Problem Setting

We study convex minimization problems:

$$\min_{x \in X} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).$$
We study convex minimization problems:

\[ \min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x). \]

Ubiquitous in machine learning, e.g.:

- Least-squares regression: \( g_i(x) = (a_i^T x - b_i)^2 \)
- Logistic regression: \( g_i(x) = \log(1 + \exp(-b_i a_i^T x)) \)
**Typical Approaches**

\[
\min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

**Batch:**
- Apply generic convex minimization methods to \( f \)
- E.g.: Gradient Descent, accelerated variants, L-BFGS, ...
Typical Approaches

\[
\min_{x \in X} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

**Stochastic:**

- Treat \( f \) as an expectation of simpler functions
- E.g.: Stochastic GD, accelerated and averaged variants, ...
Typical Approaches

\[
\min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

**Can we do better?**

- Spate of new methods to further exploit structure
- E.g.: SAG, SVRG, SDCA, ... 
- Show improvements over both Batch and Stochastic
This Work

\[
\min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

How much better can we do?

- New complexity model: Incremental First-Order Oracle (IFO)
- Worst-case lower bounds for any IFO algorithm
- Average-case analysis of algorithms
This Work

\[
\min_{x \in X} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

How much better can we do?

- New complexity model: Incremental First-Order Oracle (IFO)
- Worst-case lower bounds for any IFO algorithm
- Average-case analysis of algorithms
- Lots of open questions!
Formal Setup

\[
\min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

**Structural assumptions:**

\(g_i\) are convex in \(x\) on \(\mathcal{X} \subseteq \mathbb{R}^d\) and are \((L - \mu)\) smooth.
Formal Setup

\[
\min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

**Structural assumptions:**

- \(g_i\) are convex in \(x\) on \(\mathcal{X} \subseteq \mathbb{R}^d\) and are \((L - \mu)\) smooth.
- Regularization ensures, \(f\) is \(\mu\)-strongly convex.
Formal Setup

\[
\min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

Structural assumptions:

- \(g_i\) are convex in \(x\) on \(\mathcal{X} \subseteq \mathbb{R}^d\) and are \((L - \mu)\) smooth.

- Regularization ensures, \(f\) is \(\mu\)-strongly convex.

- \(f\) is \(L\)-smooth and \(\mu\)-strongly convex

- Class of all such functions denoted by \(\mathcal{F}_{n,L}(\mathcal{X})\)

- Condition number: \(\kappa = \frac{L}{\mu}\)
Incremental First-Order Oracle (IFO)

\[
\min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

**Definition (Incremental First-order Oracle (IFO))**

For a function \( f \) of the considered form, the Incremental First-order Oracle (IFO) takes as input a point \( x \in \mathcal{X} \) and index \( i \in \{1, 2, \ldots, n\} \) and returns the pair \((g_i(x), \nabla g_i(x))\).
\[
\min_{x \in X} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

**Observations:**
- IFO algorithms access \( f \) only through IFO
Incremental First-Order Oracle (IFO)

\[
\min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

Observations:
- IFO algorithms access \( f \) only through IFO
- Admits many common algorithms
Incremental First-Order Oracle (IFO)

\[
\min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

Observations:
- IFO algorithms access \( f \) only through IFO
- Admits many common algorithms

Batch optimization:
- For \( i = 1, 2, \ldots, n \), query IFO with \((i, x_t)\)
- Compute \( \nabla f(x_t) = \mu x_t + \frac{1}{n} \sum_{i=1}^{n} \nabla g_i(x_t) \)
- Update \( x_t \) to \( x_{t+1} \) using \( \nabla f(x_t) \)
Incremental First-Order Oracle (IFO)

\[
\min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

**Observations:**
- IFO algorithms access \( f \) only through IFO
- Admits many common algorithms

**Stochastic optimization:**
- Pick \( i \) uniformly at random, query IFO with \((i, x_t)\)
- Form stochastic gradient of \( f \): \( \mu x_t + \nabla g_i(x_t) \)
- Update \( x_t \) to \( x_{t+1} \) using stochastic gradient
\[
\min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

Observations:
- IFO algorithms access \( f \) only through IFO
- Admits many common algorithms

New IFO Algorithms:
- SAG, SVRG, SAGA, … all implementable using IFO
**Incremental First-Order Oracle (IFO)**

\[
\min_{x \in \mathcal{X}} f(x), \quad \text{where} \quad f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x).
\]

**Observations:**
- IFO algorithms access \( f \) only through IFO
- Admits many common algorithms

**New IFO Algorithms:**
- SAG, SVRG, SAGA, \ldots all implementable using IFO
- Dual co-ordinate ascent methods like SDCA, ASDCA, SPDC not implementable using IFO
Consider an IFO algorithm that guarantees \( \| x_f^* - x_K \| \leq \epsilon \| x_f^* \| \) for any \( \epsilon < 1 \) and for all \( f \in \mathcal{F}_{n,L}^{\mu}(\ell_2) \). Then there is a function \( f \in \mathcal{F}_{n,L}^{\mu}(\ell_2) \) on which the algorithm must perform at least
\[
K = \Omega \left( n + \sqrt{n \left( \frac{L}{\mu} - 1 \right) \log(1/\epsilon)} \right) \text{ IFO calls.}
\]
**Theorem**

Consider an IFO algorithm that guarantees $\|x_f^* - x_K\| \leq \epsilon \|x_f^*\|$ for any $\epsilon < 1$ and for all $f \in \mathcal{F}_{\mu,L}(\ell_2)$. Then there is a function $f \in \mathcal{F}_{\mu,L}(\ell_2)$ on which the algorithm must perform at least

$$K = \Omega \left( n + \sqrt{n \left( \frac{L}{\mu} - 1 \right) \log(1/\epsilon)} \right)$$

IFO calls.

**Remarks:**

- $n \to \infty$ approaches stochastic optimization, lower bound of $\Omega(1/\epsilon)$
- $n = 1$ is batch optimization, matches previous lower bound (Nermiovytsky and Yudin, 1983)
**Lower Bound**

**Theorem**

Consider an IFO algorithm that guarantees $\|x_f^* - x_K\| \leq \epsilon \|x_f^*\|$ for any $\epsilon < 1$ and for all $f \in \mathcal{F}_{n, L}^{\mu, L}(\ell_2)$. Then there is a function $f \in \mathcal{F}_{n, L}^{\mu, L}(\ell_2)$ on which the algorithm must perform at least

$$K = \Omega \left( n + \sqrt{n \left( \frac{L}{\mu} - 1 \right) \log(1/\epsilon)} \right)$$

IFO calls.

**Remarks:**

- $n \to \infty$ approaches stochastic optimization, lower bound of $\Omega(1/\epsilon)$
- $n = 1$ is batch optimization, matches previous lower bound (Nemirovsky and Yudin, 1983)
- One oracle call involves one data point in typical setups
- IFO complexity roughly measures number of data points touched
Comparison with upper bounds

**Batch complexity:** iteration complexity divided by \( n \), typically number of passes over data

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Batch complexity</th>
<th>Adaptive?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower bound</td>
<td>( O\left(1 + \sqrt{\frac{L - \mu}{\mu n}} \log\frac{1}{\epsilon}\right) )</td>
<td></td>
</tr>
</tbody>
</table>

Note: \( L_f, \mu_f \) are smoothness and strong convexity constants of \( f \), \( L_f \leq L \) and \( \mu_f \geq \mu \).
Batch complexity: iteration complexity divided by \( n \), typically number of passes over data

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Batch complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower bound</td>
<td>( O \left( 1 + \sqrt{\frac{L - \mu}{\mu n}} \log \frac{1}{\epsilon} \right) )</td>
</tr>
<tr>
<td>Batch</td>
<td>( \tilde{O} \left( \sqrt{\frac{L_f}{\mu_f}} \log \frac{1}{\epsilon} \right) ) to ( \mu_f, L_f )</td>
</tr>
</tbody>
</table>

**Note:** \( L_f, \mu_f \) are smoothness and strong convexity constants of \( f \),

\[ L_f \leq L \quad \text{and} \quad \mu_f \geq \mu. \]
Comparison with upper bounds

**Batch complexity:** iteration complexity divided by $n$, typically number of passes over data

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Batch complexity</th>
<th>Adaptive?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower bound</td>
<td>$\mathcal{O}\left(1 + \sqrt{\frac{L - \mu}{\mu n}} \log \frac{1}{\epsilon}\right)$</td>
<td></td>
</tr>
<tr>
<td>Batch</td>
<td>$\tilde{\mathcal{O}}\left(\sqrt{\frac{L_f}{\mu_f}} \log \frac{1}{\epsilon}\right)$</td>
<td>to $\mu_f, L_f$</td>
</tr>
<tr>
<td>SAG, SVRG</td>
<td>$\tilde{\mathcal{O}}\left(1 + \frac{L}{\mu_f n} \log \frac{1}{\epsilon}\right)$</td>
<td>to $\mu_f$</td>
</tr>
</tbody>
</table>

**Note:** $L_f, \mu_f$ are smoothness and strong convexity constants of $f$, $L_f \leq L$ and $\mu_f \geq \mu$. 
Comparison with upper bounds

**Batch complexity:** iteration complexity divided by \( n \), typically number of passes over data

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Batch complexity</th>
<th>Adaptive?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower bound</td>
<td>( \mathcal{O}\left(1 + \sqrt{\frac{L-\mu}{\mu n}} \log \frac{1}{\epsilon}\right) )</td>
<td></td>
</tr>
<tr>
<td>Batch</td>
<td>( \tilde{\mathcal{O}}\left(\sqrt{\frac{L_f}{\mu_f}} \log \frac{1}{\epsilon}\right) )</td>
<td>to ( \mu_f, L_f )</td>
</tr>
<tr>
<td>SAG, SVRG</td>
<td>( \tilde{\mathcal{O}}\left((1 + \frac{L}{\mu_f n}) \log \frac{1}{\epsilon}\right) )</td>
<td>to ( \mu_f )</td>
</tr>
<tr>
<td>ASDCA, SDPC</td>
<td>( \tilde{\mathcal{O}}\left((1 + \sqrt{\frac{L-\mu}{\mu n}}) \log \frac{1}{\epsilon}\right) )</td>
<td>no</td>
</tr>
</tbody>
</table>

**Note:** \( L_f, \mu_f \) are smoothness and strong convexity constants of \( f \),

\[
L_f \leq L \text{ and } \mu_f \geq \mu.
\]
Comparison with upper bounds (contd.)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Batch complexity</th>
<th>Adaptive?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower bound</td>
<td>$\mathcal{O}\left(1 + \sqrt{\frac{L-\mu}{\mu n}} \log \frac{1}{\epsilon}\right)$</td>
<td></td>
</tr>
<tr>
<td>Batch</td>
<td>$\tilde{\mathcal{O}}\left(\sqrt{\frac{L_f}{\mu_f}} \log \frac{1}{\epsilon}\right)$</td>
<td>to $\mu_f, L_f$</td>
</tr>
<tr>
<td>SAG, SVRG</td>
<td>$\tilde{\mathcal{O}}\left(\left(1 + \frac{L}{\mu_f n}\right) \log \frac{1}{\epsilon}\right)$</td>
<td>to $\mu_f$</td>
</tr>
<tr>
<td>ASDCA, SDPC</td>
<td>$\tilde{\mathcal{O}}\left(\left(1 + \sqrt{\frac{L-\mu}{\mu n}}\right) \log \frac{1}{\epsilon}\right)$</td>
<td>no</td>
</tr>
</tbody>
</table>

Remarks:
- ASDCA, SPDC are closest to lower bound, but are not IFO algorithms.
- SAG, SVRG improve upon Batch for ill-conditioned problems.
- Gap between upper and lower bounds, room for better results.
**Key idea:** Each $g_i$ acts on a disjoint subset of coordinates

Algorithm minimizes $f$ if and only if Algorithm minimizes each $g_i$ over its subset of coordinates
Proof Ideas (contd.)

Algorithm minimizes $f$

$\updownarrow$

Algorithm minimizes each $g_i$ over its subset of coordinates

- Can construct $g_i$ with condition number $n\kappa$ over its subset of coordinates
Algorithm minimizes $f$

Algorithm minimizes each $g_i$ over its subset of coordinates

- Can construct $g_i$ with condition number $n\kappa$ over its subset of coordinates
- Each $g_i$ is independently minimized by first-order oracle
- Require at least $\Omega(\sqrt{n\kappa \log(1/\epsilon)})$ iterations (Nemirovsky and Yudin, 1983)
Algorithm minimizes $f$

$\uparrow$

Algorithm minimizes each $g_i$ over its subset of coordinates

- Can construct $g_i$ with condition number $n\kappa$ over its subset of coordinates
- Each $g_i$ is independently minimized by first-order oracle
- Require at least $\Omega(\sqrt{n\kappa \log(1/\epsilon)})$ iterations (Nemirovsky and Yudin, 1983)
- Also need to query each $g_i$ at least once
Algorithm minimizes $f$

$\downarrow$

Algorithm minimizes each $g_i$ over its subset of coordinates

- Can construct $g_i$ with condition number $n\kappa$ over its subset of coordinates
- Each $g_i$ is independently minimized by first-order oracle
- Require at least $\Omega(\sqrt{n\kappa \log(1/\epsilon)})$ iterations (Nemirovsky and Yudin, 1983)
- Also need to query each $g_i$ at least once
- Glue the constructions together in the right way for overall lower bound
Consequences for ML

So far we have allowed:

- Arbitrarily complicated $g_i$ subject to smoothness and convexity
- No relationship amongst the $g_i$

In typical ML scenarios, $g_i$ are based on (subsets of) data points, all drawn from same distribution.
Example: Least-squares regression

\[ f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x) \quad \text{where} \quad g_i(x) = (a_i^T x - b_i)^2. \]

\[ a_i \in \mathbb{R}^d \text{ are drawn i.i.d. with } \|a_i\| = R \text{ and } \mathbb{E}[a_i a_i^T] = \Sigma \]
Example: Least-squares regression

\[
f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x) \quad \text{where} \quad g_i(x) = (a_i^T x - b_i)^2.
\]

\(a_i \in \mathbb{R}^d\) are drawn i.i.d. with \(\|a_i\| = R\) and \(\mathbb{E}[a_i a_i^T] = \Sigma\)

Let \(\lambda_{\max} = \lambda_{\max}(\Sigma), \lambda_{\min} = \lambda_{\min}(\Sigma)\)

Assume \(n\) is large enough
Example: Least-squares regression

\[ f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x) \quad \text{where} \quad g_i(x) = (a_i^T x - b_i)^2. \]

\[ a_i \in \mathbb{R}^d \text{ are drawn i.i.d. with } \|a_i\| = R \text{ and } \mathbb{E}[a_i a_i^T] = \Sigma \]

Let \( \lambda_{\text{max}} = \lambda_{\text{max}}(\Sigma), \lambda_{\text{min}} = \lambda_{\text{min}}(\Sigma) \)

Assume \( n \) is large enough

- **Smoothness of \( g_i \):** \( L = \mu + R^2 \)
Example: Least-squares regression

\[ f(x) = \frac{\mu}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^{n} g_i(x) \quad \text{where} \quad g_i(x) = (a_i^T x - b_i)^2. \]

\( a_i \in \mathbb{R}^d \) are drawn i.i.d. with \( \|a_i\| = R \) and \( \mathbb{E}[a_i a_i^T] = \Sigma \)

Let \( \lambda_{\text{max}} = \lambda_{\text{max}}(\Sigma), \lambda_{\text{min}} = \lambda_{\text{min}}(\Sigma) \)

Assume \( n \) is large enough

- **Smoothness of** \( g_i \): \( L = \mu + R^2 \)
- **Condition number for IFO**: \( \kappa = 1 + \frac{R^2}{\mu} \)
- **Condition number for Batch**: \( \kappa_f = O\left(\frac{\mu + \lambda_{\text{max}}}{\mu + \lambda_{\text{min}}}\right) \)
## Comparison of worst-case upper bounds

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Batch complexity</th>
<th>Adaptive?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower bound</td>
<td>$\mathcal{O}\left(1 + \sqrt{\frac{L-\mu}{\mu n}} \log \frac{1}{\epsilon}\right)$</td>
<td></td>
</tr>
<tr>
<td>Batch</td>
<td>$\tilde{\mathcal{O}}\left(\sqrt{\frac{L_f}{\mu_f}} \log \frac{1}{\epsilon}\right)$</td>
<td>to $\mu_f, L_f$</td>
</tr>
<tr>
<td>SAG, SVRG</td>
<td>$\tilde{\mathcal{O}}\left((1 + \frac{L}{\mu_f n}) \log \frac{1}{\epsilon}\right)$</td>
<td>to $\mu_f$</td>
</tr>
<tr>
<td>ASDCA, SDPC</td>
<td>$\tilde{\mathcal{O}}\left((1 + \sqrt{\frac{L-\mu}{\mu n}}) \log \frac{1}{\epsilon}\right)$</td>
<td>no</td>
</tr>
</tbody>
</table>
Comparison of methods for least-squares

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>( \kappa = \mathcal{O}(n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGM</td>
<td>( \tilde{O}\left(\sqrt{\kappa_f \log \frac{1}{\epsilon}}\right) )</td>
</tr>
<tr>
<td>SAG, SVRG</td>
<td>( \tilde{O}\left(\log \frac{1}{\epsilon}\right) )</td>
</tr>
<tr>
<td>ASDCA, SPDC</td>
<td>( \tilde{O}\left(\log \frac{1}{\epsilon}\right) )</td>
</tr>
</tbody>
</table>

\[
\kappa = 1 + \frac{R^2}{\mu} \quad \kappa_f = \mathcal{O}\left(\frac{(\mu + \lambda_{\text{max}})}{(\mu + \lambda_{\text{min}})}\right)
\]

- New methods still improve over AGM in average case
Comparison of methods for least-squares

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$\kappa = \mathcal{O}(n)$</th>
<th>$\kappa \gg n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGM</td>
<td>$\tilde{\mathcal{O}}\left(\sqrt{\kappa_f \log \frac{1}{\epsilon}}\right)$</td>
<td>$\tilde{\mathcal{O}}\left(\sqrt{\kappa_f \log \frac{1}{\epsilon}}\right)$</td>
</tr>
<tr>
<td>SAG, SVRG</td>
<td>$\tilde{\mathcal{O}}\left(\log \frac{1}{\epsilon}\right)$</td>
<td>$\tilde{\mathcal{O}}\left(\log \frac{1}{\epsilon}\right)$</td>
</tr>
<tr>
<td>ASDCA, SPDC</td>
<td>$\tilde{\mathcal{O}}\left(\log \frac{1}{\epsilon}\right)$</td>
<td>$\tilde{\mathcal{O}}\left(\sqrt{\kappa \log \frac{1}{\epsilon}}\right)$</td>
</tr>
</tbody>
</table>

$$\kappa = 1 + \frac{R^2}{\mu} \quad \kappa_f = \mathcal{O}\left(\frac{(\mu + \lambda_{\text{max}})}{(\mu + \lambda_{\text{min}})}\right)$$

- New methods still improve over AGM in average case
- **Worst-case and average case can be rather different**
- Adaptivity to $\mu_f$ (and $L_f$ if possible) crucial
Conclusions

- New lower bound, no matching upper bound yet
- IFO algorithms can be better in worst and average cases
Conclusions

- New lower bound, no matching upper bound yet
- IFO algorithms can be better in worst and average cases
- Matching upper bounds?
- What do we get for test error?
Thank You