathbf{w}^\top\mathbf{x}, y)]$$

Empirical Regularized Loss Minimization

$$\min_{\mathbf{w}\in\mathcal{W}} \underbrace{\frac{1}{n}\sum_{i=1}^{n}}_{\mathrm{E}_{\xi=i}} \Big[\mathrm{Loss}(\mathbf{w}^\top\mathbf{x}_i, y_i) + \lambda R(\mathbf{w})\Big]$$

Machine Learning is Stochastic Optimization

Goal:

$$\min_{\mathbf{w}\in\mathcal{W}} \mathrm{E}_{\xi=(\mathbf{x},y)}[\mathrm{Loss}(\mathbf{w}^\top\mathbf{x}, y)]$$

Empirical Regularized Loss Minimization

$$\min_{\mathbf{w}\in\mathcal{W}} \underbrace{\frac{1}{n}\sum_{i=1}^{n}}_{\mathrm{E}_{\xi=i}} \Big[\mathrm{Loss}(\mathbf{w}^\top\mathbf{x}_i, y_i) + \lambda R(\mathbf{w})\Big]$$

The Simplest Method for Stochastic Optimization

Stochastic Optimization:

$$\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) = \mathrm{E}_\xi[f(\mathbf{w};\xi)]$$

Stochastic Gradient Descent (Nemirovski & Yudin, 1978)

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \gamma_t \nabla f(\mathbf{w}_{t-1};\xi_t)$$

Stochastic Gradient

$$\mathrm{E}_{\xi_t}[\nabla f(\mathbf{w};\xi_t)] = \nabla F(\mathbf{w})$$

The Simplest Method for Stochastic Optimization

Stochastic Optimization:

$$\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) = \mathrm{E}_\xi[f(\mathbf{w};\xi)]$$

Stochastic Gradient Descent (Nemirovski & Yudin, 1978)

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \gamma_t \nabla f(\mathbf{w}_{t-1};\xi_t)$$

Stochastic Gradient

$$\mathrm{E}_{\xi_t}[\nabla f(\mathbf{w};\xi_t)] = \nabla F(\mathbf{w})$$

The Simplest Method for Stochastic Optimization

Stochastic Optimization:

$$\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) = \mathrm{E}_\xi[f(\mathbf{w};\xi)]$$

step size

Stochastic Gradient Descent (Nemirovski & Yudin, 1978)

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \gamma_t \nabla f(\mathbf{w}_{t-1};\xi_t)$$

Stochastic Gradient

$$\mathrm{E}_{\xi_t}[\nabla f(\mathbf{w};\xi_t)] = \nabla F(\mathbf{w})$$

Stochastic Gradient in Machine Learning

$$F(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{w}^\top \mathbf{x}_i, y_i) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

- let $i_t \in \{1, \ldots, n\}$ uniformly randomly sampled
- key equation: $\mathrm{E}_{i_t}[\nabla \ell(\mathbf{w}^\top \mathbf{x}_{i_t}, y_{i_t}) + \lambda \mathbf{w}] = \nabla F(\mathbf{w})$
- computation is very cheap $O(d)$ compared with full gradient $O(nd)$

$$\mathbf{w}_t = (1 - \gamma_t \lambda)\mathbf{w}_{t-1} - \gamma_t \nabla \ell(\mathbf{w}_{t-1}^\top \mathbf{x}_{i_t}, y_{i_t})$$

Warm-up: 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
- $x, y \in \mathcal{X}$ finite dimensional variable, $\mathcal{X}$ a normed space

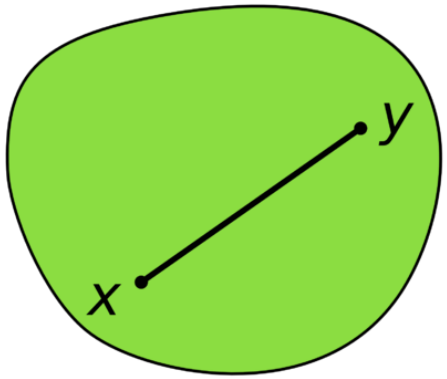
- norm $\|\mathbf{w}\|$: $\mathbb{R}^d \to \mathbb{R}_+$. e.g.
  1. ℓ_2 norm $\|\mathbf{w}\|_2 = \sqrt{\sum_{i=1}^d w_i^2}$
  2. ℓ_1 norm $\|\mathbf{w}\|_1 = \sum_{i=1}^d |w_i|$
  3. ℓ_∞ norm $\|\mathbf{w}\|_\infty = \max_i |w_i|$

- inner product $\langle \mathbf{x}, \mathbf{w} \rangle = \mathbf{x}^\top \mathbf{w} = \sum_{i=1}^d x_i w_i$

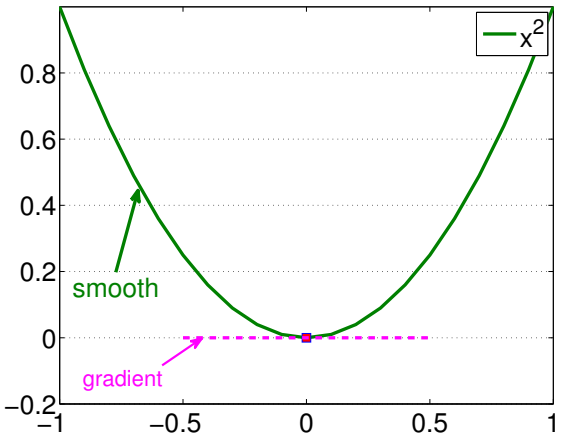
- dual norm $\|\mathbf{w}\|_* = \max_{\|\mathbf{x}\| \le 1} \mathbf{x}^\top \mathbf{w}$.
  1. $\|\mathbf{x}\|_2 \Longleftrightarrow \|\mathbf{w}\|_2$
  2. $\|\mathbf{x}\|_1 \Longleftrightarrow \|\mathbf{w}\|_\infty$

Warm-up: Convex Optimization

$$\min_{x \in \mathcal{X}} f(x)$$



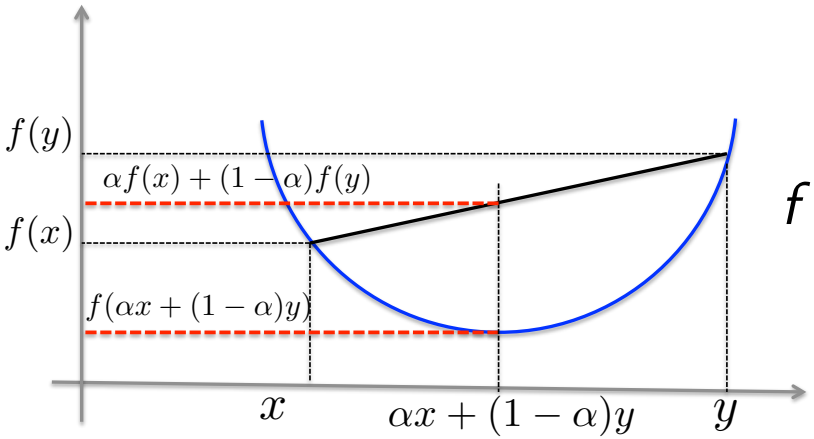
- $\mathcal{X}$ is a convex domain



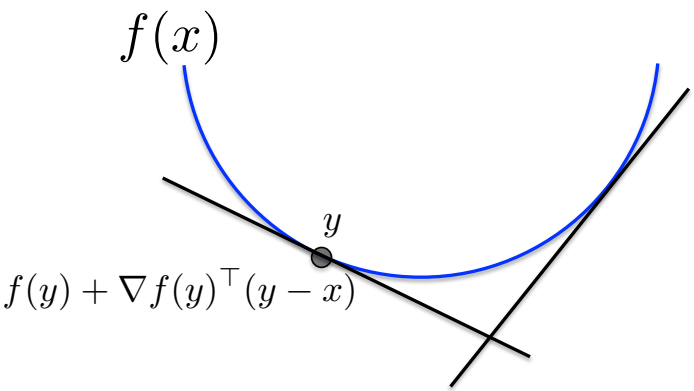
- $f(x)$ is a convex function

Warm-up: Convex Function

Characterization of Convex Function



$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y),$$
$$\forall x, y \in \mathcal{X}, \alpha \in [0,1]$$

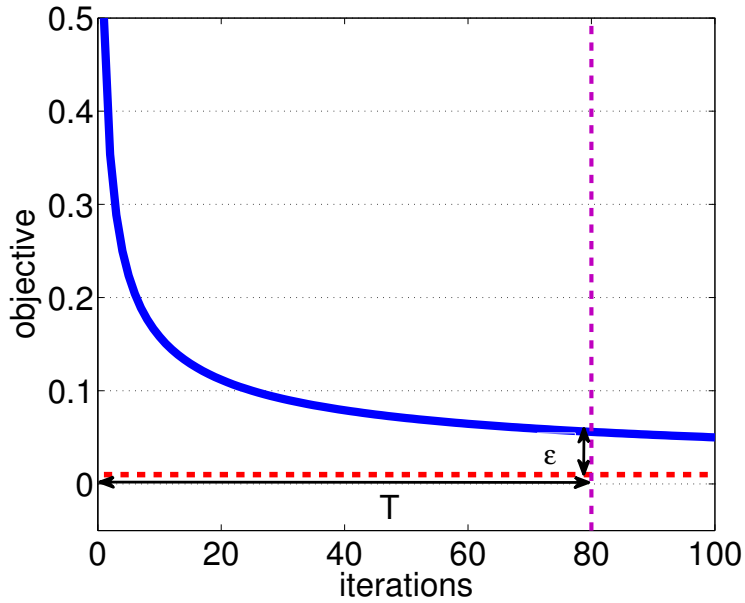


$$f(x) \geq f(y) + \nabla f(y)^\top (x - y) \, \forall x, y \in \mathcal{X}$$

Warm-up: Convergence Measure

- Most optimization algorithms are iterative

$$x_{t+1} = x_t + \Delta x_t$$



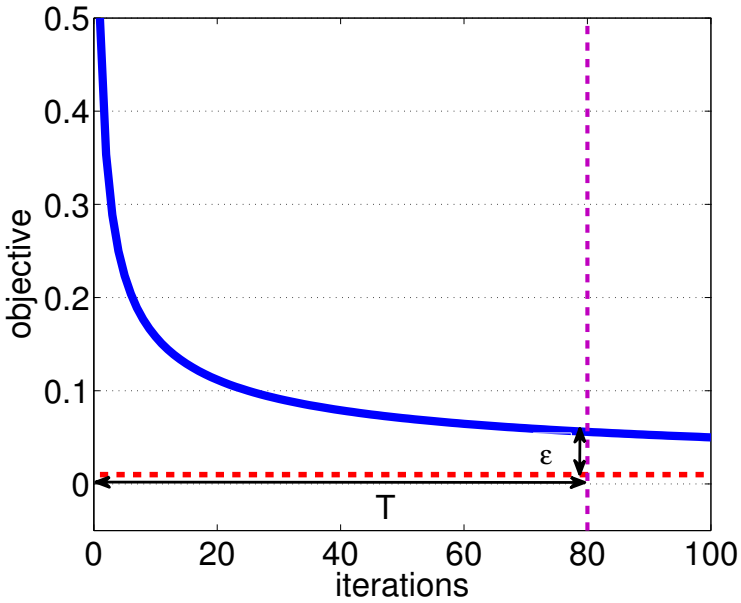
Warm-up: Convergence Measure

- Most optimization algorithms are iterative

$$x_{t+1} = x_t + \Delta x_t$$

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

$$f(x_T) - \min_{x \in \mathcal{X}} f(x) \leq \epsilon \quad (\epsilon \ll 1)$$



Warm-up: Convergence Measure

- Most optimization algorithms are iterative

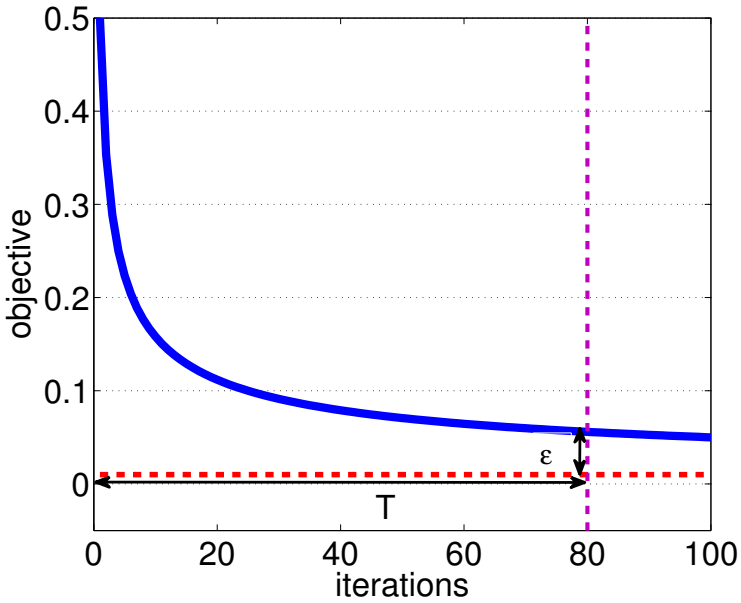
$$x_{t+1} = x_t + \Delta x_t$$

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

$$f(x_T) - \min_{x \in \mathcal{X}} f(x) \leq \epsilon \quad (\epsilon \ll 1)$$

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

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



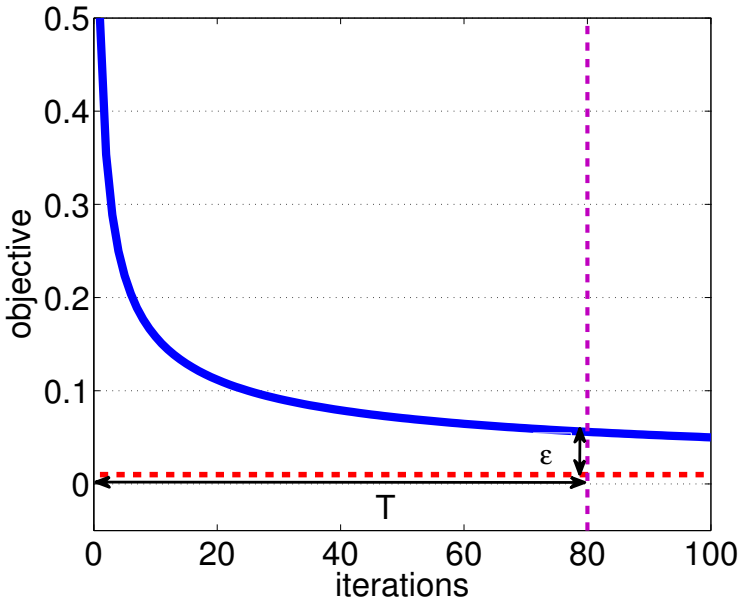
Warm-up: Convergence Measure

- Most optimization algorithms are iterative

$$x_{t+1} = x_t + \Delta x_t$$

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

$$f(x_T) - \min_{x \in \mathcal{X}} f(x) \leq \epsilon \quad (\epsilon \ll 1)$$



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

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

- Total Runtime = Per-iteration Cost × Iteration Complexity

More on Convergence Measure

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

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

Why are we interested in Bounds?

More on Convergence Measure

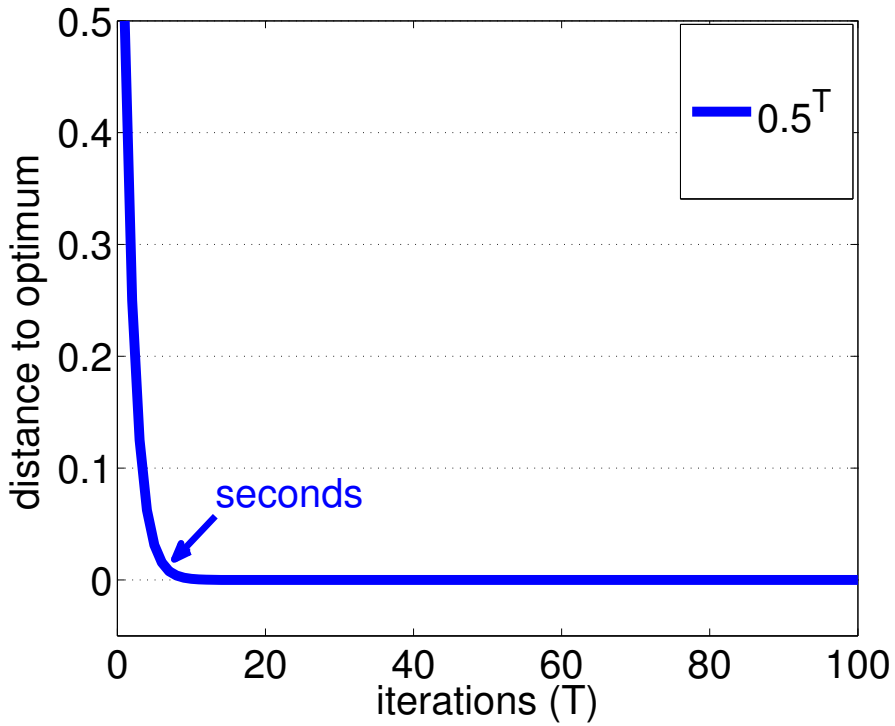
- Big $O(\cdot)$ notation: explicit dependence on T or ϵ

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

Why are we interested in Bounds?

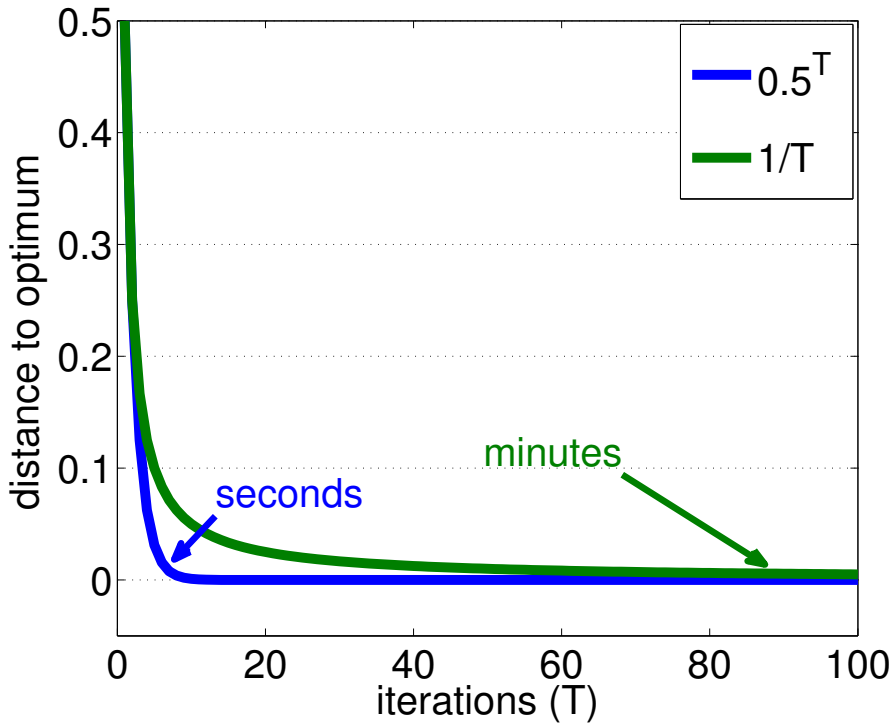
More on Convergence Measure

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



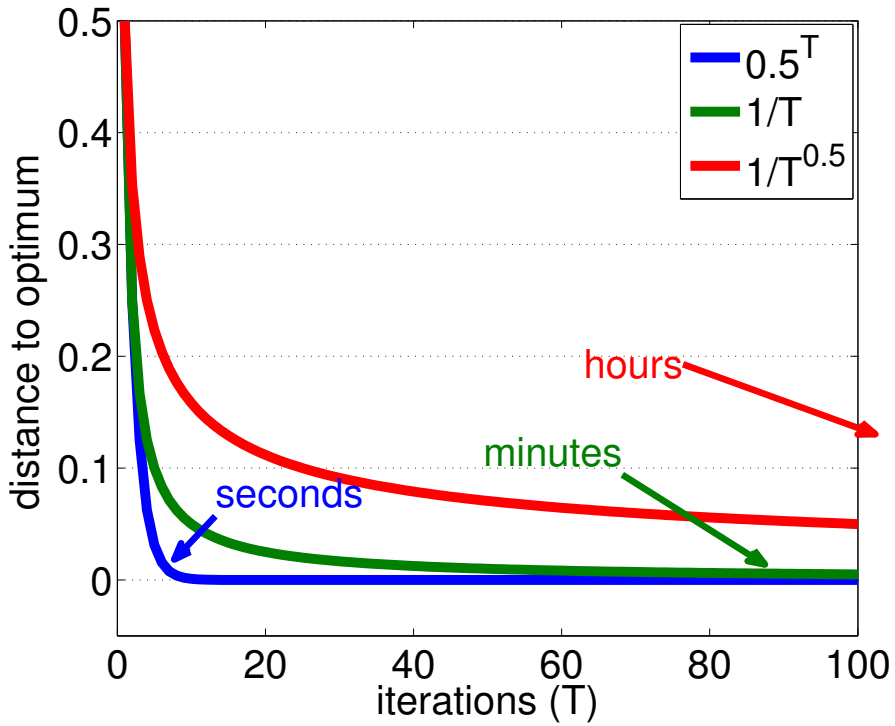
More on Convergence Measure

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



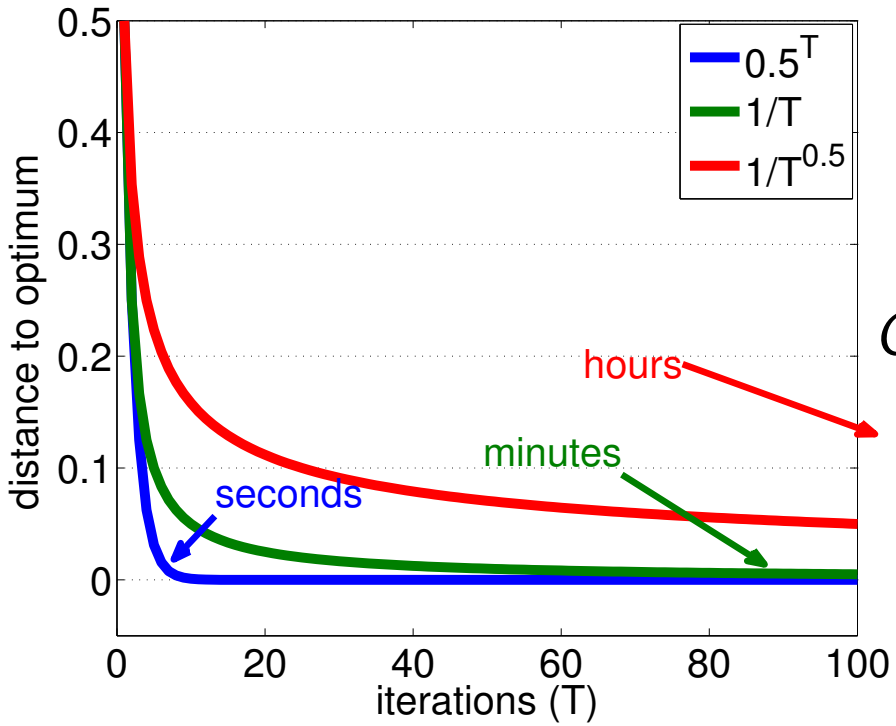
More on Convergence Measure

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



More on Convergence Measure

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

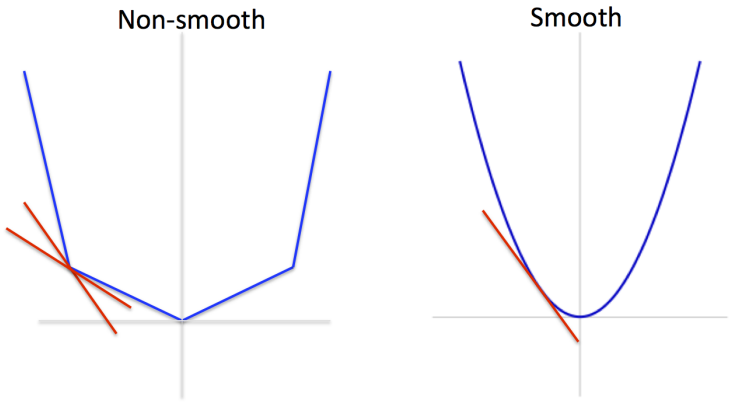


Theoretically, we consider

$$O(\mu^T) \prec O\left(\frac{1}{T^2}\right) \prec O\left(\frac{1}{T}\right) \prec O\left(\frac{1}{\sqrt{T}}\right)$$

Factors that affect Iteration Complexity

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

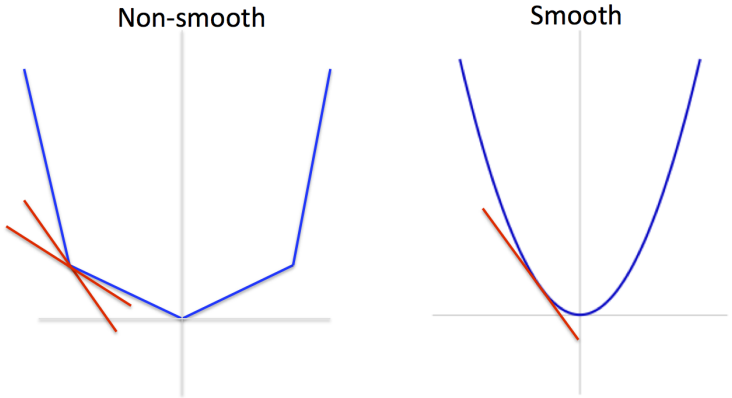


- Domain $\mathcal{X}$: size and geometry

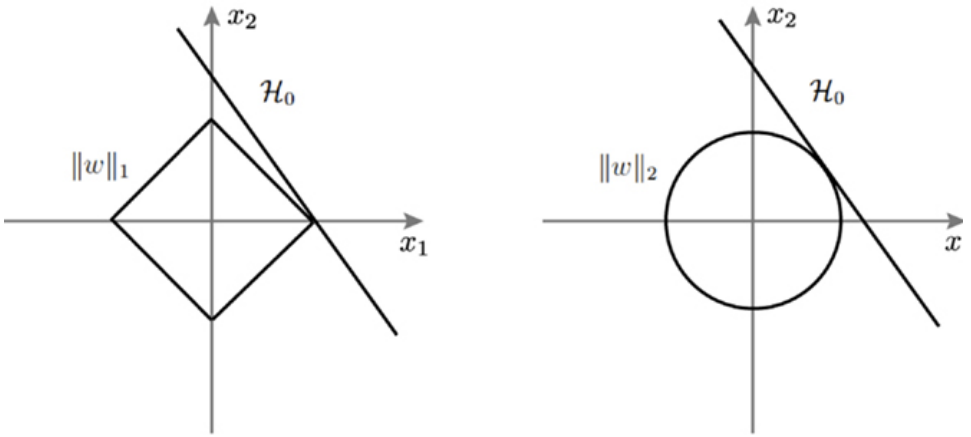
- Size of problem: dimension and number of data points

Factors that affect Iteration Complexity

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



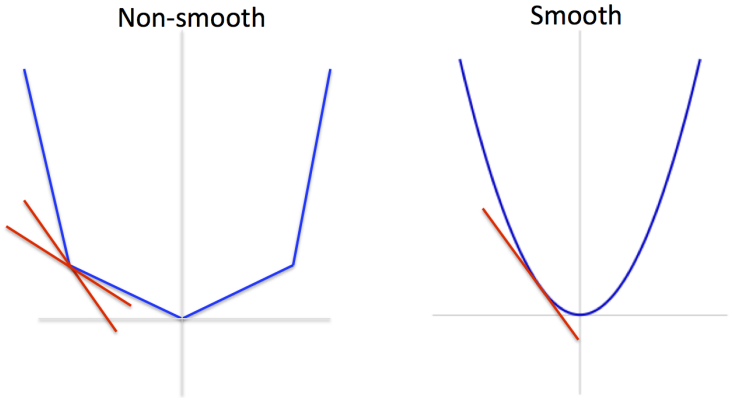
- Domain $\mathcal{X}$: size and geometry



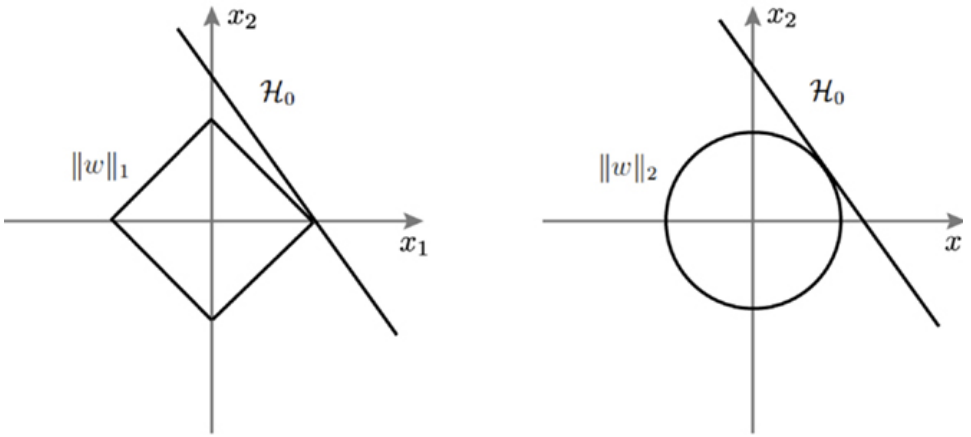
- Size of problem: dimension and number of data points

Factors that affect Iteration Complexity

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



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



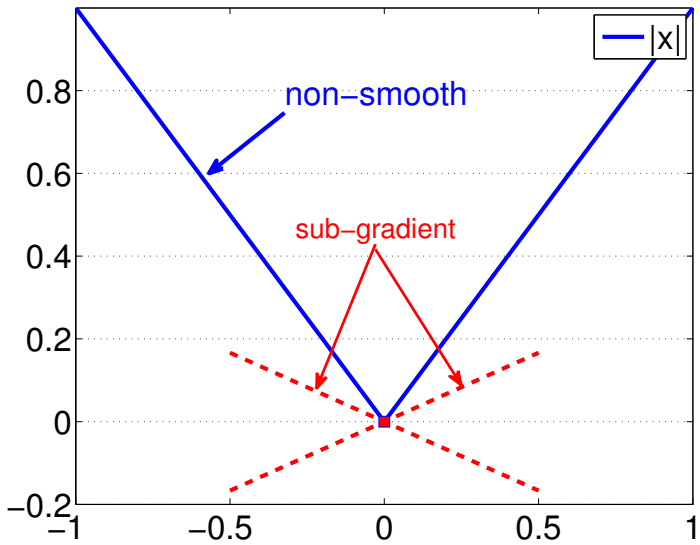
- Size of problem: dimension and number of data points

Warm-up: Non-smooth function

- Lipschitz continuous: e.g. absolute loss $f(x) = |x|$

$$|f(x) - f(y)| \leq G\|x - y\|_2$$

Subgradient: $f(x) \geq f(y) + \partial f(y)^\top (x - y)$



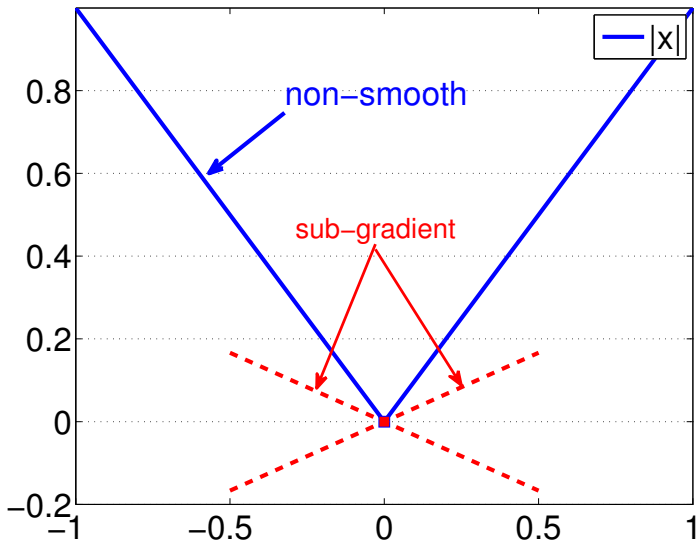
Warm-up: Non-smooth function

- Lipschitz continuous: e.g. absolute loss $f(x) = |x|$

Lipschitz constant

$$|f(x) - f(y)| \leq G\|x - y\|_2$$

Subgradient: $f(x) \geq f(y) + \partial f(y)^\top (x - y)$

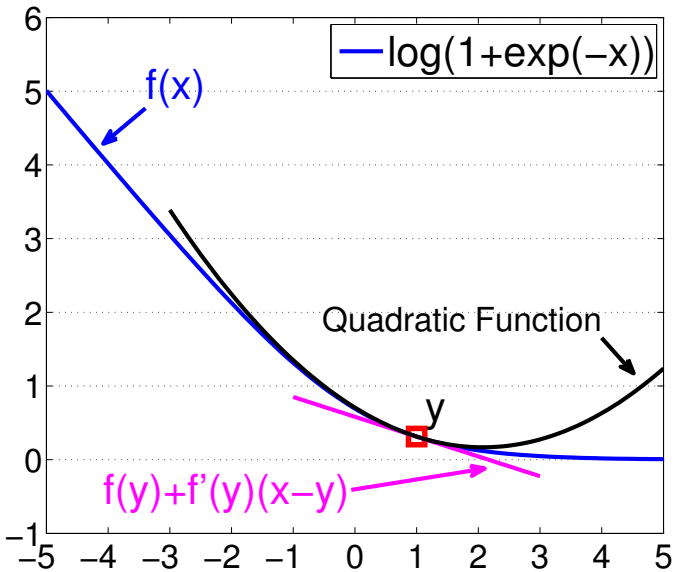


Warm-up: Smooth Convex function

- smooth: e.g. logistic loss $f(x) = \log(1 + \exp(-x))$

$\|\nabla f(x) - \nabla f(y)\|_2 \leq \beta \|x - y\|_2$
where $\beta > 0$

Second Order Derivative is upper bounded $\|\nabla^2 f(x)\|_2 \leq \beta$



Warm-up: Smooth Convex function

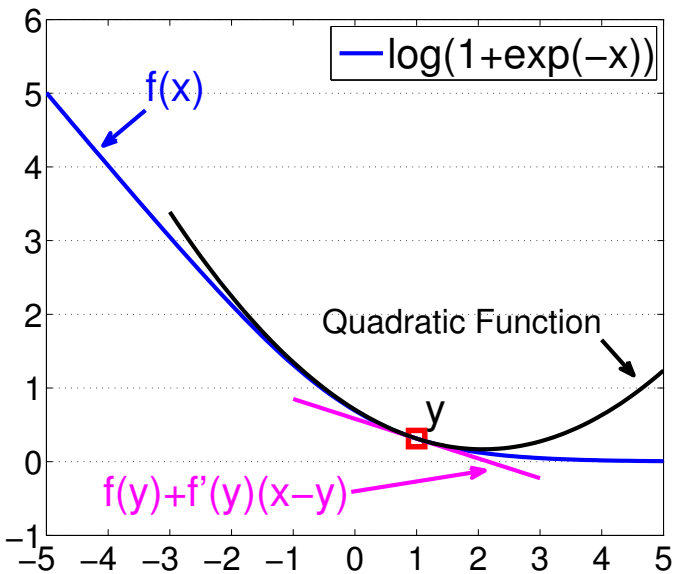
- smooth: e.g. logistic loss $f(x) = \log(1 + \exp(-x))$

smoothness constant

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq \beta \|x - y\|_2$$

where $\beta > 0$

Second Order Derivative is upper bounded $\|\nabla^2 f(x)\|_2 \leq \beta$

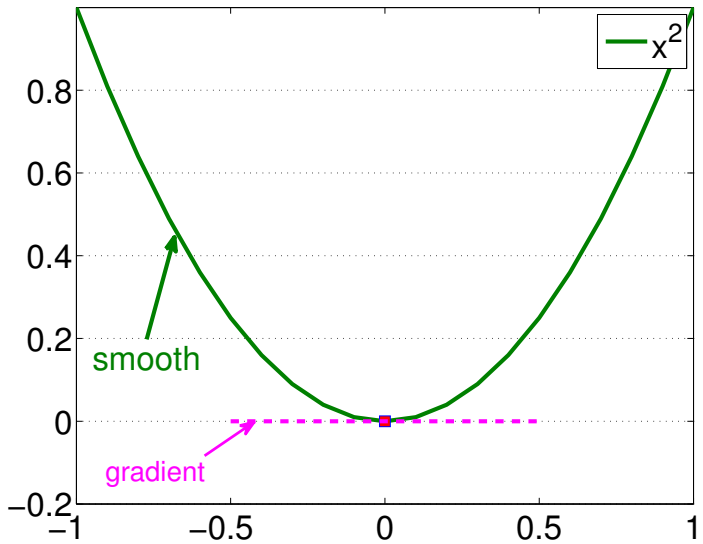


Warm-up: Strongly Convex function

- strongly convex: e.g. Euclidean norm $f(x) = \frac{1}{2}\|x\|_2^2$

$\|\nabla f(x) - \nabla f(y)\|_2 \geq \lambda \|x - y\|_2$
where $\lambda > 0$

Second Order Derivative is lower bounded $\|\nabla^2 f(x)\|_2 \geq \lambda$



Warm-up: Strongly Convex function

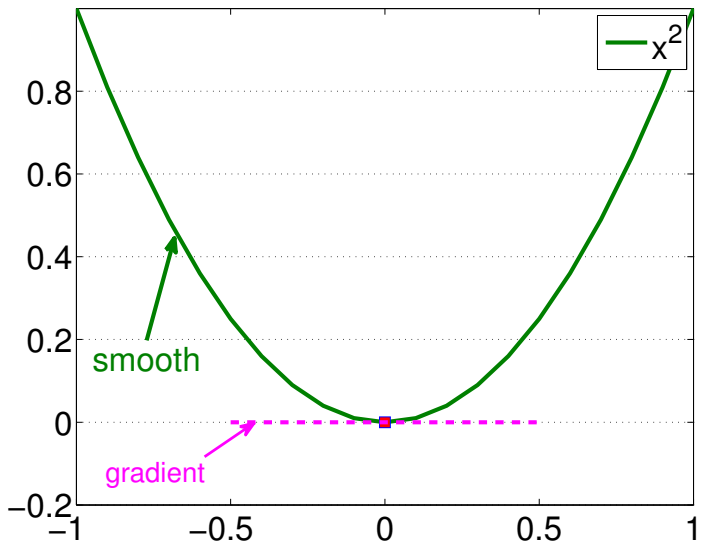
- strongly convex: e.g. Euclidean norm $f(x) = \frac{1}{2}\|x\|_2^2$

strong convexity constant

$$\|\nabla f(x) - \nabla f(y)\|_2 \geq \lambda \|x - y\|_2$$

where $\lambda > 0$

Second Order Derivative is lower bounded $\|\nabla^2 f(x)\|_2 \geq \lambda$



Warm-up: Smooth and Strongly Convex function

- smooth and strongly convex: e.g. quadratic function: $f(z) = \frac{1}{2}(z-1)^2$

$$\lambda\|x-y\|_2 \leq \|\nabla f(x) - \nabla f(y)\|_2 \leq \beta\|x-y\|_2, \quad \beta \geq \lambda > 0$$

Outline

STOP Algorithms for Big Data Classification and Regression

- Stochastic Gradient Descent (Pegasos) for SVM
- Stochastic Average Gradient (SAG) for Logistic Regression and Regression
- Stochastic Dual Coordinate Ascent (SDCA)
- Stochastic Optimization for Lasso

Classification and Regression

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

- Classification (red indicates smooth function)
  1. SVM (hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})$
  2. Smooth SVM (squared hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})^2$
  3. Equivalent to C-SVM formulations $C = n\lambda$
  4. Logistic Regression (logistic loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \log(1 + \exp(-y\mathbf{w}^\top \mathbf{x}))$

- Regression
  1. Least Square Regression (square loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = (\mathbf{w}^\top \mathbf{x} - y)^2$
  2. Least Absolute Deviation (absolute loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = |\mathbf{w}^\top \mathbf{x} - y|$

Classification and Regression

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

- Classification (red indicates smooth function)
  1. SVM (hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})$
  2. Smooth SVM (squared hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})^2$
  3. Equivalent to C-SVM formulations $C = n\lambda$
  4. Logistic Regression (logistic loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \log(1 + \exp(-y\mathbf{w}^\top \mathbf{x}))$

- Regression
  1. Least Square Regression (square loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = (\mathbf{w}^\top \mathbf{x} - y)^2$
  2. Least Absolute Deviation (absolute loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = |\mathbf{w}^\top \mathbf{x} - y|$

Classification and Regression

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

- Classification (red indicates smooth function)
  1. SVM (hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})$
  2. Smooth SVM (squared hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})^2$
  3. Equivalent to C-SVM formulations $C = n\lambda$
  4. Logistic Regression (logistic loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \log(1 + \exp(-y\mathbf{w}^\top \mathbf{x}))$

- Regression
  1. Least Square Regression (square loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = (\mathbf{w}^\top \mathbf{x} - y)^2$
  2. Least Absolute Deviation (absolute loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = |\mathbf{w}^\top \mathbf{x} - y|$

Classification and Regression

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

- Classification (red indicates smooth function)
  1. SVM (hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})$
  2. Smooth SVM (squared hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})^2$
  3. Equivalent to C-SVM formulations $C = n\lambda$
  4. Logistic Regression (logistic loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \log(1 + \exp(-y\mathbf{w}^\top \mathbf{x}))$

- Regression
  1. Least Square Regression (square loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = (\mathbf{w}^\top \mathbf{x} - y)^2$
  2. Least Absolute Deviation (absolute loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = |\mathbf{w}^\top \mathbf{x} - y|$

Classification and Regression

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

- Classification (red indicates smooth function)
  1. SVM (hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})$
  2. Smooth SVM (squared hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})^2$
  3. Equivalent to C-SVM formulations $C = n\lambda$
  4. Logistic Regression (logistic loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \log(1 + \exp(-y\mathbf{w}^\top \mathbf{x}))$

- Regression
  1. Least Square Regression (square loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = (\mathbf{w}^\top \mathbf{x} - y)^2$
  2. Least Absolute Deviation (absolute loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = |\mathbf{w}^\top \mathbf{x} - y|$

Classification and Regression

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

- Classification (red indicates smooth function)
  1. SVM (hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})$
  2. Smooth SVM (squared hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})^2$
  3. Equivalent to C-SVM formulations $C = n\lambda$
  4. Logistic Regression (logistic loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \log(1 + \exp(-y\mathbf{w}^\top \mathbf{x}))$

- Regression
  1. Least Square Regression (square loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = (\mathbf{w}^\top \mathbf{x} - y)^2$
  2. Least Absolute Deviation (absolute loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = |\mathbf{w}^\top \mathbf{x} - y|$

Classification and Regression

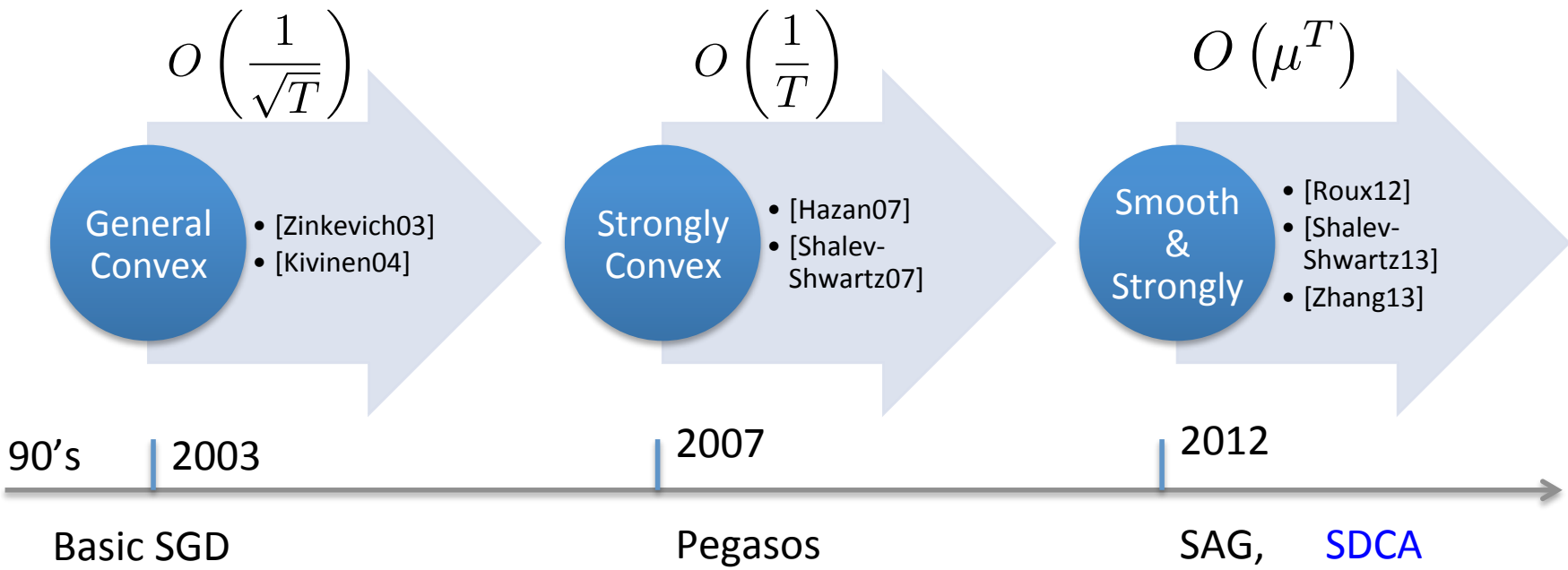
$$\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{w}^\top \mathbf{x}_i, y_i) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

- Classification (red indicates smooth function)
  1. SVM (hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})$
  2. Smooth SVM (squared hinge loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})^2$
  3. Equivalent to C-SVM formulations $C = n\lambda$
  4. Logistic Regression (logistic loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = \log(1 + \exp(-y\mathbf{w}^\top \mathbf{x}))$

- Regression
  1. Least Square Regression (square loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = (\mathbf{w}^\top \mathbf{x} - y)^2$
  2. Least Absolute Deviation (absolute loss): $\ell(\mathbf{w}^\top \mathbf{x}, y) = |\mathbf{w}^\top \mathbf{x} - y|$

Timeline of Stochastic Optimization in Machine Learning

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



$$\mathbf{w}_t = (1-\gamma_t\lambda)\mathbf{w}_{t-1} - \gamma_t \nabla \ell(\mathbf{w}_{t-1}^T \mathbf{x}_{i_t}, y_{i_t})$$

Basic SGD

- Leveraging only convexity

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

update: $\mathbf{w}_t = (1-\gamma_t\lambda)\mathbf{w}_{t-1} - \gamma_t \nabla \ell(\mathbf{w}_{t-1}^T \mathbf{x}_{i_t}, y_{i_t}), \quad \gamma_t = \frac{c}{\sqrt{t}}$

output solution: $\overline{\mathbf{w}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}_t \Rightarrow O\left(\frac{1}{\sqrt{T}}\right)$

Basic SGD

- Leveraging only convexity

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

update: $\mathbf{w}_t = (1-\gamma_t\lambda)\mathbf{w}_{t-1} - \gamma_t \nabla \ell(\mathbf{w}_{t-1}^T \mathbf{x}_{i_t}, y_{i_t}), \quad \gamma_t = \frac{c}{\sqrt{t}}$

output solution: $\overline{\mathbf{w}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}_t \Rightarrow O\left(\frac{1}{\sqrt{T}}\right)$

Pegasos (Shalev-Shwartz et al. (2007))

- Leveraging strongly convex regularizer

$$\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \max(0, 1 - y_i \mathbf{w}^\top \mathbf{x}_i) + \underbrace{\frac{\lambda}{2}\|\mathbf{w}\|_2^2}_{\text{strongly convex}}$$

update: $\mathbf{w}_t = (1 - \gamma_t \lambda)\mathbf{w}_{t-1} - \gamma_t \nabla \ell(\mathbf{w}_{t-1}^T \mathbf{x}_{i_t}, y_{i_t}), \quad \gamma_t = \frac{1}{\lambda t}$

output solution: $\overline{\mathbf{w}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}_t \Rightarrow O\left(\frac{1}{\lambda T}\right)$

Pegasos (Shalev-Shwartz et al. (2007))

- Leveraging strongly convex regularizer

$$\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \max(0, 1 - y_i \mathbf{w}^\top \mathbf{x}_i) + \underbrace{\frac{\lambda}{2}\|\mathbf{w}\|_2^2}_{\text{strongly convex}}$$

update: $\mathbf{w}_t = (1 - \gamma_t \lambda)\mathbf{w}_{t-1} - \gamma_t \nabla \ell(\mathbf{w}_{t-1}^T \mathbf{x}_{i_t}, y_{i_t}), \quad \gamma_t = \frac{1}{\lambda t}$

output solution: $\overline{\mathbf{w}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}_t \Rightarrow O\left(\frac{1}{\lambda T}\right)$

Pegasos (Shalev-Shwartz et al. (2007))

- e.g. hinge loss (SVM), absolute loss (Least Absolute Deviation)

$$\text{stochastic gradient: } \partial \ell(\mathbf{w}^\top \mathbf{x}_{i_t}, y_{i_t}) = \begin{cases} -y_{i_t}\mathbf{x}_{i_t}, & 1 - y_{i_t}\mathbf{w}^\top \mathbf{x}_{i_t} > 0 \\ 0, & \text{otherwise} \end{cases}$$

- computation cost per-iteration: $O(d)$

SAG (Roux et al. (2012))

- Leveraging smoothness of loss

$$\min_{\mathbf{w}\in\mathbb{R}^d} f(\mathbf{w}) = \underbrace{\frac{1}{n}\sum_{i=1}^{n}\ell(\mathbf{w}^\top\mathbf{x}_i, y_i) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2}_{\text{smooth and strongly convex}}$$

- Estimated Average Gradient

$$G_t = \frac{1}{n}\sum_{i=1}^{n} g_i^t, \quad g_i^t = \begin{cases} \partial\ell(\mathbf{w}_t^\top\mathbf{x}_{i_t}, y_{i_t}), & \text{if } i_t \text{ is selected} \\ g_i^{t-1}, & \text{otherwise} \end{cases}$$

update: $\mathbf{w}_t = (1-\gamma_t\lambda)\mathbf{w}_{t-1} - \gamma_t G_t, \quad \gamma_t = \frac{c}{\beta}$

output solution: $\mathbf{w}_T \Rightarrow O\left(\mu^T\right)$

SAG (Roux et al. (2012))

- Leveraging smoothness of loss

$$\min_{\mathbf{w}\in\mathbb{R}^d} f(\mathbf{w}) = \underbrace{\frac{1}{n}\sum_{i=1}^{n}\ell(\mathbf{w}^\top \mathbf{x}_i, y_i) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2}_{\text{smooth and strongly convex}}$$

- Estimated Average Gradient

$$G_t = \frac{1}{n}\sum_{i=1}^{n} g_i^t, \quad g_i^t = \begin{cases} \partial\ell(\mathbf{w}_t^\top \mathbf{x}_{i_t}, y_{i_t}), & \text{if } i_t \text{ is selected} \\ g_i^{t-1}, & \text{otherwise} \end{cases}$$

update: $\mathbf{w}_t = (1-\gamma_t\lambda)\mathbf{w}_{t-1} - \gamma_t G_t, \quad \gamma_t = \frac{c}{\beta}$

output solution: $\mathbf{w}_T \Rightarrow O\left(\mu^T\right)$

SAG: efficient update of averaged gradient

- logistic regression, least square regression, smooth SVM
- individual gradient

$$g_i = \partial \ell(\mathbf{w}^\top \mathbf{x}_i, y_i) = \alpha_i \mathbf{x}_i$$

- update of averaged gradient

$$G_t = \frac{1}{n}\sum_{i=1}^n g_i^t = \frac{1}{n}\sum_{i=1}^n \alpha_i^t \mathbf{x}_i = G_{t-1} + \frac{1}{n}(\alpha_i^t - \alpha_i^{t-1})\mathbf{x}_{i_t}$$

- computation cost per-iteration: $O(d)$

SDCA (Shalev-Shwartz & Zhang (2013))

- Stochastic Dual Coordinate Ascent (liblinear (Hsieh et al., 2008))
- non-smooth loss $O(1/\epsilon)$ and smooth loss $O(\log(1/\epsilon))$
- Dual Problem:

$$\max_{\alpha \in \mathcal{Q}} D(\alpha) = \frac{1}{n} \sum_{i=1}^{n} -\phi^*(-\alpha_i) - \frac{\lambda}{2} \left\| \frac{1}{\lambda n} \sum_{i=1}^{n} \alpha_i \mathbf{x}_i \right\|_2^2$$

- primal solution: $\mathbf{w}_t = \frac{1}{\lambda n} \sum_{i=1}^{n} \alpha_i^t \mathbf{x}_i$

Dual Coordinate Updates

$$\Delta\alpha_i = \max_{\alpha_i^t + \Delta\alpha_i \in \mathcal{Q}} -\phi^*(-\alpha_i^t - \Delta\alpha_i) - \frac{\lambda n}{2} \left\| \mathbf{w}_t + \frac{1}{\lambda n} \Delta\alpha_i \mathbf{x}_i \right\|_2^2$$

SDCA (Shalev-Shwartz & Zhang (2013))

- Stochastic Dual Coordinate Ascent (liblinear (Hsieh et al., 2008))
- non-smooth loss $O(1/\epsilon)$ and smooth loss $O(\log(1/\epsilon))$
- Dual Problem:

$$\max_{\alpha \in \mathcal{Q}} D(\alpha) = \frac{1}{n} \sum_{i=1}^{n} -\phi^*(-\alpha_i) - \frac{\lambda}{2} \left\| \frac{1}{\lambda n} \sum_{i=1}^{n} \alpha_i \mathbf{x}_i \right\|_2^2$$

- primal solution: $\mathbf{w}_t = \frac{1}{\lambda n} \sum_{i=1}^{n} \alpha_i^t \mathbf{x}_i$

Dual Coordinate Updates

$$\Delta\alpha_i = \max_{\alpha_i^t + \Delta\alpha_i \in \mathcal{Q}} -\phi^*(-\alpha_i^t - \Delta\alpha_i) - \frac{\lambda n}{2} \left\| \mathbf{w}_t + \frac{1}{\lambda n} \Delta\alpha_i \mathbf{x}_i \right\|_2^2$$

SDCA (Shalev-Shwartz & Zhang (2013))

- Stochastic Dual Coordinate Ascent (liblinear (Hsieh et al., 2008))
- non-smooth loss $O(1/\epsilon)$ and smooth loss $O(\log(1/\epsilon))$
- Dual Problem:

$$\max_{\alpha \in \mathcal{Q}} D(\alpha) = \frac{1}{n} \sum_{i=1}^{n} -\phi^*(-\alpha_i) - \frac{\lambda}{2} \left\| \frac{1}{\lambda n} \sum_{i=1}^{n} \alpha_i \mathbf{x}_i \right\|_2^2$$

- primal solution: $\mathbf{w}_t = \frac{1}{\lambda n} \sum_{i=1}^{n} \alpha_i^t \mathbf{x}_i$

Dual Coordinate Updates

$$\Delta\alpha_i = \max_{\alpha_i^t + \Delta\alpha_i \in \mathcal{Q}} -\phi^*(-\alpha_i^t - \Delta\alpha_i) - \frac{\lambda n}{2} \left\| \mathbf{w}_t + \frac{1}{\lambda n} \Delta\alpha_i \mathbf{x}_i \right\|_2^2$$

SDCA updates

- close form solution: hinge loss, squared hinge loss, absolute loss and square loss (Shalev-Shwartz & Zhang (2013))
- e.g. square loss

$$\Delta\alpha_i^t = \frac{y_i - \mathbf{w}_t^\top \mathbf{x}_i - \alpha_i^{t-1}}{1 + \|\mathbf{x}_i\|_2^2/(\lambda n)}$$

- computation cost per-iteration: $O(d)$
- approximate solution: logistic loss (Shalev-Shwartz & Zhang (2013))

Summary

alg.	Pegasos	SAG	SDCA
strongly convex			
smooth			
loss			
memory cost			
computation cost*			
Iteration Complexity			
paramter			
averaging			

Table : sqh: squared hinge loss; abs.: absolute loss; * per-iteration cost

Summary

alg.	Pegasos	SAG	SDCA
strongly convex	yes	yes	yes
smooth			
loss			
memory cost			
computation cost*			
Iteration Complexity			
paramter			
averaging			

Table : sqh: squared hinge loss; abs.: absolute loss; * per-iteration cost

Summary

alg.	Pegasos	SAG	SDCA
strongly convex	yes	yes	yes
smooth	no	yes	yes/no
loss			
memory cost			
computation cost*			
Iteration Complexity			
paramter			
averaging			

Table : sqh: squared hinge loss; abs.: absolute loss; * per-iteration cost

Summary

alg.	Pegasos	SAG	SDCA
strongly convex	yes	yes	yes
smooth	no	yes	yes/no
loss	hinge, abs.	logistic, square, sqh	all left
memory cost			
computation cost*			
Iteration Complexity			
paramter			
averaging			

Table : sqh: squared hinge loss; abs.: absolute loss; * per-iteration cost

Summary

alg.	Pegasos	SAG	SDCA
strongly convex	yes	yes	yes
smooth	no	yes	yes/no
loss	hinge, abs.	logistic, square, sqh	all left
memory cost	$O(d)$	$O(d+n)$	$O(d+n)$
computation cost*			
Iteration Complexity			
paramter			
averaging			

Table : sqh: squared hinge loss; abs.: absolute loss; * per-iteration cost

Summary

alg.	Pegasos	SAG	SDCA
strongly convex	yes	yes	yes
smooth	no	yes	yes/no
loss	hinge, abs.	logistic, square, sqh	all left
memory cost	$O(d)$	$O(d+n)$	$O(d+n)$
computation cost*	$O(d)$	$O(d)$	$O(d)$
Iteration Complexity			
paramter			
averaging			

Table : sqh: squared hinge loss; abs.: absolute loss; * per-iteration cost

Summary

alg.	Pegasos	SAG	SDCA
strongly convex	yes	yes	yes
smooth	no	yes	yes/no
loss	hinge, abs.	logistic, square, sqh	all left
memory cost	$O(d)$	$O(d+n)$	$O(d+n)$
computation cost*	$O(d)$	$O(d)$	$O(d)$
Iteration Complexity	$O\left(\frac{1}{\lambda\epsilon}\right)$	$O\left(\log\left(\frac{1}{\epsilon}\right)\right)$	$O\left(\frac{1}{\lambda\epsilon}\right)$ $O\left(\log\left(\frac{1}{\epsilon}\right)\right)$
paramter			
averaging			

Table : sqh: squared hinge loss; abs.: absolute loss; * per-iteration cost

Summary

alg.	Pegasos	SAG	SDCA
strongly convex	yes	yes	yes
smooth	no	yes	yes/no
loss	hinge, abs.	logistic, square, sqh	all left
memory cost	$O(d)$	$O(d+n)$	$O(d+n)$
computation cost*	$O(d)$	$O(d)$	$O(d)$
Iteration Complexity	$O\left(\frac{1}{\lambda\epsilon}\right)$	$O\left(\log\left(\frac{1}{\epsilon}\right)\right)$	$O\left(\frac{1}{\lambda\epsilon}\right)$ $O\left(\log\left(\frac{1}{\epsilon}\right)\right)$
paramter	no	step size	no
averaging			

Table : sqh: squared hinge loss; abs.: absolute loss; * per-iteration cost

Summary

alg.	Pegasos	SAG	SDCA
strongly convex	yes	yes	yes
smooth	no	yes	yes/no
loss	hinge, abs.	logistic, square, sqh	all left
memory cost	$O(d)$	$O(d+n)$	$O(d+n)$
computation cost*	$O(d)$	$O(d)$	$O(d)$
Iteration Complexity	$O\left(\frac{1}{\lambda\epsilon}\right)$	$O\left(\log\left(\frac{1}{\epsilon}\right)\right)$	$O\left(\frac{1}{\lambda\epsilon}\right)$ $O\left(\log\left(\frac{1}{\epsilon}\right)\right)$
paramter	no	step size	no
averaging	yes	no need	no need

Table : sqh: squared hinge loss; abs.: absolute loss; * per-iteration cost

What about ℓ_1 regularization?

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{w}^\top \mathbf{x}_i, y_i) + \sigma \underbrace{\sum_{g=1}^{K} \|\mathbf{w}_g\|_1}_{\text{Lasso or Group Lasso}}$$

Issue: Regularizer is Not Strongly Convex

Adding ℓ_2 regularization (Shalev-Shwartz & Zhang, 2012)

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^n \ell(\mathbf{w}^\top \mathbf{x}_i, y_i) + \sigma \underbrace{\sum_{g=1}^K \|\mathbf{w}_g\|_1}_{\text{Lasso or Group Lasso}}$$

Issue: Not Strongly Convex Solution: Add ℓ_2^2 regularization

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^n \ell(\mathbf{w}^\top \mathbf{x}_i, y_i) + \boxed{\sigma \sum_{g=1}^K \|\mathbf{w}_g\|_1 + \frac{\lambda}{2}\|\mathbf{w}\|_2^2}$$

setting $\lambda = \Theta(1/\epsilon)$, SDCA (non-smooth or smooth)

$O\left(\frac{1}{\epsilon^2}\right)$ for general convex loss, $O\left(\frac{1}{\epsilon}\right)$ for smooth loss

Adding ℓ_2 regularization (Shalev-Shwartz & Zhang, 2012)

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^n \ell(\mathbf{w}^\top \mathbf{x}_i, y_i) + \sigma \underbrace{\sum_{g=1}^K \|\mathbf{w}_g\|_1}_{\text{Lasso or Group Lasso}}$$

Issue: Not Strongly Convex Solution: Add ℓ_2^2 regularization

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^n \ell(\mathbf{w}^\top \mathbf{x}_i, y_i) + \boxed{\sigma \sum_{g=1}^K \|\mathbf{w}_g\|_1 + \frac{\lambda}{2}\|\mathbf{w}\|_2^2}$$

setting $\lambda = \Theta(1/\epsilon)$, SDCA (non-smooth or smooth)

$O\left(\frac{1}{\epsilon^2}\right)$ for general convex loss, $O\left(\frac{1}{\epsilon}\right)$ for smooth loss

Other algorithms for ℓ_1 regularization

- (Stochastic) Proximal Gradient Descent
  - Proximal Stochastic Gradient Descent (Langford et al., 2009; Shalev-Shwartz & Tewari, 2009; Duchi & Singer, 2009)
  - sparsity can be achieved at each iteration
  - $O(1/\epsilon^2)$ iteration complexity
- Stochastic Coordinate Descent (Shalev-Shwartz & Tewari, 2009; Bradley et al., 2011; Richtárik & Takác, 2013)
  - need to compute full gradient
  - $O(n/\epsilon)$ iteration complexity for smooth loss

Outline

Be Back in 5 minutes

General Strategies for Stochastic Optimization

General strategies for STOP

- SGD and its variants for different objectives
- Parallel and Distributed Optimization
- Other Effective Strategies

Stochastic Gradient Descent

$$\min_{x \in \mathcal{X}} f(x)$$

- stochastic gradient $\nabla f(x; \xi)$: ξ is a random variable
- basic SGD updates:

$$x_t \leftarrow \Pi_{\mathcal{X}} \left[x_{t-1} - \gamma_t \nabla f(x_{t-1}; \xi_t) \right]$$

$\Pi_{\mathcal{X}}[\hat{x}] = \min_{x \in \mathcal{X}} \|x - \hat{x}\|_2^2$

Issue: How to determine learning rate (step size) γ_t?

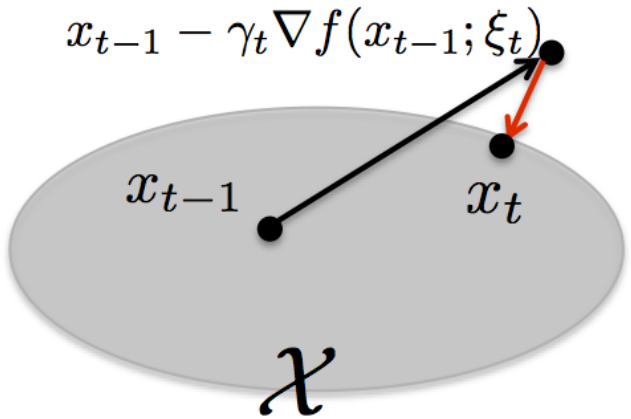
Stochastic Gradient Descent

$$\min_{x \in \mathcal{X}} f(x)$$

- stochastic gradient $\nabla f(x; \xi)$: ξ is a random variable
- basic SGD updates:

$$x_t \leftarrow \Pi_{\mathcal{X}} \left[x_{t-1} - \gamma_t \nabla f(x_{t-1}; \xi_t) \right]$$

$$\Pi_{\mathcal{X}}[\hat{x}] = \min_{x \in \mathcal{X}} \|x - \hat{x}\|_2^2$$



Issue: How to determine learning rate (step size) γ_t?

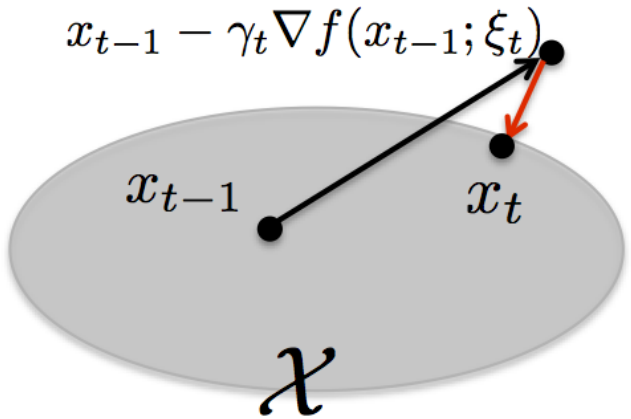
Stochastic Gradient Descent

$$\min_{x \in \mathcal{X}} f(x)$$

- stochastic gradient $\nabla f(x; \xi)$: ξ is a random variable
- basic SGD updates:

$$x_t \leftarrow \Pi_{\mathcal{X}} \left[x_{t-1} - \gamma_t \nabla f(x_{t-1}; \xi_t) \right]$$

$$\Pi_{\mathcal{X}}[\hat{x}] = \min_{x \in \mathcal{X}} \|x - \hat{x}\|_2^2$$



Issue: How to determine learning rate (step size) γ_t?

Convergence of final solution

Iterative updates

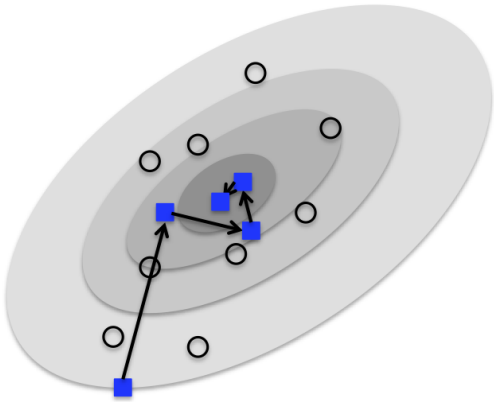
$$x_t = \Pi_{\mathcal{X}} \left[x_{t-1} - \gamma_t \Delta_t \right]$$

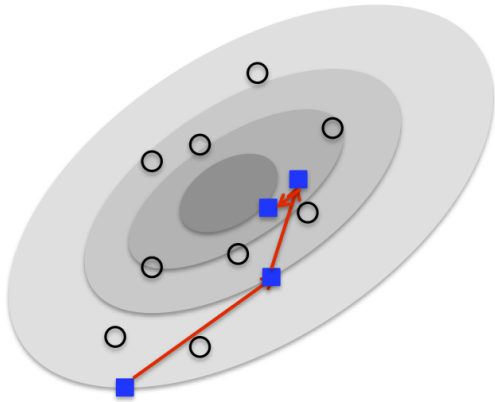
to have convergence, intuitively $\gamma_t \Delta_t \to 0$

Convergence of (S)GD

iterative updates

$$x_t = \Pi_{\mathcal{X}} \left[x_{t-1} - \gamma_t \Delta_t \right]$$





GD: $x_t = x_{t-1} - \gamma_t \nabla f(x_{t-1})$

$\nabla f(x_{t-1}) \to 0, x_t \to x_*$

SGD: $x_t = x_{t-1} - \gamma_t \nabla f(x_{t-1}; \xi_t)$

$\gamma_t \nabla f(x_{t-1}; \xi_t) \to 0$

Three Schemes of Step size

- General Convex Optimization $\gamma_t \propto 1/\sqrt{t} \to 0$
- Strongly Convex Optimization $\gamma_t \propto 1/t \to 0$
- Smooth Optimization $\gamma_t = c, \Delta_t \to 0$

SGD for General Convex Function

- Step size: $\gamma_t = \frac{c}{\sqrt{t}}$, c usually needed to be tuned
- Convergence rate of final solution $\mathbf{x}_T$ (Shamir & Zhang, 2013):

$$\mathbb{E}[f(x_T) - f(x_*)] \leq O\left(\frac{DG \log T}{\sqrt{T}}\right)$$

- $\|x - y\|_2 \leq D$ and $\|\partial f(x; \xi)\|_2 \leq G$, $\forall x, y \in \mathcal{X}$.
- Close to Optimal : $O\left(\frac{DG}{\sqrt{T}}\right)$

SGD for General Convex Function

- Step size: $\gamma_t = \frac{c}{\sqrt{t}}$, c usually needed to be tuned
- Convergence rate of final solution $\mathbf{x}_T$ (Shamir & Zhang, 2013):

$$\mathrm{E}[f(x_T) - f(x_*)] \leq O\left(\frac{DG \log T}{\sqrt{T}}\right)$$

- $\|x - y\|_2 \leq D$ and $\|\partial f(x; \xi)\|_2 \leq G$, $\forall x, y \in \mathcal{X}$.
- Close to Optimal : $O\left(\frac{DG}{\sqrt{T}}\right)$

SGD for General Convex Function

- Step size: $\gamma_t = \frac{c}{\sqrt{t}}$, c usually needed to be tuned
- Convergence rate of final solution $\mathbf{x}_T$ (Shamir & Zhang, 2013):

$$\mathrm{E}[f(x_T) - f(x_*)] \leq O\left(\frac{DG \log T}{\sqrt{T}}\right)$$

- $\|x - y\|_2 \leq D$ and $\|\partial f(x;\xi)\|_2 \leq G$, $\forall x, y \in \mathcal{X}$.
- Close to Optimal : $O\left(\frac{DG}{\sqrt{T}}\right)$

SGD for Strongly Convex Function

- $f(x)$ is λ-strongly convex
- Step size: $\gamma_t = \dfrac{1}{\lambda t}$
- Convergence Rate of x_T (Shamir & Zhang, 2013):

$$\mathrm{E}[f(x_T) - f(x_*)] \leq O\left(\frac{G^2 \log T}{\lambda T}\right)$$

- Close to Optimal : $O\left(\dfrac{G^2}{\lambda T}\right)$

SGD for Smooth Convex Function

- A sub-class of general convex function
- SGD with $\gamma_t \propto 1/\sqrt{t}$ has $O\left(\frac{\log T}{\sqrt{T}}\right)$
- Gradient Descent with $\gamma_t = c$ has $O\left(\frac{1}{T}\right)$ (Nesterov, 2004)
- Generally SGD can't bridge the gap (Lan, 2012)
- Special case, e.g.

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x), \quad \nabla f(x_t) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x_t), \quad \nabla f(x_t; \xi_t) = \nabla f_{i_t}(x_t)$$

- Constant step size of GD is due to $\nabla f(x_*) = 0$

SGD for Smooth Convex Function

- A sub-class of general convex function
- SGD with $\gamma_t \propto 1/\sqrt{t}$ has $O\left(\frac{\log T}{\sqrt{T}}\right)$
- Gradient Descent with $\gamma_t = c$ has $O\left(\frac{1}{T}\right)$ (Nesterov, 2004)
- Generally SGD can't bridge the gap (Lan, 2012)
- Special case, e.g.

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x), \quad \nabla f(x_t) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x_t), \quad \nabla f(x_t; \xi_t) = \nabla f_{i_t}(x_t)$$

- Constant step size of GD is due to $\nabla f(x_*) = 0$

SGD for Smooth Convex Function

- A sub-class of general convex function
- SGD with $\gamma_t \propto 1/\sqrt{t}$ has $O\left(\frac{\log T}{\sqrt{T}}\right)$
- Gradient Descent with $\gamma_t = c$ has $O\left(\frac{1}{T}\right)$ (Nesterov, 2004)
- Generally SGD can't bridge the gap (Lan, 2012)
- Special case, e.g.

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x), \quad \nabla f(x_t) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x_t), \quad \nabla f(x_t; \xi_t) = \nabla f_{i_t}(x_t)$$

- Constant step size of GD is due to $\nabla f(x_*) = 0$

SGD for Smooth Convex Function

- A sub-class of general convex function
- SGD with $\gamma_t \propto 1/\sqrt{t}$ has $O\left(\frac{\log T}{\sqrt{T}}\right)$
- Gradient Descent with $\gamma_t = c$ has $O\left(\frac{1}{T}\right)$ (Nesterov, 2004)
- Generally SGD can't bridge the gap (Lan, 2012)
- Special case, e.g.

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x), \quad \nabla f(x_t) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x_t), \quad \nabla f(x_t; \xi_t) = \nabla f_{i_t}(x_t)$$

- Constant step size of GD is due to $\nabla f(x_*) = 0$

Accelerated SGD for smooth function (Johnson & Zhang, 2013; Mahdavi et al., 2013)

Iterate $s = 1, \ldots,$

Iterate $t = 1, \ldots, m$

$$x_t^s = x_{t-1}^s - \gamma \underbrace{(\nabla f_{i_t}(x_{t-1}^s) - \nabla f_{i_t}(\bar{x}^{s-1}) + \nabla f(\bar{x}^{s-1}))}_{\Delta_t = StoGrad - StoGrad + Grad}$$

update $\bar{x}^s$

- $\bar{x}^s = x_m^s$ or $\bar{x}^s = \sum_{t=1}^m x_t^s/m$, $m = O(n)$
- constant step size, $\Delta_t \to 0$
  if $\bar{x}^{s-1} \to x^*$, $\nabla f(\bar{x}^{s-1}) \to 0$, $\nabla f_{i_t}(x_{t-1}^s) - \nabla f_{i_t}(x^{s-1}) \to 0$
- Smooth function: $O(1/\epsilon)$ for smooth
- Smooth & strongly convex function: $O\left(\log\left(\frac{1}{\epsilon}\right)\right)$

Accelerated SGD for smooth function (Johnson & Zhang, 2013; Mahdavi et al., 2013)

Iterate $s = 1, \ldots,$

Iterate $t = 1, \ldots, m$

$$x_t^s = x_{t-1}^s - \gamma \underbrace{(\nabla f_{i_t}(x_{t-1}^s) - \nabla f_{i_t}(\bar{x}^{s-1}) + \nabla f(\bar{x}^{s-1}))}_{\Delta_t = StoGrad - StoGrad + Grad}$$

update $\bar{x}^s$

- $\bar{x}^s = x_m^s$ or $\bar{x}^s = \sum_{t=1}^m x_t^s/m$, $m = O(n)$
- constant step size, $\Delta_t \to 0$
  if $\bar{x}^{s-1} \to x^*$, $\nabla f(\bar{x}^{s-1}) \to 0$, $\nabla f_{i_t}(x_{t-1}^s) - \nabla f_{i_t}(x^{s-1}) \to 0$
- Smooth function: $O(1/\epsilon)$ for smooth
- Smooth & strongly convex function: $O\left(\log\left(\frac{1}{\epsilon}\right)\right)$

Accelerated SGD for smooth function (Johnson & Zhang, 2013; Mahdavi et al., 2013)

Iterate $s = 1, \ldots,$

Iterate $t = 1, \ldots, m$

$$x_t^s = x_{t-1}^s - \gamma \underbrace{(\nabla f_{i_t}(x_{t-1}^s) - \nabla f_{i_t}(\bar{x}^{s-1}) + \nabla f(\bar{x}^{s-1}))}_{\Delta_t = StoGrad - StoGrad + Grad}$$

update $\bar{x}^s$

- $\bar{x}^s = x_m^s$ or $\bar{x}^s = \sum_{t=1}^m x_t^s/m$, $m = O(n)$
- constant step size, $\Delta_t \to 0$
  if $\bar{x}^{s-1} \to x^*$, $\nabla f(\bar{x}^{s-1}) \to 0$, $\nabla f_{i_t}(x_{t-1}^s) - \nabla f_{i_t}(x^{s-1}) \to 0$
- Smooth function: $O(1/\epsilon)$ for smooth
- Smooth & strongly convex function: $O\left(\log\left(\frac{1}{\epsilon}\right)\right)$

Accelerated SGD for smooth function (Johnson & Zhang, 2013; Mahdavi et al., 2013)

Iterate $s = 1, \ldots,$

Iterate $t = 1, \ldots, m$

$$x_t^s = x_{t-1}^s - \gamma \underbrace{(\nabla f_{i_t}(x_{t-1}^s) - \nabla f_{i_t}(\bar{x}^{s-1}) + \nabla f(\bar{x}^{s-1}))}_{\Delta_t = StoGrad - StoGrad + Grad}$$

update $\bar{x}^s$

- $\bar{x}^s = x_m^s$ or $\bar{x}^s = \sum_{t=1}^m x_t^s/m$, $m = O(n)$
- constant step size, $\Delta_t \to 0$
  if $\bar{x}^{s-1} \to x^*$, $\nabla f(\bar{x}^{s-1}) \to 0$, $\nabla f_{i_t}(x_{t-1}^s) - \nabla f_{i_t}(x^{s-1}) \to 0$
- Smooth function: $O(1/\epsilon)$ for smooth
- Smooth & strongly convex function: $O\left(\log\left(\frac{1}{\epsilon}\right)\right)$

Averaged Stochastic Gradient Descent

Averaging usually speed-up convergence:

$$\bar{x}_t = \left(1 - \frac{1+\eta}{t+\eta}\right)\bar{x}_{t-1} + \frac{1+\eta}{t+\eta}x_t, \quad \eta \geq 0$$

- $\eta = 0$ simple averaging $\bar{x}_T = (x_1 + \ldots + x_T)/T$

Averaged Stochastic Gradient Descent

Averaging usually speed-up convergence:

$$\bar{x}_t = \left(1 - \frac{1+\eta}{t+\eta}\right)\bar{x}_{t-1} + \frac{1+\eta}{t+\eta}x_t, \quad \eta \geq 0$$

- $\eta = 0$ simple averaging $\bar{x}_T = (x_1 + \ldots + x_T)/T$
- General Convex Optimization (Nemirovski et al., 2009): $\eta = 0 \Rightarrow O\left(\frac{1}{\sqrt{T}}\right)$ vs $O\left(\frac{\log T}{\sqrt{T}}\right)$

Averaged Stochastic Gradient Descent

Averaging usually speed-up convergence:

$$\bar{x}_t = \left(1 - \frac{1+\eta}{t+\eta}\right)\bar{x}_{t-1} + \frac{1+\eta}{t+\eta}x_t, \quad \eta \geq 0$$

- $\eta = 0$ simple averaging $\bar{x}_T = (x_1 + \ldots + x_T)/T$
- Strongly Convex Optimization (Shamir & Zhang, 2013; Zhu, 2013):
  $\eta > 0 \Rightarrow O\left(\frac{1}{\lambda T}\right)$ vs $O\left(\frac{\log T}{\lambda T}\right)$

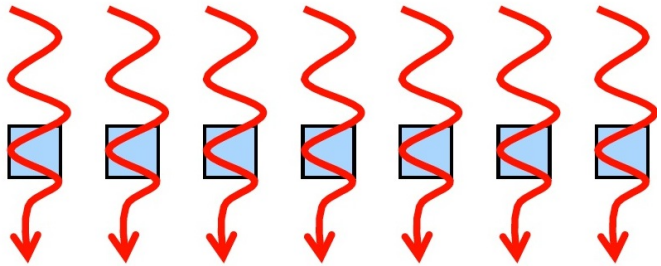
General Strategies for Stochastic Optimization

General strategies for STOP

- SGD and its variants for different objectives
- Parallel and Distributed Optimization
- Other Effective Strategies

Parallel and Distributed Optimization

Parallel (shared memory)

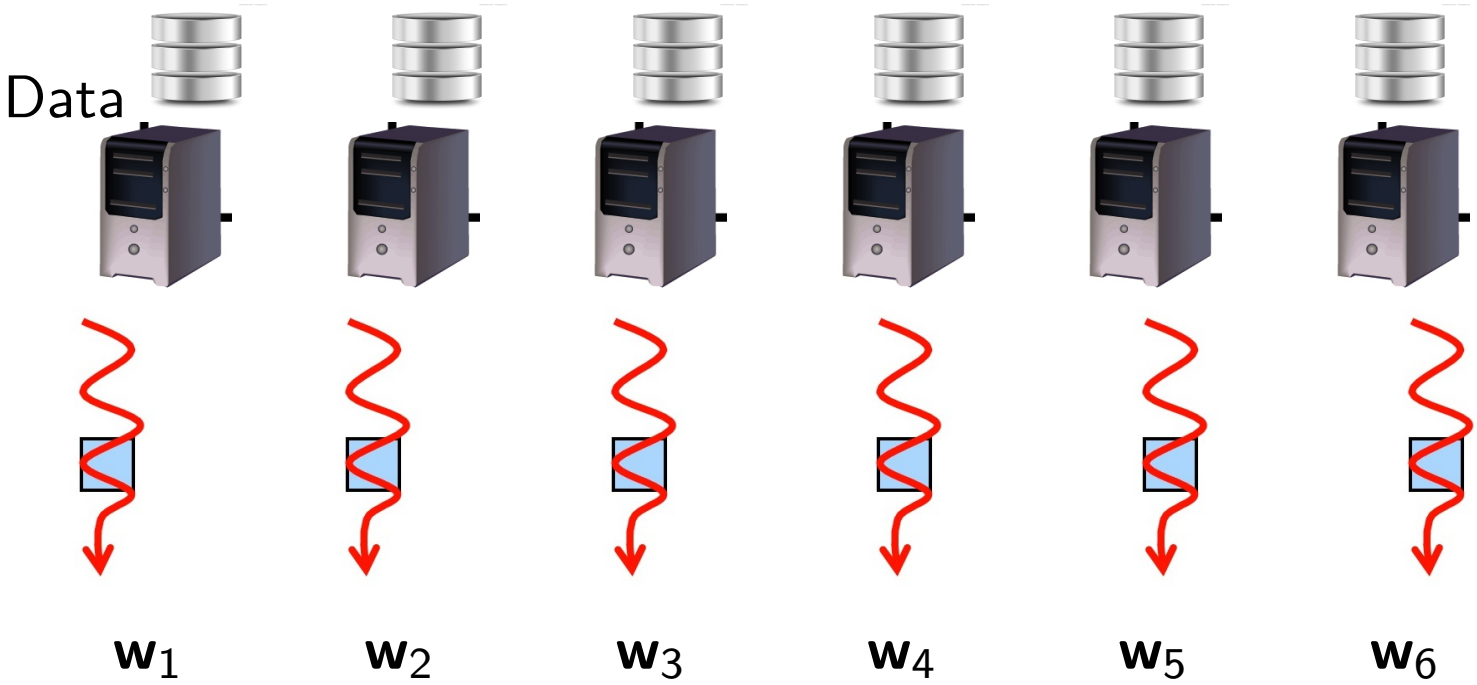


Distributed (not shared)

- To speed-up convergence
- data distributed over multiple machines
- moving to single machine suffers
  - low network bandwidth
  - limited disk or memory
- benefits from
  - cluster of machines
  - multi-core machine, GPU

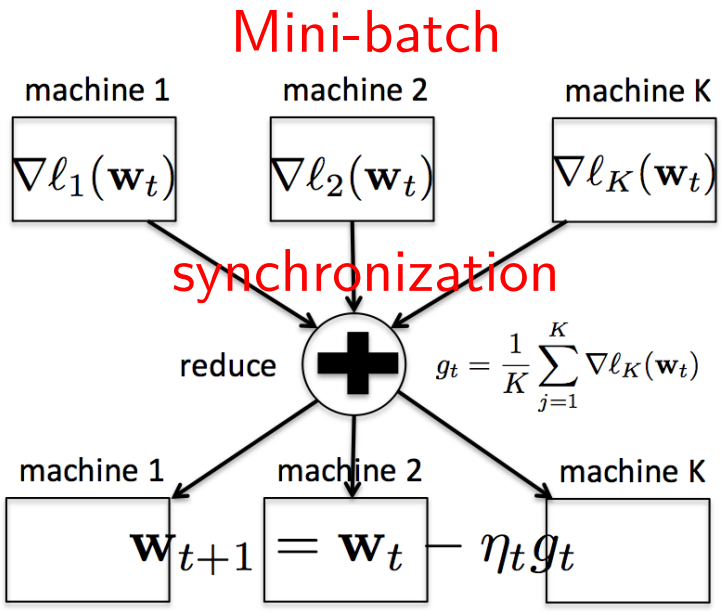
A simple solution: Average Runs

- multi-core machine
- cluster of machines



$$\widehat{\mathbf{w}} = \frac{1}{k}\sum_{i=1}^{k}\mathbf{w}_i, \quad \text{Issue: Not the Optimal}$$

Parallel SGD: Average Gradients



Mini-batch SGD

- multi-core or cluster
- Good: reduced variance, faster conv.
- Bad: synchronization is expensive
- Solutions:
  - asynchronized update: HogWild!
  - lesser synchronizations: DisDCA

Lock-free Parallel SGD: HOGWILD! (Niu et al., 2011)

$$\min_{\mathbf{x}} \sum_{e \in E} f_e(\mathbf{x}_e)$$

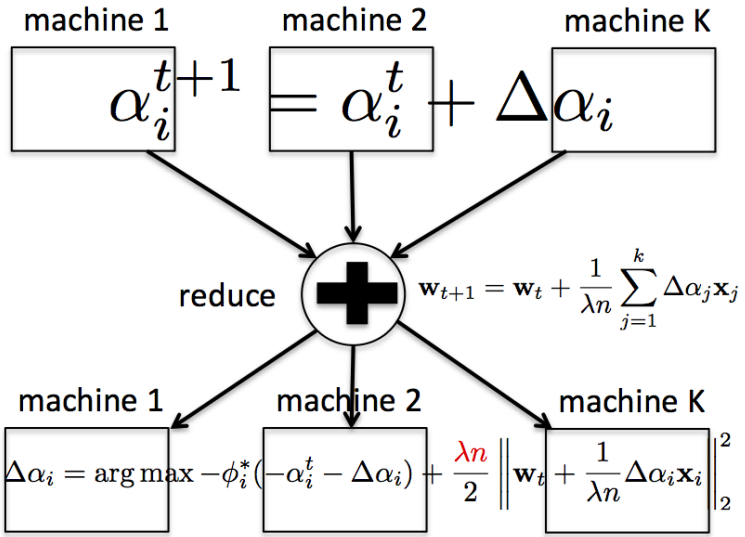
Each processor independently runs:

1. Sample e from E
2. Read current state of x_e
3. **for** v in e **do** $x_v \leftarrow x_v - \alpha[\nabla f_e(x_e)]_v$

Only assume atomicity of $x_v \leftarrow x_v - a$

- multi-core with shared-memory access
- each e is a small subset of $[d]$
  - sparse SVM, matrix completion, graph-cut
- robust $1/T$ convergence rate for strongly convex objective

Distributed SDCA (Yang, 2013)



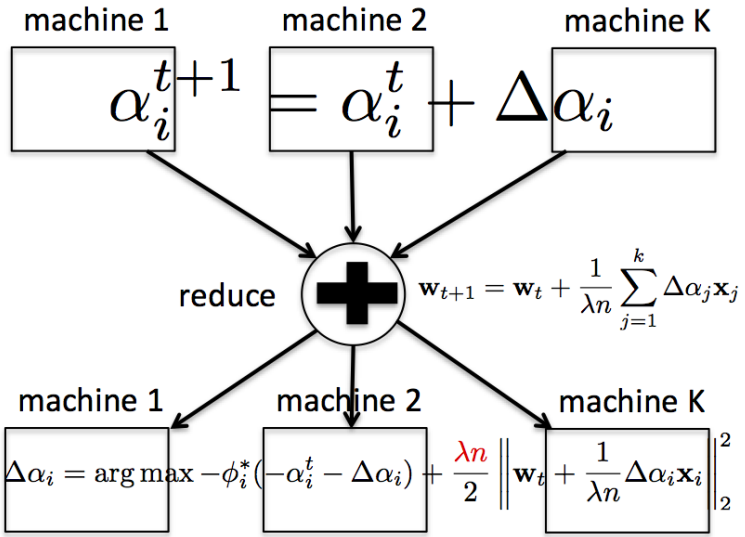
$$\Delta\alpha_i = \arg\max -\phi_i^*(-\alpha_i^t - \Delta\alpha_i) - \frac{\lambda n}{2}\left\|\mathbf{w}_t + \frac{1}{\lambda n}\Delta\alpha_i\mathbf{x}_i\right\|_2^2$$

convergence is not guaranteed: data are correlated

$$\Delta\alpha_i = \arg\max -\phi_i^*(-\alpha_i^t - \Delta\alpha_i) - \frac{\lambda n}{2K}\left\|\mathbf{w}_t + \frac{K}{\lambda n}\Delta\alpha_i\mathbf{x}_i\right\|_2^2$$

guaranteed convergence; limited speed-up

Distributed SDCA (Yang, 2013)



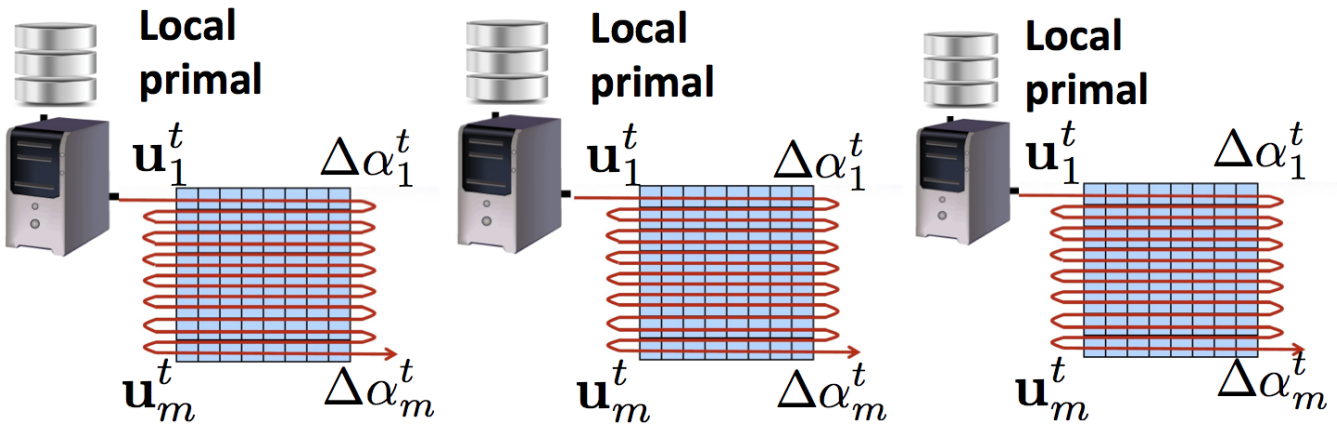
$$\Delta\alpha_i = \arg\max -\phi_i^*(-\alpha_i^t - \Delta\alpha_i) - \frac{\lambda n}{2}\left\|\mathbf{w}_t + \frac{1}{\lambda n}\Delta\alpha_i\mathbf{x}_i\right\|_2^2$$

convergence is not guaranteed: data are correlated

$$\Delta\alpha_i = \arg\max -\phi_i^*(-\alpha_i^t - \Delta\alpha_i) - \frac{\lambda n}{2K}\left\|\mathbf{w}_t + \frac{K}{\lambda n}\Delta\alpha_i\mathbf{x}_i\right\|_2^2$$

guaranteed convergence; limited speed-up

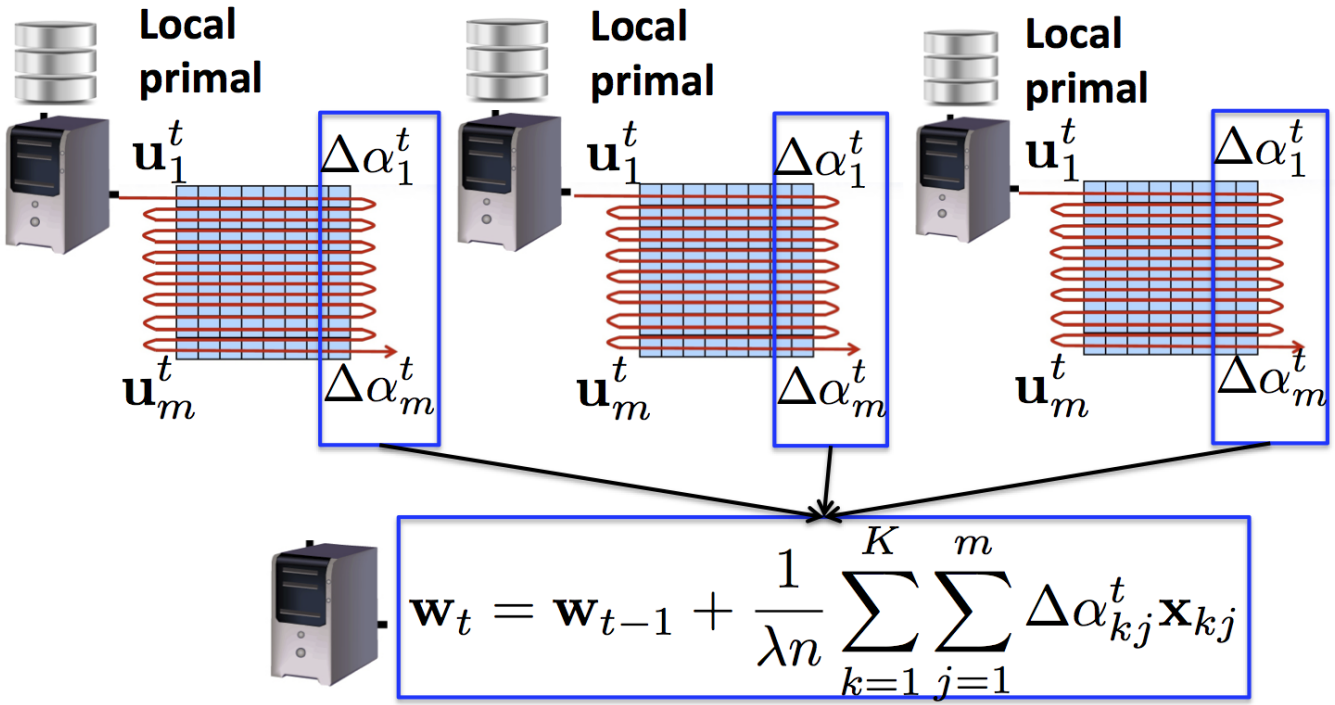
DisDCA: Trading Computation for Communication



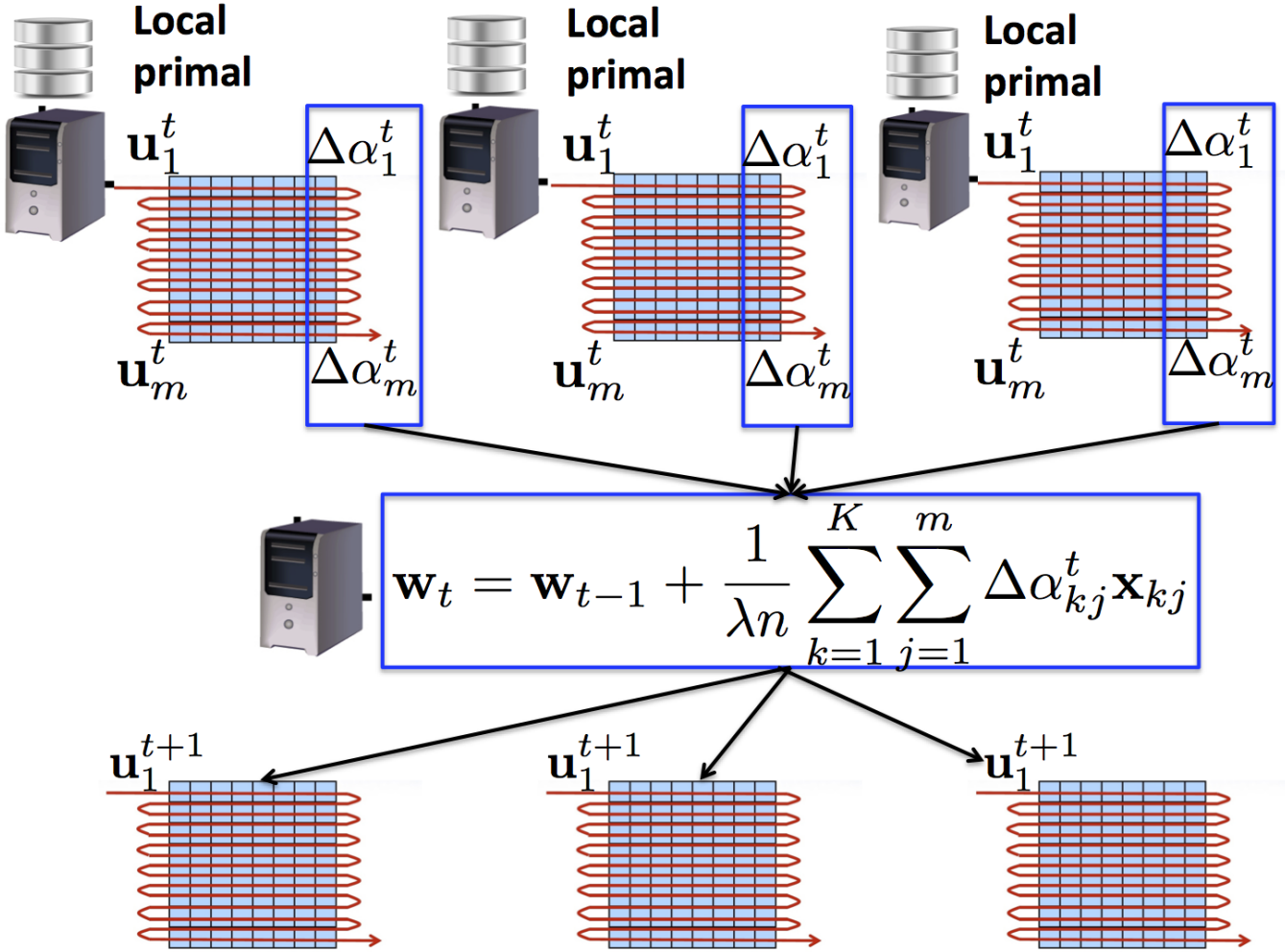
$$\Delta\alpha_{i_j} = \arg\max -\phi_{i_j}^*(-\alpha_{i_j}^t - \Delta\alpha_{i_j}) - \frac{\lambda n}{2K}\left\|\mathbf{u}_j^t + \frac{K}{\lambda n}\Delta\alpha_{i_j}\mathbf{x}_{i_j}\right\|_2^2$$

$$\mathbf{u}_{j+1}^t = \mathbf{u}_j^t + \frac{K}{\lambda n}\Delta\alpha_{i_j}\mathbf{x}_{i_j}$$

DisDCA: Trading Computation for Communication

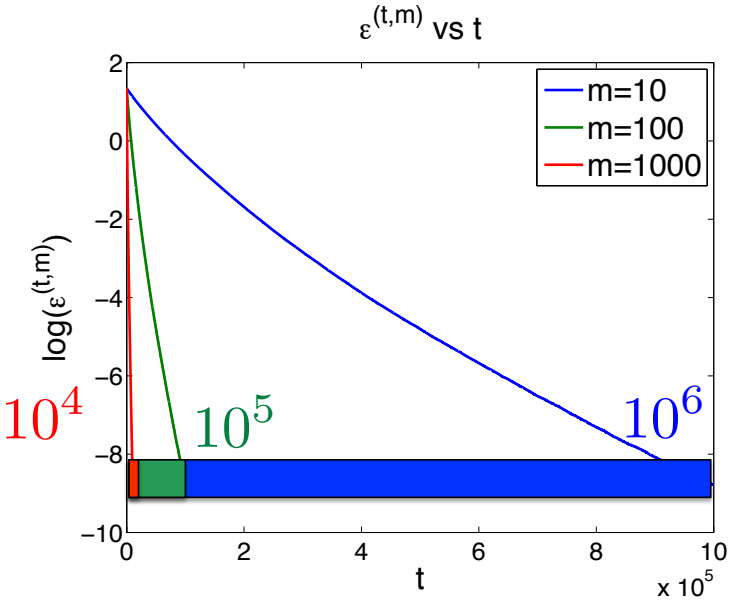


DisDCA: Trading Computation for Communication

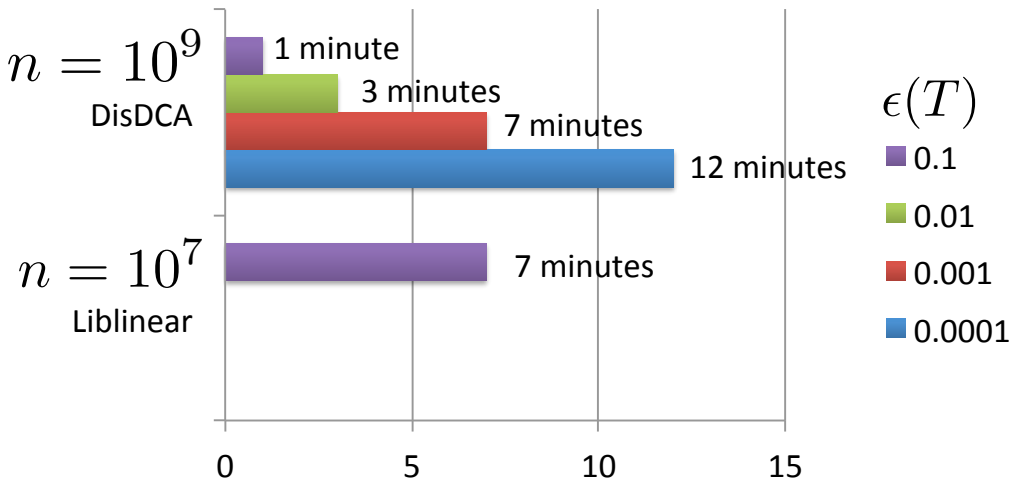


DisDCA

- increasing m could lead to nearly linear speed-up
- increasing K leads to parallel speed-up



(a) 1 million syn. data for regression



(b) 1 billion syn. data for classification; 400 GB, 50*2 processors

- The Distributed Library: Birds

General Strategies for Stochastic Optimization

General strategies for STOP

- SGD and its variants for different objectives
- Parallel and Distributed Optimization
- Other Effective Strategies

Factors that affect Iteration Complexity

- Property of function: smoothness of function
- Size of problem: dimension and number of data points
- Domain $\mathcal{X}$: size and geometry

Screening for Lasso and Support Vector Machine

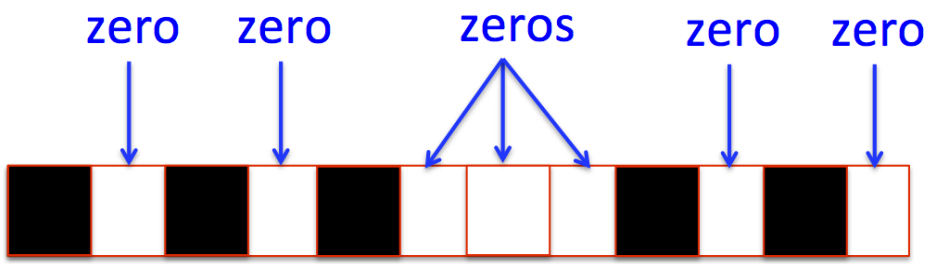
Factors that affect Iteration Complexity

- Property of function: smoothness of function
- Size of problem: dimension and number of data points
- Domain $\mathcal{X}$: size and geometry

Screening for Lasso and Support Vector Machine

Screening for Lasso

$$\text{Lasso:} \quad \min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{2}\|\mathbf{y} - X\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_1$$



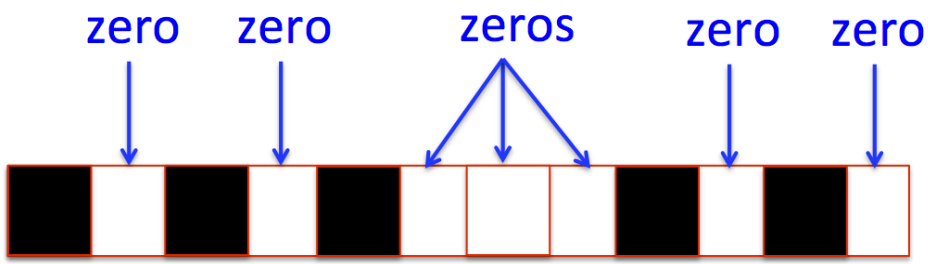
- $\mathbf{y} = (y_1, \ldots, y_n)^\top \in \mathbb{R}^n$
- $X = (\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_d) \in \mathbb{R}^{n\times d}$
- $\mathcal{I}_0 = \{i : w_i^* = 0\}$, $\mathcal{I} = [d]\backslash\mathcal{I}_0$

$$\text{Lasso:} \quad \min_{\mathbf{w}_\mathcal{I}} \frac{1}{2}\|\mathbf{y} - X_\mathcal{I}\mathbf{w}_\mathcal{I}\|_2^2 + \lambda\|\mathbf{w}_\mathcal{I}\|_1$$

SAFE rule (Ghaoui et al., 2010), DPP (Wang et al., 2012).

Screening for Lasso

$$\text{Lasso:} \quad \min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{2}\|\mathbf{y} - X\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_1$$



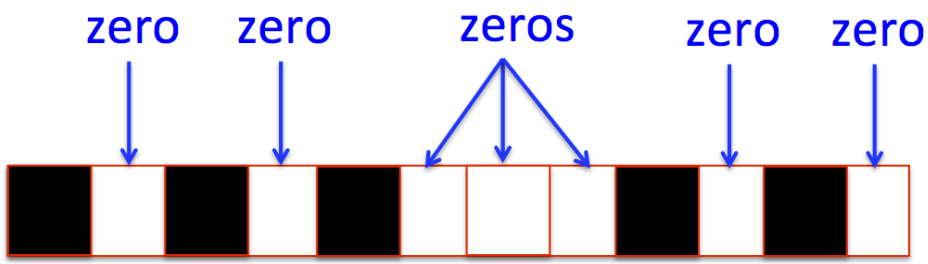
- $\mathbf{y} = (y_1, \ldots, y_n)^\top \in \mathbb{R}^n$
- $X = (\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_d) \in \mathbb{R}^{n\times d}$
- $\mathcal{I}_0 = \{i : w_i^* = 0\}$, $\mathcal{I} = [d]\backslash\mathcal{I}_0$

$$\text{Lasso:} \quad \min_{\mathbf{w}_\mathcal{I}} \frac{1}{2}\|\mathbf{y} - X_\mathcal{I}\mathbf{w}_\mathcal{I}\|_2^2 + \lambda\|\mathbf{w}_\mathcal{I}\|_1$$

SAFE rule (Ghaoui et al., 2010), DPP (Wang et al., 2012).

Screening for Lasso

$$\text{Lasso:} \quad \min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{2}\|\mathbf{y} - X\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_1$$



- $\mathbf{y} = (y_1, \ldots, y_n)^\top \in \mathbb{R}^n$
- $X = (\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_d) \in \mathbb{R}^{n\times d}$
- $\mathcal{I}_0 = \{i : w_i^* = 0\}$, $\mathcal{I} = [d]\backslash\mathcal{I}_0$

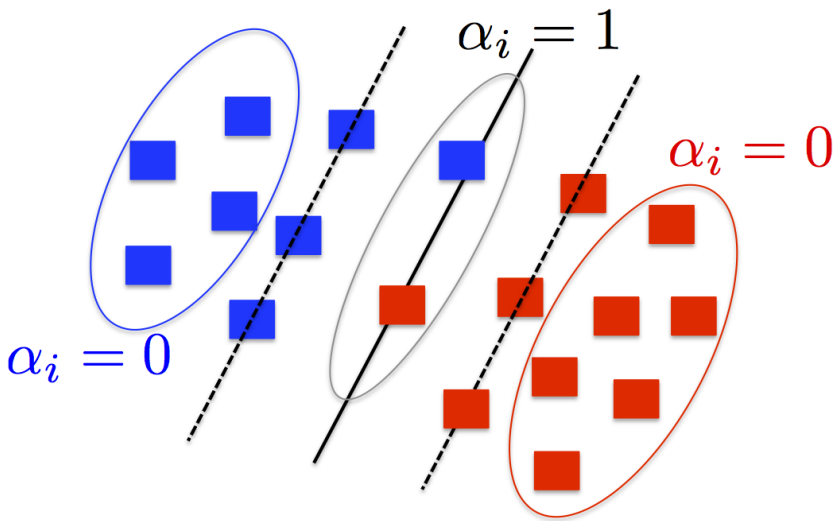
$$\text{Lasso:} \quad \min_{\mathbf{w}_\mathcal{I}} \frac{1}{2}\|\mathbf{y} - X_\mathcal{I}\mathbf{w}_\mathcal{I}\|_2^2 + \lambda\|\mathbf{w}_\mathcal{I}\|_1$$

SAFE rule (Ghaoui et al., 2010), DPP (Wang et al., 2012).

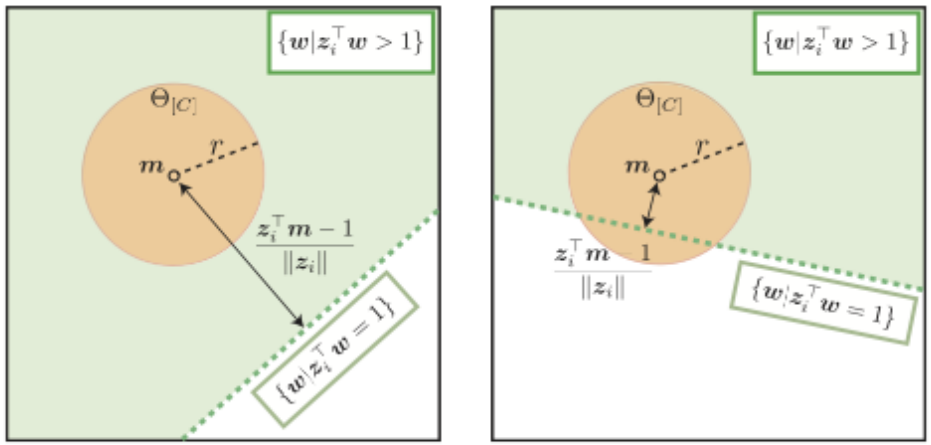
Screening for Support Vector Machine

Dual SVM: $$\max_{\alpha\in[0,1]^n} \frac{1}{n}\sum_{i=1}^n \alpha_i - \frac{\lambda}{2}\left\|\frac{1}{\lambda n}\sum_{i=1}^n \alpha_i y_i \mathbf{x}_i\right\|_2^2$$

$$y_i \mathbf{w}^* \mathbf{x}_i < 1 \Rightarrow \alpha_i^* = 1, \quad y_i \mathbf{w}^* \mathbf{x}_i > 1 \Rightarrow \alpha_i^* = 0$$



Ball Test (Ogawa et al., 2014; Wang et al., 2013)



(a) Success (b) Fail

Factors that affect Iteration Complexity

- Property of function: smoothness of function
- Size of problem: dimension and number of data points
- Domain $\mathcal{X}$: size and geometry

Stochastic Mirror Descent

Reducing G, D

Iteration Complexity of SGD depends on:

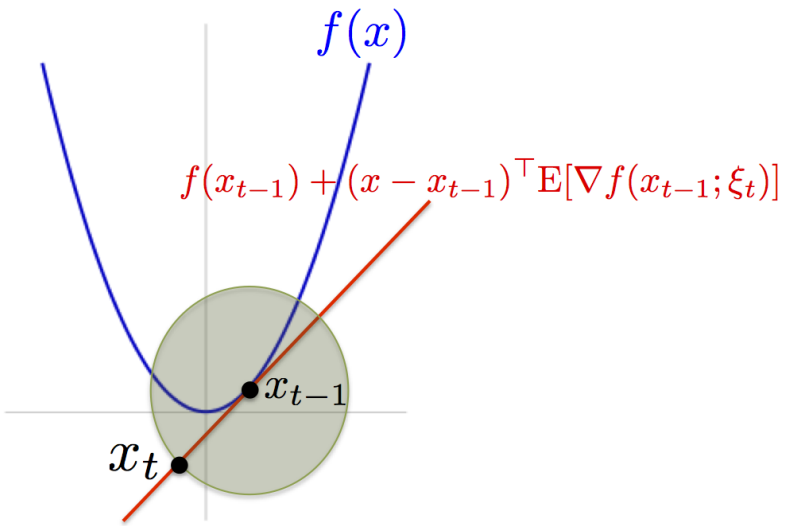
- $\|\mathbf{x} - \mathbf{x}_*\|_2 \leq D$, $\|\nabla f(\mathbf{x}; \xi)\|_2 \leq G$: positive correlation

Interpretation of Gradient Descent

$$x_t = \prod_{\mathcal{X}} [x_{t-1} - \gamma_t \nabla f(x_{t-1}; \xi_t)]$$

$$= \min_{x \in \mathcal{X}} \underbrace{f(x_{t-1}) + (x - x_{t-1})^\top \nabla f(x_{t-1}; \xi_t)}_{\text{linear approximation}}$$

$$+ \frac{1}{2\gamma_t} \underbrace{\|x - x_{t-1}\|_2^2}_{\text{distance to last solution}}$$



$$(x - x_{t-1})^\top \nabla f(x_{t-1}; \xi_t) \leq \|x - x_{t-1}\|_2 \|\nabla f(x_{t-1}; \xi_t)\|_2 \leq GD$$

Stochastic Mirror Descent (Nemirovski et al., 2009)

$$x_t = \min_{x \in \mathcal{X}} \underbrace{f(x_{t-1}) + (x - x_{t-1})^\top \nabla f(x_{t-1}; \xi_t)}_{\text{linear approximation}} + \frac{1}{\gamma_t} \underbrace{B(x, x_{t-1})}_{\text{Bregman Divergence}}$$

- $B(x, x_t) = \omega(x) - \omega(x_t) - \nabla\omega(x_t)^\top (x - x_t)$
- $B(x, x_t) \geq \frac{\alpha}{2}\|x - x_t\|^2$: strongly convex w.r.t general norm
- $(x - x_{t-1})^\top \nabla f(x_{t-1}; \xi_t) \leq \|x - x_{t-1}\| \|\nabla f(x_{t-1}; \xi_t)\|_*$

$$\mathrm{E}[f(\bar{x}_T)] - f(x_*) \leq O\left(\frac{DG}{\sqrt{T}}\right)$$

Stochastic Mirror Descent (Nemirovski et al., 2009)

$$x_t = \min_{x \in \mathcal{X}} \underbrace{f(x_{t-1}) + (x - x_{t-1})^\top \nabla f(x_{t-1}; \xi_t)}_{\text{linear approximation}} + \frac{1}{\gamma_t} \underbrace{B(x, x_{t-1})}_{\text{Bregman Divergence}}$$

- $B(x, x_t) = \omega(x) - \omega(x_t) - \nabla\omega(x_t)^\top (x - x_t)$
- $B(x, x_t) \geq \frac{\alpha}{2}\|x - x_t\|^2$: strongly convex w.r.t general norm
- $(x - x_{t-1})^\top \nabla f(x_{t-1}; \xi_t) \leq \|x - x_{t-1}\| \|\nabla f(x_{t-1}; \xi_t)\|_*$

$\|B(x, x_*)\| \leq D$

$$\mathrm{E}[f(\bar{x}_T)] - f(x_*) \leq O\left(\frac{DG}{\sqrt{T}}\right)$$

Stochastic Mirror Descent (Nemirovski et al., 2009)

$$x_t = \min_{x \in \mathcal{X}} \underbrace{f(x_{t-1}) + (x - x_{t-1})^\top \nabla f(x_{t-1}; \xi_t)}_{\text{linear approximation}} + \frac{1}{\gamma_t} \underbrace{B(x, x_{t-1})}_{\text{Bregman Divergence}}$$

- $B(x, x_t) = \omega(x) - \omega(x_t) - \nabla\omega(x_t)^\top (x - x_t)$
- $B(x, x_t) \geq \frac{\alpha}{2}\|x - x_t\|^2$: strongly convex w.r.t general norm
- $(x - x_{t-1})^\top \nabla f(x_{t-1}; \xi_t) \leq \|x - x_{t-1}\| \|\nabla f(x_{t-1}; \xi_t)\|_*$

$\|\nabla f(x; \xi)\|_* \leq G$

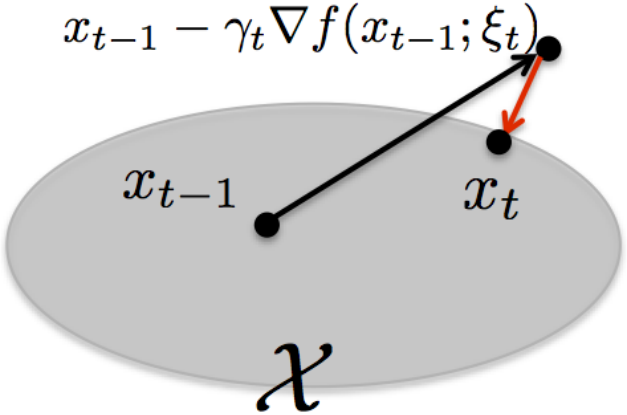
$$\mathrm{E}[f(\bar{x}_T)] - f(x_*) \leq O\left(\frac{DG}{\sqrt{T}}\right)$$

Reducing Projections

Factors that affect Iteration Complexity

- Property of function: smoothness of function
- Size of problem: dimension and number of data points
- Domain $\mathcal{X}$: size and geometry

$$x_t = \prod_{\mathcal{X}}[x_{t-1} - \nabla f(x_{t-1}; \xi_t)]$$



Complex Domain $\mathcal{X}$ leads to Expensive computations

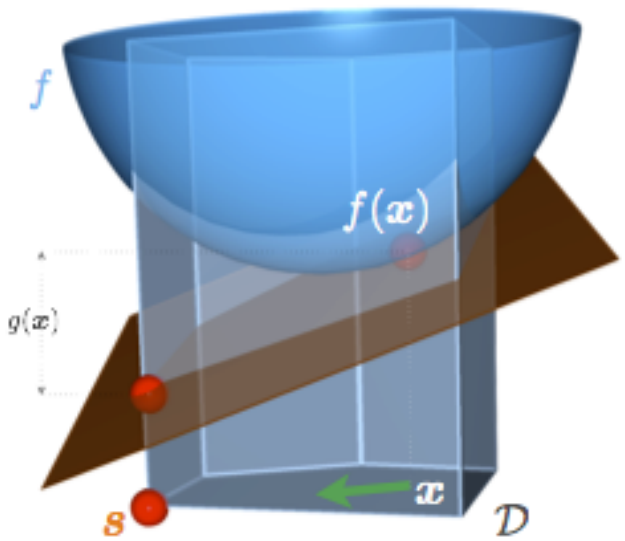
e.g.,PSD cone

Reducing Projections

- Linear Optimization over the Domain: Frank-Wolfe Algorithm (Jaggi, 2013; Lacoste-Julien et al., 2013; Hazan, 2008)

$$s_t = \arg\max_{s \in \mathcal{X}} \langle s, \nabla f(x_{t-1}) \rangle : \quad \text{linear optimization}$$

$$x_t = (1 - \eta_t) x_{t-1} + \eta_t s_t : \quad x_t \in \mathcal{X}$$



Reducing Projections

- Few Projections: SGD with only-one or log T projection (Mahdavi et al., 2012; Yang & Zhang, 2013)

$$x \in \mathcal{X} \Longleftrightarrow g(x) \leq 0$$

SGD for min-max

objective

violation of constraints

$$\min_x \max_{\lambda \geq 0} \quad f(x) + \boxed{\lambda g(x)}$$

Final projection

$$\widetilde{x}_T = \Pi_{\mathcal{X}} \left[\frac{1}{T} \Sigma_{t=1}^{T} x_t \right]$$

How about kernel methods?

Linearization + STOP for linear methods

- the Nyström method (Drineas & Mahoney, 2005)
- Random Fourier Features (Rahimi & Recht, 2007)
- Comparison of two (Yang et al., 2012)
  - the Nyström method: data dependent sampling, better approximation error under large eigen-gap and power law eigen-distribution
  - Random Fourier Features: data independent sampling

Outline

Implementations and A Distributed Library

Efficient implementations and a practical library

- Efficient averaging
- Gradient sparsification
- Distributed (parallel) optimization library

Efficient Averaging

- Update rule:

$$x_t = (1 - \gamma_t \lambda) x_{t-1} + \gamma_t g_t$$
$$\bar{x}_t = (1 - \alpha_t) \bar{x}_{t-1} + \alpha_t x_t$$

- Efficient update when g_t has many 0, or g_t is sparse,

$$S_t = \begin{pmatrix} 1 - \lambda\gamma_t & 0 \\ \alpha_t(1 - \lambda\gamma_t) & 1 - \alpha_t \end{pmatrix} S_{t-1}, \quad S_1 = I$$
$$y_t = y_{t-1} - [S_t^{-1}]_{11} \gamma_t g_t$$
$$\tilde{y}_t = \tilde{y}_{t-1} - ([S_t^{-1}]_{21} + [S_t^{-1}]_{22} \alpha_t) \gamma_t g_t$$

$$x_T = [S_T]_{11} y_T$$
$$\bar{x}_T = [S_T]_{21} y_T + [S_T]_{22} \tilde{y}_T$$

When Gradient is Sparse

Gradient sparsification

- Sparsification by importance sampling

$$R_{ti} = \mathrm{unif}(0, 1)$$
$$\tilde{g}_{ti} = g_{ti}[|g_{ti}| \geq \hat{g}_i] + \hat{g}\mathrm{sign}(g_{ti})[\hat{g}_i R_{ti} \leq |g_{ti}| < \hat{g}_i]$$

- Unbiased sample: $\mathbb{E}\tilde{g}_t = g_t$.
- Tradeoff variance increase for the efficient computation.

Especially useful for Logistic Regression

Distributed Optimization Library: Birds

- The birds library implements distributed stochastic dual coordinate ascent (DisDCA) for classification and regression with a broad support.
- For technical details see:
  - "Trading Computation for Communication: Distributed Stochastic Dual Coordinate Ascent." Tianbao Yang. NIPS 2013.
  - "Analysis of Distributed Stochastic Dual Coordinate Ascent" Tianbao Yang, etc. Tech Report 2013, arxiv.
- The code is distributed under GNU General Publich License (see license.txt for details).

Distributed Optimization Library: Birds

What problems does it solve?

- Classification and Regression
- Loss
  1. Hinge loss and squared hinge loss (SVM)
  2. Logistic loss (Logistic Regression)
  3. Least Square Regression (Ridge Regression)
- Regularizer
  1. ℓ_2 norm: SVM, Logistic Regression, Ridge Regression
  2. ℓ_1 norm: Lasso, SVM, LR with ℓ_1 norm
- Multi-class : one-vs-all

Distributed Optimization Library: Birds

What data does it support?

- dense, sparse
- txt, binary

What environment does it support?

- Prerequisites: Boost.MPI and Boost.Serialization Library
- Tested on A cluster of Linux machines (up to hundreds of processors)

Thank You!

References I

Bradley, Joseph K., Kyrola, Aapo, Bickson, Danny, and Guestrin, Carlos. Parallel coordinate descent for l1-regularized loss minimization. *CoRR*, 2011.

Drineas, Petros and Mahoney, Michael W. On the nystrom method for approximating a gram matrix for improved kernel-based learning. *J. Mach. Learn. Res.*, 6:2153–2175, 2005.

Duchi, John and Singer, Yoram. Efficient online and batch learning using forward backward splitting. *J. Mach. Learn. Res.*, 10:2899–2934, 2009.

Ghaoui, Laurent El, Viallon, Vivian, and Rabbani, Tarek. Safe feature elimination in sparse supervised learning. *CoRR*, abs/1009.3515, 2010.

Hazan, Elad. Sparse approximate solutions to semidefinite programs. In *LATIN*, pp. 306–316, 2008.

References II

Hsieh, Cho-Jui, Chang, Kai-Wei, Lin, Chih-Jen, Keerthi, S. Sathiya, and Sundararajan, S. A dual coordinate descent method for large-scale linear svm. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pp. 408–415, 2008.

Jaggi, Martin. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *ICML 2013 - Proceedings of the 30th International Conference on Machine Learning*, 2013.

Johnson, Rie and Zhang, Tong. Accelerating stochastic gradient descent using predictive variance reduction. In *NIPS*, pp. 315–323, 2013.

Lacoste-Julien, Simon, Jaggi, Martin, Schmidt, Mark W., and Pletscher, Patrick. Block-coordinate frank-wolfe optimization for structural svms. In *ICML (1)*, volume 28, pp. 53–61, 2013.

Lan, Guanghui. An optimal method for stochastic composite optimization. *Math. Program.*, 133(1-2):365–397, 2012.

References III

Langford, John, Li, Lihong, and Zhang, Tong. Sparse online learning via truncated gradient. *J. Mach. Learn. Res.*, 10:777–801, June 2009.

Mahdavi, Mehrdad, Yang, Tianbao, Jin, Rong, Zhu, Shenghuo, and Yi, Jinfeng. Stochastic gradient descent with only one projection. In *NIPS*, pp. 503–511, 2012.

Mahdavi, Mehrdad, Zhang, Lijun, and Jin, Rong. Mixed optimization for smooth functions. In *NIPS*, pp. 674–682, 2013.

Nemirovski, A. and Yudin, D. On cezari?s convergence of the steepest descent method for approximating saddle point of convex-concave functons. *Soviet Math Dkl.*, 19:341–362, 1978.

Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A. Robust stochastic approximation approach to stochastic programming. *SIAM J. on Optimization*, pp. 1574–1609, 2009.

References IV

Nesterov, Yurii. *Introductory Lectures on Convex Optimization: A Basic Course (Applied Optimization)*. Springer Netherlands, 2004.

Niu, Feng, Recht, Benjamin, Re, Christopher, and Wright, Stephen J. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. *CoRR*, 2011.

Ogawa, Kohei, Suzuki, Yoshiki, Suzumura, Shinya, and Takeuchi, Ichiro. Safe sample screening for support vector machines. *CoRR*, 2014.

Rahimi, Ali and Recht, Benjamin. Random features for large-scale kernel machines. In *NIPS*, 2007.

Richtárik, Peter and Takác, Martin. Distributed coordinate descent method for learning with big data. *CoRR*, abs/1310.2059, 2013.

Roux, Nicolas Le, Schmidt, Mark, and Bach, Francis. A stochastic gradient method with an exponential convergence rate for strongly-convex optimization with finite training sets. *CoRR*, 2012.

References V

Shalev-Shwartz, Shai and Tewari, Ambuj. Stochastic methods for l1 regularized loss minimization. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 929–936, 2009.

Shalev-Shwartz, Shai and Zhang, Tong. Proximal stochastic dual coordinate ascent. *CoRR*, abs/1211.2717, 2012.

Shalev-Shwartz, Shai and Zhang, Tong. Stochastic dual coordinate ascent methods for regularized loss. *Journal of Machine Learning Research*, 14: 567–599, 2013.

Shalev-Shwartz, Shai, Singer, Yoram, and Srebro, Nathan. Pegasos: Primal estimated sub-gradient solver for svm. In *Proceedings of the 24th International Conference on Machine Learning*, pp. 807–814, 2007.

References VI

Shalev-Shwartz, Shai, Singer, Yoram, Srebro, Nathan, and Cotter, Andrew. Pegasos: primal estimated sub-gradient solver for svm. *Math. Program.*, 127(1):3–30, 2011.

Shamir, Ohad and Zhang, Tong. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *ICML (1)*, pp. 71–79, 2013.

Wang, Jie, Lin, Binbin, Gong, Pinghua, Wonka, Peter, and Ye, Jieping. Lasso screening rules via dual polytope projection. *CoRR*, abs/1211.3966, 2012.

Wang, Jie, Wonka, Peter, and Ye, Jieping. Scaling svm and least absolute deviations via exact data reduction. *CoRR*, abs/1310.7048, 2013. URL http://dblp.uni-trier.de/db/journals/corr/corr1310.html#WangWY13.

References VII

Yang, Tianbao. Trading computation for communication: Distributed stochastic dual coordinate ascent. *NIPS'13*, pp. –, 2013.

Yang, Tianbao and Zhang, Lijun. Efficient stochastic gradient descent for strongly convex optimization. *CoRR*, abs/1304.5504, 2013.

Yang, Tianbao, Li, Yu-Feng, Mahdavi, Mehrdad, Jin, Rong, and Zhou, Zhi-Hua. "nystrom method vs random fourier features: A theoretical and empirical comparison". In *NIPS*, pp. 485–493, 2012.

Zhu, Shenghuo. Stochastic gradient descent algorithms for strongly convex functions at o(1/t) convergence rates. *CoRR*, abs/1305.2218, 2013.