First-order methods
(Optml++ Meeting 2)

Suvrit Sra
Massachusetts Institute of Technology

OPTML++, Fall 2015
Outline

- Lect 1: Recap on convexity
- Lect 1: Recap on duality, optimality
- First-order optimization algorithms
- Proximal methods, operator splitting
Descent methods

$$\min_x f(x)$$
Descent methods

$$\min_x f(x)$$

$$x_k$$

$$x_{k+1}$$

$$x^*$$

$$\nabla f(x^*) = 0$$
Descent methods
Descent methods

∇f(x) – ∇f(x)

Suvrit Sra (MIT) Optimization for ML and beyond: OPTML++
Descent methods

\[ \nabla f(x) \]

\[ x - \alpha \nabla f(x) \]

\[ -\nabla f(x) \]

\[ x - \delta \nabla f(x) \]
Descent methods

\[ \nabla f(x) \]

\[ x - \alpha \nabla f(x) \]

\[ x + \alpha_2 d \]

\[ x - \delta \nabla f(x) \]

\[ -\nabla f(x) \]
Algorithm

1. Start with some guess $x^0$;
2. For each $k = 0, 1, \ldots$
   - $x^{k+1} \leftarrow x^k + \alpha_k d^k$
   - Check when to stop (e.g., if $\nabla f(x^{k+1}) = 0$)
Gradient methods

\[ x^{k+1} = x^k + \alpha_k d^k, \quad k = 0, 1, \ldots \]

- **stepsize** \( \alpha_k \geq 0 \), usually ensures \( f(x^{k+1}) < f(x^k) \)
Gradient methods

\[ x^{k+1} = x^k + \alpha_k d^k, \quad k = 0, 1, \ldots \]

- **stepsize** \( \alpha_k \geq 0 \), usually ensures \( f(x^{k+1}) < f(x^k) \)
- **Descent direction** \( d^k \) satisfies

\[ \langle \nabla f(x^k), d^k \rangle < 0 \]
Gradient methods

\[ x^{k+1} = x^k + \alpha_k d^k, \quad k = 0, 1, \ldots \]

- **stepsize** \( \alpha_k \geq 0 \), usually ensures \( f(x^{k+1}) < f(x^k) \)
- **Descent direction** \( d^k \) satisfies
  \[ \langle \nabla f(x^k), d^k \rangle < 0 \]

Numerous ways to select \( \alpha_k \) and \( d^k \)
Gradient methods

\[ x^{k+1} = x^k + \alpha_k d^k, \quad k = 0, 1, \ldots \]

- **stepsize** \( \alpha_k \geq 0 \), usually ensures \( f(x^{k+1}) < f(x^k) \)
- **Descent direction** \( d^k \) satisfies
  \[
  \langle \nabla f(x^k), d^k \rangle < 0
  \]

Numerous ways to select \( \alpha_k \) and \( d^k \)

Usually methods **seek** monotonic descent

\[ f(x^{k+1}) < f(x^k) \]
Gradient methods – direction

\[ x^{k+1} = x^k + \alpha_k d^k, \quad k = 0, 1, \ldots \]

Different choices of direction \( d^k \)

- **Scaled gradient**: \( d^k = -D^k \nabla f(x^k), \quad D^k \succ 0 \)
- **Newton’s method**: \( (D^k = [\nabla^2 f(x^k)]^{-1}) \)
- **Quasi-Newton**: \( D^k \approx [\nabla^2 f(x^k)]^{-1} \)
- **Steepest descent**: \( D^k = I \)
- **Diagonally scaled**: \( D^k \) diagonal with \( D^k_{ii} \approx \left( \frac{\partial^2 f(x^k)}{(\partial x_i)^2} \right)^{-1} \)
- **Discretized Newton**: \( D^k = [H(x^k)]^{-1}, \quad H \) via finite-diff.
Gradient methods – direction

\[ x^{k+1} = x^k + \alpha_k d^k, \quad k = 0, 1, \ldots \]

- Different choices of direction \( d^k \)

  - **Scaled gradient:** \( d^k = -D^k \nabla f(x^k), \; D^k \succ 0 \)
  - **Newton’s method:** \( (D^k = [\nabla^2 f(x^k)]^{-1}) \)
  - **Quasi-Newton:** \( D^k \approx [\nabla^2 f(x^k)]^{-1} \)
  - **Steepest descent:** \( D^k = I \)
  - **Diagonally scaled:** \( D^k \) diagonal with \( D^k_{ii} \approx \left( \frac{\partial^2 f(x^k)}{\partial x_i^2} \right)^{-1} \)
  - **Discretized Newton:** \( D^k = [H(x^k)]^{-1}, \; H \) via finite-diff.
  - . . .

**Exercise:** Verify that \( \langle \nabla f(x^k), d^k \rangle < 0 \) for above choices
Gradient methods – stepsize

▶ **Exact:** \( \alpha_k := \arg \min_{\alpha \geq 0} f(x^k + \alpha d^k) \)
Gradient methods – stepsize

- **Exact:** \( \alpha_k := \arg\min_{\alpha \geq 0} f(x^k + \alpha d^k) \)

- **Limited min:** \( \alpha_k = \arg\min_{0 \leq \alpha \leq s} f(x^k + \alpha d^k) \)

- **Armijo-rule.** Given fixed scalars, \( s, \beta, \sigma \) with \( 0 < \beta < 1 \) and \( 0 < \sigma < 1 \) (chosen experimentally). Set \( \alpha_k = \beta m s \), where we try \( \beta m s \) for \( m = 0, 1, ... \) until sufficient descent \( f(x^k) - f(x^k + \beta m s d^k) \geq -\sigma \beta m s \langle \nabla f(x^k), d^k \rangle \). If \( \langle \nabla f(x^k), d^k \rangle < 0 \), stepsize guaranteed to exist. Usually, \( \sigma \) small \( \in [10^{-5}, 0.1] \), while \( \beta \) from \( \frac{1}{2} \) to \( \frac{1}{10} \) depending on how confident we are about initial stepsize \( s \).

- **Constant:** \( \alpha_k = \frac{1}{L} \) (for suitable value of \( L \))

- **Diminishing:** \( \alpha_k \to 0 \) but \( \sum_k \alpha_k = \infty \).
Gradient methods – stepsize

- **Exact:** \( \alpha_k := \arg\min_{\alpha \geq 0} f(x^k + \alpha d^k) \)

- **Limited min:** \( \alpha_k = \arg\min_{0 \leq \alpha \leq s} f(x^k + \alpha d^k) \)

- **Armijo-rule.** Given fixed scalars, \( s, \beta, \sigma \) with \( 0 < \beta < 1 \) and \( 0 < \sigma < 1 \) (chosen experimentally). Set

\[
\alpha_k = \beta^{m_k} s,
\]

where we **try** \( \beta^m s \) for \( m = 0, 1, \ldots \) until **sufficient descent**

\[
f(x^k) - f(x + \beta^m s d^k) \geq -\sigma \beta^m s \langle \nabla f(x^k), d^k \rangle
\]
Gradient methods – stepsize

- **Exact:** \( \alpha_k := \arg\min_{\alpha \geq 0} f(x_k + \alpha d^k) \)
- **Limited min:** \( \alpha_k = \arg\min_{0 \leq \alpha \leq s} f(x_k + \alpha d^k) \)
- **Armijo-rule.** Given fixed scalars, \( s, \beta, \sigma \) with \( 0 < \beta < 1 \) and \( 0 < \sigma < 1 \) (chosen experimentally). Set \( \alpha_k = \beta^{m_k} s \), where we try \( \beta^m s \) for \( m = 0, 1, \ldots \) until sufficient descent

\[
f(x^k) - f(x + \beta^m s d^k) \geq -\sigma \beta^m s \langle \nabla f(x^k), d^k \rangle
\]

If \( \langle \nabla f(x^k), d^k \rangle < 0 \), stepsize guaranteed to exist.
Gradient methods – stepsize

- **Exact:** \( \alpha_k := \arg\min_{\alpha \geq 0} f(x^k + \alpha d^k) \)

- **Limited min:** \( \alpha_k = \arg\min_{0 \leq \alpha \leq s} f(x^k + \alpha d^k) \)

- **Armijo-rule.** Given fixed scalars, \( s, \beta, \sigma \) with \( 0 < \beta < 1 \) and \( 0 < \sigma < 1 \) (chosen experimentally). Set \( \alpha_k = \beta^{m_k} s \), where we try \( \beta^m s \) for \( m = 0, 1, \ldots \) until sufficient descent

\[
f(x^k) - f(x + \beta^m s d^k) \geq -\sigma \beta^m s \langle \nabla f(x^k), d^k \rangle
\]

If \( \langle \nabla f(x^k), d^k \rangle < 0 \), stepsize guaranteed to exist

Usually, \( \sigma \) small \( \in [10^{-5}, 0.1] \), while \( \beta \) from 1/2 to 1/10 depending on how confident we are about initial stepsize \( s \).

- **Constant:** \( \alpha_k = 1/L \) (for suitable value of \( L \))

- **Diminishing:** \( \alpha_k \to 0 \) but \( \sum_k \alpha_k = \infty \).
Gradient methods – nonmonotonic steps

- Stepsize computation can be expensive
- Convergence analysis depends on monotonic descent

\[
\begin{align*}
\alpha_k &= \frac{\langle u_k, v_k \rangle}{\|v_k\|^2}, \\
\alpha_k &= \frac{\|u_k\|^2}{\langle u_k, v_k \rangle}, \\
u_k &= x_k - x_{k-1}, \\
v_k &= \nabla f(x_k) - \nabla f(x_{k-1})
\end{align*}
\]
Gradient methods – nonmonotonic steps

- Stepsize computation can be expensive
- Convergence analysis depends on monotonic descent
- Give up search for stepsizes
- Use closed-form formulae for stepsizes
- Don’t insist on monotonic descent?
  (e.g., diminishing stepsizes do not give monotonic descent)

\[
x_{k+1} = x_k - \alpha_k \nabla f(x_k), \quad k = 0, 1, \ldots
\]

\[
\alpha_k = \frac{\langle u_k, v_k \rangle}{\|v_k\|^2}, \quad \alpha_k = \frac{\|u_k\|^2}{\langle u_k, v_k \rangle}
\]

\[
u_k = x_k - x_{k-1}, \quad v_k = \nabla f(x_k) - \nabla f(x_{k-1})
\]
Gradient methods – nonmonotonic steps

- Stepsize computation can be expensive
- Convergence analysis depends on monotonic descent
- Give up search for stepsizes
- Use closed-form formulae for stepsizes
- Don’t insist on monotonic descent?
  (e.g., diminishing stepsizes do not give monotonic descent)

Barzilai & Borwein stepsizes

\[ x^{k+1} = x^k - \alpha^k \nabla f(x^k), \quad k = 0, 1, \ldots \]
Gradient methods – nonmonotonic steps

- Stepsize computation can be expensive
- Convergence analysis depends on monotonic descent
- Give up search for stepsizes
- Use closed-form formulae for stepsizes
- Don’t insist on monotonic descent?
- (e.g., diminishing stepsizes do not give monotonic descent)

**Barzilai & Borwein stepsizes**

\[ x^{k+1} = x^k - \alpha^k \nabla f(x^k), \quad k = 0, 1, \ldots \]

\[ \alpha_k = \frac{\langle u^k, v^k \rangle}{\| v^k \|^2}, \quad \alpha_k = \frac{\| u^k \|^2}{\langle u^k, v^k \rangle} \]

\[ u^k = x^k - x^{k-1}, \quad v^k = \nabla f(x^k) - \nabla f(x^{k-1}) \]
Least-squares
Nonnegative least squares

$$\min \frac{1}{2} \|Ax - b\|^2 + [x \geq 0]$$

intensities, concentrations, frequencies, ... 

Applications

Machine learning  Physics
Statistics  Bioinformatics
Image Processing  Remote Sensing
Computer Vision  Engineering
Medical Imaging  Inverse problems
Astronomy  Finance
NNLS: $\|Ax - b\|^2$ s.t. $x \geq 0$

Unconstrained solution

Solve $\nabla f(x) = 0 \implies x_{uc} = (A^T A)^{-1} A^T b$

Cannot just truncate $x = (x_{uc})^+$

$x \geq 0$ makes problem trickier as problem size $\uparrow$
Solving NNLS scalably

\[ x \leftarrow (x - \alpha \nabla f(x))^+ \]

Good choice of \( \alpha \) crucial

- Backtracking line-search
- Armijo
- and many others
Solving NNLS scalably

\[ x \leftarrow (x - \alpha \nabla f(x))^+ \]

Good choice of \( \alpha \) crucial

- Backtracking line-search
- Armijo
- and many others

Too slow!
### NNLS: long studied problem

<table>
<thead>
<tr>
<th>Method</th>
<th>Remarks</th>
<th>Scalability</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>NNLS (1976)</td>
<td>MATLAB default</td>
<td>poor</td>
<td>high</td>
</tr>
<tr>
<td>FNNLS (1989)</td>
<td>fast NNLS</td>
<td>poor</td>
<td>high</td>
</tr>
<tr>
<td>LBFGS-B (1997)</td>
<td>famous solver</td>
<td>fair</td>
<td>medium</td>
</tr>
<tr>
<td>TRON (1999)</td>
<td>TR newton</td>
<td>poor</td>
<td>high</td>
</tr>
<tr>
<td>SPG (2000)</td>
<td>spectral proj</td>
<td>fair+</td>
<td>medium</td>
</tr>
<tr>
<td>ASA (2006)</td>
<td>prev state-of-art</td>
<td>fair+</td>
<td>medium</td>
</tr>
<tr>
<td>SBB (2011)</td>
<td>subspace BB steps</td>
<td>very good</td>
<td>medium</td>
</tr>
</tbody>
</table>
Spectacular failure of projection

\[ x' = (x - \alpha \nabla f(x))^+ \]
Rescue: occasional line-search?

Mix BB-step with linesearch
Can we completely avoid linesearch?

Do not use all coordinates to compute $\alpha$!

"Subspace-BB" (SBB)
Kim, Sra, Dhillon (OMS, 2011)

Identify **fixed** variables
(those likely to satisfy $x_i = 0$)
Compute $\alpha$ using **free** variables
Most crucial step!
Can we completely avoid linesearch?

Do not use all coordinates to compute $\alpha$!

“Subspace-BB” (SBB)
Kim, Sra, Dhillon (OMS, 2011)

Identify **fixed** variables
(those likely to satisfy $x_i = 0$)
Compute $\alpha$ using **free** variables
Most crucial step!

SBB convergence theorem
Global rate – open problem
Empirically great!
SBB: simplicity and scalability

![Graph showing the objective function value and running time for different methods: Naive BB+Projxn, Naive BB + Linesearch, and SBB. The x-axis represents running time in seconds, and the y-axis represents the objective function value on a logarithmic scale. The graph compares the performance of these methods over time, with SBB generally showing better scalability and simplicity.](image)
### Numerical result

| Algorithm             | Time   | $||Ax - b||^2$ | Convg. tol. |
|-----------------------|--------|---------------|-------------|
| LBFGS-B (FORTRAN)     | 19000s | 20.2          | 1.0E-03     |
| SPG (FORTRAN)         | 8600s  | 20.5          | 3.8E-01     |
| ASA (C++)             | 1001s  | 24.5          | 4.8e-02     |

“medium” $20,000 \times 1,350,000$ matrix
### Numerical result

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Time</th>
<th>$|Ax - b|^2$</th>
<th>Convg. tol.</th>
</tr>
</thead>
<tbody>
<tr>
<td>LBFGS-B (FORTRAN)</td>
<td>19000s</td>
<td>20.2</td>
<td>1.0E-03</td>
</tr>
<tr>
<td>SPG (FORTRAN)</td>
<td>8600s</td>
<td>20.5</td>
<td>3.8E-01</td>
</tr>
<tr>
<td>ASA (C++)</td>
<td>1001s</td>
<td>24.5</td>
<td>4.8e-02</td>
</tr>
<tr>
<td>SBB (MATLAB)</td>
<td>201s</td>
<td>21.2</td>
<td>8.7E-03</td>
</tr>
</tbody>
</table>

“medium” $20,000 \times 1,350,000$ matrix
Assumption: Lipschitz continuous gradient; denoted $f \in C^1_L$

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2$$
Back to gradient-descent

**Assumption:** Lipschitz continuous gradient; denoted $f \in C^1_L$

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

- ♣ Gradient vectors of closeby points are close to each other
- ♣ Objective function has “bounded curvature”
- ♣ Speed at which gradient varies is bounded
Back to gradient-descent

**Assumption:** Lipschitz continuous gradient; denoted $f \in C^1_L$

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

♣ Gradient vectors of closeby points are close to each other
♣ Objective function has “bounded curvature”
♣ Speed at which gradient varies is bounded

**Lemma** (Descent). Let $f \in C^1_L$. Then,

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|y - x\|_2^2$$

**Theorem** Let $f \in C^1_L$ and $\{x^k\}$ be sequence generated as above, with $\alpha_k = 1/L$. Then, $f(x^{k+1}) - f(x^*) = O(1/k)$. 
Linear convergence

Assumption: **Strong convexity**; denote $f \in S^{1}_{L,\mu}$

$$f(x) \geq f(y) + \langle \nabla f(y), x - y \rangle + \frac{\mu}{2} \| x - y \|_{2}^{2}$$

- Setting $\alpha_{k} = 2/(\mu + L)$ yields linear rate ($\mu > 0$)
Theorem. If $f \in S^1_{L,\mu}$, $0 < \alpha < 2/(L + \mu)$, then the gradient method generates a sequence $\{x^k\}$ that satisfies

$$\|x^k - x^*\|_2^2 \leq \left(1 - \frac{2\alpha \mu L}{\mu + L}\right)^k \|x^0 - x^*\|_2.$$

Moreover, if $\alpha = 2/(L + \mu)$ then

$$f(x^k) - f^* \leq \frac{L}{2} \left(\frac{\kappa - 1}{\kappa + 1}\right)^{2k} \|x^0 - x^*\|_2^2,$$

where $\kappa = L/\mu$ is the condition number.
Gradient methods – lower bounds

\[ x^{k+1} = x^k - \alpha_k \nabla f(x^k) \]

**Theorem** Lower bound I (Nesterov) For any \( x^0 \in \mathbb{R}^n \), and \( 1 \leq k \leq \frac{1}{2}(n - 1) \), there is a smooth \( f \), s.t.

\[
f(x^k) - f(x^*) \geq \frac{3L\|x^0 - x^*\|_2^2}{32(k + 1)^2}\]

**Theorem** Lower bound II (Nesterov). For class of smooth, strongly convex, i.e., \( S_\infty^\mu \), \( \mu > 0, \kappa > 1 \)

\[
f(x^k) - f(x^*) \geq \frac{\mu}{2} (\sqrt{\kappa} - 1/\sqrt{\kappa} + 1)^2 k \|x^0 - x^*\|_2^2.
\]
Gradient methods – lower bounds

\[ x^{k+1} = x^k - \alpha_k \nabla f(x^k) \]

**Theorem** Lower bound I (Nesterov) For any \( x^0 \in \mathbb{R}^n \), and \( 1 \leq k \leq \frac{1}{2}(n - 1) \), there is a smooth \( f \), s.t.

\[
 f(x^k) - f(x^*) \geq \frac{3L\|x^0 - x^*\|^2_2}{32(k + 1)^2}
\]

**Theorem** Lower bound II (Nesterov). For class of smooth, strongly convex, i.e., \( S_{L,\mu}^{\infty} \) (\( \mu > 0, \kappa > 1 \))

\[
 f(x^k) - f(x^*) \geq \frac{\mu}{2} \left( \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^{2k} \|x^0 - x^*\|^2_2.
\]