18-660: Numerical Methods for Engineering Design and Optimization

Xin Li
Department of ECE
Carnegie Mellon University
Pittsburgh, PA 15213

Electrical & Computer ENGINEERING

Overview

- Conjugate Gradient Method (Part 1)
  - Quadratic programming
  - Gradient method
  - Orthogonal search direction

Linear Equation

- Linear equation

$$AX = B$$

  - A is symmetric and positive definite
  - Can be solved by Cholesky decomposition

- However, Cholesky decomposition (or Gaussian elimination in general) is not efficient if A is large and sparse

Linear Equation

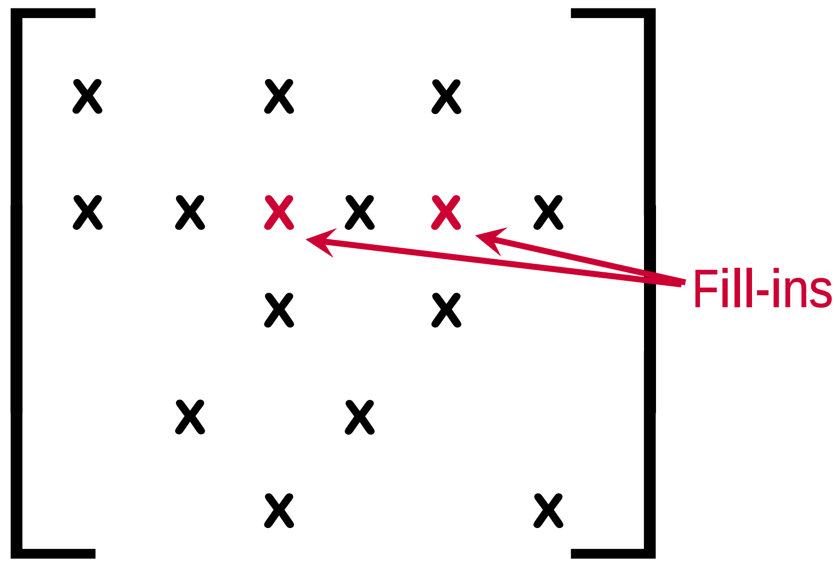
- A matrix is sparse if it contains a large number of zero elements

$$
\begin{bmatrix}
\times & & & \times & & & \\
& \times & & & & & \\
& & \times & & \times & & \\
\times & & & \times & & & \times \\
& & \times & & \times & & \\
& & & & & \times & \\
& & & \times & & & \times
\end{bmatrix}
$$

- Sparse matrix can be saved with small memory requirements
  - We do not explicitly save zero elements
  - We only save non-zero values and their locations

Linear Equation

- Cholesky decomposition or Gaussian elimination can generate a large number of fill-ins (i.e., non-zeros)
  - Matrix becomes much less sparse and consumes much memory



- Iterative methods (e.g., conjugate gradient) are much more efficient in these cases

Quadratic Programming

- Reformulate linear equation as a quadratic programming problem

$$AX = B$$

$$\Downarrow$$

$$\min_{X} \quad f(X) = \frac{1}{2} X^T A X - B^T X + C$$

Convex since A is positive definite

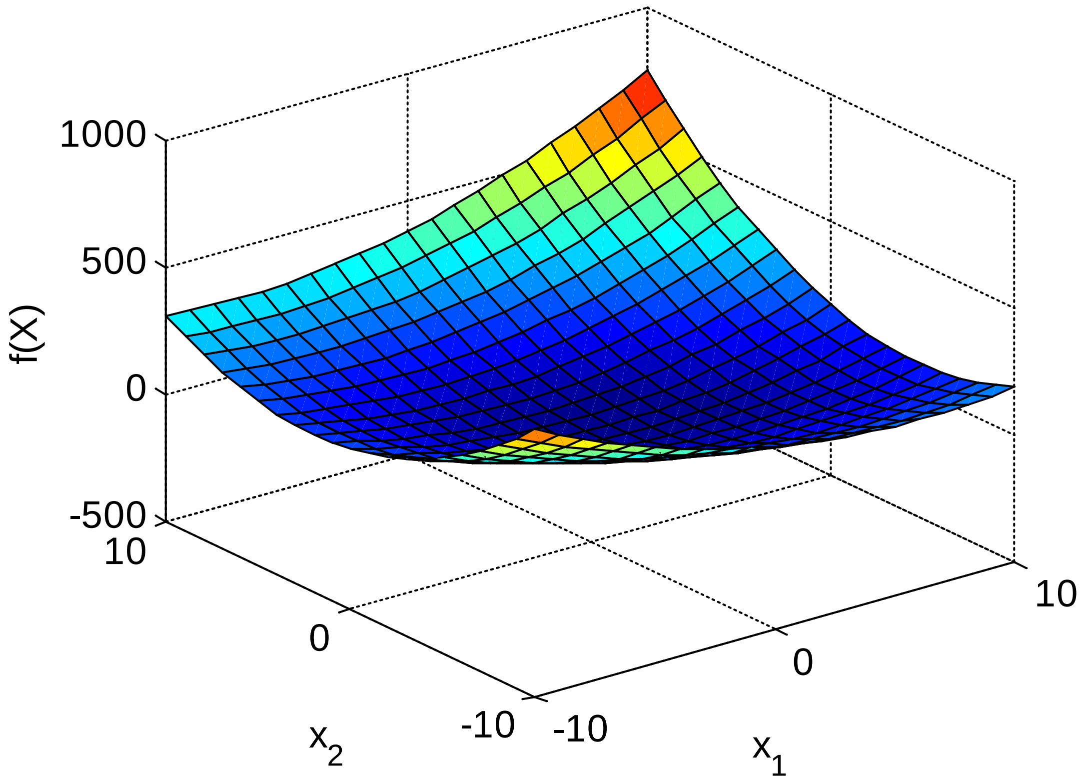
$$\nabla f(X) = AX - B = 0$$

$$X = A^{-1} B$$

Optimal X is the solution of AX = B

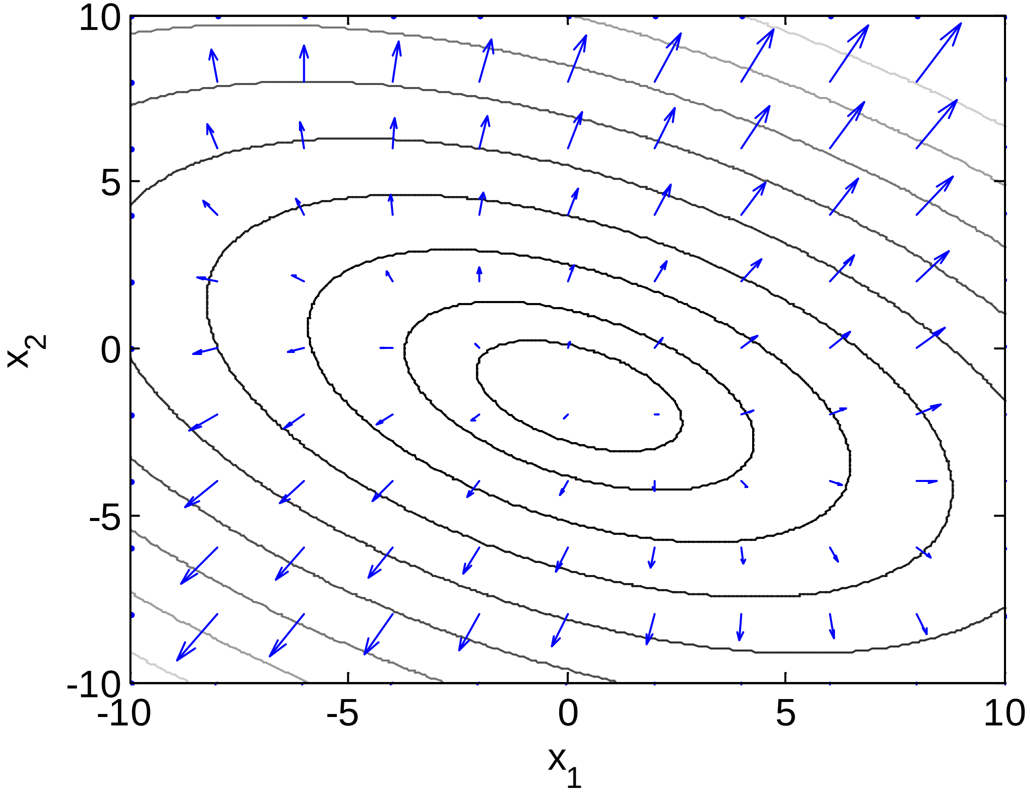
Quadratic Programming Example

$$f(X) = \frac{1}{2} X^T A X - B^T X + C \qquad A = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \quad B = \begin{bmatrix} -2 \\ -8 \end{bmatrix} \quad C = 0$$



Quadratic Programming Example

- Contour and gradient



Gradient Method

- Iteration scheme

$$\min_{X} \quad f(X) = \frac{1}{2} X^T A X - B^T X + C$$

$$\nabla f(X) = AX - B$$

$$X^{(k+1)} = X^{(k)} - \mu^{(k)} \cdot \nabla f\left[X^{(k)}\right] = X^{(k)} + \mu^{(k)} \cdot \underbrace{\left[B - AX^{(k)}\right]}_{\text{Residual } R^{(k)}}$$

$$X^{(k+1)} = X^{(k)} + \mu^{(k)} R^{(k)}$$

Gradient Method

- Optimal step size

$$f(X)=\frac{1}{2}X^{T}AX-B^{T}X+C \qquad R^{(k)}=B-AX^{(k)} \qquad X^{(k+1)}=X^{(k)}+\mu^{(k)}R^{(k)}$$

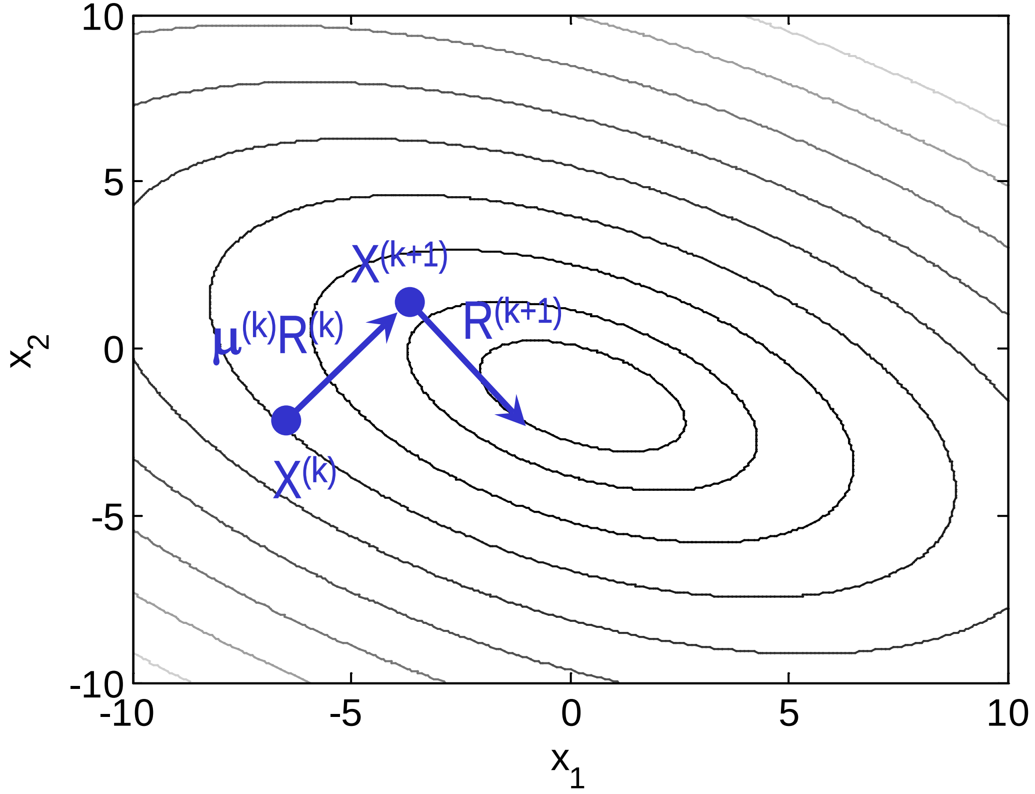
$$\min_{\mu^{(k)}} \quad f\left[X^{(k+1)}\right]=\frac{1}{2}X^{(k+1)T}AX^{(k+1)}-B^{T}X^{(k+1)}+C$$

$$\frac{d}{d\mu^{(k)}}f\left[X^{(k+1)}\right]=\left[\frac{\partial f}{\partial X^{(k+1)}}\right]^{T}\cdot\frac{\partial X^{(k+1)}}{\partial \mu^{(k)}}=\left[AX^{(k+1)}-B\right]^{T}\cdot R^{(k)}=-R^{(k+1)T}R^{(k)}=0$$

Residuals $R^{(k)}$ and $R^{(k+1)}$ are orthogonal

Quadratic Programming Example

$$f(X) = \frac{1}{2} X^T A X - B^T X + C \qquad A = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \quad B = \begin{bmatrix} -2 \\ -8 \end{bmatrix} \quad C = 0$$



Gradient Method

- Optimal step size

$$R^{(k)} = B - AX^{(k)} \qquad X^{(k+1)} = X^{(k)} + \mu^{(k)} R^{(k)} \qquad R^{(k+1)T} R^{(k)} = 0$$

$$\left[B - AX^{(k+1)}\right]^T \cdot R^{(k)} = 0$$

$$\left\{B - A \cdot \left[X^{(k)} + \mu^{(k)} R^{(k)}\right]\right\}^T \cdot R^{(k)} = 0$$

$$\left[B - AX^{(k)} - \mu^{(k)} AR^{(k)}\right]^T \cdot R^{(k)} = 0$$

$$\left[R^{(k)} - \mu^{(k)} AR^{(k)}\right]^T \cdot R^{(k)} = 0$$

$$R^{(k)T} R^{(k)} - \mu^{(k)} R^{(k)T} AR^{(k)} = 0$$

$$\mu^{(k)} = \frac{R^{(k)T} R^{(k)}}{R^{(k)T} AR^{(k)}}$$

Gradient Method

- Iteration scheme

$$\min_{X} \quad f(X) = \frac{1}{2} X^T A X - B^T X + C$$

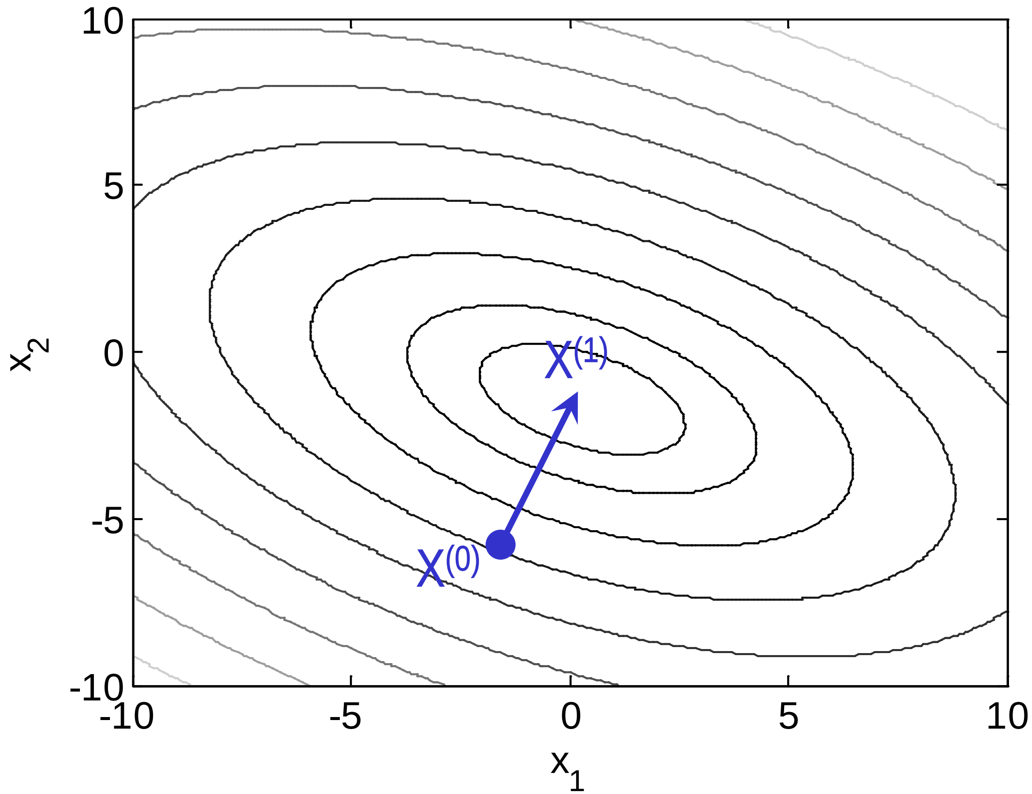
$$R^{(k)} = B - A X^{(k)}$$

$$\mu^{(k)} = \frac{R^{(k)T} R^{(k)}}{R^{(k)T} A R^{(k)}}$$

$$X^{(k+1)} = X^{(k)} + \mu^{(k)} R^{(k)}$$

Quadratic Programming Example

$$f(X) = \frac{1}{2} X^T A X - B^T X + C \qquad A = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \quad B = \begin{bmatrix} -2 \\ -8 \end{bmatrix} \quad C = 0$$



Gradient method may converge by one iteration if a "good" initial guess is selected

Initial Guess

- Gradient method converges by one iteration, if $R^{(0)}$ is an eigenvector of A

$$R^{(k)} = B - AX^{(k)} \qquad \mu^{(k)} = \frac{R^{(k)T}R^{(k)}}{R^{(k)T}AR^{(k)}} \qquad X^{(k+1)} = X^{(k)} + \mu^{(k)}R^{(k)}$$

$$AR^{(0)} = \lambda R^{(0)}$$

$$\mu^{(0)} = \frac{R^{(0)T}R^{(0)}}{\lambda R^{(0)T}R^{(0)}} = \frac{1}{\lambda}$$

$$R^{(1)} = B - AX^{(1)} = B - A\cdot\left[X^{(0)} + \frac{1}{\lambda}R^{(0)}\right] = B - AX^{(0)} - \frac{1}{\lambda}AR^{(0)}$$

$$= R^{(0)} - \frac{1}{\lambda}\cdot\lambda R^{(0)} = 0$$

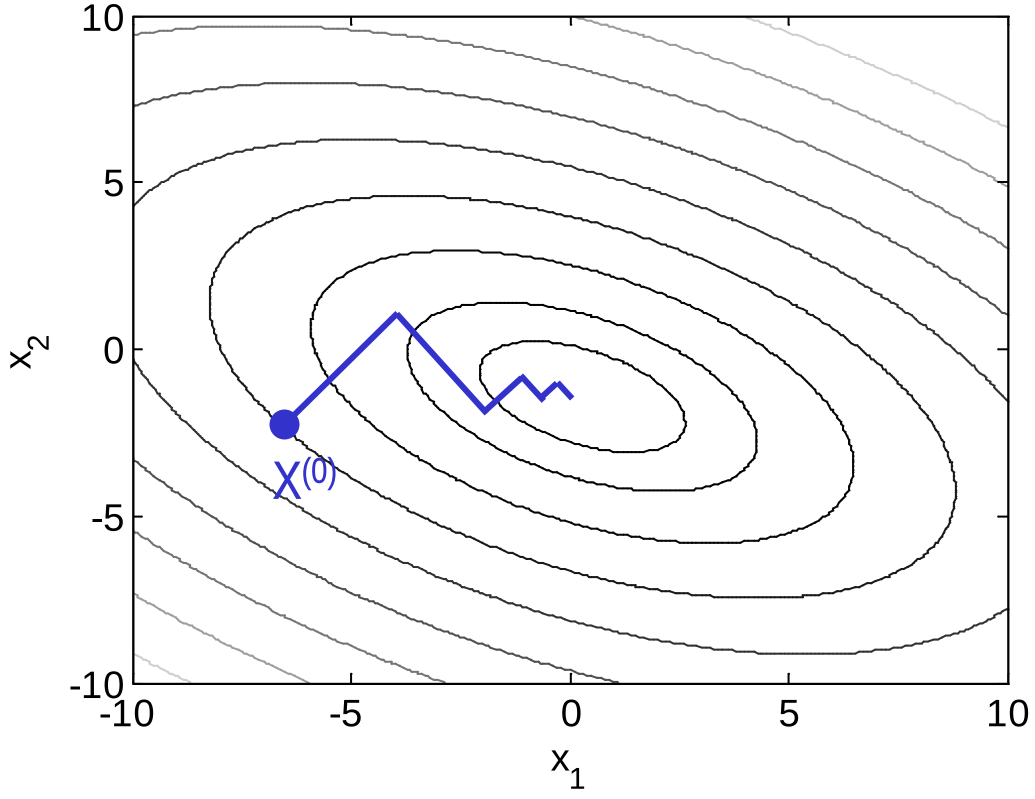
Initial Guess

- In practice, it is not possible to achieve such an ideal case
  - We do not know the exact eigenvectors of A

- Starting from a random initial guess, gradient method may take many iterations to converge
  - Gradient method has slow convergence, even though optimal step size μ is used for each iteration
  - This is a big problem of gradient method

Quadratic Programming Example

$$f(X) = \frac{1}{2} X^T A X - B^T X + C \qquad A = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \quad B = \begin{bmatrix} -2 \\ -8 \end{bmatrix} \quad C = 0$$



Newton Method

- Newton method can converge by one iteration
- However, we have to solve the linear equation $X = A^{-1}B$
  - It is exactly the problem that we try to solve at the beginning
  - Newton method does not tell us how to solve the large, sparse linear equation efficiently

$$\min_{X} \quad f(X) = \frac{1}{2} X^T A X - B^T X + C$$

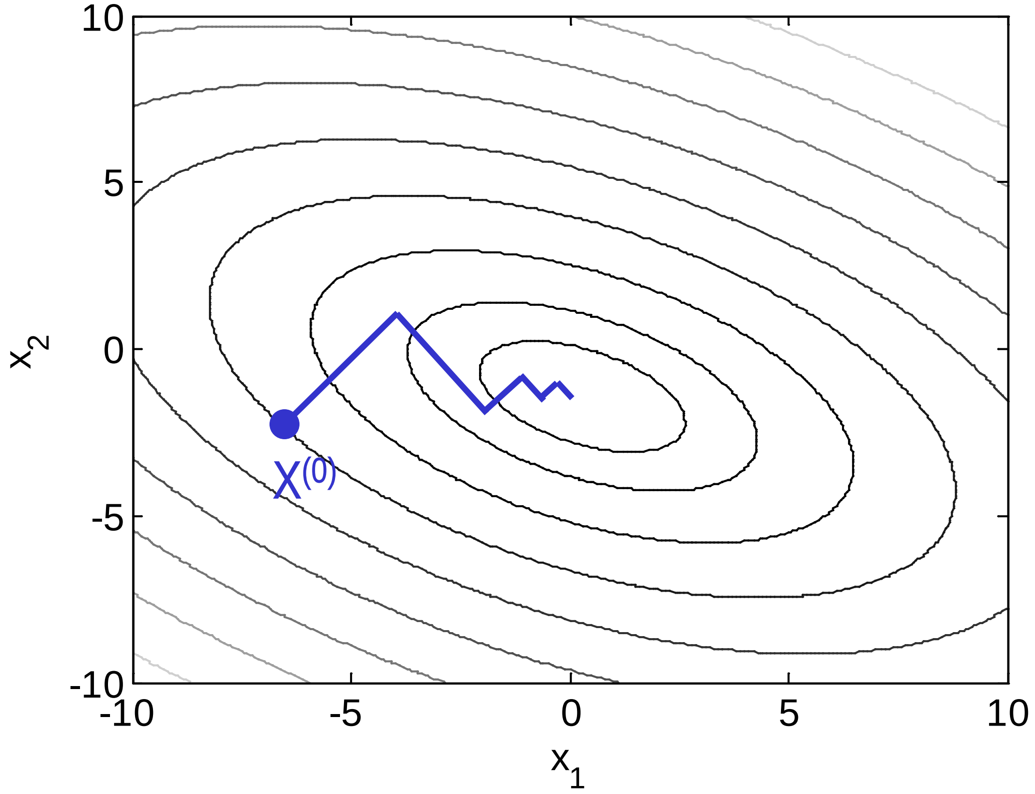
$$\Downarrow$$

$$\nabla f(X) = AX - B = 0$$

$$X = A^{-1} B$$

Orthogonal Search Direction

- Gradient method often moves towards the same direction as earlier iteration steps – BAD idea

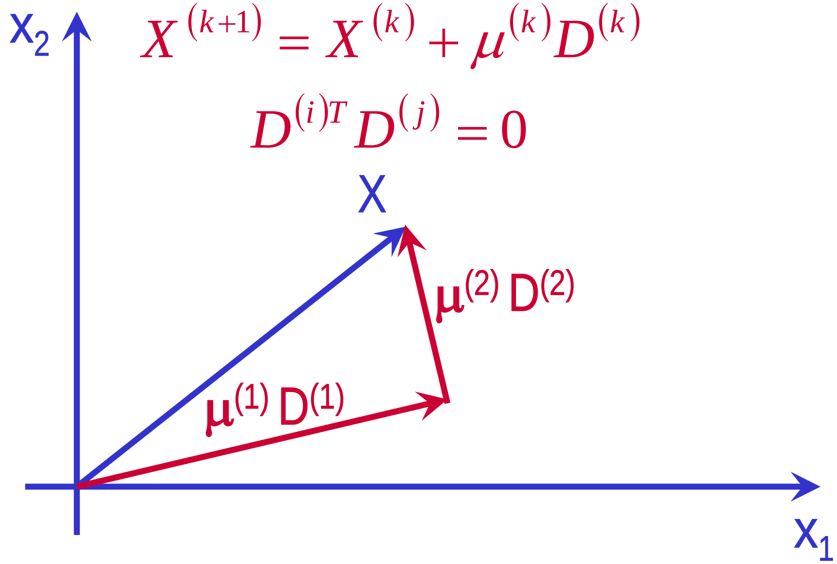


Orthogonal Search Direction

- Ideally, we want to
  - Select a set of orthogonal search directions $D^{(k)}$
  - Take exactly one iteration step for each direction
  - After at most N steps, we get the solution X
  - (N is the problem size, i.e., $A \in R^{N \times N}$)

$$X^{(k+1)} = X^{(k)} + \mu^{(k)} D^{(k)}$$

$$D^{(i)T} D^{(j)} = 0$$



How do we decide $\mu^{(k)}$ and $D^{(k)}$?

Orthogonal Search Direction

- Determine step size $\mu^{(k)}$

$$D^{(i)T}D^{(j)}=0 \qquad X^{(k+1)}=X^{(k)}+\mu^{(k)}D^{(k)} \qquad X=X^{(0)}+\mu^{(0)}D^{(0)}+\cdots+\mu^{(N-1)}D^{(N-1)}$$

$$X^{(k+1)}=X^{(0)}+\mu^{(0)}D^{(0)}+\cdots+\mu^{(k)}D^{(k)}$$

$$\Delta^{(k+1)}=X^{(k+1)}-X=-\mu^{(k+1)}D^{(k+1)}-\cdots-\mu^{(N-1)}D^{(N-1)}$$

$$D^{(k)T}\Delta^{(k+1)}=D^{(k)T}\cdot\left[-\mu^{(k+1)}D^{(k+1)}-\cdots-\mu^{(N-1)}D^{(N-1)}\right]=0$$

$\Delta^{(k+1)}$ and $D^{(k)}$ are orthogonal

Orthogonal Search Direction

- Determine step size $\mu^{(k)}$

$$X^{(k+1)} = X^{(k)} + \mu^{(k)} D^{(k)} \qquad \Delta^{(k)} = X^{(k)} - X \qquad D^{(k)T} \Delta^{(k+1)} = 0$$

$$\Delta^{(k+1)} = X^{(k+1)} - X = X^{(k)} + \mu^{(k)} D^{(k)} - X = \Delta^{(k)} + \mu^{(k)} D^{(k)}$$

$$D^{(k)T} \cdot \left[\Delta^{(k)} + \mu^{(k)} D^{(k)} \right] = 0$$

$$D^{(k)T} \Delta^{(k)} + \mu^{(k)} D^{(k)T} D^{(k)} = 0$$

$$\mu^{(k)} = -\frac{D^{(k)T} \Delta^{(k)}}{D^{(k)T} D^{(k)}}$$

However, we do not know $\Delta^{(k)}$ – otherwise, we know $X = X^{(k)} - \Delta^{(k)}$

Orthogonal Search Direction

- Orthogonal search direction is difficult to apply to many practical optimization problems
- Instead of using orthogonal directions, we can make search directions conjugate (or equivalently A-orthogonal)
- More details in next lecture

Summary

- Conjugate gradient method (Part 1)
  - Quadratic programming
  - Gradient method
  - Orthogonal search direction