EE270
Large scale matrix computation, optimization and learning

Instructor: Mert Pilanci
Stanford University

Tuesday, Feb 18 2020
Randomized Linear Algebra and Optimization
Lecture 14: Second-Order Optimization Algorithms, Randomized Newton’s Method, Strong Convexity and Acceleration
Recap: Gradient Descent with momentum

- $\min_x f(x)$
- Gradient Descent with Momentum

$$x_{t+1} = x_t - \mu_t \nabla f(x_t) + \beta_t (x_t - x_{t-1})$$

- the term $\beta_t (x_t - x_{t-1})$ is referred to as **momentum**
Computational complexity

- Gradient Descent \((\beta = 0)\) total computational cost 
  \(\kappa nd \log\left(\frac{1}{\epsilon}\right)\) for \(\epsilon\) accuracy

- Gradient Descent with Momentum total computational cost 
  \(\sqrt{\kappa nd} \log\left(\frac{1}{\epsilon}\right)\) for \(\epsilon\) accuracy

- we need to know eigenvalues of \(A^T A\) to find optimal step-sizes
Computational complexity

- Gradient Descent ($\beta = 0$) total computational cost $\kappa nd \log(\frac{1}{\epsilon})$ for $\epsilon$ accuracy
- Gradient Descent with Momentum total computational cost $\sqrt{\kappa nd} \log(\frac{1}{\epsilon})$ for $\epsilon$ accuracy
- We need to know eigenvalues of $A^T A$ to find optimal step-sizes
- Conjugate Gradient doesn’t require the eigenvalues explicitly and results in $\sqrt{\kappa nd} \log(\frac{1}{\epsilon})$ operations
Newton’s Method

- Suppose $f$ is twice differentiable, and consider a second order Taylor approximation at a point $x_t$

  \[ f(y) \approx f(x_t) + \nabla f(x_t)^T (y - x_t) + \frac{1}{2} (y - x^t) \nabla^2 f(x^t)(y - x^t) \]

- and minimize the approximation

- \[ x_{t+1} = x_t - \mu_t \left( \nabla^2 f(x) \right)^{-1} \nabla f(x) \]

- for minimizing functions $f(Ax)$ where $A \in \mathbb{R}^{n \times d}$

- complexity $O(nd^2)$ to form the Hessian and $O(d^3)$ to invert

- or alternatively $O(nd^2)$ for factorizing the Hessian
Newton’s Method in one dimension
Newton’s Method for least squares converges in one step

Consider

$$\min_x \underbrace{\frac{1}{2} \|Ax - b\|_2^2}_{f(x)}$$

- gradient $\nabla f(x) = A^T(Ax - b)$
- Hessian $\nabla^2 f(x) = A^TA$
- Gradient Descent:

$$x_{t+1} = x_t - \mu A^T(Ax_t - b)$$

- Newton’s Method:

$$x_{t+1} = x_t - \mu (A^TA)^{-1} A^T(Ax_t - b)$$

- fixed step size $\mu_t = \mu$
Newton’s Method with Random Projection

- Randomized Newton’s Method:
  \[ x_{t+1} = x_t - \mu (A^T S^T S A)^{-1} A^T (Ax_t - b) \]

- fixed step size \( \mu_t = \mu \)
- computational cost:
  - \( O(nd \log n) \) to form \( SA \) using Fast Johnson Lindenstrauss Transform and \( O(d^3) \) to invert \( (A^T S^T S A)^{-1} \)
  - alternatively \( O(md^2) \) to factorize \( SA \)
Analyzing Newton’s Method with Random Projection

- Randomized Newton’s Method:

\[ x_{t+1} = x_t - \mu (A^T S^T S A)^{-1} A^T (Ax_t - b) \]

- Define \( \Delta_t = A(x_t - x^*) \)
Analyzing Newton’s Method with Random Projection

Randomized Newton’s Method:

\[ x_{t+1} = x_t - \mu (A^T S^T S A)^{-1} A^T (A x_t - b) \]

Define \( \Delta_t = A(x_t - x^*) \)

\[ \Delta_{t+1} = \Delta_t - \mu A (A^T S^T S A)^{-1} A^T \Delta_t \]

after \( M \) iterations

\[ \Delta_M = (I - \mu A (A^T S^T S A)^{-1} A^T)^M \Delta_0 \]
Analyzing Newton’s Method with Random Projection

- Let $A = U\Sigma V^T$ be the Singular Value Decomposition of $A$
- $A(A^T S^T S A)^{-1} A^T = U(U^T S^T S U) U^T$
- $\Delta_M = (I - \mu U(U^T S^T S U)^{-1} U^T)^M \Delta_0$
- $\Delta_t \in \text{Range}(A)$ implies $UU^T \Delta_t = \Delta_t$ and $\|U^T \Delta_t\|_2 = \|\Delta_t\|_2$
Analyzing Newton’s Method with Random Projection

Let $A = U \Sigma V^T$ be the Singular Value Decomposition of $A$

$A(A^T S^T S A)^{-1} A^T = U (U^T S^T S U) U^T$

$\Delta_M = (I - \mu U (U^T S^T S U)^{-1} U^T)^M \Delta_0$

$\Delta_t \in \text{Range}(A)$ implies $UU^T \Delta_t = \Delta_t$ and $\|U^T \Delta_t\|_2 = \|\Delta_t\|_2$

$U^T \Delta_M = U^T (I - \mu U (U^T S^T S U)^{-1} U^T)^M U U^T \Delta_0$

Note that

$U^T (I - \mu U (U^T S^T S U)^{-1} U^T) = (I - \mu (U^T S^T S U)^{-1}) U^T$

$U^T \Delta_M = (I - \mu (U^T S^T S U)^{-1})^M U^T \Delta_0$
Analyzing Newton’s Method with Random Projection

- Let $A = U\Sigma V^T$ be the Singular Value Decomposition of $A$
- $A(A^T S^T S A)^{-1} A^T = U(U^T S^T S U)U^T$
- $\Delta_M = (I - \mu U(U^T S^T S U)^{-1} U^T)^M \Delta_0$
- $\Delta_t \in \text{Range}(A)$ implies $UU^T \Delta_t = \Delta_t$ and $\|U^T \Delta_t\|_2 = \|\Delta_t\|_2$
- $U^T \Delta_M = U^T (I - \mu U(U^T S^T S U)^{-1} U^T)^M U U^T \Delta_0$
- Note that $U^T (I - \mu U(U^T S^T S U)^{-1} U^T) = (I - \mu (U^T S^T S U)^{-1}) U^T$
- $U^T \Delta_M = (I - \mu (U^T S^T S U)^{-1})^M U U^T \Delta_0$
- $\|\Delta_M\|_2 \leq \sigma_{\text{max}} (I - \mu (U^T S^T S U)^{-1})^M \|\Delta_0\|_2$
- $\|\Delta_M\|_2 \leq \max_{i=1,...,d} |1 - \mu \lambda_i((U^T S^T S U)^{-1})|^M \|\Delta_0\|_2$
Eigenvalues of randomly projected matrices

- \( \lambda_i((U^T S^T SU)^{-1}) = \lambda_i(U^T S^T SU)^{-1} \)
- Approximate Matrix Multiplication for \( U^T U = I \)
  \[ \| U^T U - U^T S^T SU \|_F \leq \epsilon \] implies
  \[ \sigma_{\text{max}} (I - U^T S^T SU) \leq \epsilon \]
- which is identical to \( |1 - \lambda_i(U^T S^T SU)| \leq \epsilon \) \( \forall i = 1, \ldots, d \)
- All eigenvalues of \( U^T S^T SU \) are in the range \([1 - \epsilon, 1 + \epsilon]\)
Optimal step-size

- All eigenvalues of $U^T S^T S U$ are in the range $[1 - \epsilon, 1 + \epsilon]$
- All eigenvalues of $(U^T S^T S U)^{-1}$ are in the range $[\frac{1}{1-\epsilon}, \frac{1}{1+\epsilon}]$

\[
\|\Delta_M\|_2 \leq \max_{i=1,\ldots,d} \left| 1 - \mu \lambda_i((U^T S^T S U)^{-1}) \right|^M \|\Delta_0\|_2
\]  \hspace{1cm} (1)

\[
= \max \left( \left| 1 - \mu \frac{1}{1-\epsilon} \right|, \left| 1 - \mu \frac{1}{1+\epsilon} \right| \right)^M \|\Delta_0\|_2
\] \hspace{1cm} (2)

- optimal step-size that minimizes the upper-bound satisfies

\[
\left| 1 - \mu^* \frac{1}{1-\epsilon} \right| = \left| 1 - \mu^* \frac{1}{1+\epsilon} \right|
\]

- \[\mu^* = \frac{2}{\frac{1}{1-\epsilon} + \frac{1}{1+\epsilon}} = (1 - \epsilon)(1 + \epsilon)\]
Convergence rate

\[ \mu^* = \frac{2}{\frac{1}{1-\epsilon} + \frac{1}{1+\epsilon}} = (1 - \epsilon)(1 + \epsilon) \]

\[ \| \Delta_M \|_2 \leq \max \left( \left| 1 - \mu \frac{1}{1 - \epsilon} \right|, \left| 1 - \mu \frac{1}{1 + \epsilon} \right| \right)^M \| \Delta_0 \|_2 \]

(3)

\[ = \max (|1 - (1 + \epsilon)|, |1 - (1 - \epsilon)|)^M \| \Delta_0 \|_2 \]

(4)

\[ = \epsilon^M \| \Delta_0 \|_2 \]

(5)
Computational complexity

- For $\epsilon$ accuracy in the objective value, i.e., $\|A\hat{x} - Ax^*\|_2 \leq \epsilon$
- Gradient Descent ($\beta = 0$) total computational cost $\kappa nd \log\left(\frac{1}{\epsilon}\right)$
- Gradient Descent with Momentum total computational cost $\sqrt{\kappa nd} \log\left(\frac{1}{\epsilon}\right)$
- we need to know eigenvalues of $A^T A$ to find optimal step-sizes
Computational complexity

▷ For $\epsilon$ accuracy in the objective value, i.e., $\|A\hat{x} - Ax^*\|_2 \leq \epsilon$

▷ Gradient Descent ($\beta = 0$) total computational cost $\kappa nd \log\left(\frac{1}{\epsilon}\right)$

▷ Gradient Descent with Momentum total computational cost $\sqrt{\kappa nd} \log\left(\frac{1}{\epsilon}\right)$

▷ we need to know eigenvalues of $A^T A$ to find optimal step-sizes

▷ Conjugate Gradient doesn’t require the eigenvalues explicitly and results in $\sqrt{\kappa nd} \log\left(\frac{1}{\epsilon}\right)$ operations

▷ Randomized Newton Method ($m = \text{constant} \times d$) total computational cost $nd \log n + d^3 + nd\log\left(\frac{1}{\epsilon}\right)$
Strong convexity

A convex function $f$ is called strongly convex if there exists two positive constants $\beta_- \leq \beta_+$ such that

$$\beta_- \leq \lambda_i (\nabla^2 f(x)) \leq \beta_+$$

for every $x$ in the domain of $f$

Equivalent to

$$\lambda_{\min}(\nabla^2 f(x)) \geq \beta_-$$

$$\lambda_{\max}(\nabla^2 f(x)) \leq \beta_+$$
Gradient Descent for Strongly Convex Functions

\[ x_{t+1} = x_t - \mu_t \nabla f(x_t) \]

Suppose that \( f \) is strongly convex with parameters \( \beta_- \), \( \beta_+ \)

let \( f^* := \min_x f(x) \)

**Theorem**

Set constant step-size \( \mu_t = \frac{1}{\beta_+} \)

\[ f(x_{t+1}) - f^* \leq (1 - \frac{\beta_-}{\beta_+})(f(x_t) - f^*) \]

recursively applying we get

\[ f(x_M) - f^* \leq (1 - \frac{\beta_-}{\beta_+})^M (f(x_0) - f^*) \]
Gradient Descent for Strongly Convex Functions

\[ x_{t+1} = x_t - \mu \nabla f(x_t) \]

\[ \text{step-size } \mu = \frac{1}{\beta_+} \]

\[ f(x_M) - f^* \leq (1 - \frac{\beta_-}{\beta_+})^M (f(x_0) - f^*) \]

For optimizing functions \( f(Ax) \)

computational complexity \( O(\kappa nd \log(\frac{1}{\epsilon})) \)

where \( \kappa = \frac{\beta_+}{\beta_-} \)
Gradient Descent with Momentum (Heavy Ball Method) for Strongly Convex Functions

\[ x_{t+1} = x_t - \mu \nabla f(x_t) + \beta (x_t - x_{t-1}) \]

- step-size parameter \( \mu = \frac{4}{(\sqrt{\beta_+} + \sqrt{\beta_-})^2} \)
- momentum parameter \( \beta = \max \left( |1 - \sqrt{\mu \beta_-}|, |1 - \sqrt{\mu \beta_+}| \right)^2 \)
- For optimizing functions \( f(Ax) \)
  computational complexity \( O(\sqrt{\kappa} nd \log(\frac{1}{\epsilon})) \)
  where \( \kappa = \frac{\beta_+}{\beta_-} \)
Questions?