EE270
Large scale matrix computation, optimization and learning

Instructor : Mert Pilanci

Stanford University

Randomized Linear Algebra and Optimization
Lecture 13: Gradient Descent with Momentum and Preconditioning

Optimizing convex least squares cost

$A = U\Sigma V^T \quad A^\dagger = V\Sigma^\dagger U^T$

$x_{LS} = A^\dagger b$

$Ax_{LS} = A \cdot A^\dagger b$

$= P_{Range(A)} \cdot b$

$= UU^T b$

- Consider

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

$= (A^TA) \cdot x - A^Tb$

Another matrix B

$A \cdot A^\dagger B = P_{Range(A)} \cdot B$

Generalized

- gradient $\nabla f(x) = A^T(Ax - b)$ $O(nd)$ per iter
- Gradient Descent:

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

- fixed step size $\mu_t = \mu$

Compute once and store the $d \times d$ matrix

A^TA: $O(nd^2)$ + $d^2 \cdot T$

Sparse A: can fit A but not A^TA

not compute A^TA

$nd \cdot T$

if might not be possible to compute A^TA

load A $n \times d$

A^TA $d \times d$

Optimizing convex least squares cost

- Basic (in)equality method

  (1) x^* minimizes $f(x)$, hence $\nabla f(x^*) = A^T(Ax^* - b) = 0$

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

  (3) define error $\Delta_t = x_t - x^*$

Optimizing convex least squares cost

- Basic (in)equality method

  (1) x^* minimizes $f(x)$, hence $\nabla f(x^*) = A^T(Ax^* - b) = 0$

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

  (3) define error $\Delta_t = x_t - x^*$

- $\Delta_{t+1} = \Delta_t - \mu A^T A \Delta_t$

Optimizing convex least squares cost

$\frac{\lambda_+}{\lambda_-}$ large

$\frac{\lambda_+}{\lambda_-} \approx 1$

- run gradient descent M iterations, i.e., $t = 1, ..., M$
- $\Delta_M = (I - \mu A^T A)^M \Delta_0$
- $\|\Delta_M\|_2 \leq \sigma_{\max}\left((I - \mu A^T A)^M\right) \|\Delta_0\|_2$

  $\sigma_{\max}\left(I - \mu A^T A\right)^M = \max_{i=1,..,d} \left|1 - \mu\lambda_i(A^T A)\right|^M$

  where λ_i is the i-th eigenvalue in decreasing order

- Define

  λ_- as the smallest eigenvalue of $A^T A$

  λ_+ as the largest eigenvalue of $A^T A$

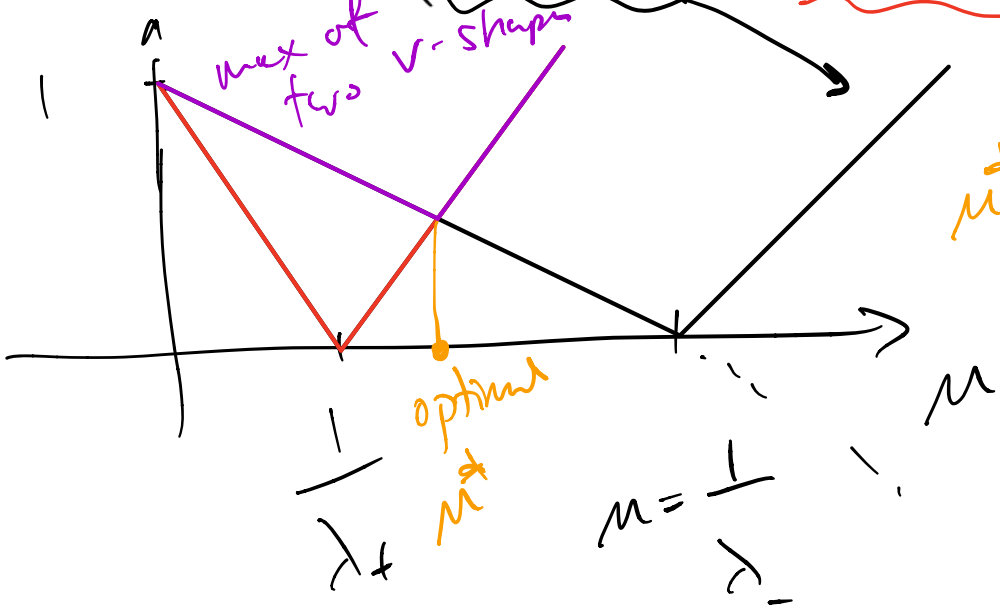
x^* unknown

$\max_{\|x^*\|_2 \leq R} \|\Delta_M\|_2 = \left(I - \mu A^T A\right)^M (x^*) =$

- $\max_{i=1,..,d} \left|1 - \mu\lambda_i(A^T A)\right| = \max\left(\left|1 - \mu\lambda_-\right|, \left|1 - \mu\lambda_+\right|\right)$
- optimal step size that minimizes above
- $\min_{\mu \geq 0} \max\left(\left|1 - \mu\lambda_-\right|, \left|1 - \mu\lambda_+\right|\right)$
- optimal $\mu = \mu^*$ satisfies $\left|1 - \mu^*\lambda_-\right| = \left|1 - \mu^*\lambda_+\right|$

  which implies $\mu^* = \frac{2}{\lambda_+ + \lambda_-}$

$$\min_{\mu \geq 0} \max\left(|1-\mu\lambda_-|, |1-\mu\lambda_+|\right)$$



$$\mu^* = \frac{1}{\frac{\lambda_+ - \lambda_-}{2}}$$

$1-\mu\lambda_1$ at optimum

$$|1-\mu\cdot\lambda_-| = |1-\mu\lambda_+|$$

$1-\mu\lambda_- = 1-\mu\lambda_+$ inconsistent $\lambda_- \neq \lambda_+$

$1-\mu\lambda_- = \mu\lambda_+ - 1 \Rightarrow \mu(\lambda_+ + \lambda_-) = 2 \Rightarrow \mu = \frac{2}{\lambda_+ + \lambda_-}$

Optimizing convex least squares cost

- Convergence rate using $\mu^* = \frac{2}{\lambda_+ + \lambda_-}$
- $\max\left(\left|1-\mu^*\lambda_-\right|, \left|1-\mu^*\lambda_+\right|\right) = \frac{\lambda_+ - \lambda_-}{\lambda_+ + \lambda_-}$
- $\|x_M - x^*\|_2 \leq \left(\frac{\lambda_+ - \lambda_-}{\lambda_+ + \lambda_-}\right)^M \|x_0 - x^*\|_2$

  convergence depends on the eigenvalues of $A^T A$

  Two extremes:
- Identical eigenvalues (extremely well conditioned) $\lambda_- = \lambda_+$, i.e., $\lambda_1 = \lambda_2 = \cdots = \lambda_d \implies$ convergence in one iteration
- Distant eigenvalues (poorly conditioned) $\lambda_+ \gg \lambda_-$ $\implies \frac{\lambda_+ - \lambda_-}{\lambda_+ + \lambda_-} \approx 1$ leads to slow convergence
- Condition number $\kappa := \frac{\lambda_+}{\lambda_-}$
- $\|x_M - x^*\|_2 \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)^M \|x_0 - x^*\|_2$

$$1 - \frac{2\cdot\lambda_-}{\lambda_+ + \lambda_-} = \frac{\lambda_+ + \lambda_- - 2\lambda_-}{\lambda_+ + \lambda_-} = \frac{\lambda_+ - \lambda_-}{\lambda_+ + \lambda_-}$$

$$\left(\frac{\lambda_+ - \lambda_-}{\lambda_+ + \lambda_-}\right)^M = \left(\frac{\frac{\lambda_+}{\lambda_-} - 1}{\frac{\lambda_+}{\lambda_-} + 1}\right)^M$$

Computational complexity

$$\left(\frac{\kappa-1}{\kappa+1}\right)^M \|x_0 - x^*\|_2 = \epsilon$$

$$M \cdot \log(\quad) = \log(\epsilon)$$

$$\|x_M - x^*\|_2 \leq \left(\frac{\kappa-1}{\kappa+1}\right)^M \|x_0 - x^*\|_2$$

- Initialize at $x_0 = 0$
- For ϵ accuracy, i.e., $\|x_M - x^*\|_2 \leq \epsilon$
- We need to set the number of iterations M to

$$-M \log\left(\frac{\kappa-1}{\kappa+1}\right) + \log\|x^*\|_2 \leq \log(\epsilon)$$

$\log(1/\epsilon)$

- $M = O\left(\frac{\log(\frac{1}{\epsilon})}{\log(\frac{\kappa+1}{\kappa-1})}\right)$

$\log(1+x) \leq x$

$\nabla f(x_t) = A^T(Ax_t - b)$

- $\log\left(\frac{\kappa+1}{\kappa-1}\right) \approx \frac{2}{\kappa-1}$ for large κ
- $M = O\left(\frac{\log(\frac{1}{\epsilon})}{\log(\frac{\kappa+1}{\kappa-1})}\right) = O\left(\kappa \log(\frac{1}{\epsilon})\right)$ for large κ
- Total computational cost $\kappa nd \log(\frac{1}{\epsilon})$ for ϵ accuracy

$O(n \cdot d)$ per iter

FJLT & GD

$\Rightarrow nd \log(1/\epsilon) + nd \log n$

FJLT $nd \log n + \frac{d^3}{\epsilon^2}$ vs GD $\kappa nd \log(1/\epsilon)$

Improving condition number dependence: 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**

Momentum

$$\frac{d\,x(t)}{dt} = \dot{x} = g(x)$$

$$\lim_{\Delta \to 0} \frac{x_{t+\Delta} - x_t}{\Delta} = g(x)$$

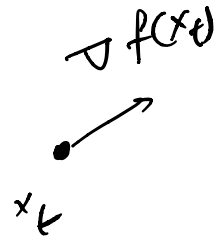
- Gradient Descent with Momentum

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

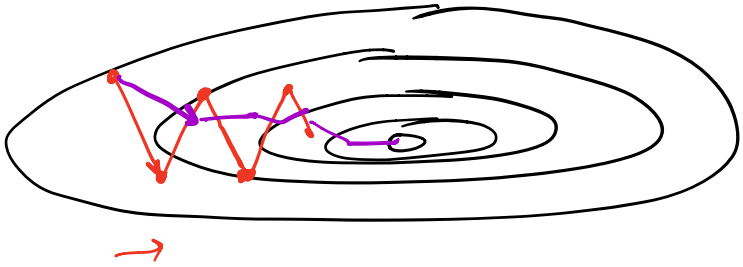
- related to a discretization of the second order ordinary differential equation

$$\ddot{x} + a\dot{x} + b\nabla f(x)$$

- which models the motion of a body in a potential field given by f



Momentum



- also called accelerated gradient descent, or heavy-ball method
- can be re-written as

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

- p_t is the search direction
- there is a short-term memory (we need to store another vector of size d)
- typically we set $p_0 = 0$

$p_{-1} = 0$

LS problems additional memory does not help.

Anderson Acceleration:

Gradient Descent with Momentum for Least Squares Problems

- $\min_x f(x)$ where $f(x) = \|Ax - b\|_2^2$
- Gradient Descent with momentum (Heavy Ball Method)

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

- Recall that when $\beta = 0$ (Gradient Descent) we defined $\Delta_t := x_t - x^*$ where $x^* = A^\dagger b$ and established the recursion

$$\Delta_{t+1} = (I - \mu A^T A)\Delta_t$$

- Since there is one time step memory, consider $V_t := \|\Delta_{t+1}\|_2^2 + \|\Delta_t\|_2^2$ instead
- we can write V_t in terms of $V_{t-1} = \|\Delta_t\|_2^2 + \|\Delta_{t-1}\|_2^2$
- **Lyapunov analysis**

  V_t is an energy function that decays to zero and upper-bounds error, i.e., $\|\Delta_t\|_2^2 \leq V_t$

Convergence analysis

- $\min_x f(x)$ where $f(x) = \|Ax - b\|_2^2$
- Gradient Descent with momentum (Heavy Ball Method)

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

- let $\Delta_t := x_t - x^*$ where $x^* = A^\dagger b$
- note that $b = Ax^* + b^\perp$ and $\nabla f(x_t) = A^T A \Delta_t$

$$\tilde{\Delta}_{t+1} = \begin{bmatrix} \Delta_{t+1} \\ \Delta_t \end{bmatrix} = \begin{bmatrix} x_t - \alpha \nabla f(x_t) + \beta(x_t - x_{t-1}) - x^* \\ \Delta_t \end{bmatrix}$$

Augmented State

$$= \begin{bmatrix} (1+\beta)I - \alpha A^T A & \beta I \\ I & 0 \end{bmatrix} \begin{bmatrix} \Delta_t \\ \Delta_{t-1} \end{bmatrix}$$

$$\tilde{\Delta}_{t+1} = \begin{bmatrix} (1+\beta)I - \alpha A^T A & \beta I \\ I & 0 \end{bmatrix} \cdot \tilde{\Delta}_t$$

Convergence analysis

- iterating for $t = 1, ..., M$

$$\begin{bmatrix} \Delta_{M+1} \\ \Delta_M \end{bmatrix} = \begin{bmatrix} (1+\beta)I - \alpha A^T A & \beta I \\ I & 0 \end{bmatrix}^M \begin{bmatrix} \Delta_1 \\ \Delta_0 \end{bmatrix}$$

- taking norms

$$\|\Delta_M\|_2 \leq \left\| \begin{bmatrix} \Delta_{M+1} \\ \Delta_M \end{bmatrix} \right\|_2 = \left\| \begin{bmatrix} (1+\beta)I - \alpha A^T A & \beta I \\ I & 0 \end{bmatrix}^M \begin{bmatrix} \Delta_t \\ \Delta_{t-1} \end{bmatrix} \right\|_2$$

$$\leq \sigma_{\max}\left(\begin{bmatrix} (1+\beta)I - \alpha A^T A & \beta I \\ I & 0 \end{bmatrix}^M \right) \left\| \begin{bmatrix} \Delta_t \\ \Delta_{t-1} \end{bmatrix} \right\|_2$$

Spectral Radius

$M = M^T = U \Lambda U^T$ Symmetric case

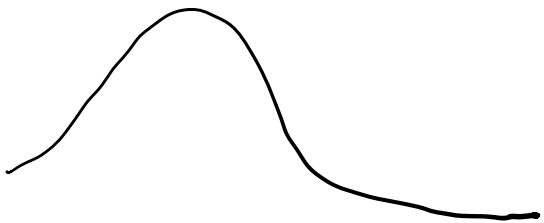
$M^k = U \Lambda^k U^T$

$\sigma_{max}(M^k) = \sigma_{max}(M)^k$

- Let M be an $d \times d$ matrix with eigenvalues $\lambda_1, ..., \lambda_d$
- spectral radius is defined as

$$\rho(M) := \max_{i=1,..,d} |\lambda_i(M)|$$

Lemma (Gelfand's formula) $\lim_{k \to \infty} \sigma_{\max}(M^k)^{\frac{1}{k}} = \rho(M)$



$$\sigma_{max}(M^k) \approx (\rho(M))^k \quad k \to \infty$$

asymmetric case

- Let λ_i denote the eigenvalues of $A^T A$ for $i = 1, ..., d$
- **Lemma** The eigenvalues of

$$\begin{bmatrix} (1+\beta)I - \alpha A^T A & \beta I \\ I & 0 \end{bmatrix}$$

are given by the eigenvalues of 2×2 matrices

$$\begin{bmatrix} 1+\beta-\alpha\lambda_i & -\beta \\ 1 & 0 \end{bmatrix}$$

- for $i = 1, ..., d$
- These are given by the roots of $u^2 - (1+\beta-\alpha\lambda_i)u + \beta = 0$
- setting $\alpha = \frac{4}{\sqrt{\lambda_+}+\sqrt{\lambda_-}}$ and $\beta = \frac{\sqrt{\lambda_+}-\sqrt{\lambda_-}}{\sqrt{\lambda_+}+\sqrt{\lambda_-}}$ yields (minimizes the abs of the max. root
- spectral radius: $\rho\left(\begin{bmatrix} (1+\beta)I - \alpha A^T A & \beta I \\ I & 0 \end{bmatrix}\right) = \frac{\sqrt{\lambda_+}-\sqrt{\lambda_-}}{\sqrt{\lambda_+}+\sqrt{\lambda_-}}$

convergence rate $\left(\frac{\sqrt{\lambda_+}-\sqrt{\lambda_-}}{\sqrt{\lambda_+}+\sqrt{\lambda_-}}\right)^k$ as $k \to \infty$

Convergence result

GD ($\beta=0$) $\quad \|x_M - x^*\|_2 \leq \left(\frac{\lambda_+ - \lambda_-}{\lambda_+ + \lambda_-}\right)^M \|x_0 - x^*\|_2$

- setting $\alpha = \frac{4}{\sqrt{\lambda_+}+\sqrt{\lambda_-}}$ and $\beta = \frac{\sqrt{\lambda_+}-\sqrt{\lambda_-}}{\sqrt{\lambda_+}+\sqrt{\lambda_-}}$ yields

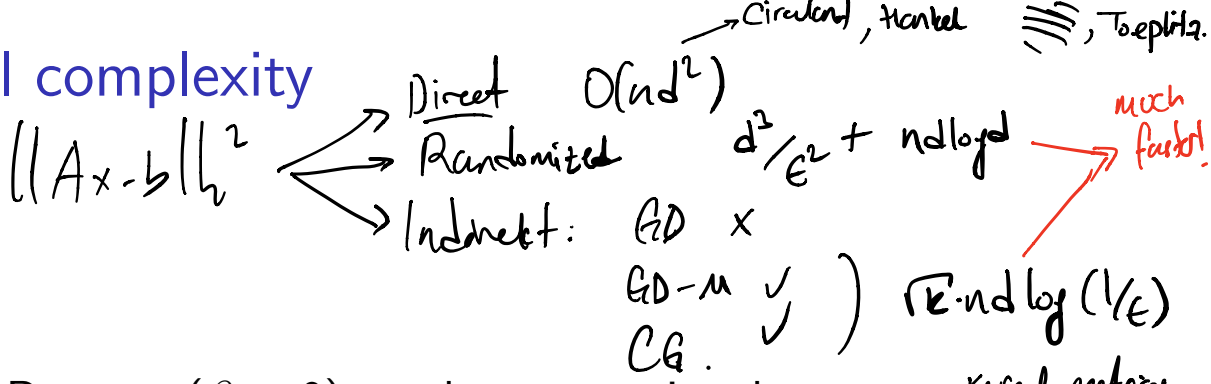
$$\|\Delta_M\|_2 \leq \left\| \begin{bmatrix} \Delta_{M-1} \\ \Delta_M \end{bmatrix} \right\|_2 \leq \left(\frac{\sqrt{\lambda_+} - \sqrt{\lambda_-}}{\sqrt{\lambda_+} + \sqrt{\lambda_-}}\right)^M \left\| \begin{bmatrix} \Delta_1 \\ \Delta_{1-1} \end{bmatrix} \right\|_2$$

$$\kappa = \frac{\lambda_+}{\lambda_-} \qquad = \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^M \cdot \left\| \begin{matrix} \Delta_1 \\ \Delta_{1-1} \end{matrix} \right\|_2 = \epsilon$$

solve for M: $\quad M \cdot \log\left(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right) = \log 1/\epsilon$

$$M = \frac{\log(1/\epsilon)}{\log\left(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right)} = \sqrt{\kappa} \cdot \log(1/\epsilon)$$

Computational complexity



- Gradient Descent ($\beta = 0$) total computational cost $\kappa nd \log(\frac{1}{\epsilon})$ for ϵ accuracy
- Gradient Descent with Momentum total computational cost $\sqrt{\kappa} nd \log(\frac{1}{\epsilon})$ for ϵ accuracy
- we need to know eigenvalues of $A^T A$ to find optimal step-sizes

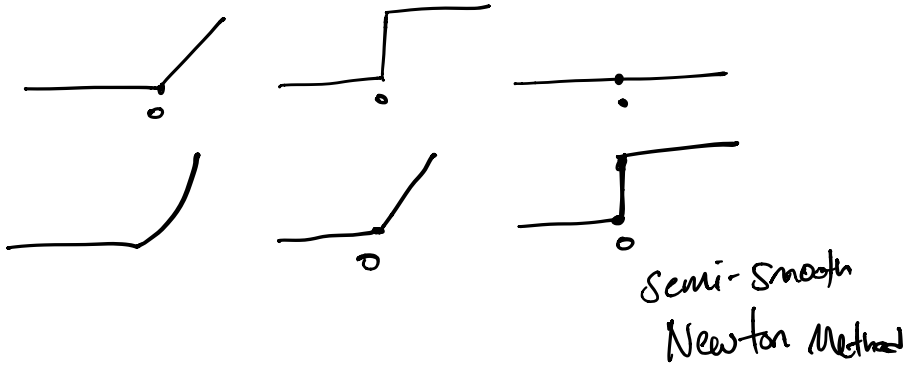
Conjugate Gradient method (automatically) tunes the step size.

- Higher order momentum can not do better than $\sqrt{\kappa}\cdot nd\log(1/\epsilon)$.
- Sketching can improve

Computational complexity

- Gradient Descent ($\beta = 0$) total computational cost $\kappa nd \log(\frac{1}{\epsilon})$ for ϵ accuracy
- Gradient Descent with Momentum total computational cost $\sqrt{\kappa} nd \log(\frac{1}{\epsilon})$ for ϵ 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

$\nabla_y (\quad) = 0$

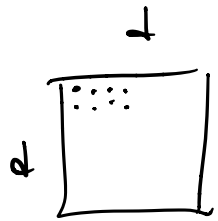
Newton's Method

$f(x) = \|Ax - b\|_2^2$

$[\nabla^2 f(x)]_{ij} = [A^T A]_{ij} = A^{(i)T} A^{(j)}$

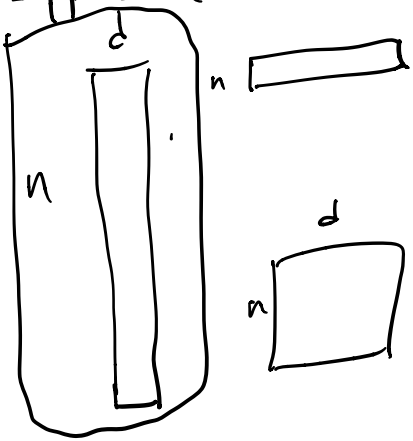
A: $n \times d$

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

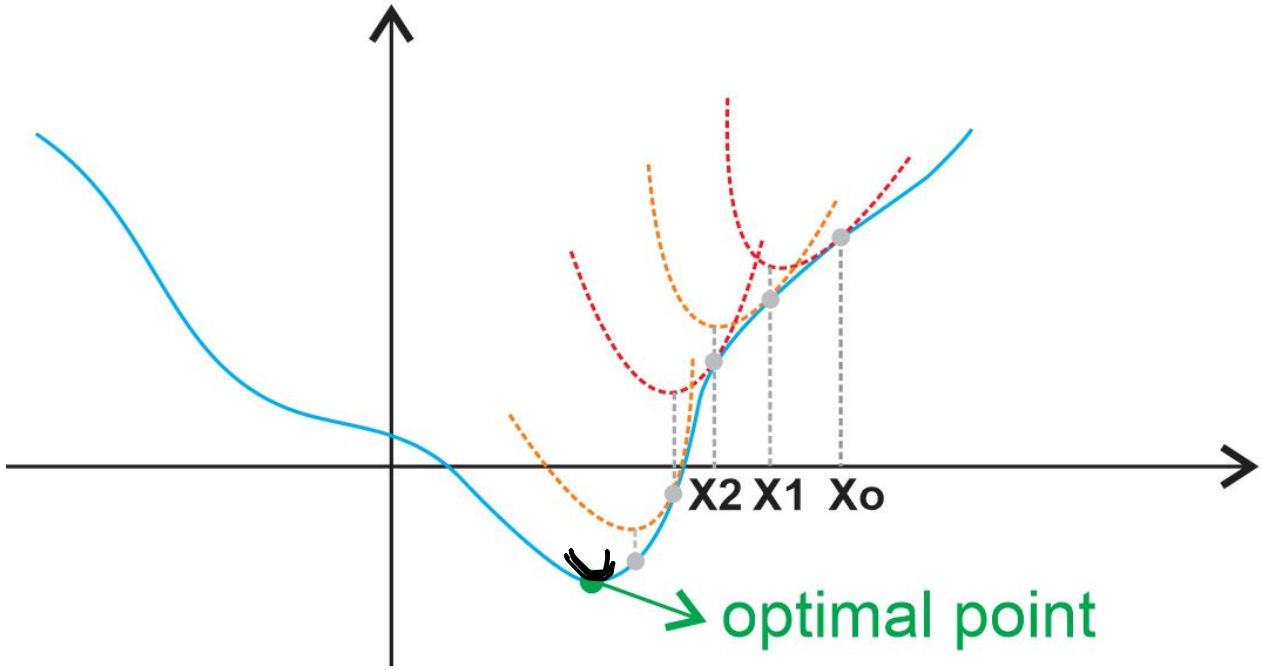
Compute $\nabla^2 f(x)$  $O(nd^2)$

Invert $\left(\nabla^2 f(x)\right)^{-1}$ $O(d^3)$

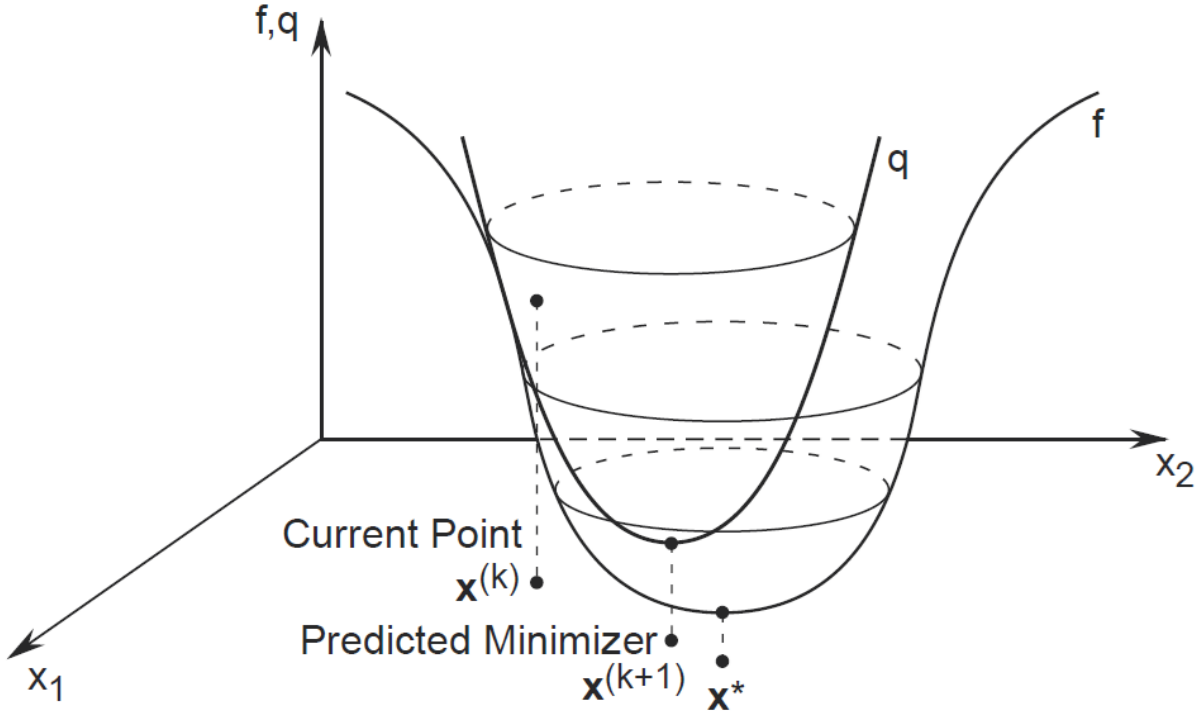
Suppose $n \gg d$



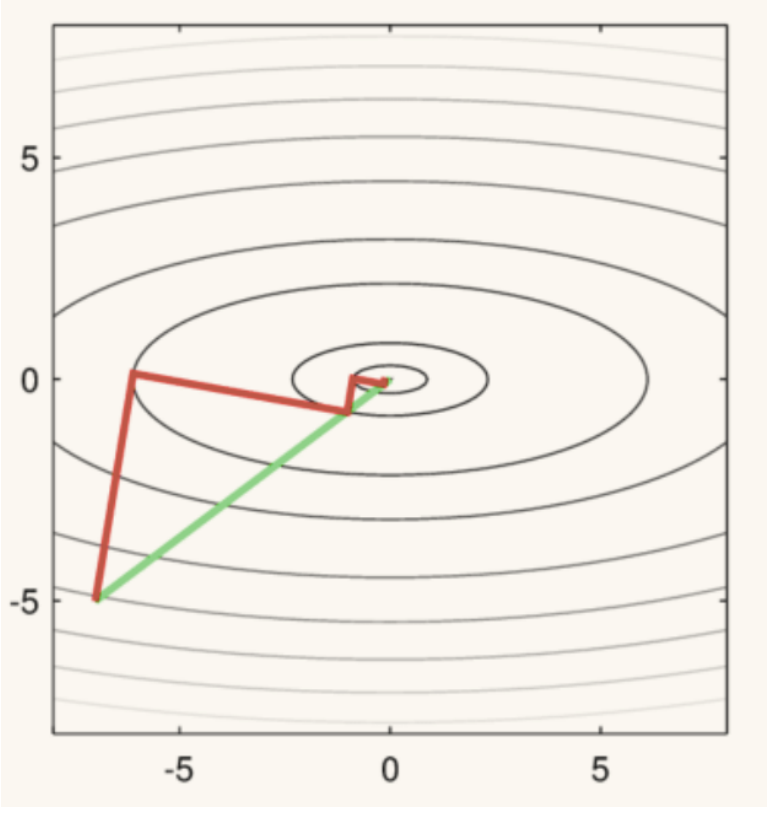
Newton's Method in one dimension



Newton's Method in higher dimensions



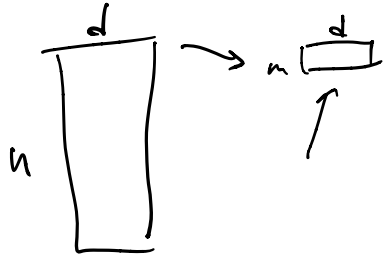
Newton's Method vs Gradient Descent



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 $\nabla^2 f(x) = A^T A$
- 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 $\mu_t = \mu$

Questions?