

CSE 245: Computer Aided Circuit Simulation and Verification

Matrix Computations: Iterative Methods (II)

Chung-Kuan Cheng

Outline

- Introduction
- Direct Methods
- Iterative Methods
 - Formulations
 - Projection Methods
 - Krylov Space Methods
 - Preconditioned Iterations
 - Multigrid Methods
 - Domain Decomposition Methods

Introduction

Direct Method

LU Decomposition

*General and Robust but
can be complicated if
 $N \geq 1M$*

Jacobi

Gauss-Seidel

Iterative Methods

Domain Decomposition

Preconditioning

*Conjugate Gradient
GMRES*

Multigrid

*Excellent choice for SPD matrices
Remain an art for arbitrary matrices*

Formulation

- Error in A norm

- *Matrix A is SPD (Symmetric and Positive Definite)*

- Minimal Residue

- *Matrix A can be arbitrary*

Formulation: Error in A norm

Min $E(x) = 1/2 x^T A x - b^T x$, A is SPD

Suppose that we knew $Ax_* = b$

$$E(x) = 1/2 (x - x_*)^T A (x - x_*).$$

For Krylov space approach,

search space: $x = x_0 + V y$,

where x_0 is an initial solution,

matrix $V_{n \times m}$ is given and full ranked

vector y_m contains the m variables.

Solution and Error

Min $E(x) = 1/2 x^T A x - b^T x$,

Search space: $x = x_0 + V y$, $r = b - A x$

Derivation

1. $V^T A V$ is nonsingular (A is SPD & V is full ranked)
2. The variable: $y = (V^T A V)^{-1} V^T r_0$ (**derivation**)
3. Thus, solution: $x = x_0 + V (V^T A V)^{-1} V^T r_0$

Property of the solution

1. Residue: $V^T r = 0$ (**proof**)
2. Error: $E(x) = E(x_0) - 1/2 r_0^T V (V^T A V)^{-1} V^T r_0$

Solution and Error

Min $E(x) = 1/2x^T Ax - b^T x$,

Search space: $x = x_0 + Vy$, $r = b - Ax$

The variable: $y = (V^T AV)^{-1} V^T r_0$

Derivation of y :

Use the condition that $dE/dy = 0$

We have $V^T AVy + V^T Ax_0 - V^T b = 0$

Thus $y = (V^T AV)^{-1} V^T (b - Ax_0)$

Solution and Error

Min $E(x) = 1/2x^T Ax - b^T x$,

Search space: $x = x_0 + Vy$, $r = b - Ax$

Property of Solution

1. Residue: $V^T r = 0$ (**proof**)

Proof:

$$\begin{aligned} V^T r &= V^T (b - Ax) = V^T (b - Ax_0 - AV(V^T AV)^{-1} V^T r_0) \\ &= V^T (r_0 - AV(V^T AV)^{-1} V^T r_0) = 0 \end{aligned}$$

The residue of the solution is orthogonal to the previous bases.

Transformation of the bases

For any $V' = VW$, where $V_{n \times m}$ is given and $W_{m \times m}$ is nonsingular, solution x remains the same for search space: $x = x_0 + V'y$

Proof:

1. $V'^T A V'$ is nonsingular (A is SPD & V' is full ranked)
2. $x = x_0 + V'(V'^T A V')^{-1} V'^T r_0 = x_0 + V(V^T A V)^{-1} V^T r_0$
3. $V'^T r = W^T V^T r = 0$
4. $E(x) = E(x_0) - 1/2 r_0^T V(V^T A V)^{-1} V^T r_0$

Steepest Descent: Error in A norm

$$\text{Min } E(x) = 1/2 x^T A x - b^T x,$$

$$\text{Gradient: } -r = Ax - b$$

$$\text{Set } x = x_0 + y r_0$$

1. $r_0^T A r_0$ is nonsingular (A is SPD)
2. $y = (r_0^T A r_0)^{-1} r_0^T r_0$
3. $x = x_0 + r_0 (r_0^T A r_0)^{-1} r_0^T r_0$
4. $r_0^T r = 0$
5. $E(x) = E(x_0) - 1/2 (r_0^T r_0)^2 / (r_0^T A r_0)$

Lanczos: Error in A norm

$$\text{Min } E(x) = 1/2 x^T A x - b^T x,$$

$$\text{Set } x = x_0 + V y$$

1. $v_1 = r_0$
2. v_i is in $K\{r_0, A, i\}$
3. $V = [v_1, v_2, \dots, v_m]$ is orthogonal
4. $AV = VH_m + v_{m+1} e_m^T$
5. $V^T AV = H_m$

Note since A is SPD, H_m is T_m Tridiagonal

Lanczos: Derive from Arnoldi Process

Arnoldi Process: $AV = VH + h_{m+1}v_{m+1}e_m^T$

Input A , $v_1 = r_0 / |r_0|$

Output $V = [v_1, v_2, \dots, v_m]$, H and h_{m+1}, v_{m+1}

$k=0$, $h_{10}=1$

While $h_{k+1,k} \neq 0$, $k \leq m$

$v_{k+1} = r_k / h_{k+1,k}$

$k = k + 1$,

$r_k = Av_k$

For $i=1, k$

$h_{ik} = v_i^T r_k$

$r_k = r_k - h_{ik} v_i$

End

$h_{k+1,k} = |r_k|_2$

End

Lanczos:

Create $V = [v_1, v_2, \dots, v_m]$ which is orthogonal

Input A, r_0 , Output V and $T = V^T A V$

Initial: $k=0, \beta_0 = |r_0|_2, v_0=0, v_1 = r_0 / \beta_0$

while $k \leq K$ or $\beta_k \neq 0$

$$v_{k+1} = r_k / \beta_k$$

$$k = k + 1$$

$$a_k = v_k^T A v_k$$

$$r_k = A v_k - a_k v_k - \beta_{k-1} v_{k-1}$$

$$\beta_k = |r_k|_2$$

End

Lanczos:

Create $V = [v_1, v_2, \dots, v_m]$ which is orthogonal

Input A, r_0 , Output V and $T = V^T A V$

$T_{ii} = a_i, T_{ij} = b_i (j = i + 1), = b_j (j = i - 1), = 0$ (else)

Proof: By induction that

$$v_j^T A v_j = \begin{cases} 0, & \text{if } i < j - 1 \\ b_j, & \text{if } i = j - 1 \\ a_j, & \text{if } i = j \end{cases}$$

Since $v_i v_j = 0$ if $i \neq j$

$$\begin{aligned} v_j^T A v_i &= v_j^T (b_{i-1} v_{i-1} + a_i v_i + b_i v_{i+1}) \\ &= b_{i-1} v_j^T v_{i-1} + a_i v_j^T v_i + b_i v_j^T v_{i+1} \end{aligned}$$

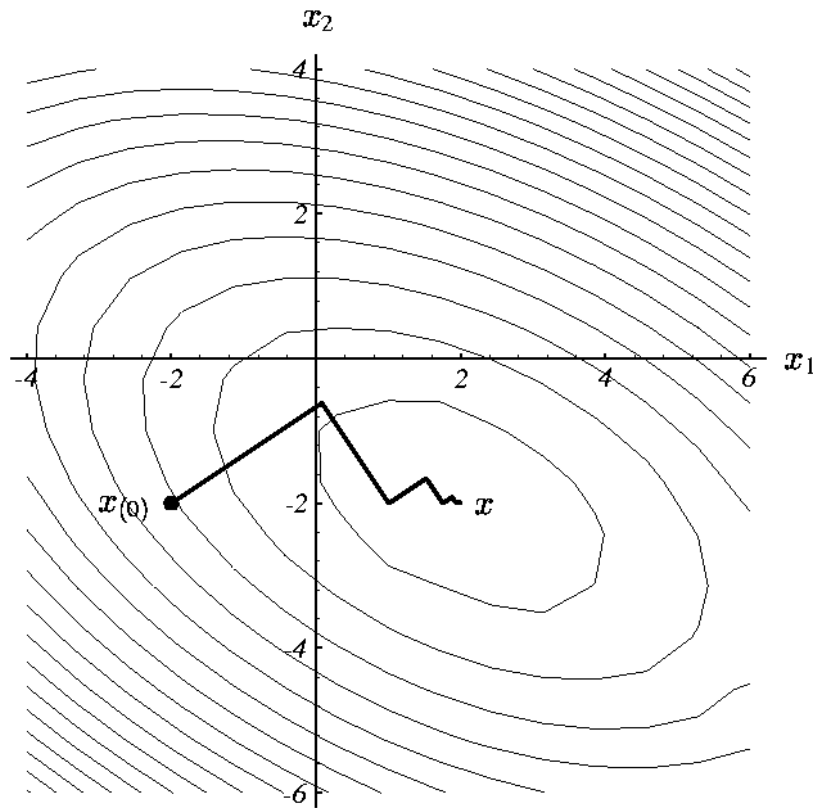
Conjugate Gradient:

$$\text{Min } E(x) = 1/2 x^T A x - b^T x,$$

$$\text{Set } x = x_0 + V y$$

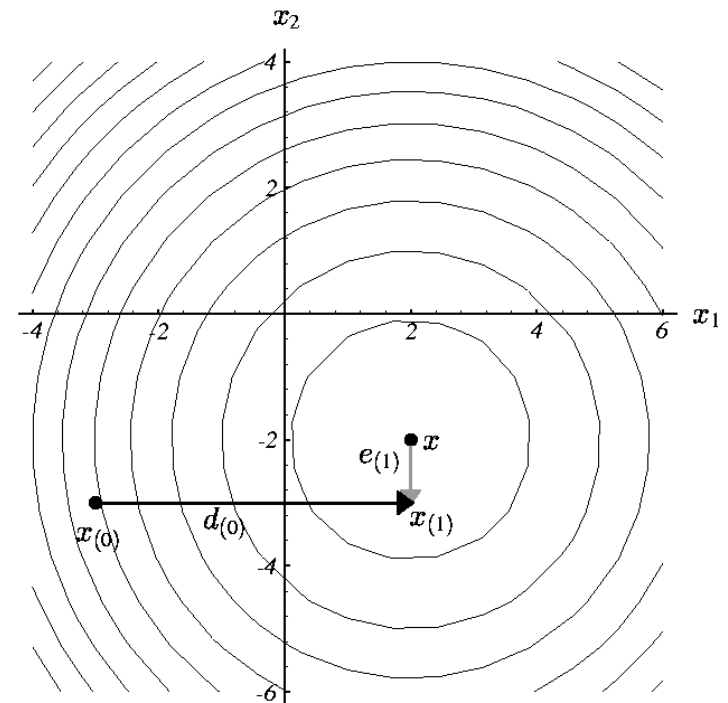
1. $v_1 = r_0$
2. v_i is in $K\{r_0, A, i\}$
3. Set $V = [v_1, v_2, \dots, v_m]$ orthogonal in A norm, i.e. $V^T A V = [\text{diag}(v_i^T A v_i)] = D$
4. $x = x_0 + V D^{-1} V^T r_0,$
5. $x = x_0 + \sum_{i=1, m} d_i v_i v_i^T r_0,$ where $d_i = (v_i^T A v_i)^{-1}$

Conjugate Gradient Method



Search direction of Steepest descent method

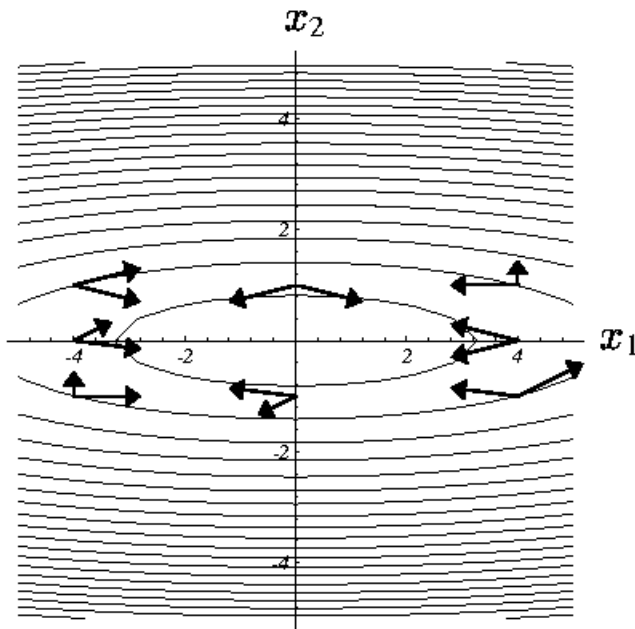
- Steepest Descent
 - Repeat search direction
- Why take exact one step for each direction?



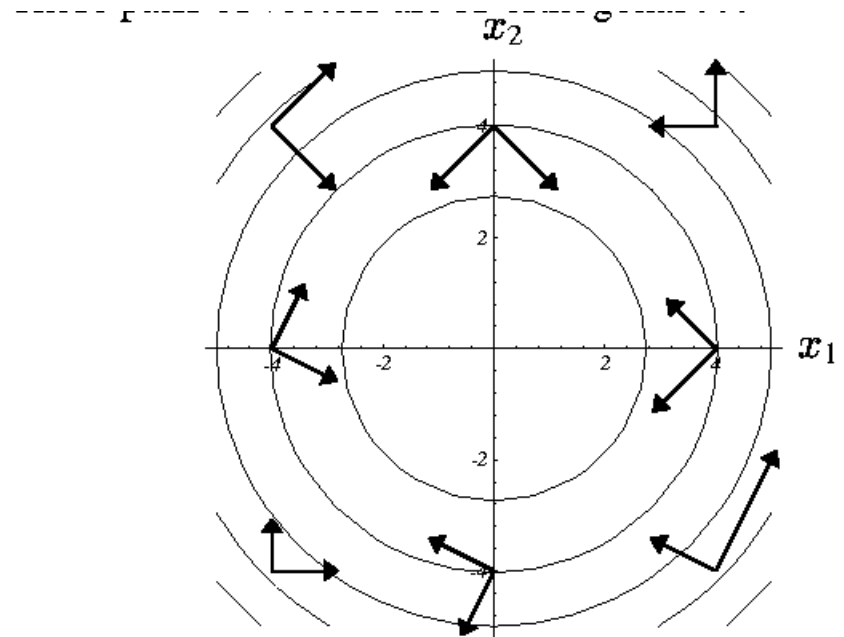
Orthogonal \rightarrow A-orthogonal

- Instead of orthogonal search direction, we make search direction A -orthogonal (conjugate)

$$d_{(j)}^T A d_{(i)} = 0$$



These pairs of vectors are A-orthogonal ...



... because these pairs of vectors are orthogonal.

Conjugate Search Direction

- How to construct A-orthogonal search directions, given a set of n linear independent vectors.
- Since the residue vector r_0, r_1, \dots, r_{n-1} in steepest descent method is orthogonal, a good candidate to start with

Conjugate Gradient:

Min $E(x) = 1/2 x^T A x - b^T x$,

Set $x = x_0 + V y$

Lanczos Method to derive V and T

$$x = x_0 + V T^{-1} V^T r_0$$

Decompose T to $LDU = T$ ($U = L^T$)

Thus, we have

$$x = x_0 + V (LDL^T)^{-1} V^T r_0 = x_0 + V L^{-T} D^{-1} L^{-1} V^T r_0$$

Conjugate Gradient Algorithm

Given x_0 , iterate until residue is smaller than error tolerance

$$d_{(0)} = r_{(0)} = b - Ax_{(0)}$$

$$a_{(i)} = \frac{r_{(i)}^T r_{(i)}}{d_{(i)}^T A d_{(i)}}$$

$$x_{(i+1)} = x_{(i)} + a_{(i)} d_{(i)}$$

$$r_{(i+1)} = r_{(i)} - a_{(i)} A d_{(i)}$$

$$\beta_{(i+1)} = \frac{r_{(i+1)}^T r_{(i+1)}}{r_{(i)}^T r_{(i)}}$$

$$d_{(i+1)} = r_{(i+1)} + \beta_{(i+1)} d_{(i)}$$

Conjugate gradient: Convergence

- ❑ In exact arithmetic, CG converges in n steps (**completely unrealistic!!**)
- ❑ Accuracy after k steps of CG is related to:
 - consider polynomials of degree k that is equal to 1 at step 0.
 - how small can such a polynomial be at all the eigenvalues of A ?
- ❑ Eigenvalues close together are good.
- ❑ **Condition number:** $\kappa(A) = \|A\|_2 \|A^{-1}\|_2 = \lambda_{\max}(A) / \lambda_{\min}(A)$
- ❑ Residual is reduced by a constant factor by $O(\kappa^{1/2}(A))$ iterations of CG.

Preconditioners

- Suppose you had a matrix B such that:
 1. condition number $\kappa(B^{-1}A)$ is small
 2. $By = z$ is easy to solve

- Then you could solve $(B^{-1}A)x = B^{-1}b$ instead of $Ax = b$

- $B = A$ is great for (1), not for (2)
- $B = I$ is great for (2), not for (1)
- Domain-specific approximations sometimes work
- $B = \text{diagonal of } A$ sometimes works

- Better: blend in some direct-methods ideas. . .

Preconditioned conjugate gradient iteration

$$\mathbf{x}_0 = \mathbf{0}, \quad \mathbf{r}_0 = \mathbf{b}, \quad \mathbf{d}_0 = \mathbf{B}^{-1} \mathbf{r}_0, \quad \mathbf{y}_0 = \mathbf{B}^{-1} \mathbf{r}_0$$

for $k = 1, 2, 3, \dots$

$$\alpha_k = (\mathbf{y}_{k-1}^T \mathbf{r}_{k-1}) / (\mathbf{d}_{k-1}^T \mathbf{A} \mathbf{d}_{k-1}) \quad \text{step length}$$

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \alpha_k \mathbf{d}_{k-1} \quad \text{approx solution}$$

$$\mathbf{r}_k = \mathbf{r}_{k-1} - \alpha_k \mathbf{A} \mathbf{d}_{k-1} \quad \text{residual}$$

$$\mathbf{y}_k = \mathbf{B}^{-1} \mathbf{r}_k \quad \text{preconditioning solve}$$

$$\beta_k = (\mathbf{y}_k^T \mathbf{r}_k) / (\mathbf{y}_{k-1}^T \mathbf{r}_{k-1}) \quad \text{improvement}$$

$$\mathbf{d}_k = \mathbf{y}_k + \beta_k \mathbf{d}_{k-1} \quad \text{search direction}$$

- One matrix-vector multiplication per iteration
- One solve with preconditioner per iteration

Other Krylov subspace methods

- Nonsymmetric linear systems:
 - GMRES:
for $i = 1, 2, 3, \dots$
find $x_i \in K_i(A, b)$ such that $r_i = (Ax_i - b) \perp K_i(A, b)$
But, no short recurrence \Rightarrow save old vectors \Rightarrow lots more space (Usually “restarted” every k iterations to use less space.)
 - BiCGStab, QMR, etc.:
Two spaces $K_i(A, b)$ and $K_i(A^T, b)$ w/ mutually orthogonal bases Short recurrences $\Rightarrow O(n)$ space, but less robust
 - Convergence and preconditioning more delicate than CG
 - Active area of current research
- Eigenvalues: **Lanczos** (symmetric), **Arnoldi** (nonsymmetric)

Formulation: Residual

Min $\|r\|_2 = \|b - Ax\|_2$, for an arbitrary square matrix A

Min $R(x) = (b - Ax)^T (b - Ax)$

Search space: $x = x_0 + Vy$

where x_0 is an initial solution,

matrix $V_{n \times m}$ has m bases of subspace K

vector y_m contains the m variables.

Solution: Residual

$$\text{Min } R(x) = (b - Ax)^T (b - Ax)$$

$$\text{Search space: } x = x_0 + Vy$$

1. $V^T A^T A V$ is nonsingular if A is nonsingular and V is full ranked.
2. $y = (V^T A^T A V)^{-1} V^T A^T r_0$
3. $x = x_0 + V (V^T A^T A V)^{-1} V^T A^T r_0$
4. $V^T A^T r = 0$
5. $R(x) = R(x_0) - r_0^T A V (V^T A^T A V)^{-1} V^T A^T r_0$

Steepest Descent: Residual

$$\text{Min } R(x) = (b - Ax)^T(b - Ax)$$

$$\text{Gradient: } -2A^T(b - Ax) = -2A^T r$$

$$\text{Let } x = x_0 + yA^T r_0$$

1. $V^T A^T A V$ is nonsingular if A is nonsingular where $V = A^T r_0$.
2. $y = (V^T A^T A V)^{-1} V^T A^T r_0$
3. $x = x_0 + V(V^T A^T A V)^{-1} V^T A^T r_0$
4. $V^T A^T r = 0$
5. $R(x) = R(x_0) - r_0^T A V (V^T A^T A V)^{-1} V^T A^T r_0$

GMRES: Residual

$$\text{Min } R(x) = (b - Ax)^T (b - Ax)$$

$$\text{Gradient: } -2A^T(b - Ax) = -2A^T r$$

$$\text{Let } x = x_0 + yA^T r_0$$

1. $v_1 = r_0$
2. v_i is in $K\{r_0, A, i\}$
3. $V = [v_1, v_2, \dots, v_m]$ is orthogonal
4. $AV = VH_m + v_{m+1} e_m^T = V_{m+1} \underline{H}_m$
5. $x = x_0 + V(V^T A^T A V)^{-1} V^T A^T r_0$
6. $= x_0 + V(\underline{H}_m^T \underline{H}_m)^{-1} \underline{H}_m^T e_1 \|r_0\|_2$

Conjugate Residual: Residual

$$\text{Min } R(x) = (b - Ax)^T (b - Ax)$$

$$\text{Gradient: } -2A^T(b - Ax) = -2A^T r$$

$$\text{Let } x = x_0 + yA^T r_0$$

1. $v_1 = r_0$
2. v_i is in $K\{r_0, A, i\}$
3. $(AV)^T AV = D$ Diagonal Matrix
4. $x = x_0 + V(V^T A^T AV)^{-1} V^T A^T r_0$
5. $= x_0 + VD^{-1} V^T A^T r_0$

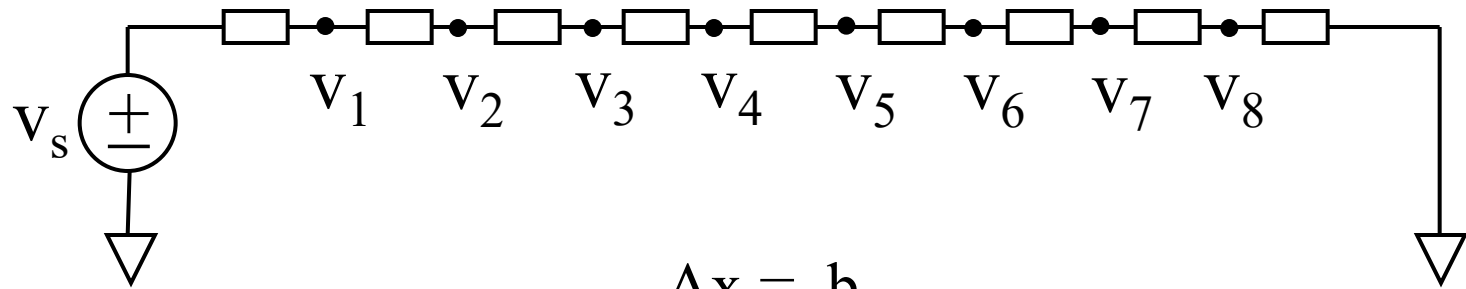
Outline

- Iterative Method
 - Stationary Iterative Method (SOR, GS, Jacob)
 - Krylov Method (CG, GMRES)
 - Multigrid Method

What is the multigrid

- A multilevel iterative method to solve
 - $Ax=b$
- Originated in PDEs on geometric grids
- Expand the multigrid idea to unstructured problem – Algebraic MG
- Geometric multigrid for presenting the basic ideas of the multigrid method.

The model problem



$$Ax = b$$
$$\begin{bmatrix} 2 & -1 & & & & & & \\ -1 & 2 & -1 & & & & & \\ & -1 & 2 & -1 & & & & \\ & & -1 & 2 & -1 & & & \\ & & & -1 & 2 & -1 & & \\ & & & & -1 & 2 & -1 & \\ & & & & & -1 & 2 & -1 \\ & & & & & & -1 & 2 \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \end{bmatrix} = \begin{bmatrix} v_s \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Simple iterative method

□ $\mathbf{x}^{(0)} \rightarrow \mathbf{x}^{(1)} \rightarrow \dots \rightarrow \mathbf{x}^{(k)}$

□ Jacobi iteration

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

□ Matrix form : $\mathbf{x}^{(k)} = \mathbf{R}_j\mathbf{x}^{(k-1)} + \mathbf{C}_j$

□ General form: $\mathbf{x}^{(k)} = \mathbf{R}\mathbf{x}^{(k-1)} + \mathbf{C} \quad (1)$

□ Stationary: $\mathbf{x}^* = \mathbf{R}\mathbf{x}^* + \mathbf{C} \quad (2)$

Error and Convergence

Definition: **error** $e = x^* - x$ (3)

residual $r = b - Ax$ (4)

e, r relation: $Ae = r$ (5) ((3)+(4))

$$e^{(1)} = x^* - x^{(1)} = Rx^* + C - Rx^{(0)} - C \\ = Re^{(0)}$$

Error equation $e^{(k)} = R^k e^{(0)}$ (6)
((1)+(2)+(3))

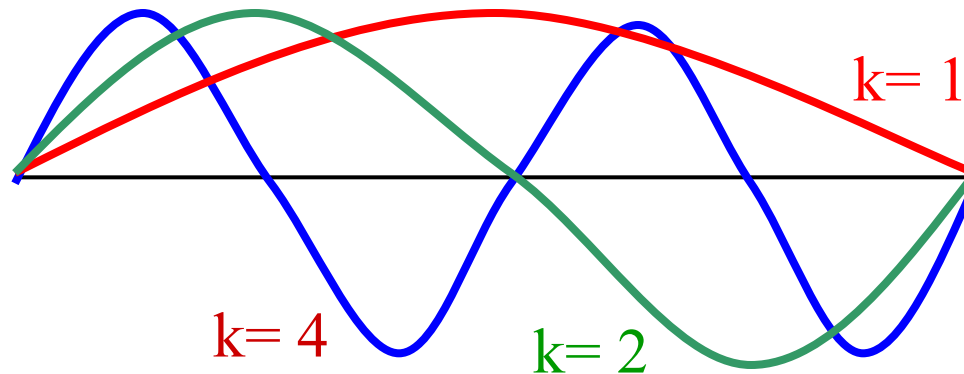
$$\lim_{k \rightarrow \infty} \|e^{(k)}\| = 0$$

Convergence:

Error of different frequency

- Wavenumber k and frequency ω

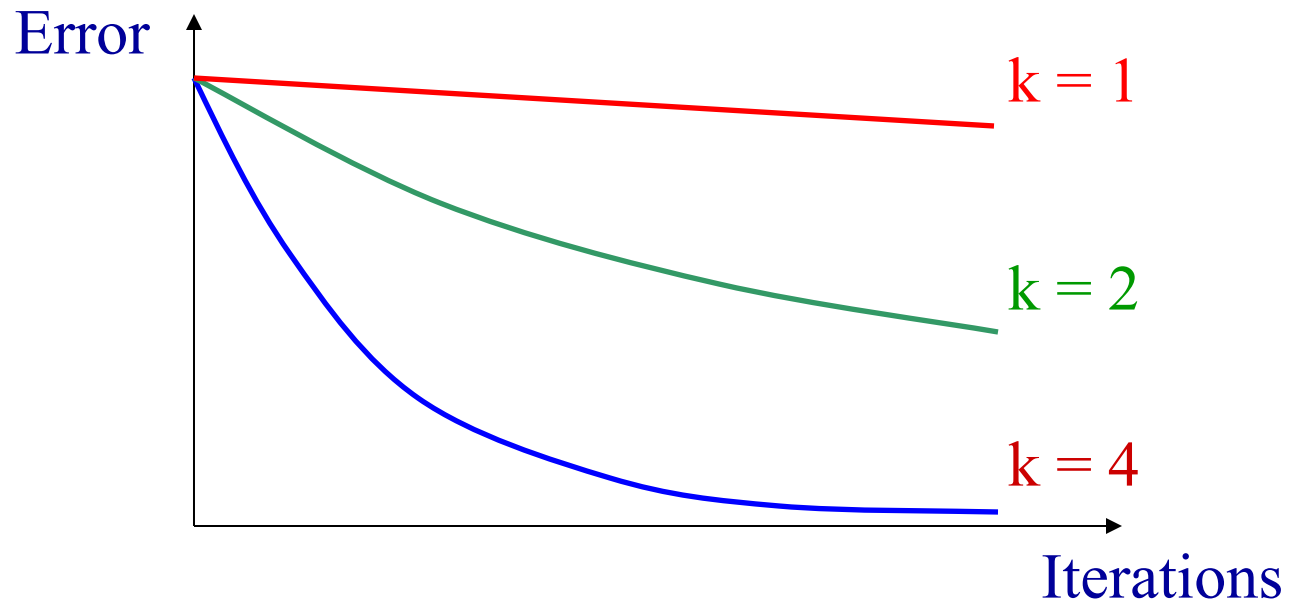
$$\omega = k\pi/n$$



- High frequency error is more oscillatory between points

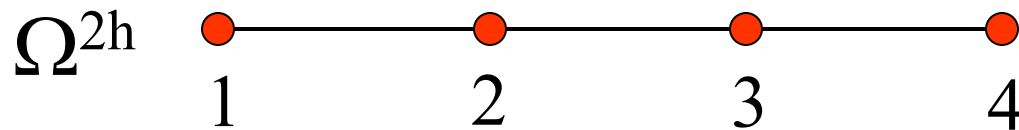
Iteration reduce low frequency error efficiently

- Smoothing iteration reduce high frequency error efficiently, but not low frequency error

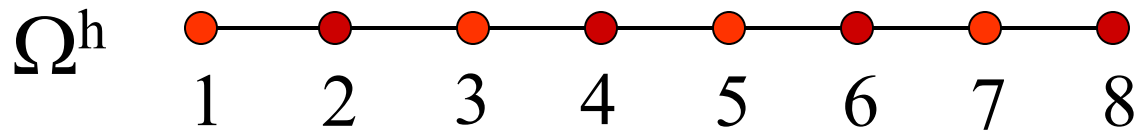


Multigrid – a first glance

- Two levels : coarse and fine grid



$$A^{2h}x^{2h}=b^{2h}$$



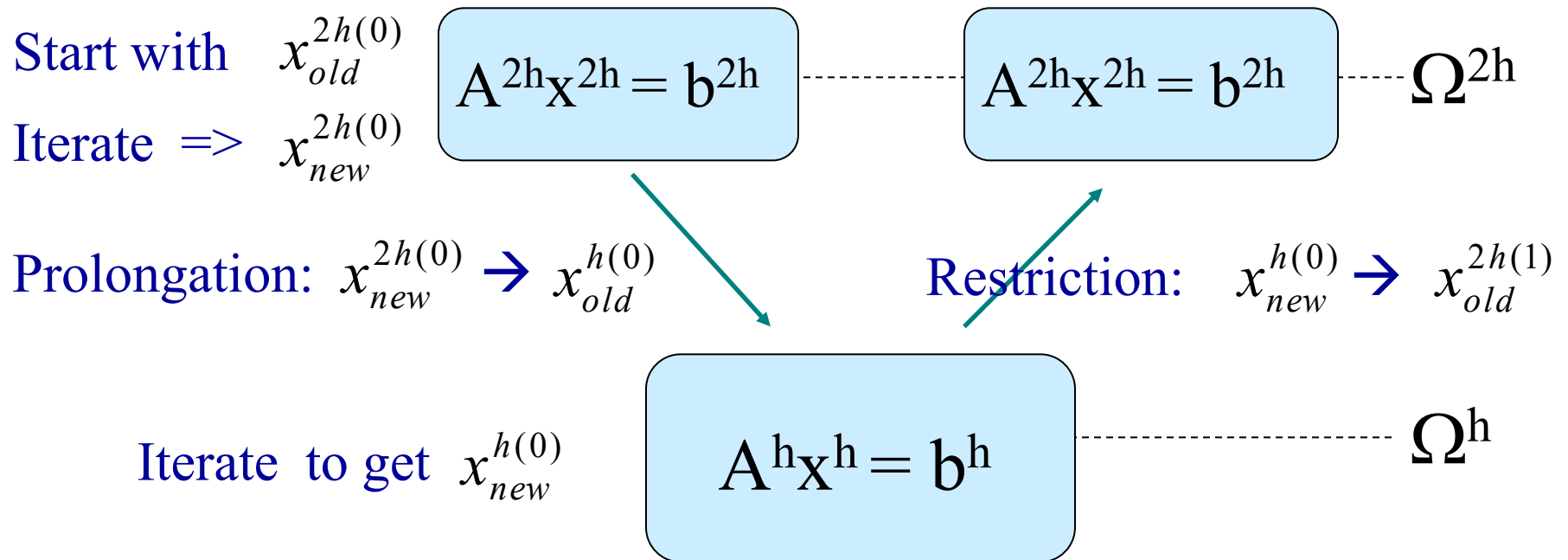
$$A^h x^h = b^h$$



$$Ax=b$$

Idea 1: the V-cycle iteration

□ Also called the nested iteration



Question 1: Why we need the coarse grid ?

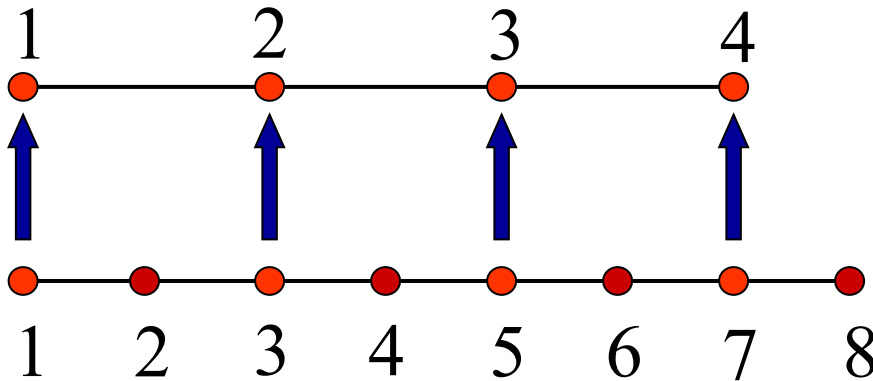
Restriction

□ Restriction operator

$$\mathbf{x}^h = \mathbf{x}^{2h}$$

$$I_h^{2h}$$

$$I_h^{2h}$$



$$I_h^{2h} = \begin{bmatrix} 1 & 0 & & & & & & \\ & 0 & 1 & 0 & & & & \\ & & & 0 & 1 & 0 & & \\ & & & & & 0 & 1 & 0 \\ & & & & & & & 0 & 1 & 0 \end{bmatrix}$$

Smoothing

- The basic iterations in each level

$$\text{In } \Omega^{\text{ph}}: x_{\text{old}}^{\text{ph}} \rightarrow x_{\text{new}}^{\text{ph}}$$

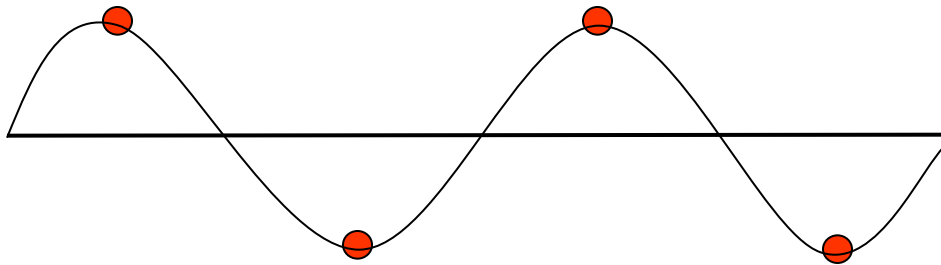
- Iteration reduces the error, makes the error **smooth** geometrically.
So the iteration is called **smoothing**.

Why multilevel ?

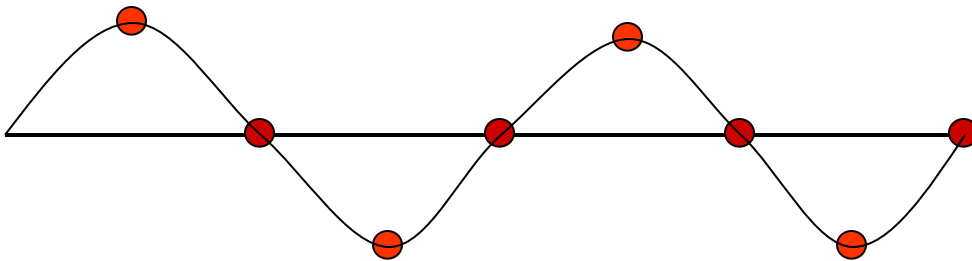
- Coarse level iteration is cheap.
- More than this...
 - Coarse level smoothing reduces the error more efficiently than fine level in some way .
 - Why ? (Question 2)

Error restriction

- Map error to coarse grid will make the error more oscillatory



$$K = 4, \omega = \pi$$



$$K = 4, \omega = \pi/2$$

Idea 2: Residual correction

- Known current solution x
- Solve $Ax=b$ eq. to $\begin{cases} \text{Solve } Ae^* = r \\ x^* = x + e^* \end{cases}$
- MG do **NOT** map x directly between levels

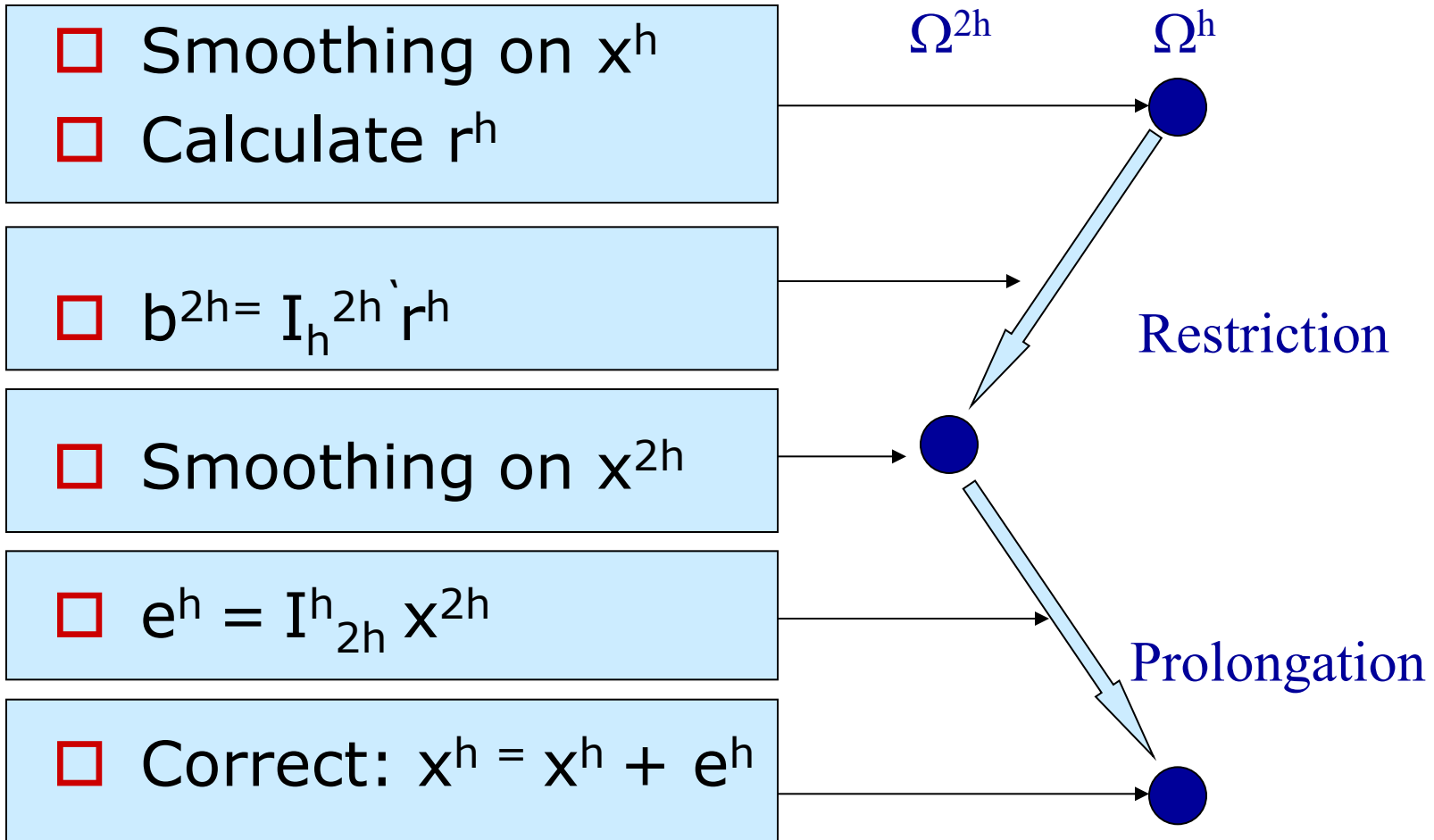
Map residual equation to coarse level

1. Calculate r^h
2. $b^{2h} = I_h^{2h} r^h$ (Restriction)
3. $e^h = I_{2h}^h x^{2h}$ (Prolongation)
4. $x^h = x^h + e^h$

Why residual correction ?

- ❑ Error is smooth at fine level, but the actual solution may not be.
- ❑ Prolongation results in a **smooth** error in fine level, which is suppose to be a **good** evaluation of the fine level error.
- ❑ If the solution is not smooth in fine level, prolongation will introduce more high frequency error.

Revised V-cycle with idea 2



What is A^{2h}

□ Galerkin condition

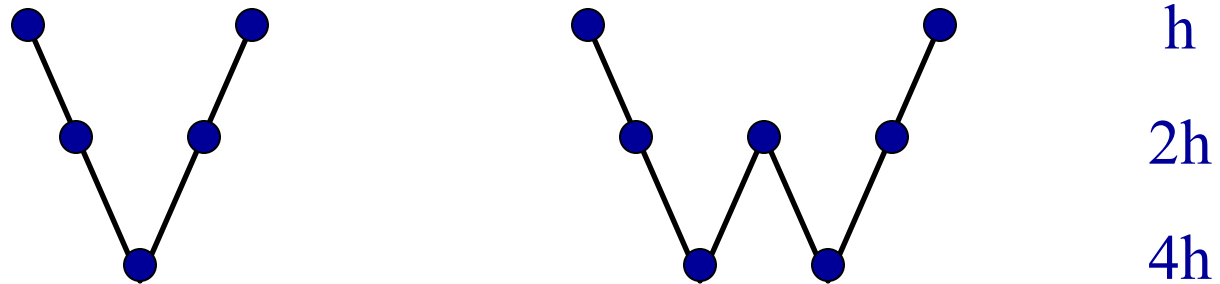
$$A^{2h} = I_h^{2h} A^h I_{2h}^h$$

$$e^h = I_{2h}^h x^{2h} \quad A^h e^h = A^h I_{2h}^h x^{2h} = r^h$$

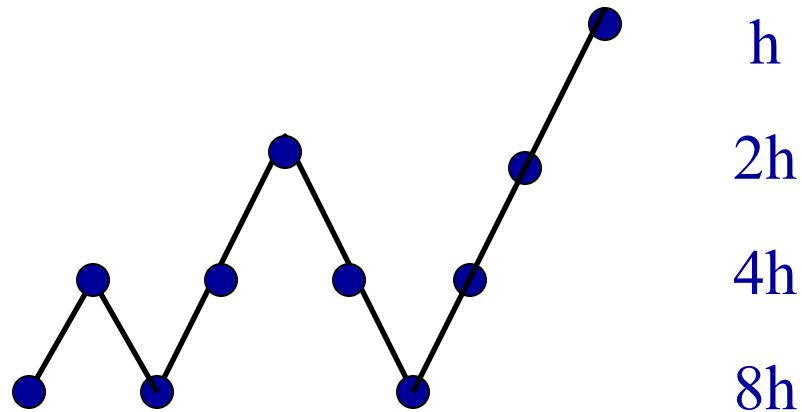
$$I_h^{2h} A^h I_{2h}^h x^{2h} = I_h^{2h} r^h \quad \Leftrightarrow \quad A^{2h} x^{2h} = b^{2h}$$

Going to multilevels

□ V-cycle and W-cycle



□ Full Multigrid V-cycle



Performance of Multigrid

□ Complexity comparison

Gaussian elimination	$O(N^2)$
Jacobi iteration	$O(N^2 \log \varepsilon)$
Gauss-Seidel	$O(N^2 \log \varepsilon)$
SOR	$O(N^{3/2} \log \varepsilon)$
Conjugate gradient	$O(N^{3/2} \log \varepsilon)$
Multigrid (iterative)	$O(N \log \varepsilon)$
Multigrid (FMG)	$O(N)$

Summary of MG ideas

Important ideas of MG

1. Hierarchical iteration
2. Residual correction
3. Galerkin condition
4. Smoothing the error:
high frequency : fine grid
low frequency : coarse grid

AMG :for unstructured grids

- $Ax=b$, no regular grid structure
- Fine grid defined from A

$A =$

$$\begin{bmatrix} 3 & -1 & -1 & -1 & 0 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ -1 & -1 & 4 & 0 & 0 & 0 \\ -1 & 0 & 0 & 2 & 0 & -1 \\ 0 & -1 & 0 & 0 & 3 & -1 \\ 0 & 0 & 0 & -1 & -1 & 3 \end{bmatrix}$$

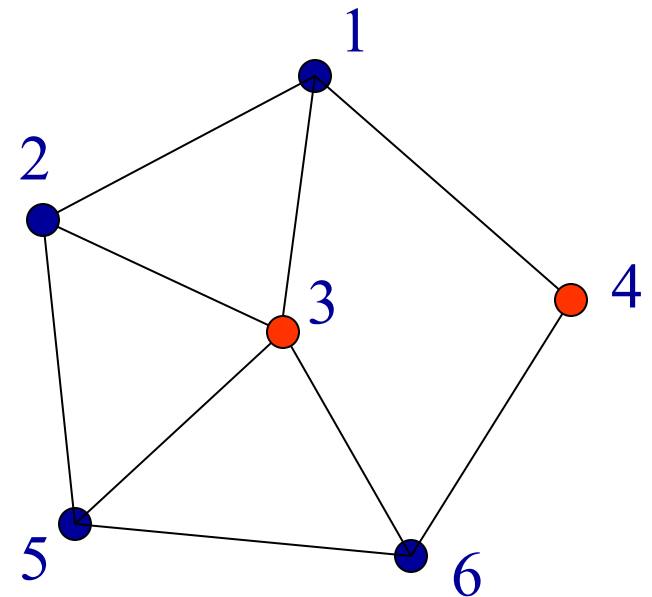


Three questions for AMG

- How to choose coarse grid
- How to define the smoothness of errors
- How are interpolation and prolongation done

How to choose coarse grid

- Idea:
 - C/F splitting
 - As few coarse grid point as possible
 - For each F-node, at least one of its neighbor is a C-node
 - Choose node with strong coupling to other nodes as C-node



How to define the smoothness of error

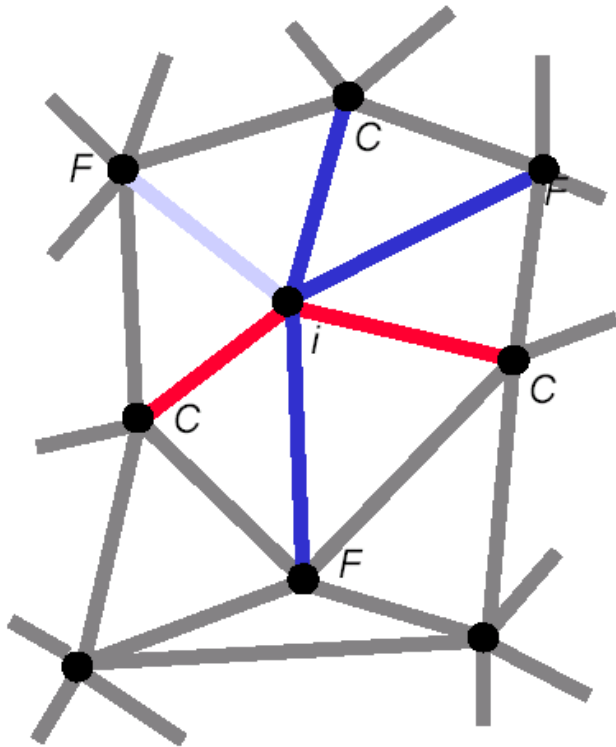
□ AMG fundamental concept:

Smooth error = small residuals

□ $\|r\| \ll \|e\|$

How are Prolongation and Restriction done

- Prolongation is based on smooth error and strong connections



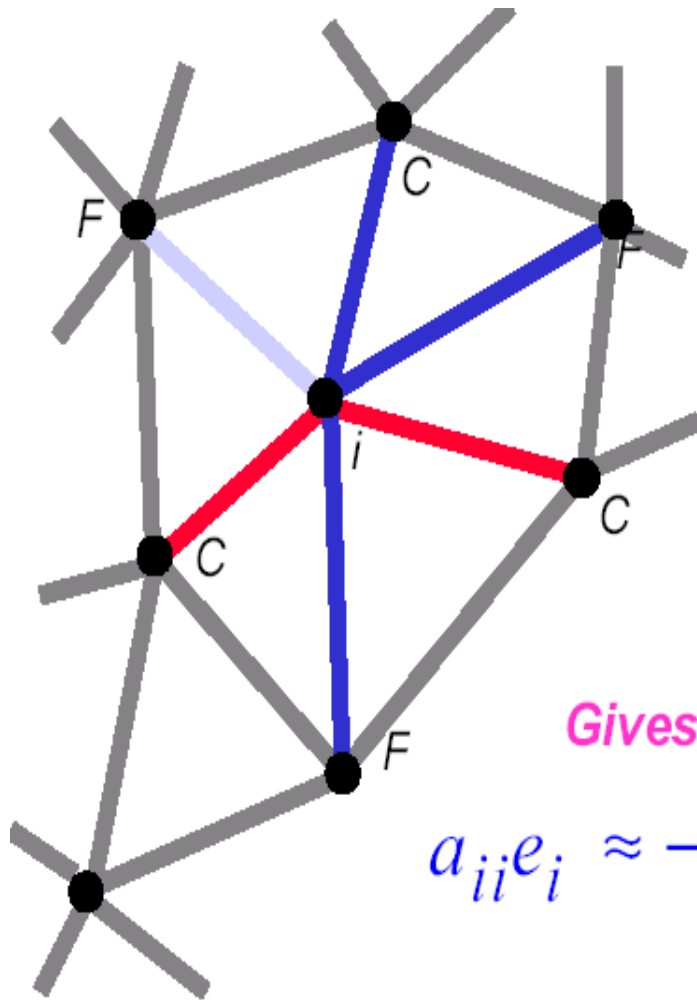
Smooth error is given by:

$$r_i = a_{ii} e_i + \sum_{j \in N_i} a_{ij} e_j \approx 0$$

Prolongation :

$$(Pe)_i = \begin{cases} e_i & , i \in C \\ \sum_{k \in C_i} \omega_{ik} e_k & , i \in F \end{cases}$$

AMG Prolongation (2)



Sets:

- C_i — Strongly connected **C**-pts.
- D_i^s — Strongly connected **F**-pts.
- D_i^w — Weakly connected points.

The definition of smooth error,

$$a_{ii}e_i \approx - \sum_{j \neq i} a_{ij}e_j$$

Gives:

$$a_{ii}e_i \approx - \sum_{j \in C_i} a_{ij}e_j - \sum_{j \in D_i^s} a_{ij}e_j - \sum_{j \in D_i^w} a_{ij}e_j$$

Strong C

Strong F

Weak pts.

AMG Prolongation (3)

- In the smooth-error relation, use $e_j = e_i$ for weak connections. For the strong F -points use :

$$e_j = \left(\sum_{k \in C_i} a_{jk} e_k \right) / \left(\sum_{k \in C_i} a_{jk} \right)$$

yielding the prolongation weights:

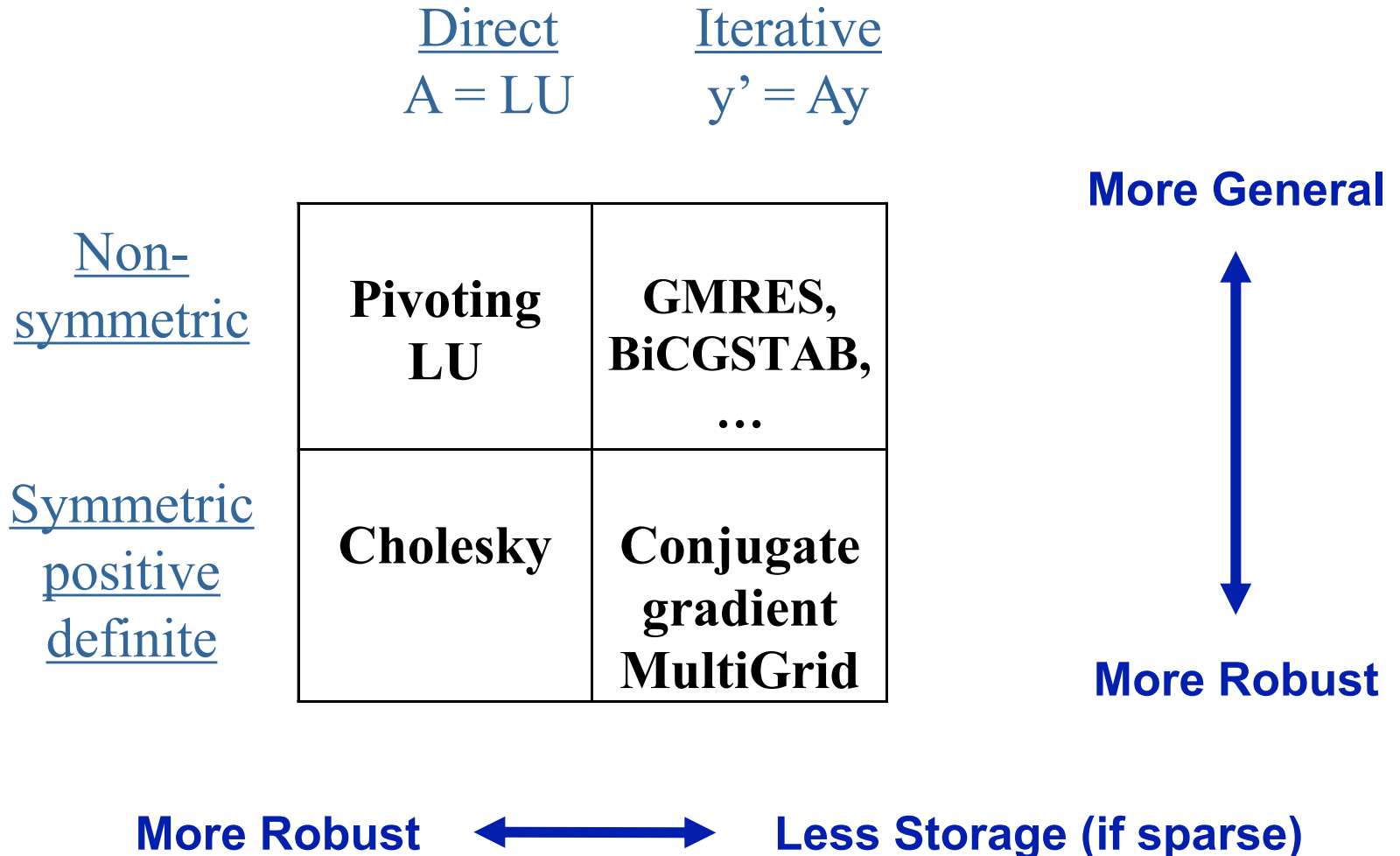
□ Restriction : $w_{ij} = - \frac{a_{ij} + \sum_{j \in D_i^s} \frac{a_{ik} a_{kj}}{\sum_{m \in C_i} a_{km}}}{a_{ii} + \sum_{n \in D_i^w} a_{in}}$

$$I_h^{2h} = C(I_{2h}^h)^T$$

Summary

- ❑ Multigrid is a multilevel iterative method.
- ❑ Advantage: scalable
- ❑ If no geometrical grid is available, try Algebraic multigrid method

The landscape of Solvers



References

- G.H. Golub and C.F. Van Loan, Matrix Computations, 4th Edition, Johns Hopkins, 2013
- Y. Saad, Iterative Methods for Sparse Linear Systems, Second Edition, SIAM, 2003.