Stochastic Variance-Reduced Optimization for Machine Learning

Part I

Francis Bach

INRIA - Ecole Normale Supérieure, Paris, France

Joint tutorial with Mark Schmidt, UBC - SIOPT - 2017
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.
Visual object recognition
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 recent progress
Outline

1. **Introduction/motivation: Supervised machine learning**
   - Optimization of finite sums
   - Existing optimization methods for finite sums

2. **Convex finite-sum problems**
   - Linearly-convergent stochastic gradient method
   - SAG, SAGA, SVRG, SDCA, MISO, etc.
   - From lazy gradient evaluations to variance reduction

3. **Non-convex problems**

4. **Non-independent and identically distributed**

5. **Non-stochastic: other types of structures**

6. **Non-serial: parallel and distributed settings**
References

• Textbooks and tutorials
  – Nesterov (2004): *Introductory lectures on convex optimization*
  – Bertsekas (2016): *Nonlinear programming*
  – Bottou et al. (2016): *Optimization methods for large-scale machine learning*

• Research papers
  – See end of slides
  – Slides available at www.ens.fr/~fbach/
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 $\theta \in \mathbb{R}^d$
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 $\theta \in \mathbb{R}^d$

• **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 $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}, i = 1, \ldots, n$

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

- **Motivating examples**
  - Linear predictions: $h(x, \theta) = \theta^T \Phi(x)$ with features $\Phi(x) \in \mathbb{R}^d$
  - Neural networks: $h(x, \theta) = \theta_m^T \sigma(\theta_{m-1}^T \sigma(\cdots \theta_2^T \sigma(\theta_1^T x)))$
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

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

data fitting term + regularizer
Usual losses

- **Regression**: $y \in \mathbb{R}$
  
  - Quadratic loss $\ell(y, h(x, \theta)) = \frac{1}{2}(y - h(x, \theta))^2$
Usual losses

- **Regression**: \( y \in \mathbb{R} \)
  - Quadratic loss \( \ell(y, h(x, \theta)) = \frac{1}{2}(y - h(x, \theta))^2 \)

- **Classification**: \( y \in \{-1, 1\} \)
  - Logistic loss \( \ell(y, h(x, \theta)) = \log(1 + \exp(-yh(x, \theta))) \)
Usual losses

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

- **Classification**: $y \in \{-1, 1\}$
  - Logistic loss $\ell(y, h(x, \theta)) = \log(1 + \exp(-yh(x, \theta)))$

- **Structured prediction**
  - Complex outputs $y$ ($k$ classes/labels, graphs, trees, or $\{0, 1\}^k$, etc.)
  - Prediction function $h(x, \theta) \in \mathbb{R}^k$
  - Conditional random fields (Lafferty et al., 2001)
  - Max-margin (Taskar et al., 2003; Tsochantaridis et al., 2005)
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

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

data fitting term + regularizer
Parametric supervised machine learning

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

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

- **(regularized) empirical risk minimization**: find $\hat{\theta}$ solution of

\[
\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \left\{ \ell(y_i, h(x_i, \theta)) + \lambda \Omega(\theta) \right\} = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)
\]

data fitting term + regularizer
Parametric supervised machine learning

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

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

- **(regularized) empirical risk minimization**: find $\hat{\theta}$ solution of

\[
\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \left\{ \ell(y_i, h(x_i, \theta)) + \lambda \Omega(\theta) \right\} = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)
\]

  data fitting term + regularizer

- **Optimization**: optimization of regularized risk training cost
Parametric supervised machine learning

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

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

- **(regularized) empirical risk minimization**: find $\hat{\theta}$ solution of

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

  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
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

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

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

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

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

$$\forall \theta \in \mathbb{R}^d, \text{ eigenvalues} [g''(\theta)] \geq 0$$
Smoothness and (strong) convexity

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

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

convex

strongly convex
Smoothness and (strong) convexity

- A twice differentiable function \( g : \mathbb{R}^d \rightarrow \mathbb{R} \) is \( \mu \)-strongly convex if and only if
  \[
  \forall \theta \in \mathbb{R}^d, \text{ eigenvalues} \left[ g''(\theta) \right] \geq \mu
  \]
- Condition number \( \kappa = L/\mu \geq 1 \)

(small \( \kappa = L/\mu \)) \hspace{1cm} (large \( \kappa = L/\mu \))
Smoothness and (strong) convexity

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

- 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 $g : \mathbb{R}^d \rightarrow \mathbb{R}$ is $\mu$-strongly convex if and only if
  \[
  \forall \theta \in \mathbb{R}^d, \text{ eigenvalues } [g''(\theta)] \geq \mu
  \]

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 $g : \mathbb{R}^d \rightarrow \mathbb{R}$ is $\mu$-strongly convex if and only if
  $$\forall \theta \in \mathbb{R}^d, \text{eigenvalues}[g''(\theta)] \geq \mu$$

- **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 $g : \mathbb{R}^d \rightarrow \mathbb{R}$ is $\mu$-strongly convex if and only if
  $$\forall \theta \in \mathbb{R}^d, \text{eigenvalues}[g''(\theta)] \geq \mu$$

- **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 $\frac{1}{n} \sum_{i=1}^{n} \Phi(x_i)\Phi(x_i)^\top \Rightarrow n \geq d$
  - Even when $\mu > 0$, $\mu$ may be arbitrarily small!
Smoothness and (strong) convexity

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

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

- **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 $\frac{1}{n} \sum_{i=1}^{n} \Phi(x_i)\Phi(x_i)^\top \Rightarrow n \geq d$
  - Even when $\mu > 0$, $\mu$ may be arbitrarily small!

- Adding regularization by $\frac{\mu}{2} \| \theta \|^2$
  
  - creates additional bias unless $\mu$ is small, but reduces variance
  - Typically $L/\sqrt{n} \geq \mu \geq L/n$
Iterative methods for minimizing smooth functions

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

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

Two diagrams illustrate the convergence of the gradient descent method for two different values of $\kappa = L/\mu$:

- **Small $\kappa = L/\mu$** (left)
- **Large $\kappa = L/\mu$** (right)
Iterative methods for minimizing smooth functions

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

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

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

\begin{align*}
\text{(small } \kappa = L/\mu) & \\
\text{(large } \kappa = L/\mu) & 
\end{align*}
Iterative methods for minimizing smooth functions

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

- **Gradient descent:** $\theta_t = \theta_{t-1} - \gamma_t \cdot g'(\theta_{t-1})$
  - $O(1/t)$ 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:** $\theta_t = \theta_{t-1} - g''(\theta_{t-1})^{-1}g'(\theta_{t-1})$
  
  - $O(e^{-\rho^2 t})$ *quadratic* rate
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 $\iff O(\kappa \log \frac{1}{\varepsilon})$ iterations

- **Newton method**: $\theta_t = \theta_{t-1} - g''(\theta_{t-1})^{-1}g'(\theta_{t-1})$
  - $O(e^{-\rho^2 t})$ quadratic rate $\iff O(\log \log \frac{1}{\varepsilon})$ 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 $= O(nd \cdot \kappa \log \frac{1}{\epsilon})$

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

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

• 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 $\iff$ complexity $= O(nd \cdot \kappa \log \frac{1}{\varepsilon})$

- **Newton method**: $\theta_t = \theta_{t-1} - g''(\theta_{t-1})^{-1} g'(\theta_{t-1})$
  - $O(e^{-\rho^2 t})$ quadratic rate $\iff$ 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 \( \{1, \ldots , n\} \)
  - 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 \( \{1, \ldots, n\} \)
  - 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 \( g \) \( \mu \)-strongly-convex:

\[ \mathbb{E} g(\bar{\theta}_t) - g(\theta^*) \leq \begin{cases} 
  O\left( \frac{1}{\sqrt{t}} \right) & \text{if } \gamma_t = \frac{1}{(L\sqrt{t})} \\
  O\left( \frac{L}{(\mu t)} \right) = O\left( \frac{\kappa}{t} \right) & \text{if } \gamma_t = \frac{1}{(\mu t)}
\end{cases} \]

- No adaptivity to strong-convexity in general
- Adaptivity with self-concordance assumption (Bach, 2014)
- Running-time complexity: \( O(d \cdot \kappa/\varepsilon) \)
Outline

1. Introduction/motivation: Supervised machine learning
   – Optimization of finite sums
   – Existing optimization methods for finite sums

2. Convex finite-sum problems
   – Linearly-convergent stochastic gradient method
   – SAG, SAGA, SVRG, SDCA, etc.
   – From lazy gradient evaluations to variance reduction

3. Non-convex problems

4. Parallel and distributed settings

5. Perspectives
Stochastic vs. deterministic methods

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

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

- 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 \( f_i(\theta) = \ell(y_i, h(x_i, \theta)) + \lambda \Omega(\theta) \)

- 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 \( f_i(\theta) = \ell(y_i, h(x_i, \theta)) + \lambda \Omega(\theta) \)

- 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 \( \{1, \ldots, n\} \)
  - 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 $f_i(\theta) = \ell(y_i, h(x_i, \theta)) + \lambda \Omega(\theta)$

- 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

![Graph comparing stochastic and deterministic methods](image)
• **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)

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

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

- Good choice of momentum term \( \delta_t \in [0, 1) \)
  
  \[
g(\theta_t) - g(\theta^*) \leq O\left(\frac{1}{t^2}\right)
  \]

  \[
g(\theta_t) - g(\theta^*) \leq O\left(e^{-t\sqrt{\mu/L}}\right) = O\left(e^{-t/\sqrt{\kappa}}\right) \quad \text{if} \ \mu\text{-strongly convex}
  \]

- Optimal rates after \( t = O(d) \) iterations (Nesterov, 2004)
Accelerating gradient methods - Related work

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

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

  - Good choice of momentum term \( \delta_t \in [0, 1) \)
    \[ g(\theta_t) - g(\theta_*) \leq O\left(1/t^2\right) \]
    \[ g(\theta_t) - g(\theta_*) \leq O\left(e^{-t\sqrt{\mu/L}}\right) = O\left(e^{-t/\sqrt{\kappa}}\right) \text{ if } \mu\text{-strongly convex} \]
  
  - **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 $f_i$, $i = 1, \ldots, n$
  - Random selection $i(t) \in \{1, \ldots, n\}$ with replacement
  - Iteration: $\theta_t = \theta_{t-1} - \frac{\gamma_t}{n} \sum_{i=1}^{n} y_t^i$ with $y_t^i = \begin{cases} f_i'(\theta_{t-1}) & \text{if } i = i(t) \\ y_{i-1}^i & \text{otherwise} \end{cases}$
• **Stochastic average gradient (SAG) iteration**
  
  – Keep in memory the gradients of all functions $f_i$, $i = 1, \ldots, n$
  
  – Random selection $i(t) \in \{1, \ldots, n\}$ with replacement
  
  – Iteration: $\theta_t = \theta_{t-1} - \frac{\gamma_t}{n} \sum_{i=1}^{n} y^t_i$ with $y^t_i = \begin{cases} f'_i(\theta_{t-1}) & \text{if } i = i(t) \\ y^t_{i-1} & \text{otherwise} \end{cases}$

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

gradients $\in \mathbb{R}^d \quad \frac{1}{n} \sum_{i=1}^{n} y^t_i \quad y^t_1 \quad y^t_2 \quad y^t_3 \quad y^t_4 \quad \ldots \quad y^t_{n-1} \quad y^t_n$
Stochastic average gradient (SAG) iteration

- Keep in memory the gradients of all functions $f_i, i = 1, \ldots, n$
- Random selection $i(t) \in \{1, \ldots, n\}$ 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 $g = \frac{1}{n} \sum_{i=1}^{n} f_i$  

gradients $\in \mathbb{R}^d$ $\frac{1}{n} \sum_{i=1}^{n} y_i^t$  

$y_1^t \quad y_2^t \quad y_3^t \quad y_4^t \quad \cdots \quad y_{n-1}^t \quad y_n^t$
**Stochastic average gradient**

*(Le Roux, Schmidt, and Bach, 2012)*

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

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

gradients $\in \mathbb{R}^d$  
$\frac{1}{n} \sum_{i=1}^{n} y^t_i$  
$y^t_1\,\,y^t_2\,\,y^t_3\,\,y^t_4\,\,\ldots\,\,y^t_{n-1}\,\,y^t_n$
Stochastic average gradient
(Le Roux, Schmidt, and Bach, 2012)

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

- 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 $f_i, i = 1, \ldots, n$
  – Random selection $i(t) \in \{1, \ldots, n\}$ 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 $f_i'(\theta) = \ell'(y_i, \Phi(x_i)^\top \theta) \Phi(x_i)$
Stochastic average gradient - Convergence analysis

• Assumptions
  
  – Each $f_i$ is $L$-smooth, $i = 1, \ldots, n$
  
  – $g = \frac{1}{n} \sum_{i=1}^{n} f_i$ is $\mu$-strongly convex
  
  – constant step size $\gamma_t = 1/(16L)$ - no need to know $\mu$
**Stochastic average gradient - Convergence analysis**

- **Assumptions**
  - Each $f_i$ is $L$-smooth, $i = 1, \ldots, n$
  - $g = \frac{1}{n} \sum_{i=1}^{n} f_i$ is $\mu$-strongly convex
  - constant step size $\gamma_t = 1/(16L)$ - no need to know $\mu$

- **Strongly convex case** (Le Roux et al., 2012; Schmidt et al., 2016)
  \[
  \mathbb{E}[g(\theta_t) - g(\theta^*)] \leq \text{cst} \times \left(1 - \min \left\{ \frac{1}{8n}, \frac{\mu}{16L} \right\} \right)^t
  \]
  - Linear (exponential) convergence rate with $O(d)$ iteration cost
  - After one pass, reduction of cost by $\exp\left(- \min \left\{ \frac{1}{8}, \frac{n\mu}{16L} \right\} \right)$
  - NB: in machine learning, may often restrict to $\mu \geq L/n$
    $\Rightarrow$ constant error reduction after each effective pass
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 \( g \) \( \mu \)-strongly convex

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Running time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stochastic gradient descent</td>
<td>( d \times \frac{L}{\mu} \times \frac{1}{\varepsilon} )</td>
</tr>
<tr>
<td>Gradient descent</td>
<td>( d \times n \frac{L}{\mu} \times \log \frac{1}{\varepsilon} )</td>
</tr>
<tr>
<td>Accelerated gradient descent</td>
<td>( d \times n \sqrt{\frac{L}{\mu}} \times \log \frac{1}{\varepsilon} )</td>
</tr>
<tr>
<td>SAG</td>
<td>( d \times (n + \frac{L}{\mu}) \times \log \frac{1}{\varepsilon} )</td>
</tr>
</tbody>
</table>

- 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 \( g \) \( \mu \)-strongly convex

<table>
<thead>
<tr>
<th>Method</th>
<th>Time Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stochastic gradient descent</td>
<td>( d \times \frac{L}{\mu} \times \frac{1}{\varepsilon} )</td>
</tr>
<tr>
<td>Gradient descent</td>
<td>( d \times n\frac{L}{\mu} \times \log\frac{1}{\varepsilon} )</td>
</tr>
<tr>
<td>Accelerated gradient descent</td>
<td>( d \times n\sqrt{\frac{L}{\mu}} \times \log\frac{1}{\varepsilon} )</td>
</tr>
<tr>
<td>SAG</td>
<td>( d \times (n + \frac{L}{\mu}) \times \log\frac{1}{\varepsilon} )</td>
</tr>
</tbody>
</table>

- **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 \( (\mu = 0) \)

<table>
<thead>
<tr>
<th>Method</th>
<th>( d \times )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stochastic gradient descent</td>
<td>( 1/\varepsilon^2 )</td>
</tr>
<tr>
<td>Gradient descent</td>
<td>( n/\varepsilon )</td>
</tr>
<tr>
<td>Accelerated gradient descent</td>
<td>( n/\sqrt{\varepsilon} )</td>
</tr>
<tr>
<td>SAG</td>
<td>( \sqrt{n}/\varepsilon )</td>
</tr>
</tbody>
</table>

**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 ⇒ 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**

\( n = 145751, \ d = 74 \)

**sido dataset**

\( n = 12678, \ d = 4932 \)
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
Variance reduction

- **Principle**: reducing variance of sample of $X$ by using a sample from another random variable $Y$ with known expectation

$$Z_\alpha = \alpha(X - Y) + \mathbb{E}Y$$

- $\mathbb{E}Z_\alpha = \alpha\mathbb{E}X + (1 - \alpha)\mathbb{E}Y$
- $\text{var}(Z_\alpha) = \alpha^2[\text{var}(X) + \text{var}(Y) - 2\text{cov}(X, Y)]$
- $\alpha = 1$: no bias, $\alpha < 1$: potential bias (but reduced variance)
- Useful if $Y$ positively correlated with $X$
Variance reduction

- **Principle**: reducing variance of sample of $X$ by using a sample from another random variable $Y$ with known expectation

  $$Z_\alpha = \alpha(X - Y) + \mathbb{E}Y$$

  $$\mathbb{E}Z_\alpha = \alpha\mathbb{E}X + (1 - \alpha)\mathbb{E}Y$$

  $$\text{var}(Z_\alpha) = \alpha^2[\text{var}(X) + \text{var}(Y) - 2\text{cov}(X, Y)]$$

  - $\alpha = 1$: no bias, $\alpha < 1$: potential bias (but reduced variance)
  - Useful if $Y$ positively correlated with $X$

- **Application to gradient estimation** (Johnson and Zhang, 2013; Zhang, Mahdavi, and Jin, 2013)

  - SVRG: $X = f'_{i(t)}(\theta_{t-1})$, $Y = f'_{i(t)}(\tilde{\theta})$, $\alpha = 1$, with $\tilde{\theta}$ stored
  - $\mathbb{E}Y = \frac{1}{n}\sum_{i=1}^{n} f'_i(\tilde{\theta})$ full gradient at $\tilde{\theta}$, $X - Y = f'_{i(t)}(\theta_{t-1}) - f'_i(t)(\tilde{\theta})$
Stochastic variance reduced gradient (SVRG) (Johnson and Zhang, 2013; Zhang et al., 2013)

- Initialize $\tilde{\theta} \in \mathbb{R}^d$
- For $i_{\text{epoch}} = 1$ to $\#$ of epochs
  - Compute all gradients $f'_i(\tilde{\theta})$; store $g'(\tilde{\theta}) = \frac{1}{n} \sum_{i=1}^{n} f'_i(\tilde{\theta})$
  - Initialize $\theta_0 = \tilde{\theta}$
  - For $t = 1$ to length of epochs
    $$\theta_t = \theta_{t-1} - \gamma \left[ g'(\tilde{\theta}) + (f'_{i(t)}(\theta_{t-1}) - f'_{i(t)}(\tilde{\theta})) \right]$$
  - Update $\tilde{\theta} = \theta_t$
- Output: $\tilde{\theta}$

- No need to store gradients - two gradient evaluations per inner step
- Two parameters: length of epochs + step-size
- Same linear convergence rate as SAG, simpler proof
Stochastic variance reduced gradient (SVRG)  
(Johnson and Zhang, 2013; Zhang et al., 2013)

- Initialize \( \tilde{\theta} \in \mathbb{R}^d \)
- For \( i_{\text{epoch}} = 1 \) to \# of epochs
  - Compute all gradients \( f'_i(\tilde{\theta}) \); store \( g'(\tilde{\theta}) = \frac{1}{n} \sum_{i=1}^{n} f'_i(\tilde{\theta}) \)
  - Initialize \( \theta_0 = \tilde{\theta} \)
  - For \( t = 1 \) to length of epochs
    \[ \theta_t = \theta_{t-1} - \gamma \left[ g'(\tilde{\theta}) + (f'_{i(t)}(\theta_{t-1}) - f'_{i(t)}(\tilde{\theta})) \right] \]
  - Update \( \tilde{\theta} = \theta_t \)
- Output: \( \tilde{\theta} \)

- No need to store gradients - two gradient evaluations per inner step
- Two parameters: length of epochs + step-size \( \gamma \)
- Same linear convergence rate as SAG, simpler proof
Interpretation of SAG as variance reduction

- **SAG update**: $\theta_t = \theta_{t-1} - \frac{\gamma}{n} \sum_{i=1}^{n} y^t_i$ with $y^t_i = \begin{cases} f'_i(\theta_{t-1}) & \text{if } i = i(t) \\ y^t_{i-1} & \text{otherwise} \end{cases}$

- Interpretation as lazy gradient evaluations
Interpretation of SAG as variance reduction

- **SAG update**: $\theta_t = \theta_{t-1} - \gamma \frac{1}{n} \sum_{i=1}^{n} y_i^t$ with $y_i^t = \begin{cases} f_i'(\theta_{t-1}) & \text{if } i = i(t) \\ y_i^{t-1} & \text{otherwise} \end{cases}$

  - Interpretation as lazy gradient evaluations

- **SAG update**: $\theta_t = \theta_{t-1} - \gamma \left[ \frac{1}{n} \sum_{i=1}^{n} y_i^{t-1} + \frac{1}{n} \left( f_i'(\theta_{t-1}) - y_i^{t-1} \right) \right]$

  - Biased update (expectation w.r.t. to $i(t)$ not equal to full gradient)
Interpretation of SAG as variance reduction

- **SAG update**: \( \theta_t = \theta_{t-1} - \frac{\gamma}{n} \sum_{i=1}^{n} y_t^i \) with \( y_t^i = \begin{cases} f'_i(\theta_{t-1}) & \text{if } i = i(t) \\ y_{t-1}^i & \text{otherwise} \end{cases} \)

  - Interpretation as lazy gradient evaluations

- **SAG update**: \( \theta_t = \theta_{t-1} - \gamma \left[ \frac{1}{n} \sum_{i=1}^{n} y_{t-1}^i + \frac{1}{n} (f'_i(t)(\theta_{t-1}) - y_{i(t)}^{t-1}) \right] \)

  - Biased update (expectation w.r.t. to \( i(t) \) not equal to full gradient)

- **SVRG update**: \( \theta_t = \theta_{t-1} - \gamma \left[ \frac{1}{n} \sum_{i=1}^{n} f'_i(\tilde{\theta}) + (f'_i(t)(\theta_{t-1}) - f'_i(t)(\tilde{\theta})) \right] \)

  - Unbiased update
Interpretation of SAG as variance reduction

- **SAG update**: \( \theta_t = \theta_{t-1} - \gamma \sum_{i=1}^{n} y_i^t \) with \( y_i^t = \begin{cases} f_i'(\theta_{t-1}) & \text{if } i = i(t) \\ y_{i(t)}^{t-1} & \text{otherwise} \end{cases} \)

  - Interpretation as lazy gradient evaluations

- **SAG update**: \( \theta_t = \theta_{t-1} - \gamma \left[ \frac{1}{n} \sum_{i=1}^{n} y_i^{t-1} + \frac{1}{n} (f_i'(\theta_{t-1}) - y_{i(t)}^{t-1}) \right] \)

  - Biased update (expectation w.r.t. to \( i(t) \) not equal to full gradient)

- **SVRG update**: \( \theta_t = \theta_{t-1} - \gamma \left[ \frac{1}{n} \sum_{i=1}^{n} f_i'(\tilde{\theta}) + (f_i'(\theta_{t-1}) - f_i'(\tilde{\theta})) \right] \)

  - Unbiased update

- **SAGA update**: \( \theta_t = \theta_{t-1} - \gamma \left[ \frac{1}{n} \sum_{i=1}^{n} y_i^{t-1} + (f_i'(\theta_{t-1}) - y_{i(t)}^{t-1}) \right] \)

  - Defazio, Bach, and Lacoste-Julien (2014a)
  - Unbiased update without epochs
SVRG vs. SAGA

• **SAGA update:** \( \theta_t = \theta_{t-1} - \gamma \left[ \frac{1}{n} \sum_{i=1}^{n} y_{i(t)}^{t-1} + (f'_{i(t)}(\theta_{t-1}) - y_{i(t)}^{t-1}) \right] \)

• **SVRG update:** \( \theta_t = \theta_{t-1} - \gamma \left[ \frac{1}{n} \sum_{i=1}^{n} f'_{i}(\tilde{\theta}) + (f'_{i(t)}(\theta_{t-1}) - f'_{i(t)}(\tilde{\theta})) \right] \)

<table>
<thead>
<tr>
<th>Storage of gradients</th>
<th>SAGA</th>
<th>SVRG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Epoch-based</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Parameters</td>
<td>step-size</td>
<td>step-size &amp; epoch lengths</td>
</tr>
<tr>
<td>Gradient evaluations per step</td>
<td>1</td>
<td>at least 2</td>
</tr>
<tr>
<td>Adaptivity to strong-convexity</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Robustness to ill-conditioning</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>

– See Babanezhad et al. (2015)
Proximal extensions

- **Composite optimization problems:**
  \[
  \min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} f_i(\theta) + h(\theta)
  \]

  - \(f_i\) smooth and convex
  - \(h\) convex, potentially non-smooth
Proximal extensions

- **Composite optimization problems:**

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

- $f_i$ smooth and convex
- $h$ convex, potentially non-smooth
- Constrained optimization: $h(\theta) = 0$ if $\theta \in K$, and $+\infty$ otherwise
- Sparsity-inducing norms, e.g., $h(\theta) = \|\theta\|_1$
Proximal extensions

- **Composite optimization problems**: \( \min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} f_i(\theta) + h(\theta) \)
  
  - \( f_i \) smooth and convex
  - \( h \) convex, potentially non-smooth
  - Constrained optimization: \( h(\theta) = 0 \) if \( \theta \in K \), and \(+\infty\) otherwise
  - Sparsity-inducing norms, e.g., \( h(\theta) = \|\theta\|_1 \)

- **Proximal methods (a.k.a. splitting methods)**
  
  - Extra projection / soft thresholding step after gradient update
  - See, e.g., Combettes and Pesquet (2011); Bach, Jenatton, Mairal, and Obozinski (2012); Parikh and Boyd (2014)
Proximal extensions

• **Composite optimization problems:** \( \min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} f_i(\theta) + h(\theta) \)

  – \( f_i \) smooth and convex
  – \( h \) convex, potentially non-smooth
  – Constrained optimization: \( h(\theta) = 0 \) if \( \theta \in K \), and \(+\infty\) otherwise
  – Sparsity-inducing norms, e.g., \( h(\theta) = \|\theta\|_1 \)

• **Proximal methods (a.k.a. splitting methods)**

  – Extra projection / soft thresholding step after gradient update
  – See, e.g., Combettes and Pesquet (2011); Bach, Jenatton, Mairal, and Obozinski (2012); Parikh and Boyd (2014)

• **Directly extends to variance-reduced gradient techniques**

  – Same rates of convergence
**Acceleration**

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

<table>
<thead>
<tr>
<th>Method</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gradient descent</td>
<td>$d \times n \frac{L}{\mu} \times \log \frac{1}{\varepsilon}$</td>
</tr>
<tr>
<td>Accelerated gradient descent</td>
<td>$d \times n \sqrt{\frac{L}{\mu}} \times \log \frac{1}{\varepsilon}$</td>
</tr>
<tr>
<td>SAG(A), SVRG, SDCA, MISO</td>
<td>$d \times (n + \frac{L}{\mu}) \times \log \frac{1}{\varepsilon}$</td>
</tr>
</tbody>
</table>
**Acceleration**

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

<table>
<thead>
<tr>
<th>Method</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gradient descent</td>
<td>$d \times \frac{nL}{\mu} \times \log \frac{1}{\varepsilon}$</td>
</tr>
<tr>
<td>Accelerated gradient descent</td>
<td>$d \times \frac{n\sqrt{L}}{\mu} \times \log \frac{1}{\varepsilon}$</td>
</tr>
<tr>
<td>SAG(A), SVRG, SDCA, MISO</td>
<td>$d \times (n + \frac{L}{\mu}) \times \log \frac{1}{\varepsilon}$</td>
</tr>
<tr>
<td>Accelerated versions</td>
<td>$d \times (n + \sqrt{n\frac{L}{\mu}}) \times \log \frac{1}{\varepsilon}$</td>
</tr>
</tbody>
</table>

- **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
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

<table>
<thead>
<tr>
<th>Objective minus Optimum</th>
<th>Effective Passes</th>
</tr>
</thead>
<tbody>
<tr>
<td>10^{-20}</td>
<td>0</td>
</tr>
<tr>
<td>10^{-15}</td>
<td>10</td>
</tr>
<tr>
<td>10^{-10}</td>
<td>20</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>30</td>
</tr>
<tr>
<td>0</td>
<td>40</td>
</tr>
<tr>
<td>10^{0}</td>
<td>50</td>
</tr>
</tbody>
</table>
From training to testing errors

- **rcv1** dataset \( (n = 697\,641,\ d = 47\,236) \)
  - NB: IAG, SG-C, ASG with optimal step-sizes in hindsight
SGD minimizes the testing cost!

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

- **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 $f(\theta) = \mathbb{E}_{p(x,y)} \ell(y, \theta^\top \Phi(x))$
  
  - Given $n$ independent samples $(x_i, y_i)$, $i = 1, \ldots, n$ 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 $\mu$
  - Simple choice of step-size (equal to $1/L$)
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 $\mu$
  - Simple choice of step-size (equal to $1/L$)

- Convergence in $O(1/n)$ for smooth losses with $O(d)$ online Newton step
Conclusions - Convex optimization

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

• 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
  – Extension to saddle-point problems (Balamurugan and Bach, 2016)
  – Lower bounds for finite sums (Agarwal and Bottou, 2015; Lan, 2015; Arjevani and Shamir, 2016)
  – Sampling without replacement (Gurbuzbalaban et al., 2015; Shamir, 2016)
Conclusions - Convex optimization

• 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
  – Extension to saddle-point problems (Balamurugan and Bach, 2016)
  – Lower bounds for finite sums (Agarwal and Bottou, 2015; Lan, 2015; Arjevani and Shamir, 2016)
  – Sampling without replacement (Gurbuzbalaban et al., 2015; Shamir, 2016)
  – Bounds on testing errors for incremental methods (Frostig et al., 2015; Babanezhad et al., 2015)
Conclusions - Convex optimization

• 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
  – Extension to saddle-point problems (Balamurugan and Bach, 2016)
  – Lower bounds for finite sums (Agarwal and Bottou, 2015; Lan, 2015; Arjevani and Shamir, 2016)
  – Sampling without replacement (Gurbuzbalaban et al., 2015; Shamir, 2016)
  – Bounds on testing errors for incremental methods (Frostig et al., 2015; Babanezhad et al., 2015)

• What’s next: non-convex, non-i.i.d., non-serial
References


A. Nitanda. Stochastic proximal gradient descent with acceleration techniques. In *Advances in Neural


I. Tsochantaridis, Thomas Joachims, T., Y. Altun, and Y. Singer. Large margin methods for structured

