Optimization for Machine Learning
From Stochastic to Conditional Gradient

Francis Bach

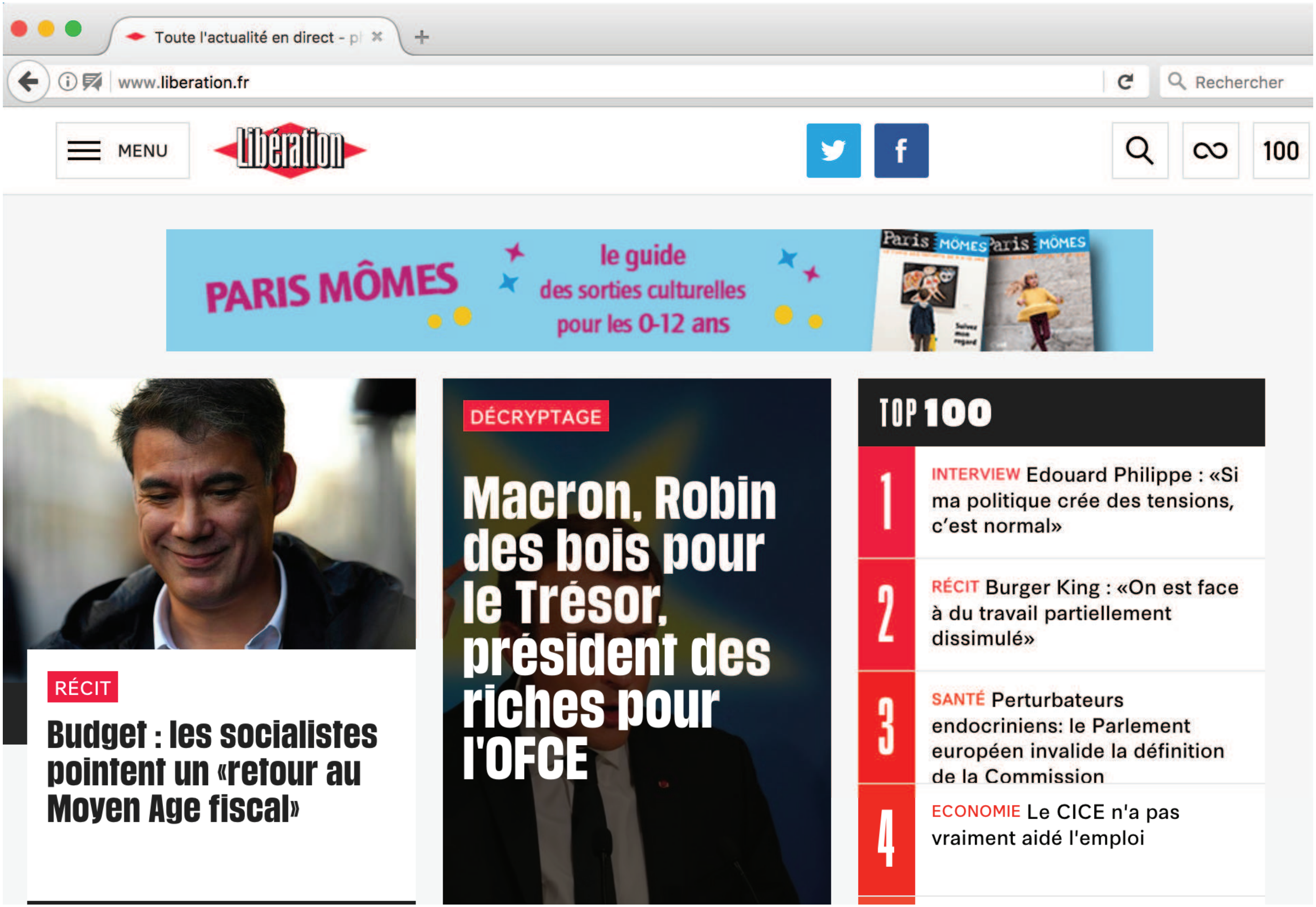
INRIA - Ecole Normale Supérieure, Paris, France

Ecole des Mines - March 2018

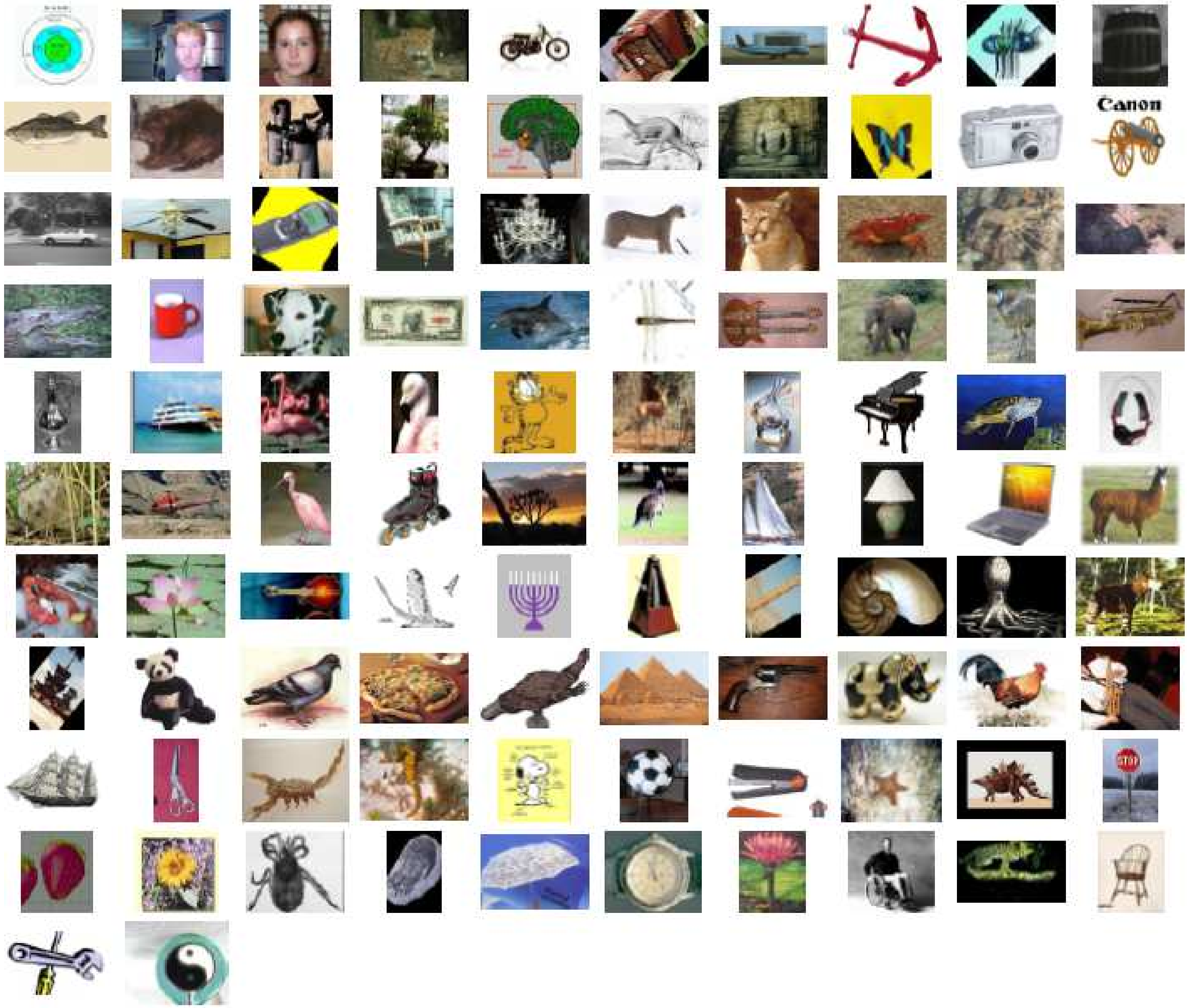
Context
Machine learning for large-scale data

- **Large-scale supervised machine learning**: **large d, large n**
  - d : dimension of each observation (input) or number of parameters
  - n : number of observations
- **Examples**: computer vision, advertising, bioinformatics, etc.

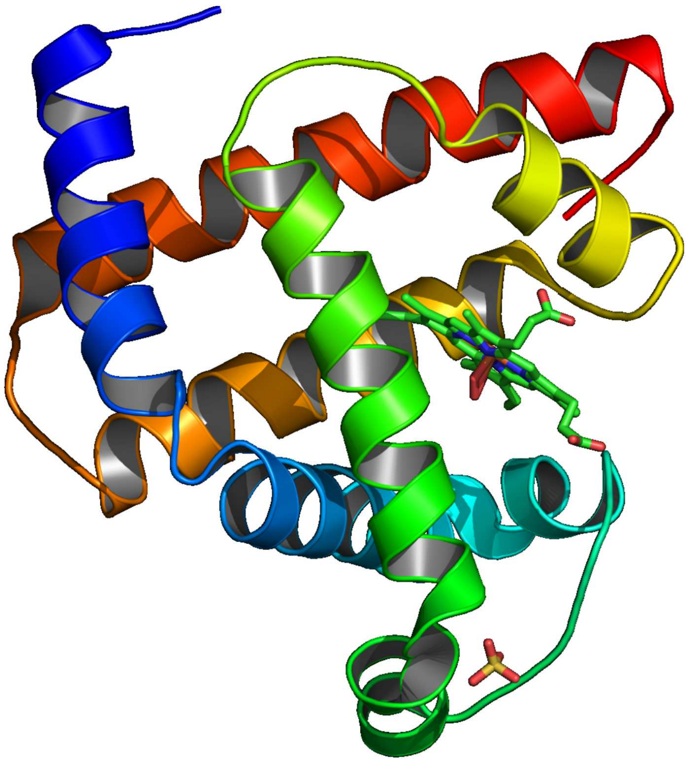
Advertising



Visual object recognition



Bioinformatics



- **Protein**: Crucial elements of cell life
- **Massive data**: 2 millions for humans
- **Complex data**

Context
Machine learning for large-scale data

- **Large-scale supervised machine learning**: **large d, large n**
  - d : dimension of each observation (input), or number of parameters
  - n : number of observations

- **Examples**: computer vision, advertising, bioinformatics, etc.

- **Ideal running-time complexity**: $O(dn)$

Context

Machine learning for large-scale data

- **Large-scale supervised machine learning**: **large d, large n**
  - d : dimension of each observation (input), or number of parameters
  - n : number of observations
- **Examples**: computer vision, advertising, bioinformatics, etc.
- **Ideal running-time complexity**: $O(dn)$
- **Going back to simple methods**
  - Stochastic gradient methods (Robbins and Monro, 1951)
- **Goal: Present classical algorithms and some recent progress**

Scaling to large problems with convex optimization
“Retour aux sources”

- **1950’s**: computers not powerful enough

IBM “1620”, 1959
CPU frequency: 50 KHz
Price $>$ 100 000 dollars

- **2010’s**: Data too massive

Scaling to large problems with convex optimization
"Retour aux sources"

- **1950's**: computers not powerful enough

IBM "1620", 1959
CPU frequency: 50 KHz
Price $>$ 100 000 dollars

- **2010's**: Data too massive

- **One pass through the data** (Robbins et Monro, 1951)
  - Algorithm: $\boxed{\theta_n = \theta_{n-1} - \gamma_n \ell'(y_n, \theta_{n-1}^\top \Phi(x_n))\Phi(x_n)}$

Outline

1. **Introduction/motivation: Supervised machine learning**
   - Optimization of finite sums
   - Batch gradient descent
   - Stochastic gradient descent

2. **Stochastic average gradient (SAG)**
   - Linearly-convergent stochastic gradient method
   - Precise convergence rates
   - From training cost to testing cost

3. **Conditional Gradient (a.k.a. Frank-Wolfe algorithm)**
   - Optimization over convex hulls
   - Application to one-hidden layer neural networks

Parametric supervised machine learning

- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \dots, n$
- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$

Parametric supervised machine learning

- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \dots, n$

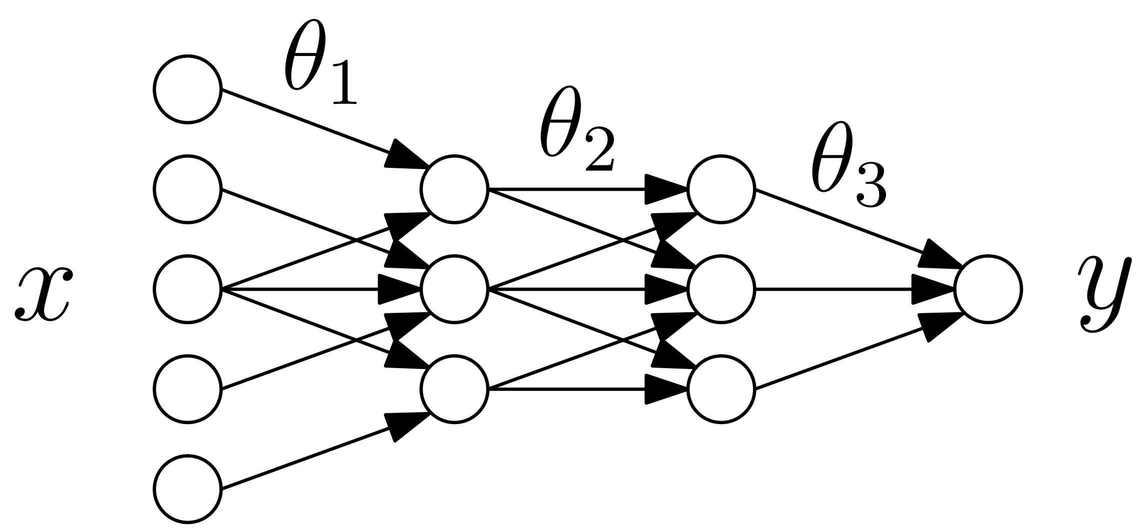
- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$

- **Motivating examples**

  – Linear predictions: $h(x, \theta) = \theta^\top \Phi(x)$ with features $\Phi(x) \in \mathbb{R}^d$

Parametric supervised machine learning

- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \dots, n$
- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$
- **Motivating examples**
  - Linear predictions: $h(x, \theta) = \theta^\top \Phi(x)$ with features $\Phi(x) \in \mathbb{R}^d$
  - Neural networks: $h(x, \theta) = \theta_m^\top \sigma(\theta_{m-1}^\top \sigma(\cdots \theta_2^\top \sigma(\theta_1^\top x))$



Parametric supervised machine learning

- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \ldots, n$
- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$
- **(regularized) empirical risk minimization**: find $\hat{\theta}$ solution of

$$\min_{\theta \in \mathbb{R}^d} \quad \frac{1}{n} \sum_{i=1}^{n} \ \ell\big(y_i, h(x_i, \theta)\big) \quad + \quad \lambda \Omega(\theta)$$

data fitting term + regularizer

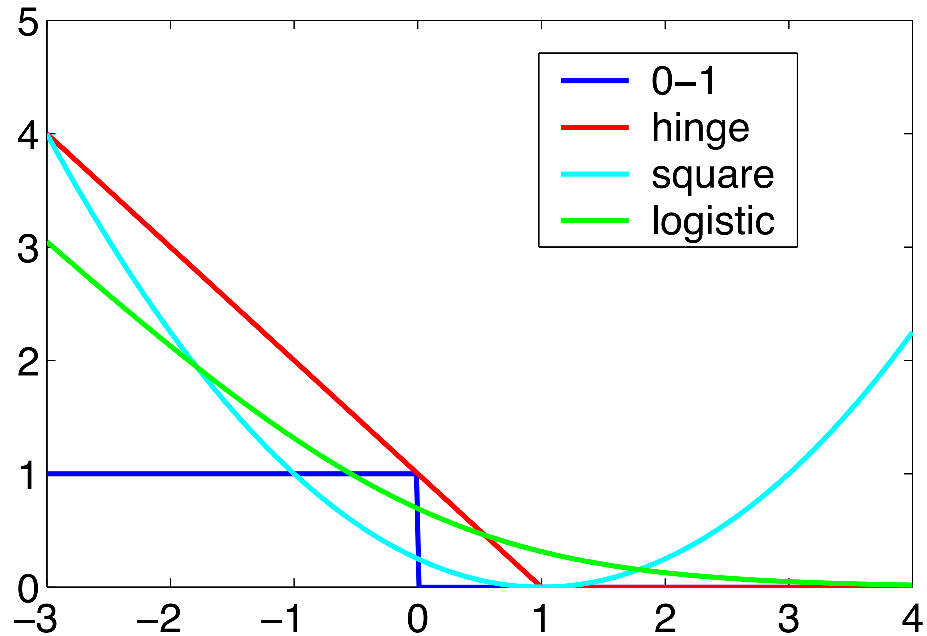
Usual losses

- **Regression**: $y \in \mathbb{R}$, prediction $\hat{y} = h(x, \theta)$
  - quadratic loss $\frac{1}{2}(y - \hat{y})^2 = \frac{1}{2}(y - h(x, \theta))^2$

Usual losses

- **Regression**: $y \in \mathbb{R}$, prediction $\hat{y} = h(x, \theta)$
  – quadratic loss $\frac{1}{2}(y - \hat{y})^2 = \frac{1}{2}(y - h(x, \theta))^2$

- **Classification** : $y \in \{-1, 1\}$, prediction $\hat{y} = \text{sign}(h(x, \theta))$
  – loss of the form $\ell(y\,h(x, \theta))$
  – "True" 0-1 loss: $\ell(y\,h(x, \theta)) = 1_{y\,h(x,\theta)<0}$
  – Usual convex losses:



Main motivating examples

- **Support vector machine** (hinge loss): non-smooth

$$\ell(Y, h(X\theta)) = \max\{1 - Yh(X,\theta), 0\}$$

- **Logistic regression**: smooth

$$\ell(Y, h(X\theta)) = \log(1 + \exp(-Yh(X,\theta)))$$

- **Least-squares regression**

$$\ell(Y, h(X\theta)) = \frac{1}{2}(Y - h(X,\theta))^2$$

- **Structured output regression**
  - See Tsochantaridis et al. (2005); Lacoste-Julien et al. (2013)

Usual regularizers

- **Main goal**: avoid overfitting

- **(squared) Euclidean norm**: $\|\theta\|_2^2 = \sum_{j=1}^d |\theta_j|^2$
  - Numerically well-behaved if $h(x,\theta) = \theta^\top \Phi(x)$
  - Representer theorem and kernel methods : $\theta = \sum_{i=1}^n \alpha_i \Phi(x_i)$
  - See, e.g., Schölkopf and Smola (2001); Shawe-Taylor and Cristianini (2004)

Usual regularizers

- **Main goal**: avoid overfitting

- **(squared) Euclidean norm**: $\|\theta\|_2^2 = \sum_{j=1}^{d} |\theta_j|^2$
  - Numerically well-behaved if $h(x, \theta) = \theta^\top \Phi(x)$
  - Representer theorem and kernel methods : $\theta = \sum_{i=1}^{n} \alpha_i \Phi(x_i)$
  - See, e.g., Schölkopf and Smola (2001); Shawe-Taylor and Cristianini (2004)

- **Sparsity-inducing norms**
  - Main example: ℓ_1-norm $\|\theta\|_1 = \sum_{j=1}^{d} |\theta_j|$
  - Perform model selection as well as regularization
  - Non-smooth optimization and structured sparsity
  - See, e.g., Bach, Jenatton, Mairal, and Obozinski (2012a,b)

Parametric supervised machine learning

- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \dots, n$
- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$
- **(regularized) empirical risk minimization**: find $\hat{\theta}$ solution of

$$\min_{\theta \in \mathbb{R}^d} \quad \frac{1}{n} \sum_{i=1}^{n} \quad \ell\big(y_i, h(x_i, \theta)\big) \quad + \quad \lambda \Omega(\theta)$$

data fitting term $+$ regularizer

Parametric supervised machine learning

- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \dots, n$
- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$
- **(regularized) empirical risk minimization**: find $\hat{\theta}$ solution of

$$\min_{\theta \in \mathbb{R}^d} \quad \frac{1}{n} \sum_{i=1}^{n} \Big\{ \ell\big(y_i, h(x_i, \theta)\big) \quad + \quad \lambda \Omega(\theta) \Big\} \qquad = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

data fitting term $+$ regularizer

Parametric supervised machine learning

- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \dots, n$
- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$
- **(regularized) empirical risk minimization**: find $\hat{\theta}$ solution of

$$\min_{\theta \in \mathbb{R}^d} \quad \frac{1}{n} \sum_{i=1}^{n} \Big\{ \ell\big(y_i, h(x_i, \theta)\big) \quad + \quad \lambda \Omega(\theta) \Big\} \qquad = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

data fitting term + regularizer

- Optimization: optimization of regularized risk training cost

Parametric supervised machine learning

- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \dots, n$, **i.i.d.**
- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$
- **(regularized) empirical risk minimization**: find $\hat{\theta}$ solution of

$$\min_{\theta \in \mathbb{R}^d} \quad \frac{1}{n} \sum_{i=1}^{n} \Big\{ \ell\big(y_i, h(x_i, \theta)\big) \quad + \quad \lambda \Omega(\theta) \Big\} \qquad = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

data fitting term + regularizer

- Optimization: optimization of regularized risk — training cost
- Statistics: guarantees on $\mathbb{E}_{p(x,y)} \ell(y, h(x, \theta))$ — testing cost

Finite sums beyond machine learning

- **Model fitting**
  - *Same optimization problem*: $\displaystyle \min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell\big(y_i, h(x_i, \theta)\big) + \lambda \Omega(\theta)$

Finite sums beyond machine learning

- **Model fitting**
  - *Same optimization problem*: $\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell\big(y_i, h(x_i, \theta)\big) + \lambda \Omega(\theta)$
  - *Differences:* (1) Typically need high precision for θ
    (2) Data (x_i, y_i) may not be i.i.d.

Finite sums beyond machine learning

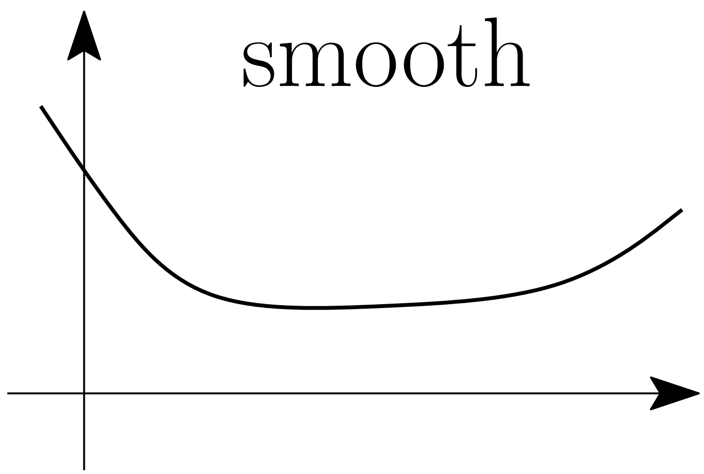
- **Model fitting**
  - *Same optimization problem*: $\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell\big(y_i, h(x_i, \theta)\big) + \lambda \Omega(\theta)$
  - *Differences:* (1) Typically need high precision for θ
    (2) Data (x_i, y_i) may not be i.i.d.

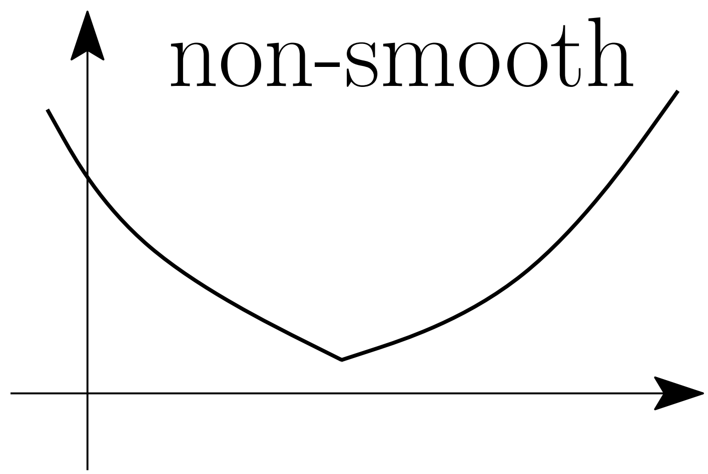
- **Structured regularization**
  - E.g., total variation $\sum_{i \sim j} |\theta_i - \theta_j|$

Smoothness and (strong) convexity

- A function $g : \mathbb{R}^d \to \mathbb{R}$ is L-smooth if and only if it is twice differentiable and

$$\forall \theta \in \mathbb{R}^d, \ \left| \text{eigenvalues}\big[g''(\theta)\big] \right| \leqslant L$$





Smoothness and (strong) convexity

- A function $g : \mathbb{R}^d \to \mathbb{R}$ is L-smooth if and only if it is twice differentiable and

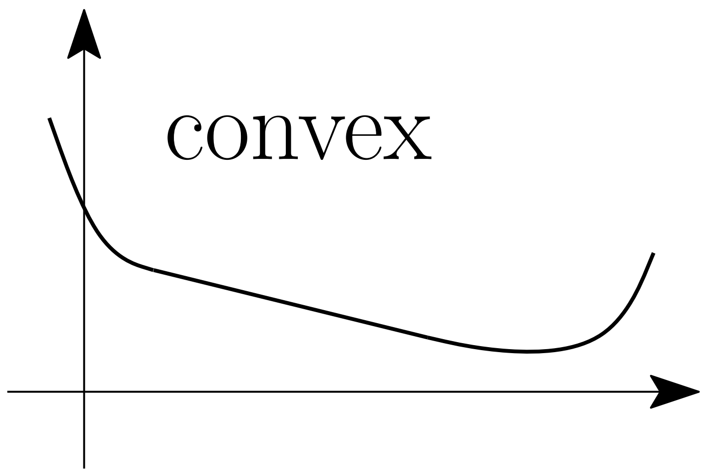
$$\forall \theta \in \mathbb{R}^d, \ \big| \text{eigenvalues}\big[g''(\theta)\big]\big| \leqslant L$$

- **Machine learning**
  - with $g(\theta) = \frac{1}{n}\sum_{i=1}^n \ell(y_i, h(x_i, \theta))$
  - Smooth prediction function $\theta \mapsto h(x_i, \theta)$ + smooth loss

Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is convex if and only if

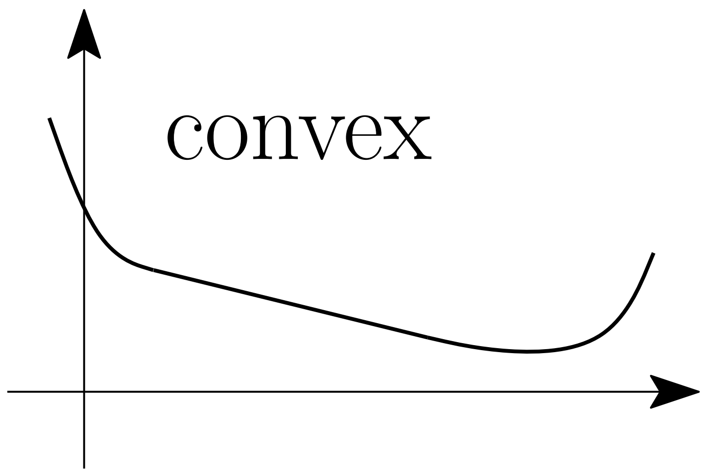
$$\forall \theta \in \mathbb{R}^d, \ \text{eigenvalues}\big[g''(\theta)\big] \geqslant 0$$

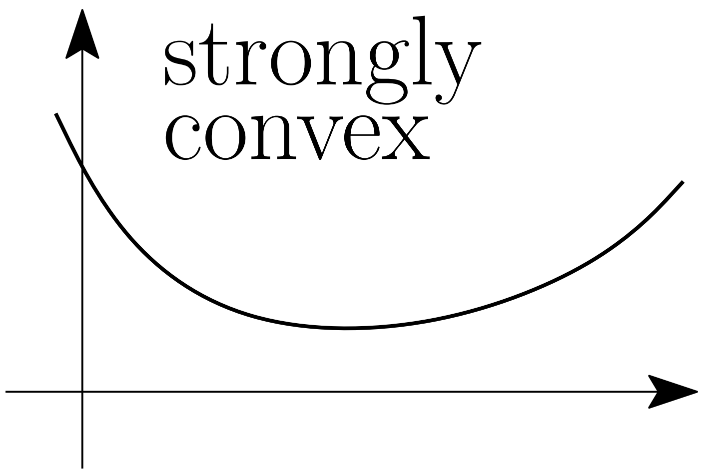


Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

$$\forall \theta \in \mathbb{R}^d,\ \text{eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$



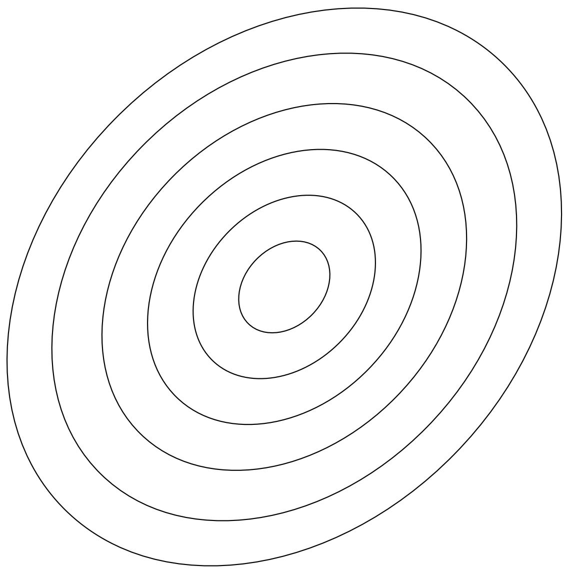


Smoothness and (strong) convexity

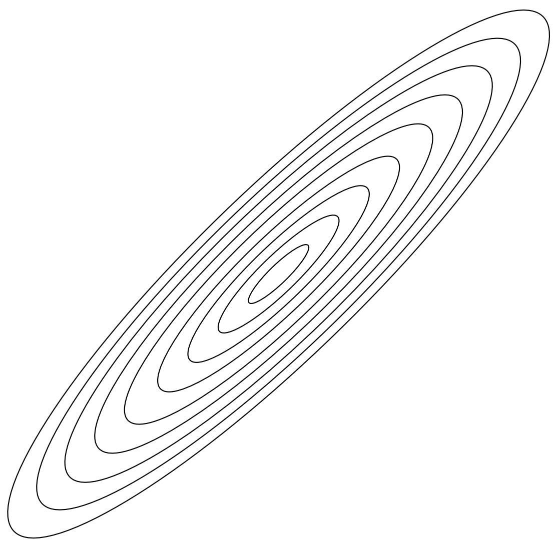
- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

$$\forall \theta \in \mathbb{R}^d, \ \text{eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$

  – Condition number $\kappa = L/\mu \geqslant 1$



(small $\kappa = L/\mu$)



(large $\kappa = L/\mu$)

Smoothness and (strong) convexity

- A twice differentiable function $g:\mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if
$$\forall \theta \in \mathbb{R}^d, \ \text{eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$

- **Convexity in machine learning**
  - With $g(\theta) = \frac{1}{n}\sum_{i=1}^n \ell(y_i, h(x_i,\theta))$
  - Convex loss and linear predictions $h(x,\theta) = \theta^\top \Phi(x)$

Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

$$\forall \theta \in \mathbb{R}^d, \ \text{eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$

- **Convexity in machine learning**
  - With $g(\theta) = \frac{1}{n}\sum_{i=1}^n \ell(y_i, h(x_i, \theta))$
  - Convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$

- **Relevance of convex optimization**
  - Easier design and analysis of algorithms
  - Global minimum vs. local minimum vs. stationary points
  - Gradient-based algorithms only need convexity for their analysis

Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

$$\forall \theta \in \mathbb{R}^d, \text{ eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$

- **Strong convexity in machine learning**
  - With $g(\theta) = \frac{1}{n}\sum_{i=1}^n \ell(y_i, h(x_i, \theta))$
  - Strongly convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$

Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

$$\forall \theta \in \mathbb{R}^d, \ \text{eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$

- **Strong convexity in machine learning**
  - With $g(\theta) = \frac{1}{n}\sum_{i=1}^n \ell(y_i, h(x_i, \theta))$
  - Strongly convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$
  - Invertible covariance matrix $\frac{1}{n}\sum_{i=1}^n \Phi(x_i)\Phi(x_i)^\top \Rightarrow n \geqslant d$ *(board)*
  - Even when $\mu > 0$, μ may be arbitrarily small!

Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

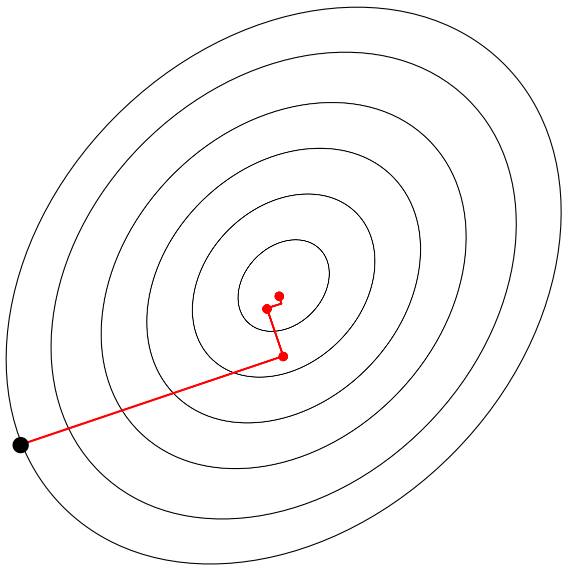
$$\forall \theta \in \mathbb{R}^d, \text{ eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$

- **Strong convexity in machine learning**
  - With $g(\theta) = \frac{1}{n}\sum_{i=1}^n \ell(y_i, h(x_i, \theta))$
  - Strongly convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$
  - Invertible covariance matrix $\frac{1}{n}\sum_{i=1}^n \Phi(x_i)\Phi(x_i)^\top \Rightarrow n \geqslant d$ *(board)*
  - Even when $\mu > 0$, μ may be arbitrarily small!

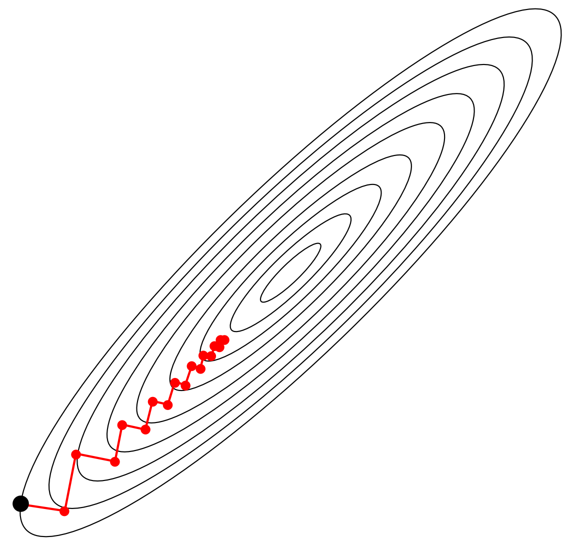
- **Adding regularization by $\frac{\mu}{2}\|\theta\|^2$**
  - creates additional bias unless μ is small, but reduces variance
  - Typically $L/\sqrt{n} \geqslant \mu \geqslant L/n$

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$
- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$ *(line search)*



(small $\kappa = L/\mu$)



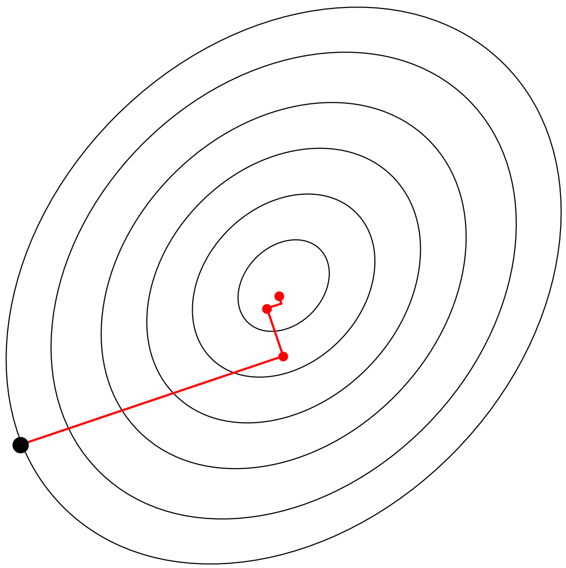
(large $\kappa = L/\mu$)

Iterative methods for minimizing smooth functions

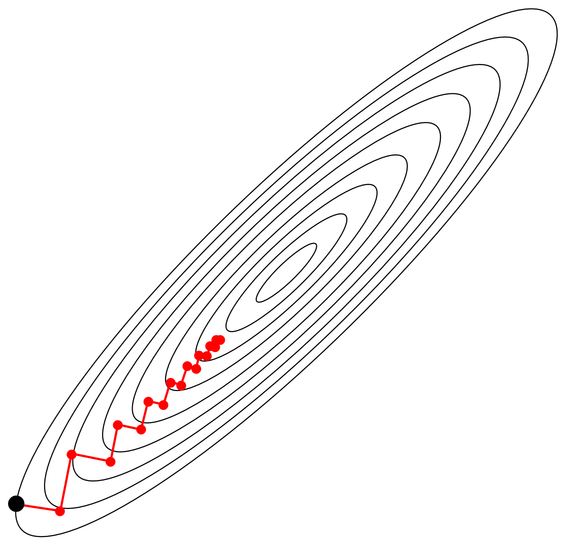
- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$

- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$ *(line search)*

  $g(\theta_t) - g(\theta_*) \leqslant O(1/t)$
  $g(\theta_t) - g(\theta_*) \leqslant O((1-\mu/L)^t) = O(e^{-t(\mu/L)})$ if μ-strongly convex



(small $\kappa = L/\mu$)



(large $\kappa = L/\mu$)

Gradient descent - Proof for quadratic functions

- Quadratic convex function: $g(\theta) = \frac{1}{2}\theta^\top H\theta - c^\top\theta$
  - μ and L are smallest largest eigenvalues of H
  - Global optimum $\theta_* = H^{-1}c$ (or $H^\dagger c$) such that $H\theta_* = c$

Gradient descent - Proof for quadratic functions

- Quadratic convex function: $g(\theta) = \frac{1}{2}\theta^\top H\theta - c^\top\theta$
  - μ and L are smallest largest eigenvalues of H
  - Global optimum $\theta_* = H^{-1}c$ (or $H^\dagger c$) such that $H\theta_* = c$

- Gradient descent with $\gamma = 1/L$:

$$\begin{aligned}
\theta_t &= \theta_{t-1} - \frac{1}{L}(H\theta_{t-1} - c) = \theta_{t-1} - \frac{1}{L}(H\theta_{t-1} - H\theta_*) \\
\theta_t - \theta_* &= (I - \frac{1}{L}H)(\theta_{t-1} - \theta_*) = (I - \frac{1}{L}H)^t(\theta_0 - \theta_*)
\end{aligned}$$

Gradient descent - Proof for quadratic functions

- Quadratic convex function: $g(\theta) = \frac{1}{2}\theta^\top H\theta - c^\top\theta$
  - μ and L are smallest largest eigenvalues of H
  - Global optimum $\theta_* = H^{-1}c$ (or $H^\dagger c$) such that $H\theta_* = c$

- Gradient descent with $\gamma = 1/L$:

$$\begin{aligned}
\theta_t &= \theta_{t-1} - \frac{1}{L}(H\theta_{t-1} - c) = \theta_{t-1} - \frac{1}{L}(H\theta_{t-1} - H\theta_*) \\
\theta_t - \theta_* &= (I - \frac{1}{L}H)(\theta_{t-1} - \theta_*) = (I - \frac{1}{L}H)^t(\theta_0 - \theta_*)
\end{aligned}$$

- **Strong convexity** $\mu > 0$: eigenvalues of $(I - \frac{1}{L}H)^t$ in $[0, (1 - \frac{\mu}{L})^t]$
  - Convergence of iterates: $\|\theta_t - \theta_*\|^2 \leqslant (1 - \mu/L)^{2t}\|\theta_0 - \theta_*\|^2$
  - Function values: $g(\theta_t) - g(\theta_*) \leqslant (1 - \mu/L)^{2t}\big[g(\theta_0) - g(\theta_*)\big]$

Gradient descent - Proof for quadratic functions

- Quadratic convex function: $g(\theta) = \frac{1}{2}\theta^\top H\theta - c^\top\theta$
  - μ and L are smallest largest eigenvalues of H
  - Global optimum $\theta_* = H^{-1}c$ (or $H^\dagger c$) such that $H\theta_* = c$

- Gradient descent with $\gamma = 1/L$:

$$\begin{aligned}\theta_t &= \theta_{t-1} - \frac{1}{L}(H\theta_{t-1} - c) = \theta_{t-1} - \frac{1}{L}(H\theta_{t-1} - H\theta_*) \\ \theta_t - \theta_* &= (I - \frac{1}{L}H)(\theta_{t-1} - \theta_*) = (I - \frac{1}{L}H)^t(\theta_0 - \theta_*)\end{aligned}$$

- **Convexity** $\mu = 0$: eigenvalues of $(I - \frac{1}{L}H)^t$ in $[0, 1]$
  - No convergence of iterates: $\|\theta_t - \theta_*\|^2 \leqslant \|\theta_0 - \theta_*\|^2$
  - Function values: $g(\theta_t) - g(\theta_*) \leqslant \max_{v\in[0,L]} v(1 - v/L)^{2t}\|\theta_0 - \theta_*\|^2$

$$g(\theta_t) - g(\theta_*) \leqslant \frac{L}{t}\|\theta_0 - \theta_*\|^2$$

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$

- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$
  - $O(1/t)$ convergence rate for convex functions
  - $O(e^{-t/\kappa})$ *linear* if strongly-convex

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$

- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$
  - $O(1/t)$ convergence rate for convex functions
  - $O(e^{-t/\kappa})$ *linear* if strongly-convex

- **Newton method**: $\theta_t = \theta_{t-1} - g''(\theta_{t-1})^{-1} g'(\theta_{t-1})$
  - $O\big(e^{-\rho 2^t}\big)$ *quadratic* rate *(see board)*

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$

- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$
  - $O(1/t)$ convergence rate for convex functions
  - $O(e^{-t/\kappa})$ *linear* if strongly-convex $\Leftrightarrow$ $O(\kappa \log \frac{1}{\varepsilon})$ iterations

- **Newton method**: $\theta_t = \theta_{t-1} - g''(\theta_{t-1})^{-1} g'(\theta_{t-1})$
  - $O\big(e^{-\rho 2^t}\big)$ *quadratic* rate $\Leftrightarrow$ $O(\log \log \frac{1}{\varepsilon})$ iterations

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$

- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$
  - $O(1/t)$ convergence rate for convex functions
  - $O(e^{-t/\kappa})$ *linear* if strongly-convex $\Leftrightarrow$ complexity $= O(nd \cdot \kappa \log \frac{1}{\varepsilon})$

- **Newton method**: $\theta_t = \theta_{t-1} - g''(\theta_{t-1})^{-1} g'(\theta_{t-1})$
  - $O\big(e^{-\rho 2^t}\big)$ *quadratic* rate $\Leftrightarrow$ complexity $= O((nd^2 + d^3) \cdot \log \log \frac{1}{\varepsilon})$

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$
- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$
  - $O(1/t)$ convergence rate for convex functions
  - $O(e^{-t/\kappa})$ *linear* if strongly-convex $\Leftrightarrow$ complexity $= O(nd \cdot \kappa \log \frac{1}{\varepsilon})$
- **Newton method**: $\theta_t = \theta_{t-1} - g''(\theta_{t-1})^{-1} g'(\theta_{t-1})$
  - $O\big(e^{-\rho 2^t}\big)$ *quadratic* rate $\Leftrightarrow$ complexity $= O((nd^2 + d^3) \cdot \log \log \frac{1}{\varepsilon})$
- **Key insights for machine learning (Bottou and Bousquet, 2008)**

1. No need to optimize below statistical error
2. Cost functions are averages
3. Testing error is more important than training error

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$

- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$
  - $O(1/t)$ convergence rate for convex functions
  - $O(e^{-t/\kappa})$ *linear* if strongly-convex $\Leftrightarrow$ complexity $= O(nd \cdot \kappa \log \frac{1}{\varepsilon})$

- **Newton method**: $\theta_t = \theta_{t-1} - g''(\theta_{t-1})^{-1} g'(\theta_{t-1})$
  - $O\big(e^{-\rho 2^t}\big)$ *quadratic* rate $\Leftrightarrow$ complexity $= O((nd^2 + d^3) \cdot \log \log \frac{1}{\varepsilon})$

- **Key insights for machine learning (Bottou and Bousquet, 2008)**
  1. No need to optimize below statistical error
  2. Cost functions are averages
  3. Testing error is more important than training error

Stochastic gradient descent (SGD) for finite sums

$$\min_{\theta \in \mathbb{R}^d} g(\theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

- **Iteration**: $\theta_t = \theta_{t-1} - \gamma_t f'_{i(t)}(\theta_{t-1})$
  - Sampling with replacement: $i(t)$ random element of $\{1, \dots, n\}$
  - Polyak-Ruppert averaging: $\bar{\theta}_t = \frac{1}{t+1} \sum_{u=0}^{t} \theta_u$

Stochastic gradient descent (SGD) for finite sums

$$\min_{\theta \in \mathbb{R}^d} g(\theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

- **Iteration**: $\theta_t = \theta_{t-1} - \gamma_t f'_{i(t)}(\theta_{t-1})$
  - Sampling with replacement: $i(t)$ random element of $\{1, \dots, n\}$
  - Polyak-Ruppert averaging: $\bar{\theta}_t = \frac{1}{t+1} \sum_{u=0}^{t} \theta_u$
- **Convergence rate** if each f_i is convex L-smooth and g μ-strongly-convex:

$$\mathbb{E} g(\bar{\theta}_t) - g(\theta_*) \leqslant \begin{cases} O(1/\sqrt{t}) & \text{if } \gamma_t = 1/(L\sqrt{t}) \\ O(L/(\mu t)) = O(\kappa/t) & \text{if } \gamma_t = 1/(\mu t) \end{cases}$$

  - No adaptivity to strong-convexity in general
  - Running-time complexity: $O(d \cdot \kappa/\varepsilon)$

Non-asymptotic analysis (Bach and Moulines, 2011)

- Stochastic gradient descent with learning rate $\gamma_t = Ct^{-\alpha}$

- **Strongly convex smooth objective functions**
  - Old: $O(1/(\mu t))$ rate achieved without averaging for $\alpha = 1$
  - New: $O(1/(\mu t))$ rate achieved with averaging for $\alpha \in [1/2, 1]$
  - Non-asymptotic analysis with explicit constants
  - Forgetting of initial conditions
  - Robustness to the choice of C

Non-asymptotic analysis (Bach and Moulines, 2011)

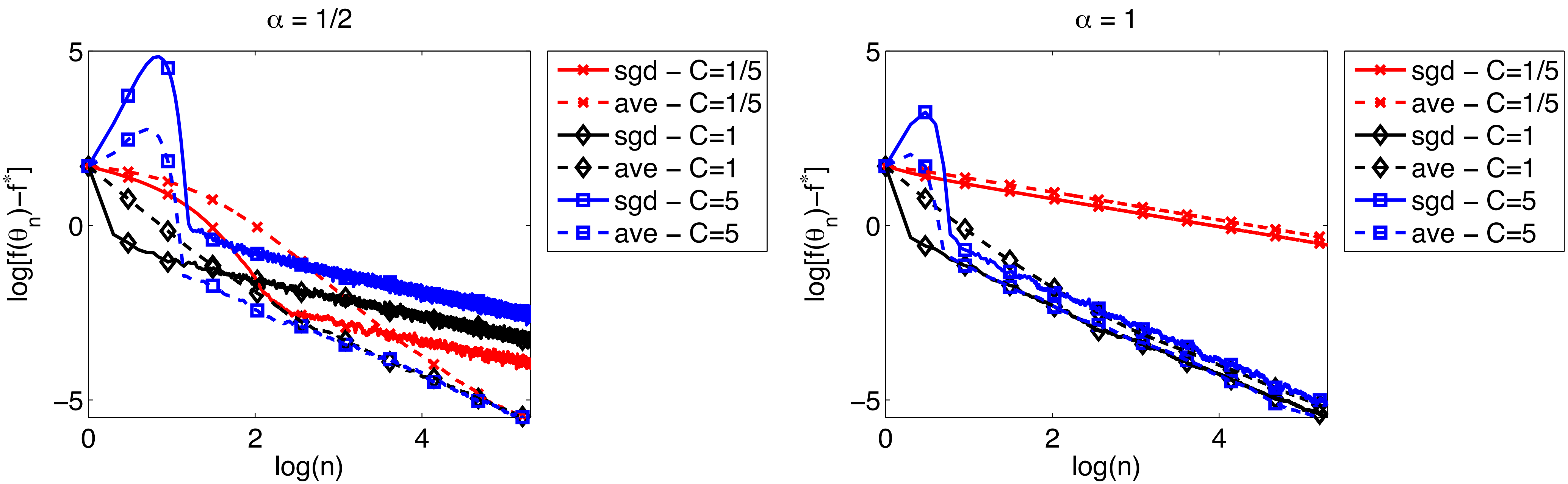
- Stochastic gradient descent with learning rate $\gamma_t = Ct^{-\alpha}$

- **Strongly convex smooth objective functions**
  - Old: $O(1/(\mu t))$ rate achieved without averaging for $\alpha = 1$
  - New: $O(1/(\mu t))$ rate achieved with averaging for $\alpha \in [1/2, 1]$
  - Non-asymptotic analysis with explicit constants
  - Forgetting of initial conditions
  - Robustness to the choice of C

- **Convergence rates** for $\mathbb{E}\|\theta_t - \theta_*\|^2$ and $\mathbb{E}\|\bar{\theta}_t - \theta_*\|^2$
  - no averaging: $O\Big(\frac{\sigma^2\gamma_t}{\mu}\Big) + O(e^{-\mu t \gamma_t})\|\theta_0 - \theta_*\|^2$
  - averaging: $\frac{\operatorname{tr} H(\theta_*)^{-1}}{t} + \mu^{-1}O(t^{-2\alpha} + t^{-2+\alpha}) + O\Big(\frac{\|\theta_0 - \theta_*\|^2}{\mu^2 t^2}\Big)$

Robustness to wrong constants for $\gamma_t = Ct^{-\alpha}$

- $f(\theta) = \frac{1}{2}|\theta|^2$ with i.i.d. Gaussian noise ($d = 1$)
- Left: $\alpha = 1/2$
- Right: $\alpha = 1$



- See also `http://leon.bottou.org/projects/sgd`

Non-asymptotic analysis (Bach and Moulines, 2011)

- Stochastic gradient descent with learning rate $\gamma_t = Ct^{-\alpha}$

- **Strongly convex smooth objective functions**
  - Old: $O(1/(\mu t))$ rate achieved without averaging for $\alpha = 1$
  - New: $O(1/(\mu t))$ rate achieved with averaging for $\alpha \in [1/2, 1]$
  - Non-asymptotic analysis with explicit constants

Non-asymptotic analysis (Bach and Moulines, 2011)

- Stochastic gradient descent with learning rate $\gamma_t = Ct^{-\alpha}$

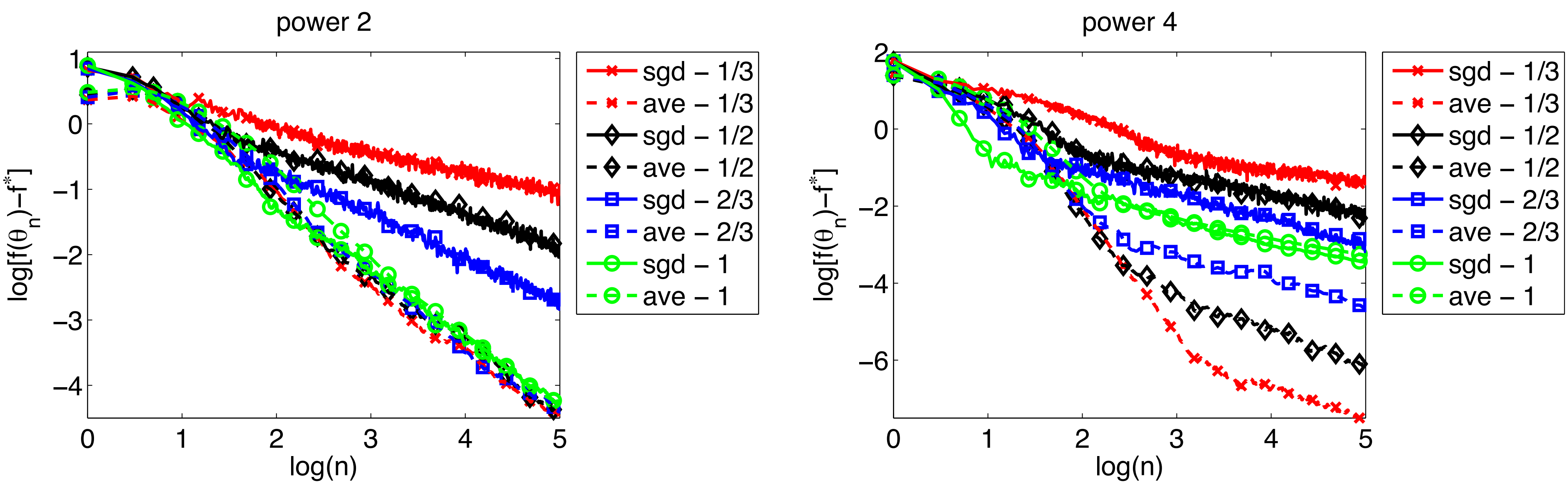
- **Strongly convex smooth objective functions**
  - Old: $O(1/(\mu t))$ rate achieved without averaging for $\alpha = 1$
  - New: $O(1/(\mu t))$ rate achieved with averaging for $\alpha \in [1/2, 1]$
  - Non-asymptotic analysis with explicit constants

- **Non-strongly convex smooth objective functions**
  - Old: $O(t^{-1/2})$ rate achieved with averaging for $\alpha = 1/2$
  - New: $O(\max\{t^{1/2-3\alpha/2}, t^{-\alpha/2}, t^{\alpha-1}\})$ rate achieved without averaging for $\alpha \in [1/3, 1]$

- **Take-home message**
  - Use $\alpha = 1/2$ with averaging to be adaptive to strong convexity

Robustness to lack of strong convexity

- Left: $f(\theta) = |\theta|^2$ between -1 and 1
- Right: $f(\theta) = |\theta|^4$ between -1 and 1
- affine outside of $[-1, 1]$, continuously differentiable.



Outline

1. **Introduction/motivation: Supervised machine learning**
   - Optimization of finite sums
   - Batch gradient descent
   - Stochastic gradient descent

2. **Stochastic average gradient (SAG)**
   - Linearly-convergent stochastic gradient method
   - Precise convergence rates
   - From training cost to testing cost

3. **Conditional Gradient (a.k.a. Frank-Wolfe algorithm)**
   - Optimization over convex hulls
   - Application to one-hidden layer neural networks

Stochastic vs. deterministic methods

- Minimizing $g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$ with $f_i(\theta) = \ell\big(y_i, h(x_i, \theta)\big) + \lambda\Omega(\theta)$

Stochastic vs. deterministic methods

- Minimizing $g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$ with $f_i(\theta) = \ell\big(y_i, h(x_i, \theta)\big) + \lambda\Omega(\theta)$

- Batch gradient descent: $\theta_t = \theta_{t-1} - \gamma_t g'(\theta_{t-1}) = \theta_{t-1} - \frac{\gamma_t}{n}\sum_{i=1}^{n} f_i'(\theta_{t-1})$

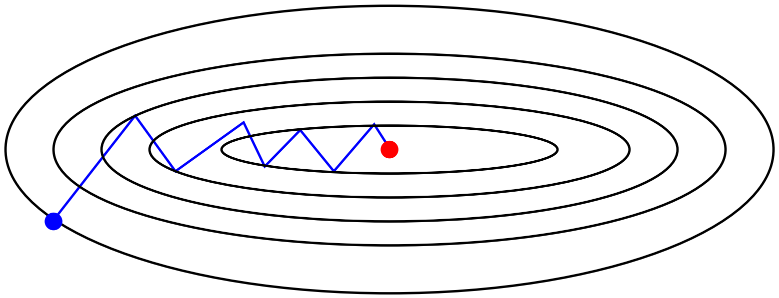
  – Linear (e.g., exponential) convergence rate in $O(e^{-t/\kappa})$

  – Iteration complexity is linear in n

Stochastic vs. deterministic methods

- Minimizing $g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$ with $f_i(\theta) = \ell\big(y_i, h(x_i,\theta)\big) + \lambda\Omega(\theta)$

- Batch gradient descent: $\theta_t = \theta_{t-1} - \gamma_t g'(\theta_{t-1}) = \theta_{t-1} - \frac{\gamma_t}{n}\sum_{i=1}^{n} f_i'(\theta_{t-1})$



Stochastic vs. deterministic methods

- Minimizing $g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$ with $f_i(\theta) = \ell\big(y_i, h(x_i,\theta)\big) + \lambda\Omega(\theta)$

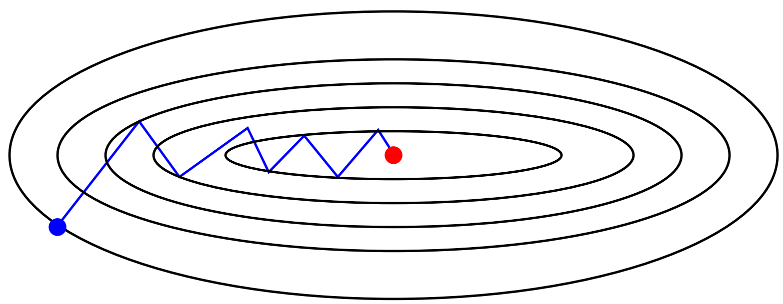
- Batch gradient descent: $\theta_t = \theta_{t-1} - \gamma_t g'(\theta_{t-1}) = \theta_{t-1} - \frac{\gamma_t}{n}\sum_{i=1}^{n} f_i'(\theta_{t-1})$
  - Linear (e.g., exponential) convergence rate in $O(e^{-t/\kappa})$
  - Iteration complexity is linear in n

- Stochastic gradient descent: $\theta_t = \theta_{t-1} - \gamma_t f'_{i(t)}(\theta_{t-1})$
  - Sampling with replacement: $i(t)$ random element of $\{1,\dots,n\}$
  - Convergence rate in $O(\kappa/t)$
  - Iteration complexity is independent of n

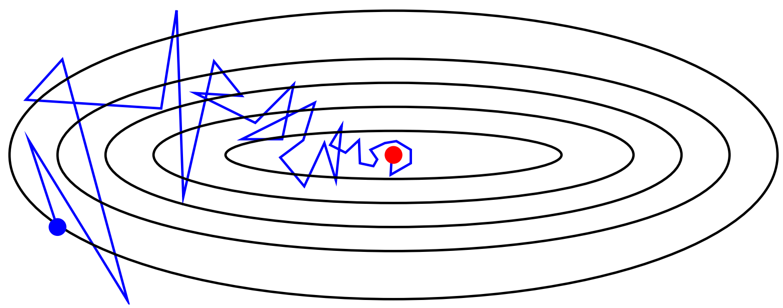
Stochastic vs. deterministic methods

- Minimizing $g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$ with $f_i(\theta) = \ell\big(y_i, h(x_i, \theta)\big) + \lambda\Omega(\theta)$

- Batch gradient descent: $\theta_t = \theta_{t-1} - \gamma_t g'(\theta_{t-1}) = \theta_{t-1} - \frac{\gamma_t}{n}\sum_{i=1}^{n} f_i'(\theta_{t-1})$

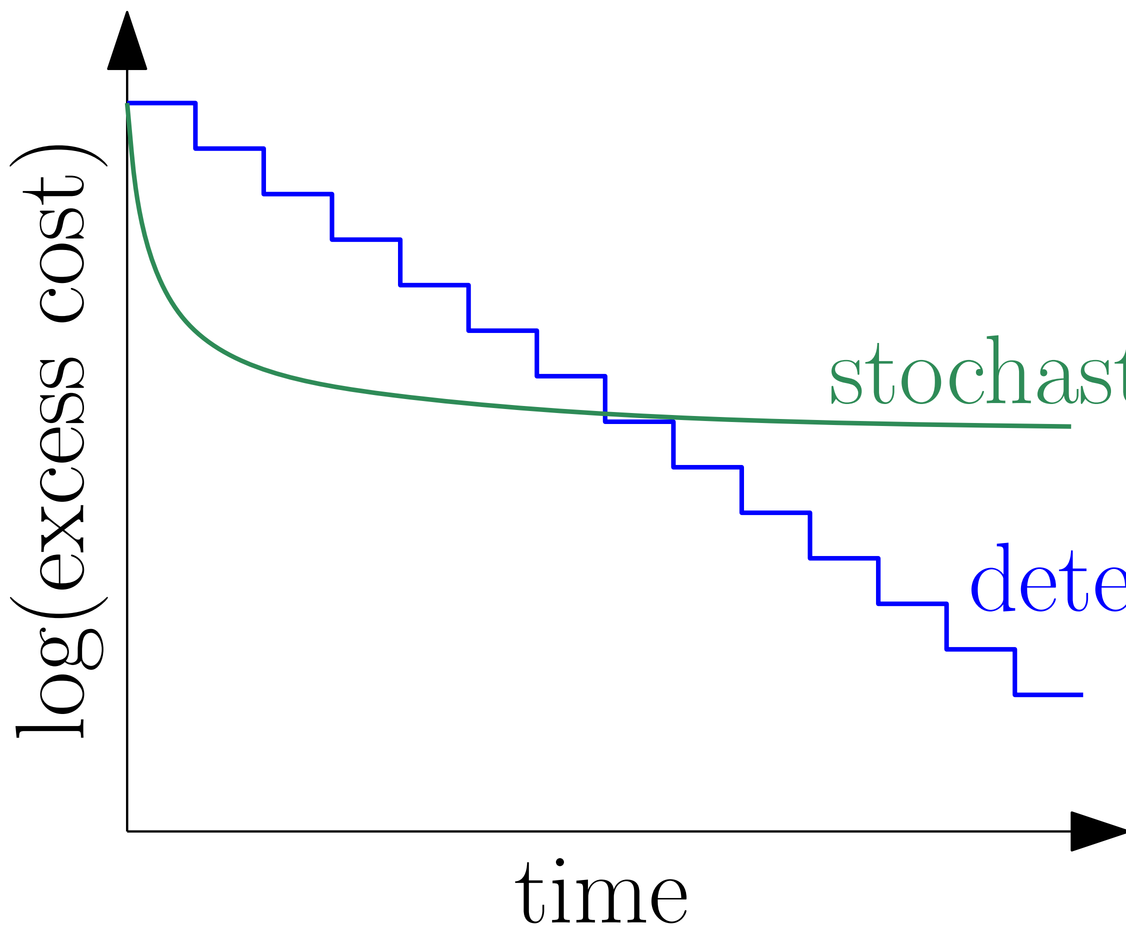


- Stochastic gradient descent: $\theta_t = \theta_{t-1} - \gamma_t f_{i(t)}'(\theta_{t-1})$



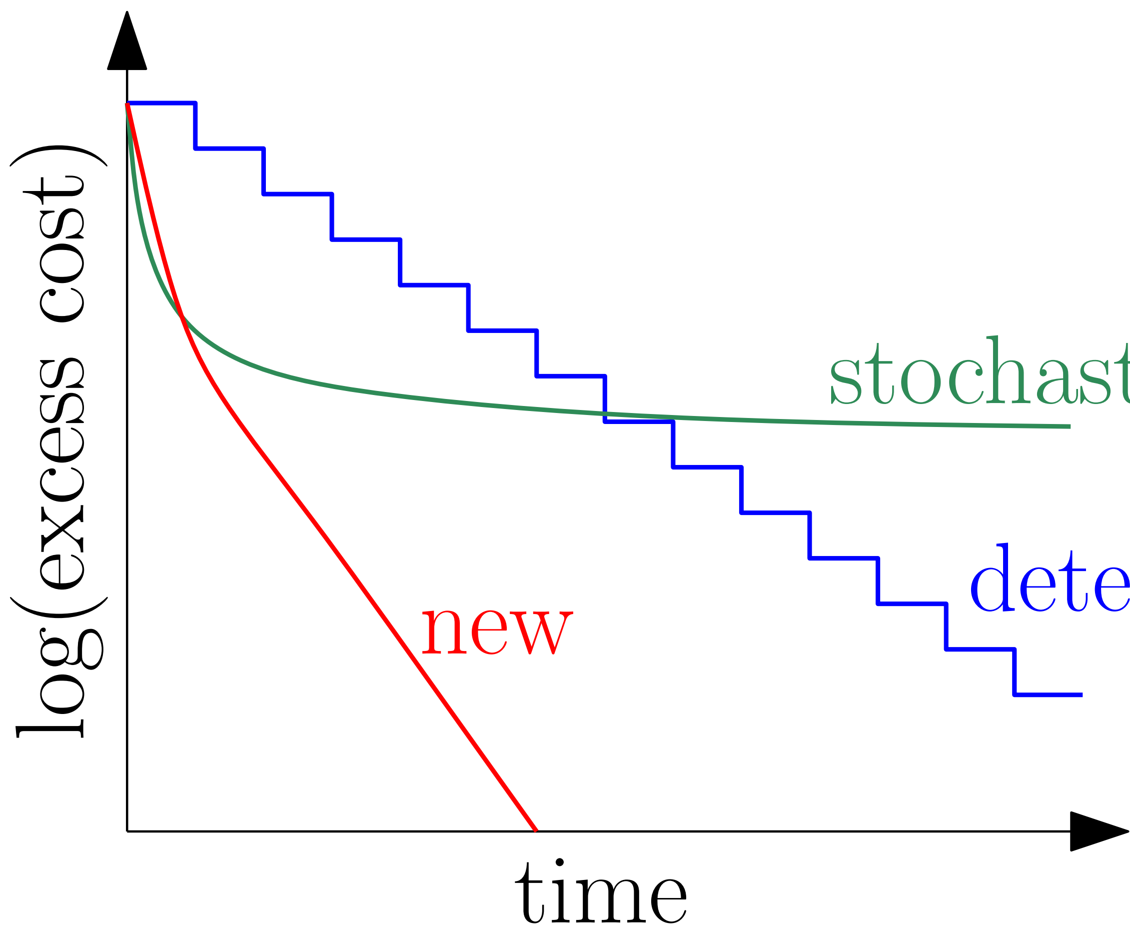
Stochastic vs. deterministic methods

- **Goal = best of both worlds**: Linear rate with $O(d)$ iteration cost
  Simple choice of step size



Stochastic vs. deterministic methods

- **Goal = best of both worlds**: Linear rate with $O(d)$ iteration cost
  Simple choice of step size



Accelerating gradient methods - Related work

- **Generic acceleration** (Nesterov, 1983, 2004)

$$\theta_t = \eta_{t-1} - \gamma_t g'(\eta_{t-1}) \text{ and } \eta_t = \theta_t + \delta_t(\theta_t - \theta_{t-1})$$

Accelerating gradient methods - Related work

- **Generic acceleration** (Nesterov, 1983, 2004)

$$\theta_t = \eta_{t-1} - \gamma_t g'(\eta_{t-1}) \text{ and } \eta_t = \theta_t + \delta_t(\theta_t - \theta_{t-1})$$

  - Good choice of momentum term $\delta_t \in [0, 1)$

    $g(\theta_t) - g(\theta_*) \leqslant O(1/t^2)$

    $g(\theta_t) - g(\theta_*) \leqslant O(e^{-t\sqrt{\mu/L}}) = O(e^{-t/\sqrt{\kappa}})$ if μ-strongly convex
  - Optimal rates after $t = O(d)$ iterations (Nesterov, 2004)

Accelerating gradient methods - Related work

- **Generic acceleration** (Nesterov, 1983, 2004)

$$\theta_t = \eta_{t-1} - \gamma_t g'(\eta_{t-1}) \text{ and } \eta_t = \theta_t + \delta_t(\theta_t - \theta_{t-1})$$

  - Good choice of momentum term $\delta_t \in [0, 1)$

    $g(\theta_t) - g(\theta_*) \leqslant O(1/t^2)$

    $g(\theta_t) - g(\theta_*) \leqslant O(e^{-t\sqrt{\mu/L}}) = O(e^{-t/\sqrt{\kappa}})$ if μ-strongly convex
  - Optimal rates after $t = O(d)$ iterations (Nesterov, 2004)
  - Still $O(nd)$ iteration cost: complexity $= O(nd \cdot \sqrt{\kappa} \log \frac{1}{\varepsilon})$

Accelerating gradient methods - Related work

- **Constant step-size stochastic gradient**
  - Solodov (1998); Nedic and Bertsekas (2000)
  - Linear convergence, but only up to a fixed tolerance

Accelerating gradient methods - Related work

- **Constant step-size stochastic gradient**
  - Solodov (1998); Nedic and Bertsekas (2000)
  - Linear convergence, but only up to a fixed tolerance

- **Stochastic methods in the dual (SDCA)**
  - Shalev-Shwartz and Zhang (2013)
  - Similar linear rate but limited choice for the f_i's
  - Extensions without duality: see Shalev-Shwartz (2016)

Accelerating gradient methods - Related work

- **Constant step-size stochastic gradient**
  - Solodov (1998); Nedic and Bertsekas (2000)
  - Linear convergence, but only up to a fixed tolerance
- **Stochastic methods in the dual (SDCA)**
  - Shalev-Shwartz and Zhang (2013)
  - Similar linear rate but limited choice for the f_i's
  - Extensions without duality: see Shalev-Shwartz (2016)
- **Stochastic version of accelerated batch gradient methods**
  - Tseng (1998); Ghadimi and Lan (2010); Xiao (2010)
  - Can improve constants, but still have sublinear $O(1/t)$ rate

Stochastic average gradient (Le Roux, Schmidt, and Bach, 2012)

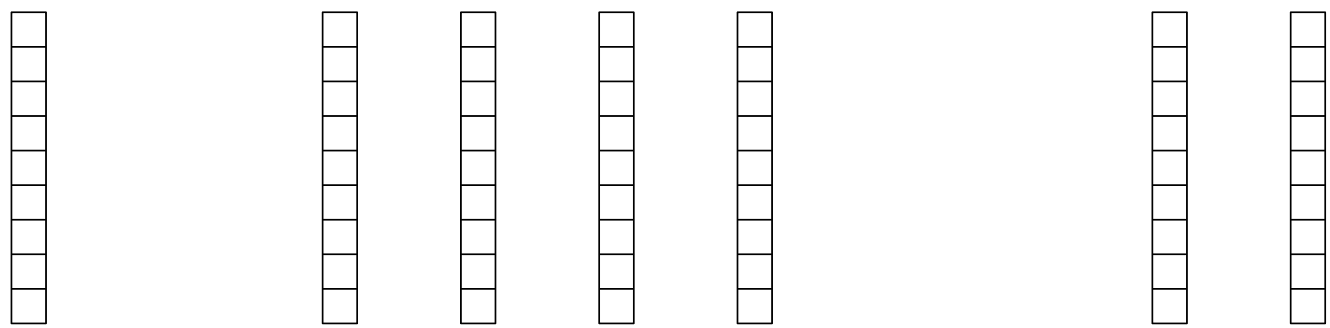
- **Stochastic average gradient** (SAG) iteration
  - Keep in memory the gradients of all functions f_i, $i = 1, \dots, n$
  - Random selection $i(t) \in \{1, \dots, n\}$ with replacement
  - Iteration: $\theta_t = \theta_{t-1} - \frac{\gamma_t}{n} \sum_{i=1}^{n} y_i^t$ with $y_i^t = \begin{cases} f_i'(\theta_{t-1}) & \text{if } i = i(t) \\ y_i^{t-1} & \text{otherwise} \end{cases}$

Stochastic average gradient
(Le Roux, Schmidt, and Bach, 2012)

- **Stochastic average gradient** (SAG) iteration
  - Keep in memory the gradients of all functions f_i, $i = 1, \dots, n$
  - Random selection $i(t) \in \{1, \dots, n\}$ with replacement
  - Iteration: $\theta_t = \theta_{t-1} - \frac{\gamma_t}{n} \sum_{i=1}^{n} y_i^t$ with $y_i^t = \begin{cases} f_i'(\theta_{t-1}) & \text{if } i = i(t) \\ y_i^{t-1} & \text{otherwise} \end{cases}$

functions $\quad g = \frac{1}{n}\sum_{i=1}^n f_i \quad f_1 \quad f_2 \quad f_3 \quad f_4 \quad \cdots \quad f_{n-1} \quad f_n$

gradients $\in \mathbb{R}^d \quad \frac{1}{n}\sum_{i=1}^n y_i^t \quad y_1^t \quad y_2^t \quad y_3^t \quad y_4^t \quad \cdots \quad y_{n-1}^t \quad y_n^t$

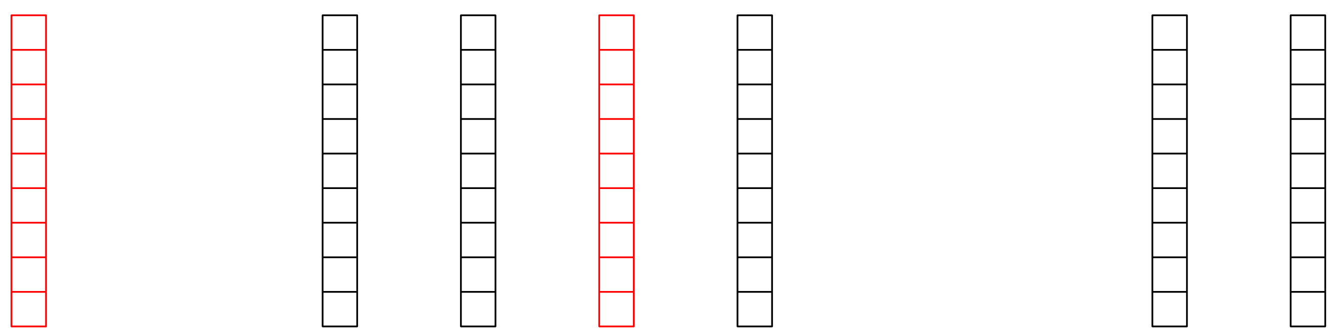


Stochastic average gradient
(Le Roux, Schmidt, and Bach, 2012)

- **Stochastic average gradient** (SAG) iteration
  - Keep in memory the gradients of all functions f_i, $i = 1, \dots, n$
  - Random selection $i(t) \in \{1, \dots, n\}$ with replacement
  - Iteration: $\theta_t = \theta_{t-1} - \frac{\gamma_t}{n} \sum_{i=1}^{n} y_i^t$ with $y_i^t = \begin{cases} f_i'(\theta_{t-1}) & \text{if } i = i(t) \\ y_i^{t-1} & \text{otherwise} \end{cases}$

functions $g = \frac{1}{n} \sum_{i=1}^{n} f_i$ f_1 f_2 f_3 f_4 $\bullet\bullet\bullet$ f_{n-1} f_n

gradients $\in \mathbb{R}^d$ $\frac{1}{n} \sum_{i=1}^{n} y_i^t$ y_1^t y_2^t y_3^t y_4^t $\bullet\bullet\bullet$ y_{n-1}^t y_n^t

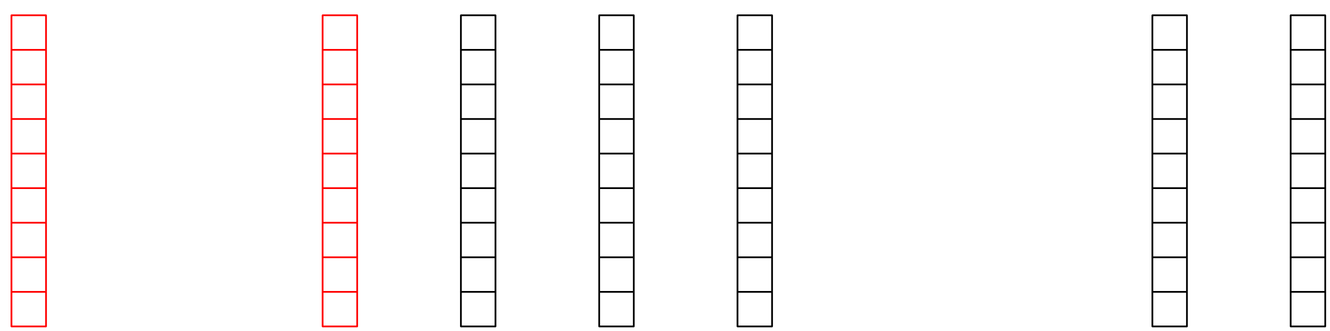


Stochastic average gradient (Le Roux, Schmidt, and Bach, 2012)

- **Stochastic average gradient** (SAG) iteration
  - Keep in memory the gradients of all functions f_i, $i = 1, \dots, n$
  - Random selection $i(t) \in \{1, \dots, n\}$ with replacement
  - Iteration: $\theta_t = \theta_{t-1} - \frac{\gamma_t}{n} \sum_{i=1}^{n} y_i^t$ with $y_i^t = \begin{cases} f_i'(\theta_{t-1}) & \text{if } i = i(t) \\ y_i^{t-1} & \text{otherwise} \end{cases}$

functions $\quad g = \frac{1}{n} \sum_{i=1}^n f_i \quad f_1 \quad f_2 \quad f_3 \quad f_4 \quad \cdots \quad f_{n-1} \quad f_n$

gradients $\in \mathbb{R}^d \quad \frac{1}{n} \sum_{i=1}^n y_i^t \quad y_1^t \quad y_2^t \quad y_3^t \quad y_4^t \quad \cdots \quad y_{n-1}^t \quad y_n^t$



Stochastic average gradient (Le Roux, Schmidt, and Bach, 2012)

- **Stochastic average gradient** (SAG) iteration
  - Keep in memory the gradients of all functions f_i, $i = 1, \dots, n$
  - Random selection $i(t) \in \{1, \dots, n\}$ with replacement
  - Iteration: $\theta_t = \theta_{t-1} - \frac{\gamma_t}{n} \sum_{i=1}^{n} y_i^t$ with $y_i^t = \begin{cases} f_i'(\theta_{t-1}) & \text{if } i = i(t) \\ y_i^{t-1} & \text{otherwise} \end{cases}$

- Stochastic version of incremental average gradient (Blatt et al., 2008)

Stochastic average gradient (Le Roux, Schmidt, and Bach, 2012)

- **Stochastic average gradient** (SAG) iteration
  - Keep in memory the gradients of all functions f_i, $i = 1, \dots, n$
  - Random selection $i(t) \in \{1, \dots, n\}$ with replacement
  - Iteration: $\theta_t = \theta_{t-1} - \frac{\gamma_t}{n} \sum_{i=1}^{n} y_i^t$ with $y_i^t = \begin{cases} f_i'(\theta_{t-1}) & \text{if } i = i(t) \\ y_i^{t-1} & \text{otherwise} \end{cases}$
- Stochastic version of incremental average gradient (Blatt et al., 2008)
- **Extra memory requirement**: n gradients in $\mathbb{R}^d$ in general
- **Linear supervised machine learning**: only n real numbers
  - If $f_i(\theta) = \ell(y_i, \Phi(x_i)^\top \theta)$, then $f_i'(\theta) = \ell'(y_i, \Phi(x_i)^\top \theta)\, \Phi(x_i)$

Running-time comparisons (strongly-convex)

- **Assumptions**: $g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$
  - Each f_i convex L-smooth and g μ-strongly convex

Stochastic gradient descent	$d\times$	$\frac{L}{\mu}$	$\times \quad \frac{1}{\varepsilon}$
Gradient descent	$d\times$	$n\frac{L}{\mu}$	$\times \log\frac{1}{\varepsilon}$
Accelerated gradient descent	$d\times$	$n\sqrt{\frac{L}{\mu}}$	$\times \log\frac{1}{\varepsilon}$
SAG	$d\times$	$(n + \frac{L}{\mu})$	$\times \log\frac{1}{\varepsilon}$

  - NB-1: for (accelerated) gradient descent, $L =$ smoothness constant of g
  - NB-2: with non-uniform sampling, $L =$ average smoothness constants of all f_i's

Running-time comparisons (strongly-convex)

- **Assumptions**: $g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$
  - Each f_i convex L-smooth and g μ-strongly convex

Stochastic gradient descent	$d\times$	$\frac{L}{\mu}$	$\times \frac{1}{\varepsilon}$
Gradient descent	$d\times$	$n\frac{L}{\mu}$	$\times \log \frac{1}{\varepsilon}$
Accelerated gradient descent	$d\times$	$n\sqrt{\frac{L}{\mu}}$	$\times \log \frac{1}{\varepsilon}$
SAG	$d\times$	$(n + \frac{L}{\mu})$	$\times \log \frac{1}{\varepsilon}$

- **Beating two lower bounds** (Nemirovski and Yudin, 1983; Nesterov, 2004): with additional assumptions

(1) stochastic gradient: exponential rate for finite sums

(2) full gradient: better exponential rate using the sum structure

Running-time comparisons (non-strongly-convex)

- **Assumptions**: $g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$
  - Each f_i convex L-smooth
  - Ill conditioned problems: g may not be strongly-convex ($\mu = 0$)

Stochastic gradient descent	$d\times$	$1/\varepsilon^2$
Gradient descent	$d\times$	n/ε
Accelerated gradient descent	$d\times$	$n/\sqrt{\varepsilon}$
SAG	$d\times$	$\sqrt{n}/\varepsilon$

- Adaptivity to potentially hidden strong convexity
- No need to know the local/global strong-convexity constant

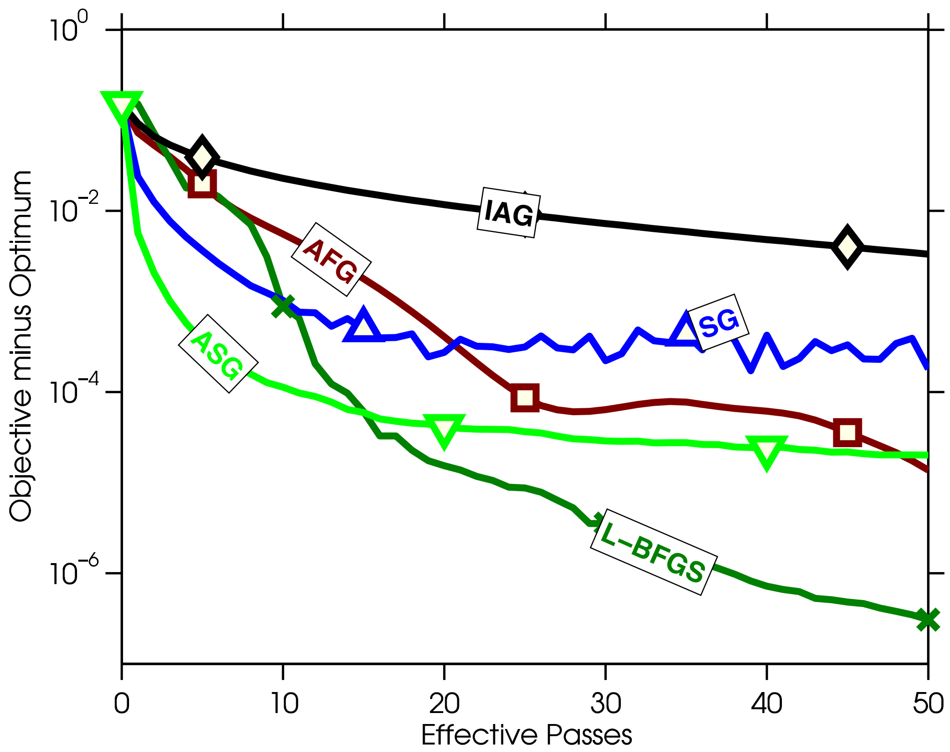
Stochastic average gradient
Implementation details and extensions

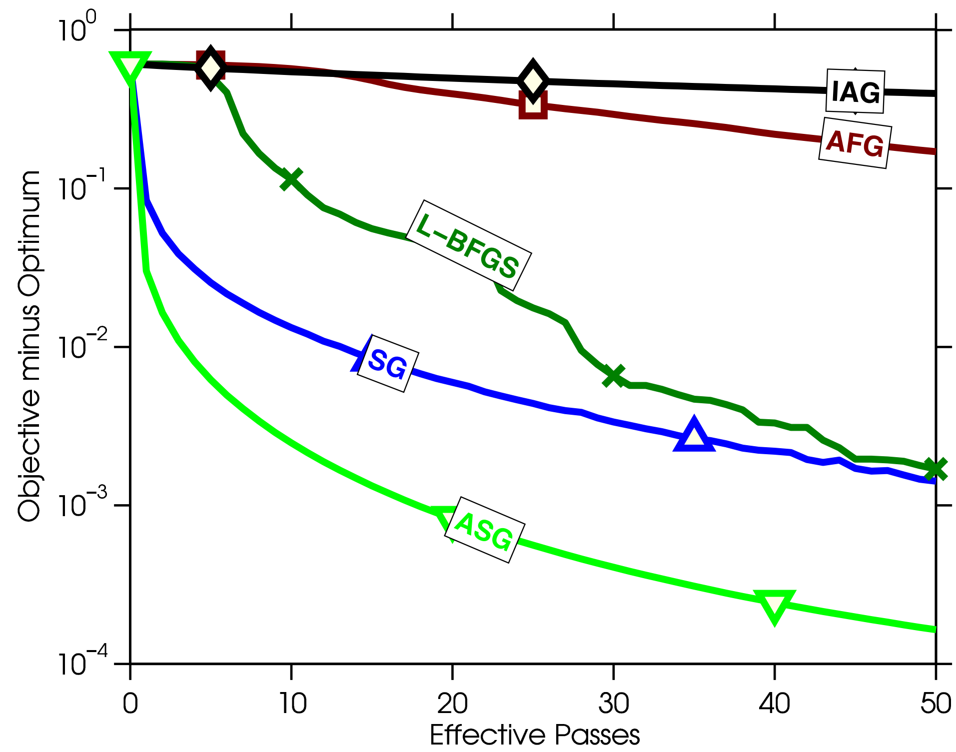
- **Sparsity in the features**
  - Just-in-time updates $\Rightarrow$ replace $O(d)$ by number of non zeros
  - See also Leblond, Pedregosa, and Lacoste-Julien (2016)
- **Mini-batches**
  - Reduces the memory requirement $+$ block access to data
- **Line-search**
  - Avoids knowing L in advance
- **Non-uniform sampling**
  - Favors functions with large variations
- See `www.cs.ubc.ca/~schmidtm/Software/SAG.html`

Experimental results (logistic regression)

quantum dataset
($n = 50\ 000$, $d = 78$)

rcv1 dataset
($n = 697\ 641$, $d = 47\ 236$)

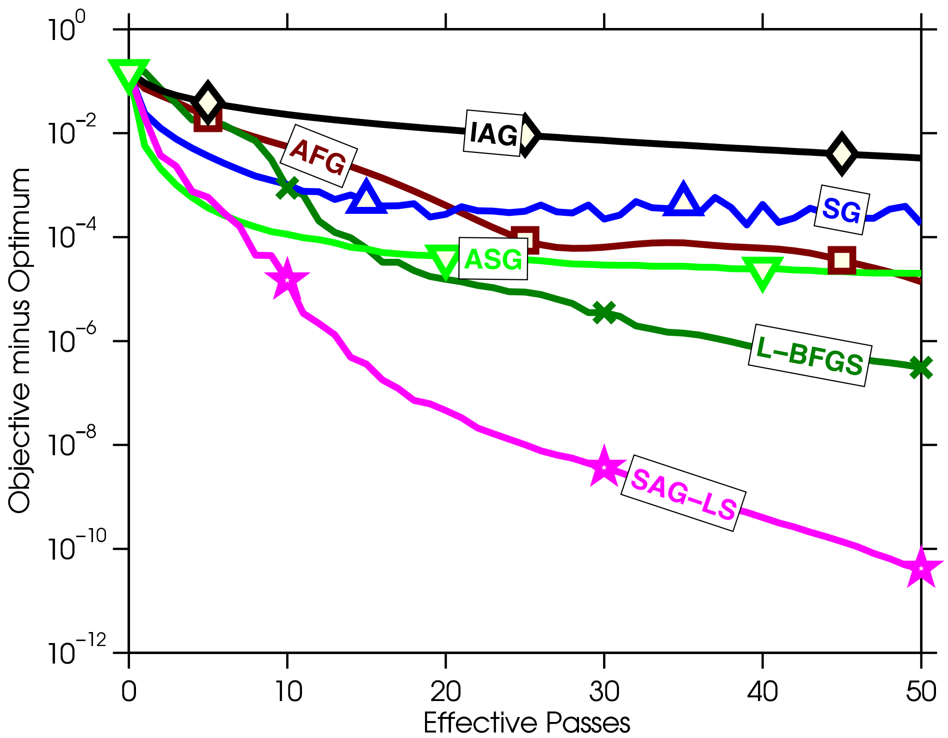


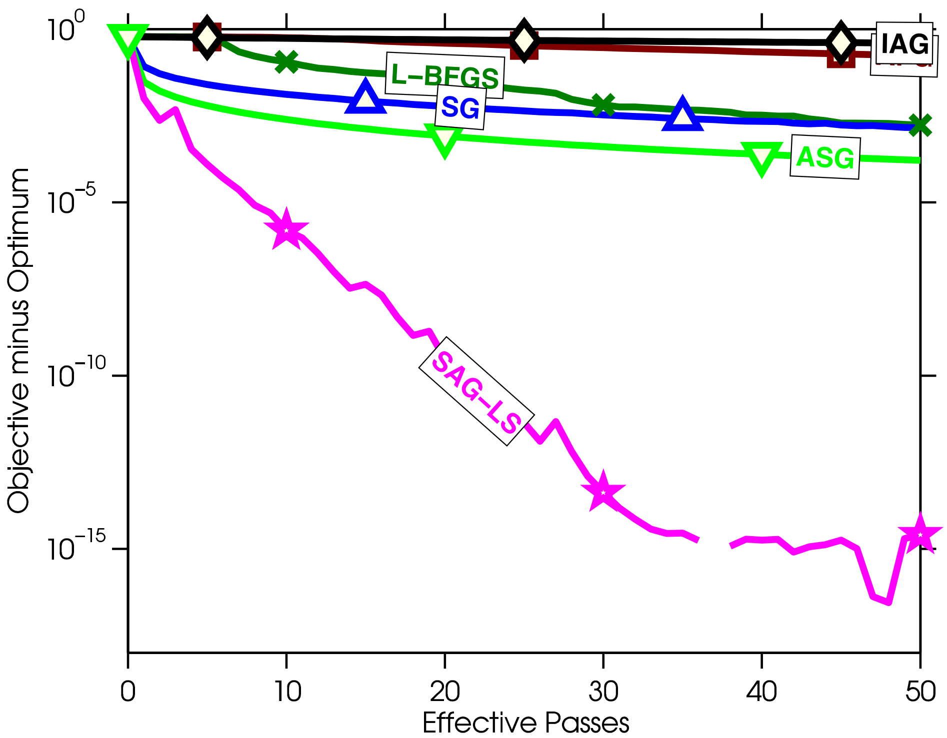


Experimental results (logistic regression)

`quantum` dataset
$(n = 50\ 000,\ d = 78)$

`rcv1` dataset
$(n = 697\ 641,\ d = 47\ 236)$





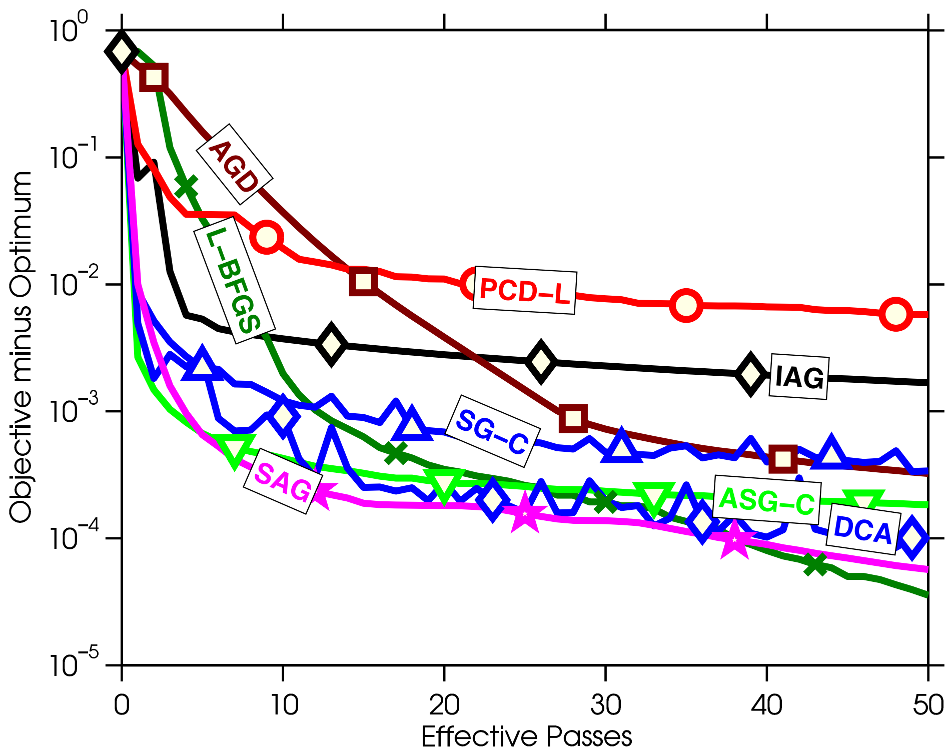
Before non-uniform sampling

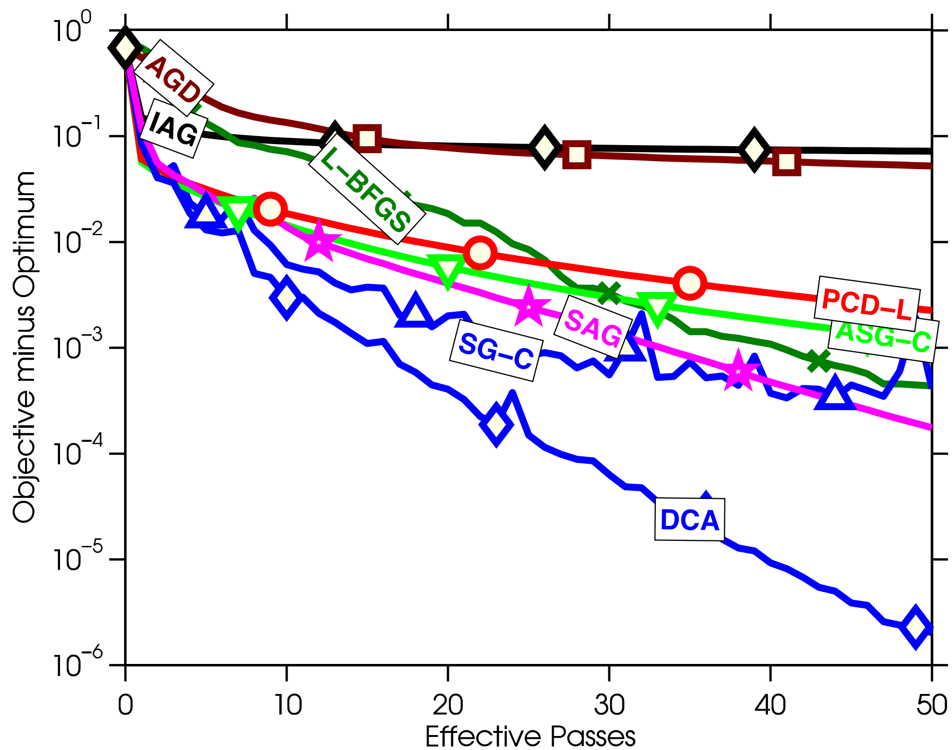
`protein` dataset

$(n = 145\ 751,\ d = 74)$

`sido` dataset

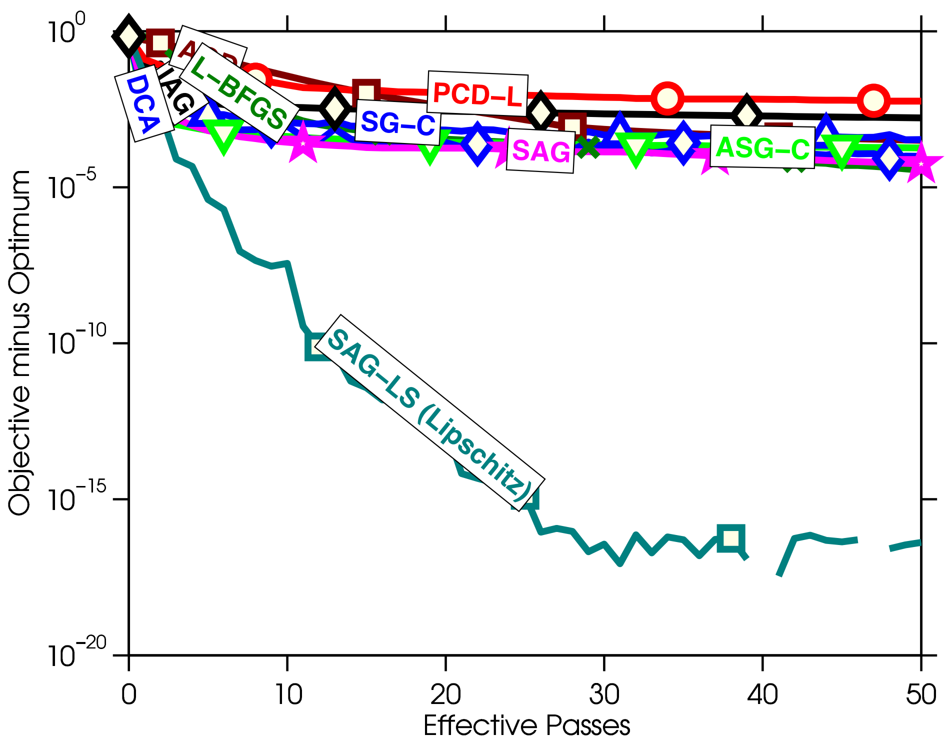
$(n = 12\ 678,\ d = 4\ 932)$



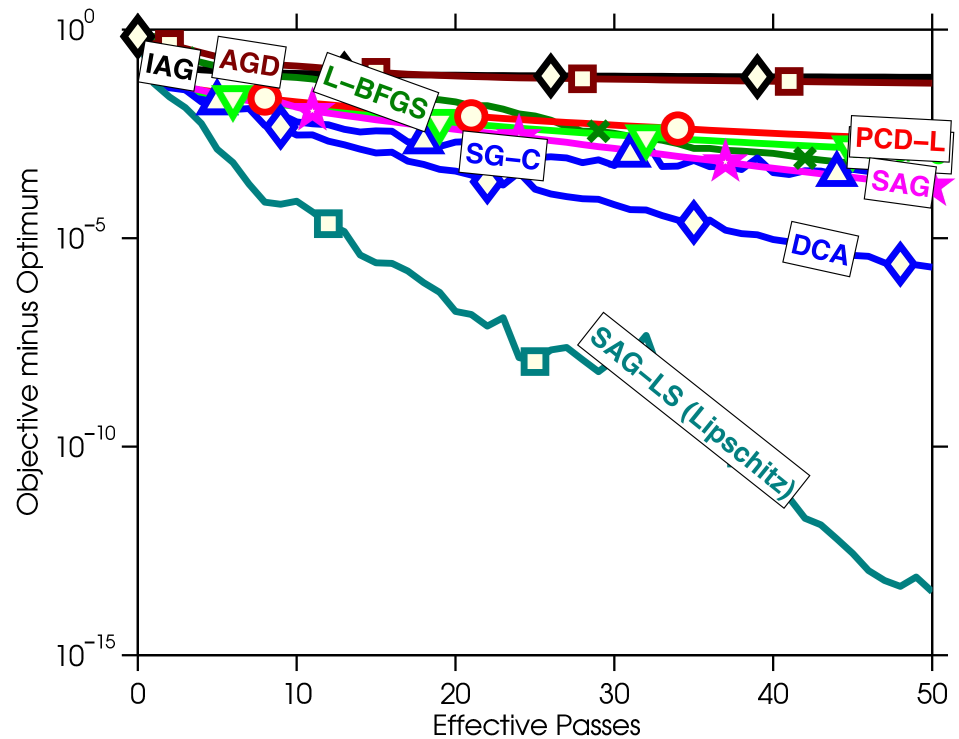


After non-uniform sampling

protein dataset
$(n = 145\ 751,\ d = 74)$



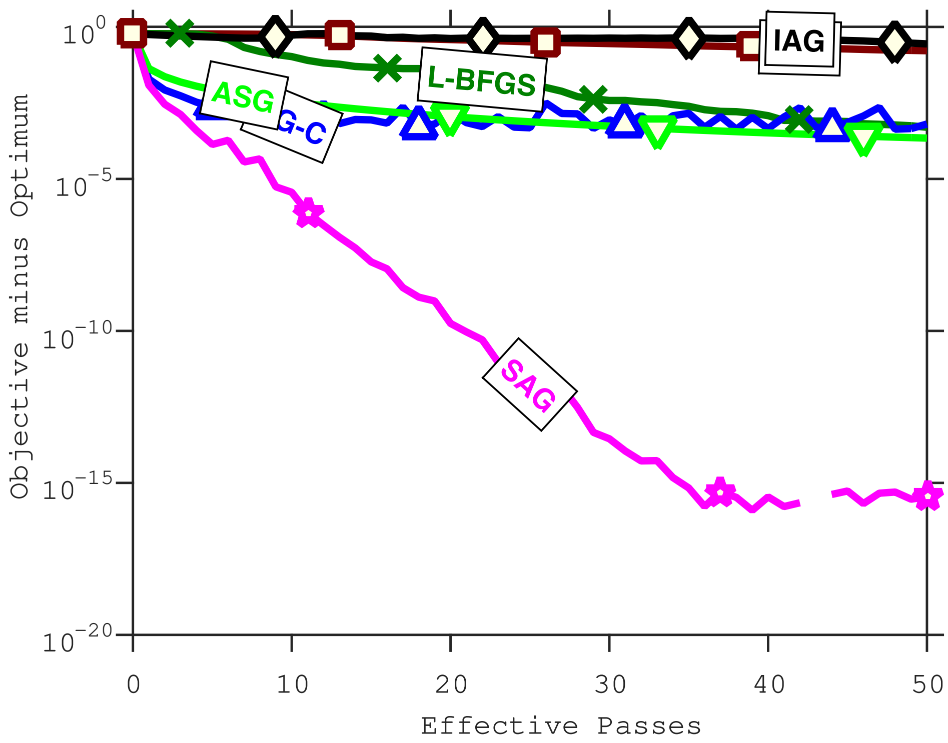
sido dataset
$(n = 12\ 678,\ d = 4\ 932)$



From training to testing errors

- `rcv1` dataset $(n = 697\ 641,\ d = 47\ 236)$
  - NB: IAG, SG-C, ASG with optimal step-sizes in hindsight

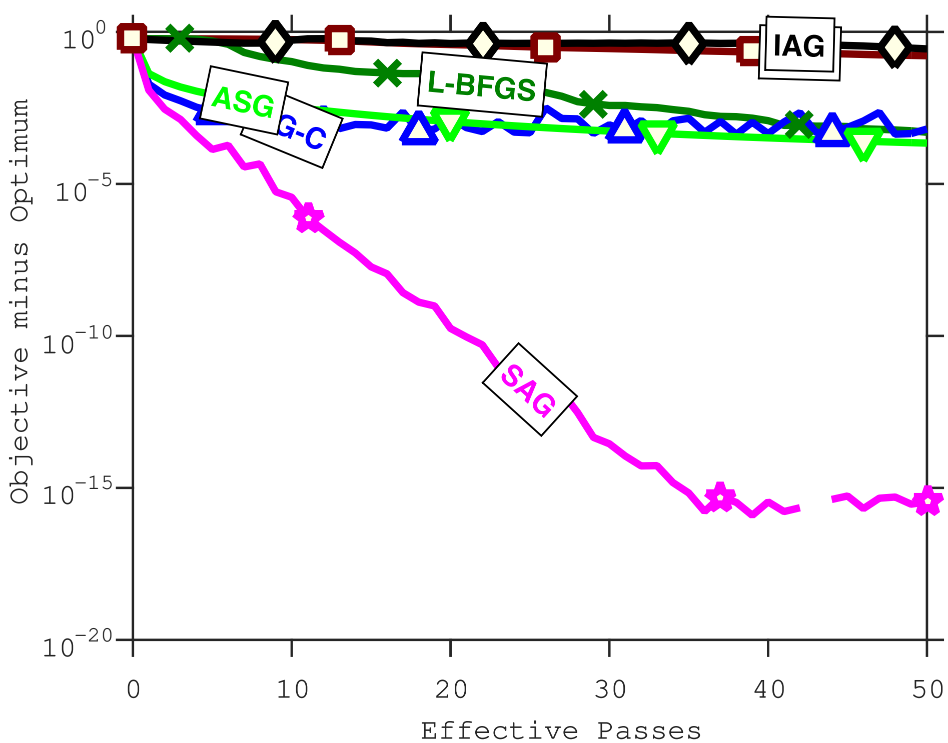
Training cost



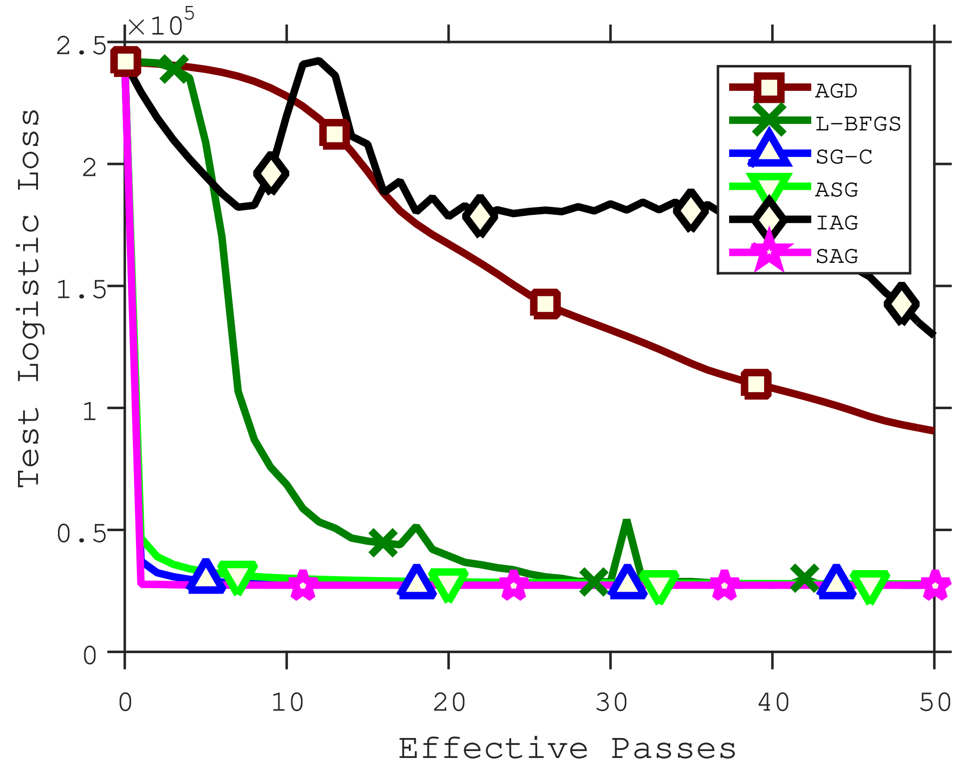
From training to testing errors

- `rcv1` dataset ($n = 697\ 641$, $d = 47\ 236$)
  - NB: IAG, SG-C, ASG with optimal step-sizes in hindsight

Training cost



Testing cost



Linearly convergent stochastic gradient algorithms

- **Many related algorithms**
  - SAG (Le Roux, Schmidt, and Bach, 2012)
  - SDCA (Shalev-Shwartz and Zhang, 2013)
  - SVRG (Johnson and Zhang, 2013; Zhang et al., 2013)
  - MISO (Mairal, 2015)
  - Finito (Defazio et al., 2014b)
  - SAGA (Defazio, Bach, and Lacoste-Julien, 2014a)
  - . . .

- **Similar rates of convergence and iterations**

Linearly convergent stochastic gradient algorithms

- **Many related algorithms**
  - SAG (Le Roux, Schmidt, and Bach, 2012)
  - SDCA (Shalev-Shwartz and Zhang, 2013)
  - SVRG (Johnson and Zhang, 2013; Zhang et al., 2013)
  - MISO (Mairal, 2015)
  - Finito (Defazio et al., 2014b)
  - SAGA (Defazio, Bach, and Lacoste-Julien, 2014a)
  - ...

- **Similar rates of convergence and iterations**

- **Different interpretations and proofs / proof lengths**
  - Lazy gradient evaluations
  - Variance reduction

Acceleration

- **Similar guarantees for finite sums**: SAG, SDCA, SVRG (Xiao and Zhang, 2014), SAGA, MISO (Mairal, 2015)

Gradient descent	$d\times \quad n\frac{L}{\mu} \quad \times \log\frac{1}{\varepsilon}$
Accelerated gradient descent	$d\times \quad n\sqrt{\frac{L}{\mu}} \quad \times \log\frac{1}{\varepsilon}$
SAG(A), SVRG, SDCA, MISO	$d\times \quad (n+\frac{L}{\mu}) \quad \times \log\frac{1}{\varepsilon}$
Accelerated versions	$d\times \quad (n+\sqrt{n\frac{L}{\mu}}) \quad \times \log\frac{1}{\varepsilon}$

- **Acceleration for special algorithms** (e.g., Shalev-Shwartz and Zhang, 2014; Nitanda, 2014; Lan, 2015; Defazio, 2016)

- **Catalyst** (Lin, Mairal, and Harchaoui, 2015)
  - Widely applicable generic acceleration scheme

SGD minimizes the testing cost!

- **Goal**: minimize $f(\theta) = \mathbb{E}_{p(x,y)} \ell(y, h(x, \theta))$
  - Given n independent samples (x_i, y_i), $i = 1, \dots, n$ from $p(x, y)$
  - Given a single pass of stochastic gradient descent
  - Bounds on the excess testing cost $\mathbb{E} f(\bar{\theta}_n) - \inf_{\theta \in \mathbb{R}^d} f(\theta)$

SGD minimizes the testing cost!

- **Goal**: minimize $f(\theta) = \mathbb{E}_{p(x,y)} \ell(y, h(x,\theta))$
  - Given n independent samples (x_i, y_i), $i = 1, \dots, n$ from $p(x,y)$
  - Given a single pass of stochastic gradient descent
  - Bounds on the excess testing cost $\mathbb{E} f(\bar{\theta}_n) - \inf_{\theta \in \mathbb{R}^d} f(\theta)$

- **Optimal convergence rates**: $O(1/\sqrt{n})$ and $O(1/(n\mu))$
  - Optimal for non-smooth losses (Nemirovski and Yudin, 1983)
  - Attained by averaged SGD with decaying step-sizes

SGD minimizes the testing cost!

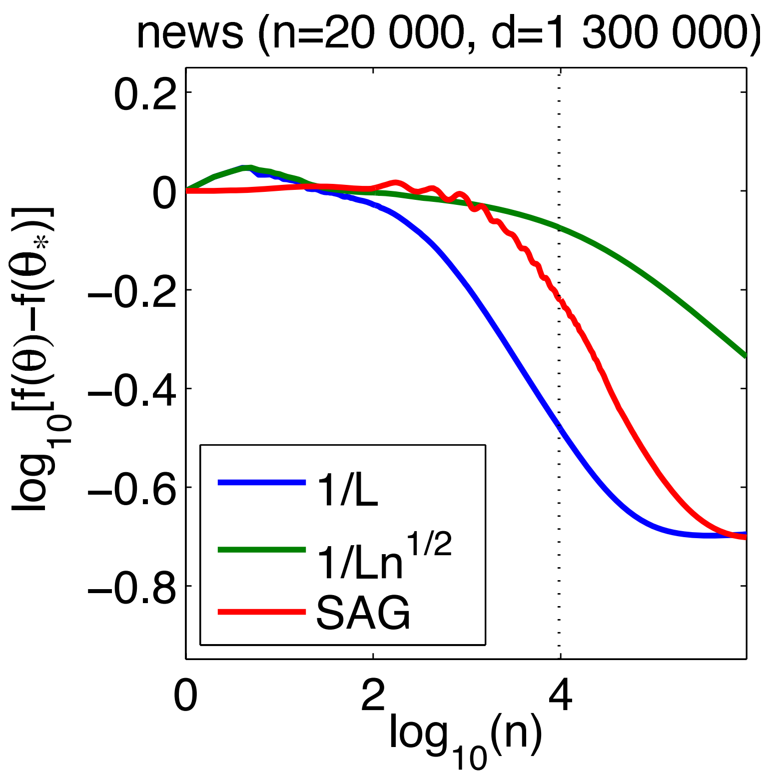
- **Goal**: minimize $f(\theta) = \mathbb{E}_{p(x,y)} \ell(y, h(x,\theta))$
  - Given n independent samples (x_i, y_i), $i = 1, \dots, n$ from $p(x,y)$
  - Given a single pass of stochastic gradient descent
  - Bounds on the excess testing cost $\mathbb{E} f(\bar{\theta}_n) - \inf_{\theta \in \mathbb{R}^d} f(\theta)$

- **Optimal convergence rates**: $O(1/\sqrt{n})$ and $O(1/(n\mu))$
  - Optimal for non-smooth losses (Nemirovski and Yudin, 1983)
  - Attained by averaged SGD with decaying step-sizes

- **Constant-step-size SGD**
  - Linear convergence up to the noise level for strongly-convex problems (Solodov, 1998; Nedic and Bertsekas, 2000)
  - Full convergence and robustness to ill-conditioning?

Robust averaged stochastic gradient (Bach and Moulines, 2013)

- **Constant-step-size SGD is convergent for least-squares**
  - Convergence rate in $O(1/n)$ without any dependence on μ
  - Simple choice of step-size (equal to $1/L$) *(see board)*



Robust averaged stochastic gradient (Bach and Moulines, 2013)

- **Constant-step-size SGD is convergent for least-squares**
  - Convergence rate in $O(1/n)$ without any dependence on μ
  - Simple choice of step-size (equal to $1/L$)

- **Constant-step-size SGD can be made convergent** *(see board)*
  - Online Newton correction with same complexity as SGD
  - Replace $\theta_n = \theta_{n-1} - \gamma f_n'(\theta_{n-1})$
    by $\quad \theta_n = \theta_{n-1} - \gamma \big[f_n'(\bar\theta_{n-1}) + f''(\bar\theta_{n-1})(\theta_{n-1} - \bar\theta_{n-1}) \big]$
  - Simple choice of step-size and convergence rate in $O(1/n)$

Robust averaged stochastic gradient (Bach and Moulines, 2013)

- **Constant-step-size SGD is convergent for least-squares**
  - Convergence rate in $O(1/n)$ without any dependence on μ
  - Simple choice of step-size (equal to $1/L$)

- **Constant-step-size SGD can be made convergent**
  - Online Newton correction with same complexity as SGD
  - Replace $\theta_n = \theta_{n-1} - \gamma f'_n(\theta_{n-1})$
    by $\theta_n = \theta_{n-1} - \gamma \big[f'_n(\bar\theta_{n-1}) + f''(\bar\theta_{n-1})(\theta_{n-1} - \bar\theta_{n-1}) \big]$
  - Simple choice of step-size and convergence rate in $O(1/n)$

- **Multiple passes still work better in practice**

Perspectives

- **Linearly-convergent stochastic gradient methods**
  - Provable and precise rates
  - Improves on two known lower-bounds (by using structure)
  - Several extensions / interpretations / accelerations

Perspectives

- **Linearly-convergent stochastic gradient methods**
  - Provable and precise rates
  - Improves on two known lower-bounds (by using structure)
  - Several extensions / interpretations / accelerations

- **Extensions and future work**
  - Lower bounds for finite sums (Lan, 2015)
  - Sampling without replacement (Gurbuzbalaban et al., 2015)

Perspectives

- **Linearly-convergent stochastic gradient methods**
  - Provable and precise rates
  - Improves on two known lower-bounds (by using structure)
  - Several extensions / interpretations / accelerations

- **Extensions and future work**
  - Lower bounds for finite sums (Lan, 2015)
  - Sampling without replacement (Gurbuzbalaban et al., 2015)
  - Bounds on testing errors for incremental methods

Perspectives

- **Linearly-convergent stochastic gradient methods**
  - Provable and precise rates
  - Improves on two known lower-bounds (by using structure)
  - Several extensions / interpretations / accelerations
- **Extensions and future work**
  - Lower bounds for finite sums (Lan, 2015)
  - Sampling without replacement (Gurbuzbalaban et al., 2015)
  - Bounds on testing errors for incremental methods
  - Parallelization (Leblond et al., 2016)
  - Non-convex problems (Reddi et al., 2016)
  - Other forms of acceleration (Scieur, d'Aspremont, and Bach, 2016)

Outline

1. **Introduction/motivation: Supervised machine learning**
   - Optimization of finite sums
   - Batch gradient descent
   - Stochastic gradient descent

2. **Stochastic average gradient (SAG)**
   - Linearly-convergent stochastic gradient method
   - Precise convergence rates
   - From training cost to testing cost

3. **Conditional Gradient (a.k.a. Frank-Wolfe algorithm)**
   - Optimization over convex hulls
   - Application to one-hidden layer neural networks

Dealing with constraints

- **Regularization**: $\mathcal{C} = \left\{\theta \in \mathbb{R}^d, \Omega(\theta) \leqslant \omega\right\}$
  - Squared ℓ_2-norm: $\Omega(\theta) = \sum_{j=1}^d |\theta_j|^2$
  - ℓ_1-norm: $\Omega(\theta) = \sum_{j=1}^d |\theta_j|$
  - Matrix norm: ℓ_1-norm of singular values *(see board)*

Dealing with constraints

- **Regularization**: $\mathcal{C} = \left\{\theta \in \mathbb{R}^d, \Omega(\theta) \leqslant \omega\right\}$
  - Squared ℓ_2-norm: $\Omega(\theta) = \sum_{j=1}^d |\theta_j|^2$
  - ℓ_1-norm: $\Omega(\theta) = \sum_{j=1}^d |\theta_j|$
  - Matrix norm: ℓ_1-norm of singular values *(see board)*

- **Projected gradient descent** for $\min\limits_{\theta \in \mathcal{C}} g(\theta)$ *(see board)*

$$\theta_t = \arg\min_{\theta \in \mathcal{C}} \left\| \theta - \big(\theta_{t-1} - \gamma g'(\theta_{t-1})\big) \right\|^2$$

  - Requires costly "quadratic oracle" $\arg\min_{\theta \in \mathcal{C}} \left\|\theta - z\right\|^2$
  - Preserved convergence rates

Dealing with constraints

- **Regularization**: $\mathcal{C} = \left\{\theta \in \mathbb{R}^d, \Omega(\theta) \leqslant \omega\right\}$
  - Squared ℓ_2-norm: $\Omega(\theta) = \sum_{j=1}^d |\theta_j|^2$
  - ℓ_1-norm: $\Omega(\theta) = \sum_{j=1}^d |\theta_j|$
  - Matrix norm: ℓ_1-norm of singular values *(see board)*

- **Projected gradient descent** for $\min_{\theta \in \mathcal{C}} g(\theta)$ *(see board)*

$$\theta_t = \arg\min_{\theta \in \mathcal{C}} \left\|\theta - \big(\theta_{t-1} - \gamma g'(\theta_{t-1})\big)\right\|^2$$

  - Requires costly "quadratic oracle" $\arg\min_{\theta \in \mathcal{C}} \left\|\theta - z\right\|^2$
  - Preserved convergence rates

- **"Linear oracle" often easier** $\arg\min_{\theta \in \mathcal{C}} z^\top \theta$

Conditional Gradient (a.k.a. Frank-Wolfe algorithm)

- **Algorithm for** $\min_{\theta \in \mathcal{C}} g(\theta)$ *(see board)*

1. Linearization: $g(\theta) \geqslant g(\theta_{t-1}) + g'(\theta_{t-1})^\top (\theta - \theta_{t-1})$

2. "FW step": $\bar{\theta}_{t-1} \in \arg\min_{\theta \in \mathcal{C}} g'(\theta_{t-1})^\top (\theta - \theta_{t-1})$

3. Line search: $\theta_t = (1 - \rho_t)\theta_{t-1} + \rho_t \bar{\theta}_{t-1}$

Conditional Gradient (a.k.a. Frank-Wolfe algorithm)

- **Algorithm for** $\min_{\theta \in \mathcal{C}} g(\theta)$ *(see board)*

1. Linearization: $g(\theta) \geqslant g(\theta_{t-1}) + g'(\theta_{t-1})^\top (\theta - \theta_{t-1})$

2. "FW step": $\bar{\theta}_{t-1} \in \arg\min_{\theta \in \mathcal{C}} g'(\theta_{t-1})^\top (\theta - \theta_{t-1})$

3. Line search: $\theta_t = (1 - \rho_t)\theta_{t-1} + \rho_t \bar{\theta}_{t-1}$

- **"Greedy" optimization**
  - Convergence rate: $g(\theta_t) - f(\theta_*) \leqslant \dfrac{2L \mathrm{diam}(\mathcal{C})^2}{t}$
  - Sparse iterates and ℓ_1-norm example *(see board)*
  - see, e.g., Jaggi (2013) and references therein

One-hidden layer neural networks

- **Replace the sum $\sum_{i=1}^k \eta_i (w_i^\top x)_+$ by an integral**

$$f(x) = \int_{\mathbb{R}^d} (w^\top x)_+ \, d\mu(w)$$

  - $d\mu$ any signed measure with finite mass (e.g., $d\mu(w) = \eta dw$)
  - Equivalence when $d\mu$ is a weighted sum of Diracs: $\sum_{i=1}^k \eta_i \delta_{w_i}$

One-hidden layer neural networks

- **Replace the sum $\sum_{i=1}^{k} \eta_i (w_i^\top x)_+$ by an integral**

$$f(x) = \int_{\mathbb{R}^d} (w^\top x)_+ \, d\mu(w)$$

  - $d\mu$ any signed measure with finite mass (e.g., $d\mu(w) = \eta dw$)
  - Equivalence when $d\mu$ is a weighted sum of Diracs: $\sum_{i=1}^{k} \eta_i \delta_{w_i}$

- **Promote sparsity with total variation of μ:** $\displaystyle\int_{\mathbb{R}^d} |\eta(w)| dw$
  - Several points of views (Barron, 1993; Kurkova and Sanguineti, 2001; Bengio, Le Roux, Vincent, Delalleau, and Marcotte, 2006; Rosset, Swirszcz, Srebro, and Zhu, 2007)

- ℓ_1**-norm in infinite dimension** $\Rightarrow$ **convex problem**

Conditional gradient for neural networks

$$\min_{\eta} \mathbb{E}_{(x,y)} \ell\Big(y, \int_{\mathbb{R}^d} (w^\top x)_+ \, \eta(w) dw\Big) \text{ such that } \int_{\mathbb{R}^d} |\eta(w)| dw \leqslant C$$

Conditional gradient for neural networks

$$\min_\eta \mathbb{E}_{(x,y)} \ell\Big(y, \int_{\mathbb{R}^d} (w^\top x)_+ \ \eta(w) dw\Big) \text{ such that } \int_{\mathbb{R}^d} |\eta(w)| dw \leqslant C$$

- **"Frank-Wolfe" step** with $g(x, y)$ gradient of the loss for (x, y) at η

$$\min_\eta \mathbb{E}_{(x,y)} g(x, y) \int_{\mathbb{R}^d} (w^\top x)_+ \ \eta(w) dw \text{ such that } \int_{\mathbb{R}^d} |\eta(w)| dw \leqslant C$$

$$\min_\eta \int_{\mathbb{R}^d} \Big(\mathbb{E}_{(x,y)} g(x, y)(w^\top x)_+\Big) \eta(w) dw \text{ such that } \int_{\mathbb{R}^d} |\eta(w)| dw \leqslant C$$

$$\Leftrightarrow \min_\eta \int_{\mathbb{R}^d} h(w) \eta(w) dw \text{ such that } \int_{\mathbb{R}^d} |\eta(w)| dw \leqslant C$$

Conditional gradient for neural networks

$$\min_{\eta} \mathbb{E}_{(x,y)} \ell\Big(y, \int_{\mathbb{R}^d} (w^\top x)_+ \, \eta(w) dw\Big) \text{ such that } \int_{\mathbb{R}^d} |\eta(w)| dw \leqslant C$$

- **"Frank-Wolfe" step** with $g(x,y)$ gradient of the loss for (x,y) at η

$$\min_{\eta} \mathbb{E}_{(x,y)} g(x,y) \int_{\mathbb{R}^d} (w^\top x)_+ \, \eta(w) dw \text{ such that } \int_{\mathbb{R}^d} |\eta(w)| dw \leqslant C$$

$$\min_{\eta} \int_{\mathbb{R}^d} \Big(\mathbb{E}_{(x,y)} g(x,y)(w^\top x)_+\Big) \eta(w) dw \text{ such that } \int_{\mathbb{R}^d} |\eta(w)| dw \leqslant C$$

$$\Leftrightarrow \min_{\eta} \int_{\mathbb{R}^d} h(w) \eta(w) dw \text{ such that } \int_{\mathbb{R}^d} |\eta(w)| dw \leqslant C$$

- **Best additional neuron**: maximizing $|h(w)|$ with respect to w
  - **Incremental learning of neural networks**

Conditional gradient for neural networks

- **Still not polynomial time**
  - Incremental step still NP-hard (Bach, 2014)
  - Classical binary classification problem (Bengio et al., 2006)

- **Precise analysis of number of neurons and sample complexity**
  - Exponential in dimension $O(\varepsilon^{-d})$ in general to reach precision ε
  - Adaptive to linear structures

Conditional gradient for neural networks

- **Still not polynomial time**
  - Incremental step still NP-hard (Bach, 2014)
  - Classical binary classification problem (Bengio et al., 2006)

- **Precise analysis of number of neurons and sample complexity**
  - Exponential in dimension $O(\varepsilon^{-d})$ in general to reach precision ε
  - Adaptive to linear structures

Linear function	$w^\top x + b$	$(\sqrt{d}/\varepsilon)^2$
Generalized additive model	$\sum_{j=1}^d f_j(x_j)$	$(\sqrt{d}/\varepsilon)^4$
One-hidden layer neural network	$\sum_{i=1}^k \eta_i \sigma(w_i^\top x + b)$	$k^2(\sqrt{d}/\varepsilon)^2$
Projection pursuit	$\sum_{i=1}^k f_i(w_i^\top x)$	$k^4(\sqrt{d}/\varepsilon)^4$
Subspace dependence	$g(W^\top x)$	$(\sqrt{d}/\varepsilon)^{\mathrm{rank(W)}+3}$

Conclusions
Optimization for machine learning

- **Well understood**
  - Convex case with a single machine
  - Matching lower and upper bounds for variants of SGD
  - Non-convex case: SGD for local risk minimization

Conclusions
Optimization for machine learning

- **Well understood**
  - Convex case with a single machine
  - Matching lower and upper bounds for variants of SGD
  - Non-convex case: SGD for local risk minimization

- **Not well understood**: many open problems
  - Step-size schedules and acceleration
  - Dealing with non-convexity (local minima and stationary points)
  - Distributed learning (multiple cores, GPUs, and cloud)

References

F. Bach and E. Moulines. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Adv. NIPS*, 2011.

F. Bach and E. Moulines. Non-strongly-convex smooth stochastic approximation with convergence rate $O(1/n)$. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends in Machine Learning*, 4(1):1–106, 2012a.

F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Structured sparsity through convex optimization, 2012b.

Francis Bach. Breaking the curse of dimensionality with convex neural networks. Technical Report 1412.8690, arXiv, 2014.

A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.

Y. Bengio, N. Le Roux, P. Vincent, O. Delalleau, and P. Marcotte. Convex neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2006.

D. Blatt, A. O. Hero, and H. Gauchman. A convergent incremental gradient method with a constant step size. *SIAM Journal on Optimization*, 18(1):29–51, 2008.

L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *Adv. NIPS*, 2008.

A. Defazio. A simple practical accelerated method for finite sums. In *Advances In Neural Information Processing Systems (NIPS)*, 2016.

A. Defazio, F. Bach, and S. Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems (NIPS)*, 2014a.

A. Defazio, J. Domke, and T. S. Caetano. Finito: A faster, permutable incremental gradient method for big data problems. In *Proc. ICML*, 2014b.

S. Ghadimi and G. Lan. Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization. *Optimization Online*, July, 2010.

M. Gurbuzbalaban, A. Ozdaglar, and P. Parrilo. On the convergence rate of incremental aggregated gradient algorithms. Technical Report 1506.02081, arXiv, 2015.

Martin Jaggi. Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 427–435, 2013.

R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, 2013.

V. Kurkova and M. Sanguineti. Bounds on rates of variable-basis and neural-network approximation. *IEEE Transactions on Information Theory*, 47(6):2659–2665, Sep 2001.

Simon Lacoste-Julien, Martin Jaggi, Mark Schmidt, and Patrick Pletscher. Block-coordinate {Frank-Wolfe} optimization for structural {SVMs}. In *Proceedings of The 30th International Conference on Machine Learning*, pages 53–61, 2013.

G. Lan. An optimal randomized incremental gradient method. Technical Report 1507.02000, arXiv, 2015.

N. Le Roux, M. Schmidt, and F. Bach. A stochastic gradient method with an exponential convergence rate for strongly-convex optimization with finite training sets. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

R. Leblond, F. Pedregosa, and S. Lacoste-Julien. Asaga: Asynchronous parallel Saga. Technical Report 1606.04809, arXiv, 2016.

H. Lin, J. Mairal, and Z. Harchaoui. A universal catalyst for first-order optimization. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

J. Mairal. Incremental majorization-minimization optimization with application to large-scale machine learning. *SIAM Journal on Optimization*, 25(2):829–855, 2015.

A. Nedic and D. Bertsekas. Convergence rate of incremental subgradient algorithms. *Stochastic Optimization: Algorithms and Applications*, pages 263–304, 2000.

A. S. Nemirovski and D. B. Yudin. *Problem complexity and method efficiency in optimization.* Wiley & Sons, 1983.

Y. Nesterov. A method for solving a convex programming problem with rate of convergence $O(1/k^2)$. *Soviet Math. Doklady*, 269(3):543–547, 1983.

Y. Nesterov. *Introductory lectures on convex optimization: a basic course*. Kluwer, 2004.

A. Nitanda. Stochastic proximal gradient descent with acceleration techniques. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

S. J. Reddi, A. Hefny, S. Sra, B. Póczós, and A. Smola. Stochastic variance reduction for nonconvex optimization. Technical Report 1603.06160, arXiv, 2016.

H. Robbins and S. Monro. A stochastic approximation method. *Ann. Math. Statistics*, 22:400–407, 1951.

S. Rosset, G. Swirszcz, N. Srebro, and J. Zhu. ℓ_1-regularization in infinite dimensional feature spaces. In *Proceedings of the Conference on Learning Theory (COLT)*, 2007.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2001.

D. Scieur, A. d'Aspremont, and F. Bach. Regularized nonlinear acceleration. In *Advances in Neural Information Processing Systems*, 2016.

S. Shalev-Shwartz. Sdca without duality, regularization, and individual convexity. Technical Report 1602.01582, arXiv, 2016.

S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(Feb):567–599, 2013.

S. Shalev-Shwartz and T. Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. In *Proc. ICML*, 2014.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

M. V. Solodov. Incremental gradient algorithms with stepsizes bounded away from zero. *Computational Optimization and Applications*, 11(1):23–35, 1998.

P. Tseng. An incremental gradient(-projection) method with momentum term and adaptive stepsize rule. *SIAM Journal on Optimization*, 8(2):506–531, 1998.

I. Tsochantaridis, Thomas Joachims, T., Y. Altun, and Y. Singer. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research*, 9:2543–2596, 2010.

L. Xiao and T. Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

L. Zhang, M. Mahdavi, and R. Jin. Linear convergence with condition number independent access of full gradients. In *Advances in Neural Information Processing Systems*, 2013.