

**Physics Based Animation Fall 2008**

# **Iterative Solver 1**

**Imaging Media Research Center**

**KIST**

**Jinwook Kim**

## Iterative Solver

# Linear System

- Given  $A \in \mathbb{R}^{n \times m}$  and  $b \in \mathbb{R}^n$ , find  $x \in \mathbb{R}^m$   
$$Ax = b$$
- If you are lucky,  $x = A^{-1}b$
- If  $A$  has no inverse but it is full rank,
  - $n > m$ ,  $x^* = (A^T A)^{-1} A^T b$  minimizes residual  $\|b - Ax^*\|$ 
    - Called **least squares solution**
  - $n < m$ ,  $x = A^T (AA^T)^{-1} b$  is one of the solutions
    - Called **minimum norm solution**

## Iterative Solver

# Direct methods to solve $Ax=b$

- Never try to get  $A^{-1}$
- Gauss elimination
- LU decomposition
  - $A = LU$
- QR decomposition
  - $A = QR$
- Cholesky decomposition
  - $A = LL^T$
- Singular value decomposition
  - $A = USV^T$

## Iterative Solver

# Direct methods to solve $Ax=b$

- Complexity
  - Decomposition =  $O(n^3)$
  - Back substitution =  $O(n^2)$
- Gives you an exact solution in machine precision
- May **NOT** suitable for large systems
  - Prone to be ill-conditioned
  - Round off error accumulates
  - High computational complexity

## Iterative Solver

# Iterative methods to solve $Ax=b$

- Approximate solution
  - Refine the solution iteratively
- Pros
  - Good for sparse matrix
  - Scalable
  - Numerically stable
  - Parallel computation
- Cons
  - Applicable to specific types of linear systems

## Iterative Solver

# General Idea

- Assume  $A$  is a square matrix and strictly diagonally dominant

$$a_{ii} > \sum_{j \neq i} a_{ij}$$

- Split the matrix

$$A = P - N$$

- $\det(P) \neq 0$
- $B = P^{-1}N$
- Sequence  $\left\{ x^{(k+1)} = Bx^{(k)} + P^{-1}b \right\}$  converges to the solution of  $Ax = b$ 
  - Iff  $\rho(B) < 1$

## Iterative Solver

# Jacobi Iteration

- $A = L + D + U$ 
  - L = lower triangular matrix
  - D = diagonal matrix
  - U = upper triangular matrix
- $P = D$ ,  $N = -L - U$
- (k+1)th iteration

$$x^{(k+1)} = D^{-1} \left( b - (L + U)x^{(k)} \right)$$

or

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left( b_i - \sum_{\substack{j=1 \\ j \neq i}}^n a_{ij} x_j^{(k)} \right)$$

## Iterative Solver

# Gauss-Seidel Iteration

- $P = D+L$ ,  $N = -U$
- (k+1)th iteration

$$x^{(k+1)} = D^{-1} \left( b - Lx^{(k+1)} - Ux^{(k)} \right)$$

or

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij} x_j^{(k)} \right)$$

## Iterative Solver

# Successive Over Relaxation(SOR)

- $\left\{ x^{(k+1)} = x^{(k)} + P^{-1}r^{(k)} \right\} = \left\{ x^{(k+1)} = P^{-1}Nx^{(k)} + P^{-1}b \right\}$

where  $r^{(k)} = b - Ax^{(k)}$

- Accelerate the correction term

$$x_{\omega}^{(k+1)} = x^{(k)} + \omega P^{-1}r^{(k)}$$

- SOR iteration

$$x_{\omega}^{(k+1)} = (1 - \omega)x^{(k)} + \omega x^{(k+1)}$$

$$x_i^{(k+1)} = (1 - \omega)x_i^{(k)} + \frac{\omega}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right)$$

## Iterative Solver

# Comments

- Jacobi iteration updates the solution using the previous approximation
  - Slow convergence
  - Easy to parallelize
  - Requires more memory to keep the previous step
- Gauss-Seidel iteration updates the solution using the latest approximation
  - Faster convergence
  - Hard to parallelize → red-black G-S iteration
  - Usually used with acceleration parameter  $0 < \omega < 2$