Large-scale machine learning: large $N$, large $P$

- $N$: number of observations (inputs)
- $P$: dimension of each observation

Examples: vision, bioinformatics, speech, language, etc.

- Pascal large-scale datasets: $N = 5 \cdot 10^5$, $P = 10^3$
- ImageNet: $N = 10^7$
- Industrial datasets: $N > 10^8$, $P > 10^7$
**Large-scale machine learning**: large $N$, large $P$

- $N$: number of observations (inputs)
- $P$: dimension of each observation

**Examples**: vision, bioinformatics, speech, language, etc.
- Pascal large-scale datasets: $N = 5 \cdot 10^5$, $P = 10^3$
- ImageNet: $N = 10^7$
- Industrial datasets: $N > 10^8$, $P > 10^7$

**Main computational challenge:**
- Design algorithms for very large $N$ and $P$. 
Example: Supervised Machine Learning

- **Data**: \( n \) observations \((a_i, b_i), i = 1, \ldots, N\).
- Prediction as linear function \( x^T a_i \) of features \( a_i \in \mathbb{R}^P \).

**Regularized empirical risk minimization**: find \( x^* \) solution of

\[
\min_{x \in \mathbb{R}^P} \frac{1}{N} \sum_{i=1}^{N} \ell(b_i, x^T a_i) + \lambda r(x)
\]

data fitting term + regularizer

Applications to any data-oriented field: Vision, bioinformatics, speech, natural language, web.

Main practical challenges: Designing/learning good features \( a_i \).
Efficiently solving the problem when \( N \) or \( P \) are very large.

Mark Schmidt

Linearly-Convergent Stochastic-Gradient Methods
Example: Supervised Machine Learning

- **Data:** \( n \) observations \((a_i, b_i), i = 1, \ldots, N\).
- Prediction as linear function \( x^T a_i \) of features \( a_i \in \mathbb{R}^P \).
- **Regularized empirical risk minimization:** find \( x^* \) solution of

\[
\min_{x \in \mathbb{R}^P} \frac{1}{N} \sum_{i=1}^{N} \ell(b_i, x^T a_i) + \lambda r(x)
\]

- **Applications to any data-oriented field:**
  - Vision, bioinformatics, speech, natural language, web.
- **Main practical challenges:**
  - Designing/learning good features \( a_i \).
  - Efficiently solving the problem when \( N \) or \( P \) are very large.
We want to minimize the sum of a finite set of smooth functions:

$$\min_{x \in \mathbb{R}^p} g(x) := \frac{1}{N} \sum_{i=1}^{N} f_i(x).$$
We want to minimize the sum of a finite set of smooth functions:

$$\min_{x \in \mathbb{R}^p} g(x) := \frac{1}{N} \sum_{i=1}^{N} f_i(x).$$

We are interested in cases where $N$ is very large.

We will focus on strongly-convex functions $g$. 
Big-N Problems

- We want to minimize the sum of a **finite** set of smooth functions:

\[
\min_{x \in \mathbb{R}^P} g(x) := \frac{1}{N} \sum_{i=1}^{N} f_i(x).
\]

- We are interested in cases where \( N \) is very large.
- We will focus on **strongly-convex** functions \( g \).
- Simplest example is \( \ell_2 \)-regularized least-squares,

\[
f_i(x) := (a_i^T x - b_i)^2 + \frac{\lambda}{2} \|x\|^2.
\]

- Other examples include any \( \ell_2 \)-regularized convex loss:
  - logistic regression, smooth SVMs, CRFs, etc.
We consider minimizing $g(x) = \frac{1}{n} \sum_{i=1}^{N} f_i(x)$.
Stochastic vs. Deterministic Gradient Methods

- We consider minimizing $g(x) = \frac{1}{n} \sum_{i=1}^{N} f_i(x)$.

- **Deterministic** gradient method [Cauchy, 1847]:

$$x_{t+1} = x_t - \alpha_t g'(x_t) = x_t - \frac{\alpha_t}{n} \sum_{i=1}^{N} f'_i(x_t).$$

- Linear convergence rate: $O(\rho^t)$.

- Iteration cost is linear in $N$.

---

Mark Schmidt

Linearly-Convergent Stochastic-Gradient Methods
We consider minimizing $g(x) = \frac{1}{n} \sum_{i=1}^{N} f_i(x)$.

**Deterministic** gradient method [Cauchy, 1847]:

$$x_{t+1} = x_t - \alpha_t g'(x_t) = x_t - \frac{\alpha_t}{n} \sum_{i=1}^{N} f'_i(x_t).$$

- **Linear** convergence rate: $O(\rho^t)$.
- Iteration cost is **linear** in $N$.

**Stochastic** gradient method [Robbins & Monro, 1951]:

- Random selection of $i(t)$ from $\{1, 2, \ldots, N\}$,

$$x_{t+1} = x_t - \alpha_t f_{i(t)}(x_t).$$

- Iteration cost is **independent** of $N$.
- **Sublinear** convergence rate: $O(1/t)$. 
Stochastic vs. Deterministic Gradient Methods

- We consider minimizing $g(x) = \frac{1}{n} \sum_{i=1}^{N} f_i(x)$.
- **Deterministic** gradient method [Cauchy, 1847]:

- **Stochastic** gradient method [Robbins & Monro, 1951]:
**Motivation for New Methods**

- **FG method** has $O(N)$ cost with linear rate.
- **SG method** has $O(1)$ cost with sublinear rate.

---

**Stochastic vs. Deterministic Methods**

- **Goal**: best of both worlds: linear rate with $O(1)$ iteration cost

![Graph showing the comparison between stochastic and deterministic methods.](image)
Motivation for New Methods

- FG method has $O(N)$ cost with linear rate.
- SG method has $O(1)$ cost with sublinear rate.

Goal is linear rate with reduced cost.
A variety of methods have been proposed to speed up SG methods:
Prior Work on Speeding up SG Methods

A variety of methods have been proposed to speed up SG methods:

- **Momentum, gradient averaging, iterate averaging, stochastic version of FG methods:**
  

  - None of these methods improve on the $O(1/t)$ rate.
A variety of methods have been proposed to speed up SG methods:

- **Momentum, gradient averaging, iterate averaging, stochastic version of FG methods:**
  
  
  - None of these methods improve on the $O(1/t)$ rate.

- **Constant step-size SG, accelerated SG:**
  
  
  - Linear rate, but only up to a fixed tolerance.
A variety of methods have been proposed to speed up SG methods:

- **Momentum, gradient averaging, iterate averaging, stochastic version of FG methods:**
  
  
  - None of these methods improve on the $O(1/t)$ rate.

- **Constant step-size SG, accelerated SG:**
  
  
  - Linear rate, but only up to a fixed tolerance.

- **Hybrid Methods, Incremental Average Gradient:**
  
  [Bertsekas, 1997, Blatt et al., 2007]
  
  - Linear rate, but iterations make full passes through the data.
Is a linear rate possible, without requiring full passes?
Overview of Contributions

Is a linear rate possible, without requiring full passes?

1. **Control the sample size** to interpolate between FG and SG.
   - Linear convergence rate.
   - Iteration cost grows from $O(1)$ to $O(N)$. 

Mark Schmidt
Linearly-Convergent Stochastic-Gradient Methods
Is a linear rate possible, without requiring full passes?

1. Control the sample size to interpolate between FG and SG.
   - Linear convergence rate.
   - Iteration cost grows from $O(1)$ to $O(N)$.

2. SAG algorithm: sequence of estimates converging to $g'(x^t)$ as $\|x^t - x^{t-1}\| \to 0$.
   - Linear convergence rate.
   - Iteration cost is $O(1)$. 
SG methods with a larger subsample

- Approach 1: control the sample size.
Approach 1: control the sample size.

The FG method uses the exact gradient,

$$\frac{1}{N} \sum_{i=1}^{N} f_i(x^t) = g'(x^t).$$

The SG method approximates it with 1 sample,

$$f_i(t)(x^t) \approx \frac{1}{N} \sum_{i=1}^{N} f_i(x^t).$$
Approach 1: control the sample size.

The FG method uses the exact gradient,

\[
\frac{1}{N} \sum_{i=1}^{N} f_i(x^t) = g'(x^t).
\]

The SG method approximates it with 1 sample,

\[
f_{i(t)}(x^t) \approx \frac{1}{N} \sum_{i=1}^{N} f_i(x^t).
\]

A common variant is to use larger sample \( B^t \),

\[
\frac{1}{|B^t|} \sum_{i \in B^t} f'_i(x^t) \approx \frac{1}{N} \sum_{i=1}^{N} f_i(x^t).
\]
The SG method with a sample $B^t$ uses iterations

$$x^{t+1} = x^t - \frac{\alpha^t}{|B^t|} \sum_{i \in B^t} f_i(x^t).$$

For a fixed sample size $|B^t|$, the rate is sublinear.
The SG method with a sample $\mathcal{B}^t$ uses iterations

$$x^{t+1} = x^t - \frac{\alpha^t}{|\mathcal{B}^t|} \sum_{i \in \mathcal{B}^t} f_i(x^t).$$

- For a fixed sample size $|\mathcal{B}^t|$, the rate is sublinear.
- Gradient error decreases as sample size $|\mathcal{B}^t|$ increases.

[Bertsekas & Tsitsiklis, 1996]
The SG method with a sample $\mathcal{B}^t$ uses iterations

$$x^{t+1} = x^t - \frac{\alpha^t}{|\mathcal{B}^t|} \sum_{i \in \mathcal{B}^t} f_i(x^t).$$

For a fixed sample size $|\mathcal{B}^t|$, the rate is sublinear.

Gradient error decreases as sample size $|\mathcal{B}^t|$ increases.

Common to gradually increase the sample size $|\mathcal{B}^t|$.

[Bertsekas & Tsitsiklis, 1996]
The SG method with a sample $\mathcal{B}^t$ uses iterations

$$x^{t+1} = x^t - \frac{\alpha_t}{|\mathcal{B}^t|} \sum_{i \in \mathcal{B}^t} f_i(x^t).$$

- For a fixed sample size $|\mathcal{B}^t|$, the rate is sublinear.
- **Gradient error decreases as sample size $|\mathcal{B}^t|$ increases.**
- Common to gradually increase the sample size $|\mathcal{B}^t|$.
  - [Bertsekas & Tsitsiklis, 1996]
- We can choose $|\mathcal{B}^t|$ to achieve a linear convergence rate.
The SG method with a sample $\mathcal{B}^t$ uses iterations

$$x^{t+1} = x^t - \frac{\alpha^t}{|\mathcal{B}^t|} \sum_{i \in \mathcal{B}^t} f_i(x^t).$$

- For a fixed sample size $|\mathcal{B}^t|$, the rate is sublinear.
- Gradient error decreases as sample size $|\mathcal{B}^t|$ increases.
- Common to gradually increase the sample size $|\mathcal{B}^t|$.

[Mark Schmidt, 2021]

We can choose $|\mathcal{B}^t|$ to achieve a linear convergence rate.

- Early iterations are cheap like SG iterations.
We first analyze the basic gradient method with error $e^t$,

$$x^{t+1} = x^t - \alpha^t (g'(x^t) + e^t).$$
We first analyze the basic gradient method with error $e^t$,

$$x^{t+1} = x^t - \alpha^t (g'(x^t) + e^t).$$

We assume:

- $g'$ is $L$-Lipschitz continuous.
- $g$ is $\mu$-strongly convex.
- The step size $\alpha^t$ is set to $1/L$. 

Convergence of gradient method with bounded error

We first analyze the basic gradient method with error $e^t$,

$$x^{t+1} = x^t - \alpha^t (g'(x^t) + e^t).$$

We assume:

- $g'$ is $L$-Lipschitz continuous.
- $g$ is $\mu$-strongly convex.
- The step size $\alpha^t$ is set to $1/L$.

For twice-differentiable $g$, equivalent to (with $\mu > 0$)

$$\mu L \preceq g''(x) \preceq L.$$
We first analyze the basic gradient method with error $e^t$,

$$x^{t+1} = x^t - \alpha^t (g'(x^t) + e^t).$$

We assume:

- $g'$ is $L$-Lipschitz continuous.
- $g$ is $\mu$-strongly convex.
- The step size $\alpha^t$ is set to $1/L$.

For twice-differentiable $g$, equivalent to (with $\mu > 0$)

$$\mu I \preceq g''(x) \preceq LI$$

We analyze how $\|e^t\|$ affects the convergence rate.
Proposition 1. If the sequence \( \{E[\|e^t\|^2]\} \) is in \( O(\gamma^t) \), then

\[
E[g(x^t) - g(x^*)] \leq (1 - \mu/L)^t[g(x^0) - g(x^*)] + O(\rho^t),
\]

where \( \rho = \max\{\gamma, 1 - \mu/L + \epsilon\} \) for any \( \epsilon \geq 0 \).
**Proposition 1.** If the sequence $\{\mathbb{E}[\|e^t\|^2]\}$ is in $O(\gamma^t)$, then

$$
\mathbb{E}[g(x^t) - g(x^*)] \leq (1 - \mu/L)^t[g(x^0) - g(x^*)] + O(\rho^t),
$$

where $\rho = \max\{\gamma, 1 - \mu/L + \epsilon\}$ for any $\epsilon \geq 0$.

- If $\gamma < 1 - \mu/L$, rate is the same as error-free case.
Proposition 1. If the sequence \( \{E[\|e^t\|^2]\} \) is in \( O(\gamma^t) \), then

\[
E[g(x^t) - g(x^*)] \leq (1 - \mu/L)^t[g(x^0) - g(x^*)] + O(\rho^t),
\]

where \( \rho = \max\{\gamma, 1 - \mu/L + \epsilon\} \) for any \( \epsilon \geq 0 \).

- If \( \gamma < 1 - \mu/L \), rate is the same as error-free case.
- If \( \gamma > 1 - \mu/L \), the rate is \( \gamma \).
Proposition 1. If the sequence \( \{\mathbb{E}[\|e^t\|^2]\} \) is in \( O(\gamma^t) \), then

\[
\mathbb{E}[g(x^t) - g(x^*)] \leq (1 - \mu/L)^t[g(x^0) - g(x^*)] + O(\rho^t),
\]

where \( \rho = \max\{\gamma, 1 - \mu/L + \epsilon\} \) for any \( \epsilon \geq 0 \).

- If \( \gamma < 1 - \mu/L \), rate is the same as error-free case.
- If \( \gamma > 1 - \mu/L \), the rate is \( \gamma \).
- We also obtain a bound on the iterates because

\[
\frac{\mu}{2}\|x^t - x^*\|^2 \leq g(x^t) - g(x^*) \leq \frac{L}{2}\|x^t - x^*\|^2.
\]
Convergence of methods with increasing sample sizes

- How can we set the sample size $|B^t|$ to control $\|e^t\|$?
How can we set the sample size $|B^t|$ to control $\|e^t\|$?

To have $\mathbb{E}[\|e^t\|^2] = O(\gamma^t)$, we need

- for sampling uniformly with replacement:
  
  \[
  \frac{1}{|B^t|} = O(\gamma^t). \]

- for sampling uniformly without replacement:

  
  \[
  \frac{N - |B^t|}{N \cdot |B^t|} = O(\gamma^t). \]
Convergence of methods with increasing sample sizes

- How can we set the sample size $|B^t|$ to control $\|e^t\|$?
- To have $\mathbb{E}[\|e^t\|^2] = O(\gamma^t)$, we need
  - for sampling uniformly with replacement:
    $$\frac{1}{|B^t|} = O(\gamma^t).$$
  - for any sampling without replacement strategy:
    $$\left[ \frac{N - |B^t|}{N} \right]^2 = O(\gamma^t).$$
- How can we set the sample size \(|\mathcal{B}^t|\) to control \(||e^t||\)?
- To have \(\mathbb{E}[||e^t||^2] = O(\gamma^t)\), we need
  - for sampling uniformly with replacement:
    \[
    \frac{1}{|\mathcal{B}^t|} = O(\gamma^t).
    \]
  - for any sampling without replacement strategy:
    \[
    \left[ \frac{N - |\mathcal{B}^t|}{N} \right]^2 = O(\gamma^t).
    \]
  - for sampling uniformly without replacement:
    \[
    \frac{N - |\mathcal{B}^t|}{N} \cdot \frac{1}{|\mathcal{B}^t|} = O(\gamma^t).
    \]
Sample Size needed for Linear Rate

![Graph showing sample size over iterations for different methods: replacement, no replacement, and deterministic.]

- **Sample size (s/m)**
- **Iteration (k)**

- **Legend:**
  - Dotted line: replacement
  - Solid line: no replacement
  - Red line: deterministic
We also give sequence $\mathbb{E}[\|e^t\|^2]$ to achieve strong linear rate:

$$\mathbb{E}[g(x^t) - g(x^*)] \leq (1 - \rho)^t [g(x^0) - g(x^*)],$$

for any $\rho > 1 - \mu/L$. 
We also give sequence $\mathbb{E}[\|e^t\|^2]$ to achieve strong linear rate:

$$\mathbb{E}[g(x^t) - g(x^*)] \leq (1 - \rho)^t [g(x^0) - g(x^*)],$$

for any $\rho > 1 - \mu / L$.

We can analyze more advance algorithms:
- Nesterov’s accelerated gradient method (faster rate).
- Newton-like second-order methods (faster rate)
- Proximal methods (constrained/non-smooth).
We also give sequence $\mathbb{E}[\|e^t\|^2]$ to achieve strong linear rate:

$$\mathbb{E}[g(x^t) - g(x^*)] \leq (1 - \rho)^t[g(x^0) - g(x^*)],$$

for any $\rho > 1 - \mu/L$.

We can analyze more advance algorithms:

- Nesterov’s accelerated gradient method (faster rate).
- Newton-like second-order methods (faster rate)
- Proximal methods (constrained/non-smooth).

We made a practical implementation:

- $L$-BFGS Hessian approximation.
- Armijo line-search on the batch.
- Eventually reduces to standard quasi-Newton method.
Results on chain-structured conditional random field:

Hybrid uses $|B^{t+1}| = \lceil \min\{1.1 \cdot |B^t| + 1, N\} \rceil$. 
Growing $|\mathcal{B}^t|$ eventually requires $O(N)$ iteration cost.
Growing $|B^t|$ eventually requires $O(N)$ iteration cost.

Is it possible to have a linearly convergent algorithm with iteration cost independent of $N$?
Growing $|\mathcal{B}^t|$ eventually requires $O(N)$ iteration cost.

Is it possible to have a linearly convergent algorithm with iteration cost independent of $N$?

YES!
Growing $|\mathcal{B}_t|$ eventually requires $O(N)$ iteration cost.

Is it possible to have a linearly convergent algorithm with iteration cost independent of $N$?

YES! The stochastic average gradient (SAG) algorithm:

- Randomly select $i(t)$ from $\{1, 2, \ldots, n\}$ and compute $f'_{i(t)}(x^t)$.

\[
x^{t+1} = x^t - \frac{\alpha^t}{N} \sum_{i=1}^{N} f_i'(x^t)
\]
Growing $|B_t|$ eventually requires $O(N)$ iteration cost.

Is it possible to have a linearly convergent algorithm with iteration cost independent of $N$?

YES! The stochastic average gradient (SAG) algorithm:

- Randomly select $i(t)$ from $\{1, 2, \ldots, n\}$ and compute $f'_{i(t)}(x^t)$.

\[
x^{t+1} = x^t - \frac{\alpha^t}{N} \sum_{i=1}^{N} f'_i(x^t)
\]
Growing $|\mathcal{B}^t|$ eventually requires $O(N)$ iteration cost.

Is it possible to have a linearly convergent algorithm with iteration cost independent of $N$?

YES! The stochastic average gradient (SAG) algorithm:

- Randomly select $i(t)$ from $\{1, 2, \ldots, n\}$ and compute $f'_{i(t)}(x^t)$.

$$x^{t+1} = x^t - \frac{\alpha^t}{N} \sum_{i=1}^{N} y_i^t$$

- Memory: $y_i^t = f'_i(x^k)$ from the last $k$ where $i$ was selected.
Growing $|\mathcal{B}^t|$ eventually requires $O(N)$ iteration cost.

Is it possible to have a linearly convergent algorithm with iteration cost independent of $N$?

YES! The stochastic average gradient (SAG) algorithm:

- Randomly select $i(t)$ from $\{1, 2, \ldots, n\}$ and compute $f'_i(x^t)$.

$$x^{t+1} = x^t - \frac{\alpha^t}{N} \sum_{i=1}^{N} y_i^t$$

- Memory: $y_i^t = f'_i(x^k)$ from the last $k$ where $i$ was selected.

Stochastic variant of increment average gradient (IAG).

[Blatt et al. 2007]

- Assumes that gradients of other examples don’t change.
Convergence Rate of SAG

- Assume each $f_i'$ is $L$-continuous, $g$ is $\mu$-strongly convex.

**Theorem.** With $\alpha_t = \frac{1}{16L}$ the SAG iterations satisfy

$$
\mathbb{E}[g(x^t) - g(x^*)] \leq \left(1 - \min\left\{\frac{\mu}{16L}, \frac{1}{8N}\right\}\right)^t C,
$$

with

$$
C = [g(x^0) - g(x^*)] + \frac{4L}{N} \|x^0 - x^*\|^2 + \frac{\sigma^2}{16L}.
$$

Linear convergence with iteration cost independent of $N$.

"Despite 60 years of extensive research on SG methods, with a significant portion of the applications focusing on finite datasets, we are not aware of any other SG method that achieves a linear convergence rate while preserving the iteration cost of standard SG methods."
Assume each $f'_i$ is $L$–continuous, $g$ is $\mu$-strongly convex.

**Theorem.** With $\alpha_t = \frac{1}{16L}$ the SAG iterations satisfy

$$\mathbb{E}[g(x^t) - g(x^*)] \leq \left(1 - \min \left\{ \frac{\mu}{16L}, \frac{1}{8N} \right\} \right)^t C,$$

with

$$C = [g(x^0) - g(x^*)] + \frac{4L}{N} \|x^0 - x^*\|^2 + \frac{\sigma^2}{16L}.$$

Linear convergence with iteration cost independent of $N$. 

**Mark Schmidt**

**Linearly-Convergent Stochastic-Gradient Methods**
Assume each $f_i'$ is $L$-continuous, $g$ is $\mu$-strongly convex.

**Theorem.** With $\alpha_t = \frac{1}{16L}$ the SAG iterations satisfy

$$\mathbb{E}[g(x^t) - g(x^*)] \leq \left(1 - \min\left\{\frac{\mu}{16L}, \frac{1}{8N}\right\}\right)^t C,$$

with

$$C = [g(x^0) - g(x^*)] + \frac{4L}{N} \|x^0 - x^*\|^2 + \frac{\sigma^2}{16L}.$$

- Linear convergence with iteration cost independent of $N$.
- “despite 60 years of extensive research on SG methods, with a significant portion of the applications focusing on finite datasets, we are not aware of any other SG method that achieves a linear convergence rate while preserving the iteration cost of standard SG methods.”
Assume each $f_i'$ is $L$-continuous, $g$ is $\mu$-strongly convex.

**Theorem.** With $\alpha_t = \frac{1}{16L}$, the SAG iterations satisfy

$$\mathbb{E}[g(x^t) - g(x^*)] \leq \left( 1 - \min \left\{ \frac{\mu}{16L}, \frac{1}{8N} \right\} \right)^t C,$$

with

$$C = [g(x^0) - g(x^*)] + \frac{4L}{N} \|x^0 - x^*\|^2 + \frac{\sigma^2}{16L}.$$
Assume each $f_i'$ is $L$-continuous, $g$ is $\mu$-strongly convex.

**Theorem.** With $\alpha_t = \frac{1}{16L}$ the SAG iterations satisfy

$$E[g(x^t) - g(x^*)] \leq \left(1 - \min\left\{\frac{\mu}{16L}, \frac{1}{8N}\right\}\right)^t C,$$

with

$$C = [g(x^0) - g(x^*)] + \frac{4L}{N} \|x^0 - x^*\|^2 + \frac{\sigma^2}{16L}.$$

Further, this rate is “very fast”:

- For well-conditioned problems, constant reduction per pass:
  $$\left(1 - \frac{1}{8N}\right)^N \leq \exp\left(-\frac{1}{8}\right) = 0.8825.$$

- For ill-conditioned problems, almost same as deterministic method. (but $N$ times faster)
Assume that $L = 100$, $\mu = .01$, and $n = 80000$: 

- Gradient method has rate $(L - \mu L + \mu)^2 = 0.9996$. 
- Accelerated gradient method has rate $(1 - \sqrt{\mu} L)^2 = 0.9900$. 
- SAG ($N$ iterations) has rate $(1 - \min\{\mu L^{16}, 18 N\})^N = 0.8825$. 

Fastest possible first-order method: $(\sqrt{L} - \sqrt{\mu} \sqrt{L} + \sqrt{\mu})^2 = 0.9608$. 

SAG beats two lower bounds: Stochastic gradient bound. 

Full gradient bound (for appropriate $L$, $\mu$, and $N$). 

$O\left(\frac{1}{k}\right)$ rate in convex case (vs. $O\left(\frac{1}{\sqrt{k}}\right)$ for SG methods). 

Algorithm is adaptive to $\mu$ around optimum. 

Can improve $C$ by clever initialization of $x_0$ and $y_i$. 
Assume that $L = 100$, $\mu = .01$, and $n = 80000$: 

- Gradient method has rate $\left(\frac{L-\mu}{L+\mu}\right)^2 = 0.9996$. 

- Accelerated gradient method has rate $\left(1 - \sqrt{\mu L}\right) = 0.9900$. 

- SAG ($N$ iterations) has rate $\left(1 - \min\{\mu 16, 18\} N\right) N = 0.8825$. 

- Fastest possible first-order method: $\left(\sqrt{L} - \sqrt{\mu} \sqrt{L} + \sqrt{\mu}\right)^2 = 0.9608$. 

SAG beats two lower bounds: 

- Stochastic gradient bound. 
- Full gradient bound (for appropriate $L$, $\mu$, and $N$). 

Algorithm is adaptive to $\mu$ around optimum. Can improve $C$ by clever initialization of $x_0$ and $y_0$. 

Mark Schmidt

Linearly-Convergent Stochastic-Gradient Methods
Assume that $L = 100$, $\mu = .01$, and $n = 80000$:

- Gradient method has rate $\left(\frac{L-\mu}{L+\mu}\right)^2 = 0.9996$.
- Accelerated gradient method has rate $\left(1 - \sqrt{\frac{\mu}{L}}\right) = 0.9900$. 

SAG ($N$ iterations) has rate $\left(1 - \min\{\frac{\mu}{16L}, \frac{1}{8N}\}\right)N = 0.8825$.

Fastest possible first-order method: $\left(\sqrt{L} - \sqrt{\mu} \sqrt{L} + \sqrt{\mu}\right)^2 = 0.9608$.

SAG beats two lower bounds: Stochastic gradient bound. Full gradient bound (for appropriate $L$, $\mu$, and $N$).

$O\left(\frac{1}{k}\right)$ rate in convex case (vs. $O\left(\frac{1}{\sqrt{k}}\right)$ for SG methods).

Algorithm is adaptive to $\mu$ around optimum.

Can improve $C$ by clever initialization of $x_0$ and $y_i$. 

Mark Schmidt

Linearly-Convergent Stochastic-Gradient Methods
Assume that $L = 100$, $\mu = .01$, and $n = 80000$:

- Gradient method has rate $\left( \frac{L-\mu}{L+\mu} \right)^2 = 0.9996$.
- Accelerated gradient method has rate $\left( 1 - \sqrt{\frac{\mu}{L}} \right) = 0.9900$.
- SAG ($N$ iterations) has rate $\left( 1 - \min \left\{ \frac{\mu}{16L}, \frac{1}{8N} \right\} \right)^N = 0.8825$. 
Assume that $L = 100$, $\mu = .01$, and $n = 80000$:

- Gradient method has rate $\left( \frac{L - \mu}{L + \mu} \right)^2 = 0.9996$.
- Accelerated gradient method has rate $(1 - \sqrt{\frac{\mu}{L}}) = 0.9900$.
- SAG ($N$ iterations) has rate $(1 - \min\left\{ \frac{\mu}{16L}, \frac{1}{8N} \right\})^N = 0.8825$.
- Fastest possible first-order method: $\left( \frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}} \right)^2 = 0.9608$.
Assume that $L = 100$, $\mu = .01$, and $n = 80000$:

- Gradient method has rate $\left(\frac{L - \mu}{L + \mu}\right)^2 = 0.9996$.
- Accelerated gradient method has rate $(1 - \sqrt{\frac{\mu}{L}}) = 0.9900$.
- SAG ($N$ iterations) has rate $(1 - \min \left\{ \frac{\mu}{16L}, \frac{1}{8N} \right\})^N = 0.8825$.
- *Fastest possible* first-order method: $\left(\frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}}\right)^2 = 0.9608$.

SAG beats two lower bounds:

- Stochastic gradient bound.
- Full gradient bound (for appropriate $L$, $\mu$, and $N$).
Assume that $L = 100$, $\mu = .01$, and $n = 80000$:

- Gradient method has rate \( \left( \frac{L-\mu}{L+\mu} \right)^2 = 0.9996 \).
- Accelerated gradient method has rate \( (1 - \sqrt{\frac{\mu}{L}}) = 0.9900 \).
- SAG (\( N \) iterations) has rate \( (1 - \min \{ \frac{\mu}{16L}, \frac{1}{8N} \})^N = 0.8825 \).
- **Fastest possible** first-order method: \( \left( \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}} \right)^2 = 0.9608 \).

SAG beats two lower bounds:

- Stochastic gradient bound.
- Full gradient bound (for appropriate \( L, \mu, \) and \( N \)).

- \( O(1/k) \) rate in convex case (vs. \( O(1/\sqrt{k}) \) for SG methods).

Algorithm is **adaptive to** \( \mu \) around optimum.

Can improve \( C \) by clever initialization of \( x^0 \) and \( y_i^0 \).
Experiments with SAG algorithm

- rcv1 data set \((n = 20242, \ p = 47236)\)
Experiments with SAG algorithm

- rcv1 data set \((n = 20242, \ p = 47236)\)
Reducing memory requirements

- SAG algorithm:
  \[ x^{t+1} = x^t - \frac{\alpha_t}{n} \sum_{i=1}^{N} y_i^t, \]
  where \( y_i^t \) is the last gradient computed on datapoint \( i \).

- Memory requirement: \( O(NP) \)
Reducing memory requirements

- **SAG algorithm:**
  \[ x^{t+1} = x^t - \frac{\alpha t}{n} \sum_{i=1}^{N} y_i^t, \]
  where \( y_i^t \) is the last gradient computed on datapoint \( i \).

- Memory requirement: \( O(NP) \)

- Smaller for structured models, e.g., linear models:
  - If \( f_i(x) = \ell(a_i^T x) \), then \( f'_i(x) = \ell'(a_i^T x) a_i \)
  - Memory requirement: \( O(N) \)
Reducing memory requirements

- SAG algorithm:

\[ x^{t+1} = x^t - \frac{\alpha_t}{n} \sum_{i=1}^{N} y_i^t, \]

where \( y_i^t \) is the last gradient computed on datapoint \( i \).

- Memory requirement: \( O(NP) \)

- Smaller for structured models, e.g., linear models:
  - If \( f_i(x) = \ell(a_i^\top x) \), then \( f_i'(x) = \ell'(a_i^\top x)a_i \)
  - Memory requirement: \( O(N) \)

- Smaller for unstructured models using batches.
Fast theoretical convergence using the ‘sum’ structure.
Conclusion and Open Problems

- Fast theoretical convergence using the ‘sum’ structure.
- Simple algorithm, empirically better than theory predicts.

Open problems:
- Large-scale distributed implementation.
- Reduce the memory requirements.
- Constrained and non-smooth problems.
- Non-uniform sampling and non-Euclidean metrics.

Thanks for coming!
Conclusion and Open Problems

- Fast theoretical convergence using the ‘sum’ structure.
- Simple algorithm, empirically better than theory predicts.
- Allows line-search and stopping criteria.

Open problems:
- Large-scale distributed implementation.
- Reduce the memory requirements.
- Constrained and non-smooth problems.
- Non-uniform sampling and non-Euclidean metrics.
Conclusion and Open Problems

- Fast theoretical convergence using the ‘sum’ structure.
- Simple algorithm, empirically better than theory predicts.
- Allows line-search and stopping criteria.
- Open problems:
  - Large-scale distributed implementation.
  - Reduce the memory requirements.
  - Constrained and non-smooth problems.
  - Non-uniform sampling and non-Euclidean metrics.
Conclusion and Open Problems

- Fast theoretical convergence using the ‘sum’ structure.
- Simple algorithm, empirically better than theory predicts.
- Allows line-search and stopping criteria.
- Open problems:
  - Large-scale distributed implementation.
  - Reduce the memory requirements.
  - Constrained and non-smooth problems.
  - Non-uniform sampling and non-Euclidean metrics.
- Thanks for coming!