Challenges for Matrix Preconditioning Methods

Matthias Bollhoefer[1]

[1]Dept. of Mathematics
TU Berlin

Preconditioning 2005, Atlanta, May 19, 2005

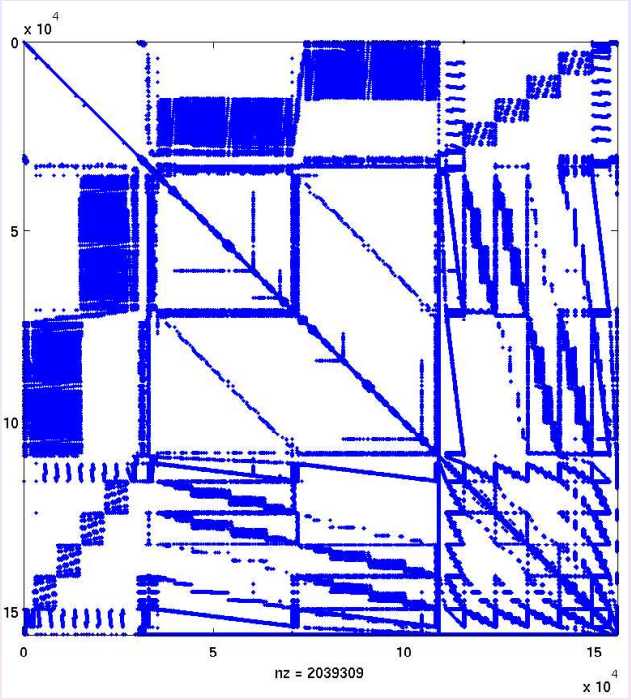
supported by the DFG research center MATHEON in Berlin

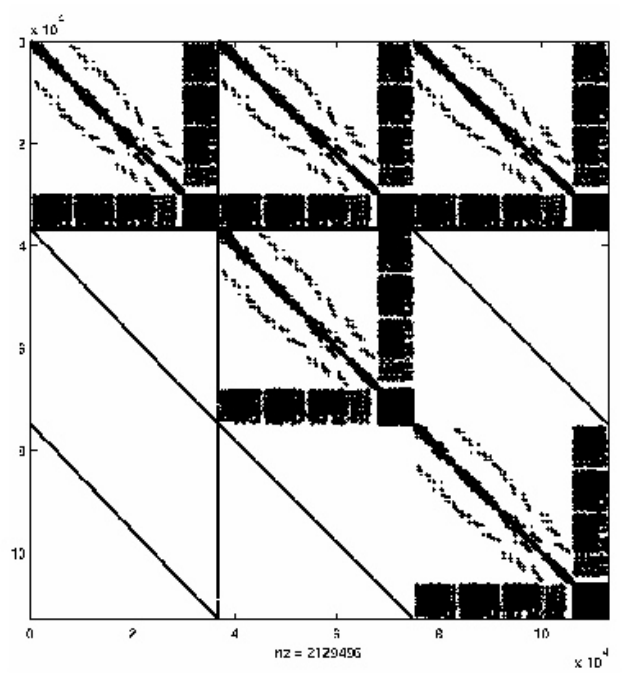
Outline

Outline

Circuit and device simulation

- Circuits
  - Modified nodal approach
  - transient analysis





- Devices
  - harmonic balance
  - drift diffusion equations

Outline

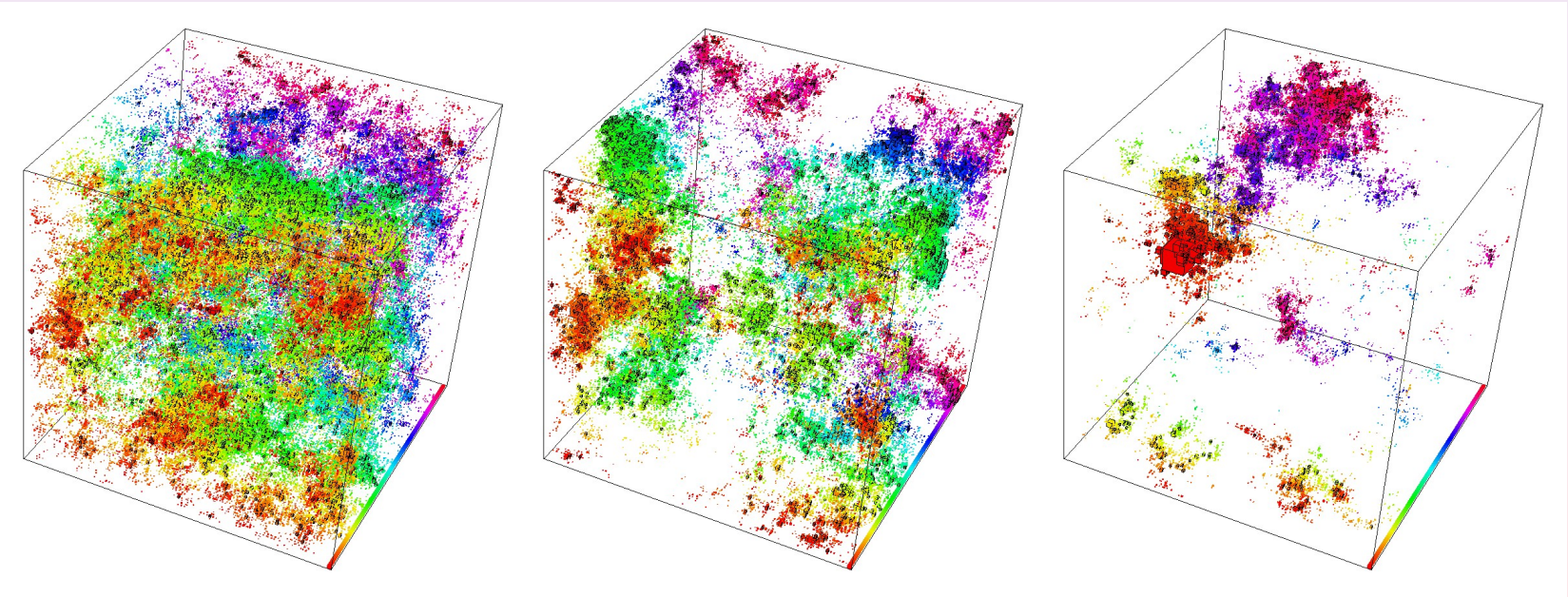
The Anderson model of localization

A Challenge for modern eigenvalue algorithms

Model describes electronic transport properties in disordered systems

Wave function probabilities

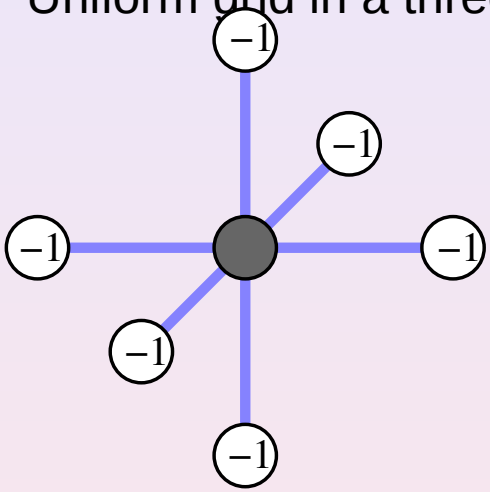
$\omega = 14.5$ $\omega = 16.5$ $\omega = 18.5$



The Anderson model of localization

A Challenge for modern eigenvalue algorithms

Uniform grid in a three–dimensional cube



$$\varepsilon_{ijk} x_{ijk} - \sum_{|p-i|=1,|q-j|=1,|r-j|=1} x_{pqr} = \lambda x_{ijk}$$

$\varepsilon_{ijk} \in [-\omega/2, \omega/2]$, ω radomly in $[1, 30]$.

- Eigenvalue problem $Ax = \lambda x$ in 3D
- x quantum mechanic wave function

- $\omega \approx 16.5$ critical range
- $\omega \ll 16.5 \rightarrow$ fluctuations, but bounded
- $\omega \gg 16.5 \rightarrow$ wave functions are localized

Challenges

- Physically sensible results require large scale simulation, $n = m^3$(e.g. $m = 100, 200, \ldots$)
- physically interesting: eigenvectors around $\lambda = 0$ at $\omega_c = 16.5$
- $\rightarrow$ requires eigenvalue solver which
  - compute some eigenvectors around $\lambda = 0$ at ω_c
  - are fast (some hours up to a few days)
  - are memory efficient (required memory scales $\sim n$)

Challenges

- Physically sensible results require large scale simulation, $n = m^3$(e.g. $m = 100, 200, \ldots$)
- physically interesting: eigenvectors around $\lambda = 0$ at $\omega_c = 16.5$
- $\rightarrow$ requires eigenvalue solver which
  - compute some eigenvectors around $\lambda = 0$ at ω_c
  - are fast (some hours up to a few days)
  - are memory efficient (required memory scales $\sim n$)

Numerical Approach

Preconditioned Eigenvalue Solver

Outline

What are Matchings

Example

$$\begin{pmatrix} * & & * & & * \\ & & * & * & \\ * & * & & * & \\ & * & * & & \\ * & & & & \end{pmatrix}$$

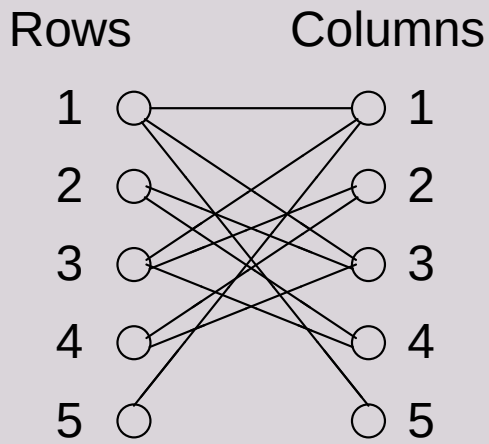
original A

What are Matchings

Example

$$\begin{pmatrix} * & & * & & * \\ & & * & * & \\ * & * & & * & \\ & * & * & & \\ * & & & & \end{pmatrix}$$

original A



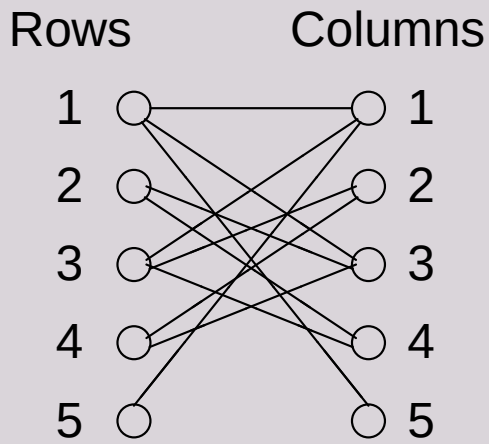
- Associated graph

What are Matchings

Example

$$\begin{pmatrix} * & & * & & * \\ & & * & * & \\ * & * & & * & \\ & * & * & & \\ * & & & & \end{pmatrix}$$

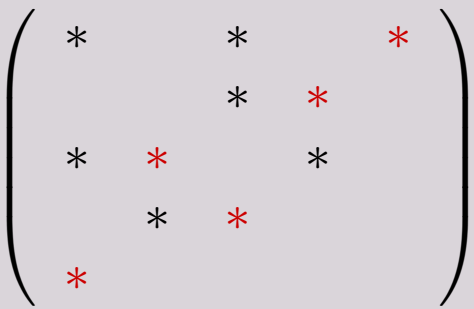
original A



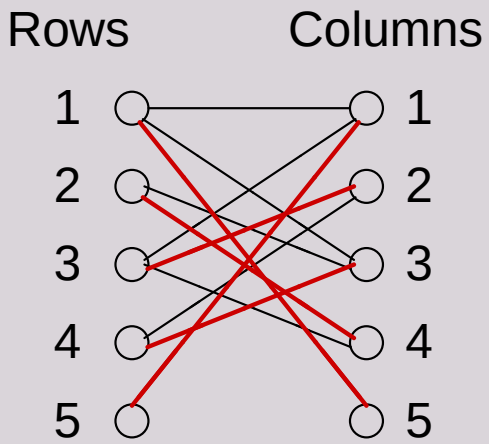
- Associated graph
- (Perfect) matching. (At most) one edge for each row/column

What are Matchings

Example



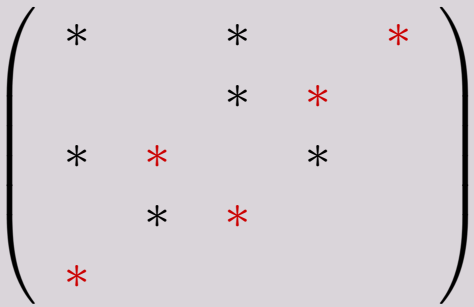
original A



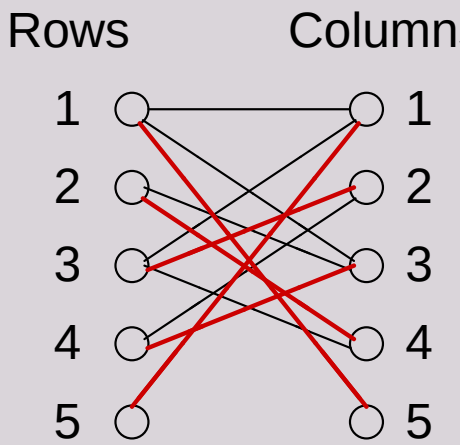
- (Perfect) matching. (At most) one edge for each row/column

What are Matchings

Example



original A

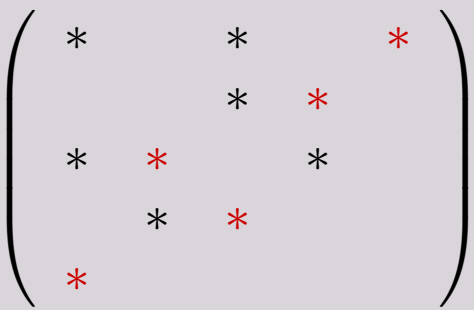


$\pi = (5, 3, 4, 2, 1)$

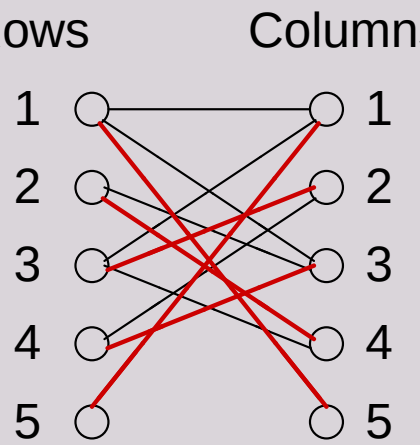
- Associated permutation

What are Matchings

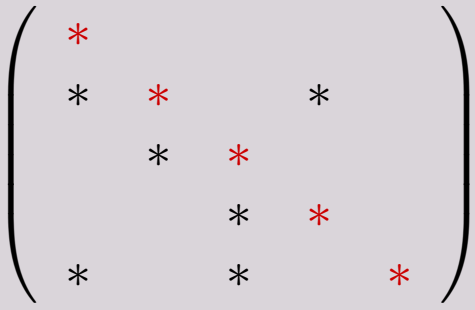
Example



original A



$\pi = (5, 3, 4, 2, 1)$



permuted $\Pi^\top A$

- Permuted matrix

Minimum Weight Matchings

- Matching $\longrightarrow$ Permutation of the matrix to zero–free representation ['MC21', Duff'77]

Refined Objective: Strengthening diagonal dominance

- Find matching such that $\prod_{i=1}^{n} |a_{\pi(i),i}|$ is maximized

Minimum Weight Matchings

- Matching $\rightarrow$ Permutation of the matrix to zero–free representation ['MC21', Duff'77]

Refined Objective: Strengthening diagonal dominance

- Find matching such that $\prod_{i=1}^{n} |a_{\pi(i),i}|$ is maximized

- dense case [Olschowka, Neumeier'96],
- sparse case ['MC64', Duff,Koster'99]

Minimum Weight Matchings

- reformulation: minimize $\sum_{i=1}^{n} |c_{\pi(i),i}|$, where

$$c_{ij} = \begin{cases} \max_j |a_{ij}| - \log |a_{ij}| & a_{ij} \neq 0 \\ \infty & \text{otherwise} \end{cases}$$

- known linear–sum assignment problem

Minimum Weight Matchings

- reformulation: minimize $\sum_{i=1}^{n} |c_{\pi(i),i}|$, where

$$c_{ij} = \begin{cases} \max_j |a_{ij}| - \log |a_{ij}| & a_{ij} \neq 0 \\ \infty & \text{otherwise} \end{cases}$$

- known linear–sum assignment problem
- solution leads to two vectors $(u_i)_i$, $(v_j)_j$ such that
  - $u_i + u_j \leqslant c_{ij}$ and if (i,j) are part of the matching $u_i + u_j = c_{ij}$
  - diagonal scalings $D_r = \mathrm{dgl}\,(e^{u_i})_i$, $D_c = \mathrm{dgl}\,(e^{v_i} / \max_k |a_{jk}|)_j$

Minimum Weight Matchings

- reformulation: minimize $\sum_{i=1}^{n} |c_{\pi(i),i}|$, where

$$c_{ij} = \begin{cases} \max_j |a_{ij}| - \log |a_{ij}| & a_{ij} \neq 0 \\ \infty & \text{otherwise} \end{cases}$$

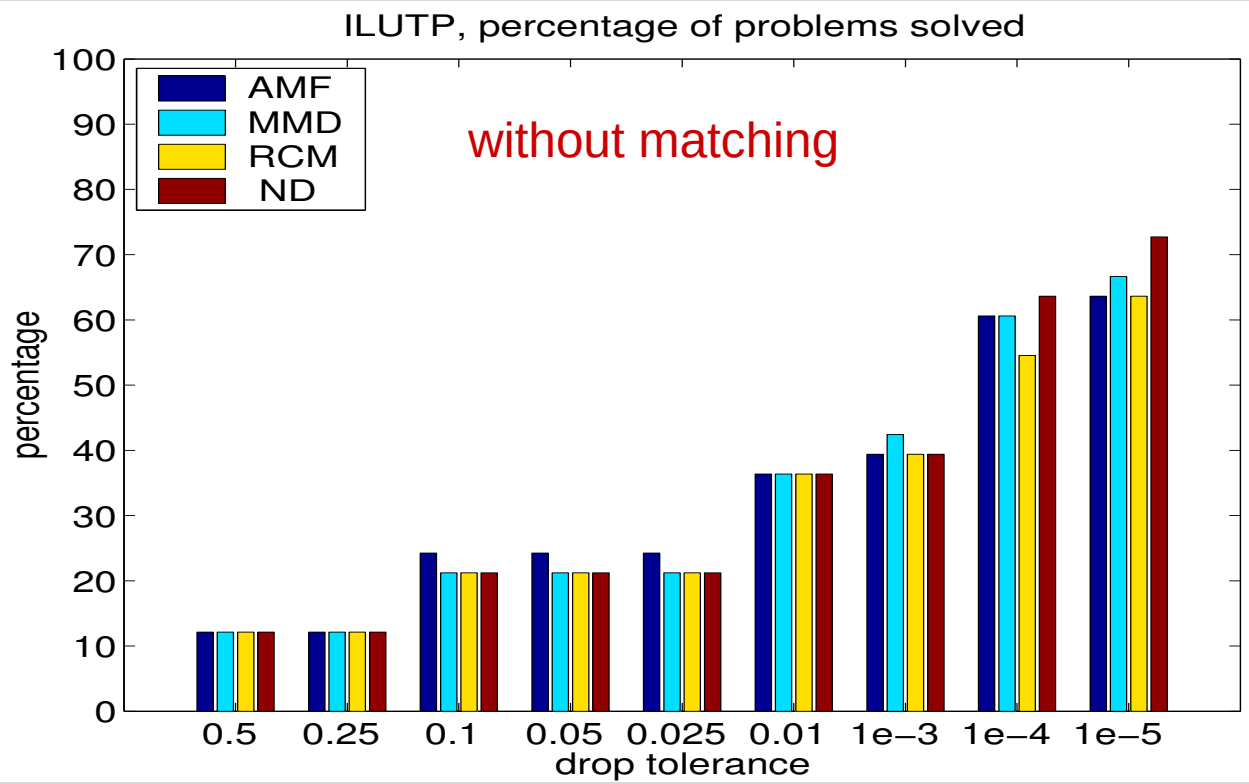
- known linear–sum assignment problem
- solution leads to two vectors $(u_i)_i$, $(v_j)_j$ such that
  - $u_i + u_j \leqslant c_{ij}$ and if (i,j) are part of the matching $u_i + u_j = c_{ij}$
  - diagonal scalings $D_r = \mathrm{dgl}\,(e^{u_i})_i$, $D_c = \mathrm{dgl}\,(e^{v_i} / \max_k |a_{jk}|)_j$
- yields permutation Π and diagonal scalings D_r and D_c.

$$A \to \Pi^\top D_r A D_c \text{ such that } |a_{ii}| = 1, |a_{ij}| \leqslant 1 \text{ for all } i,j$$

How matchings improve iterative solvers

Example

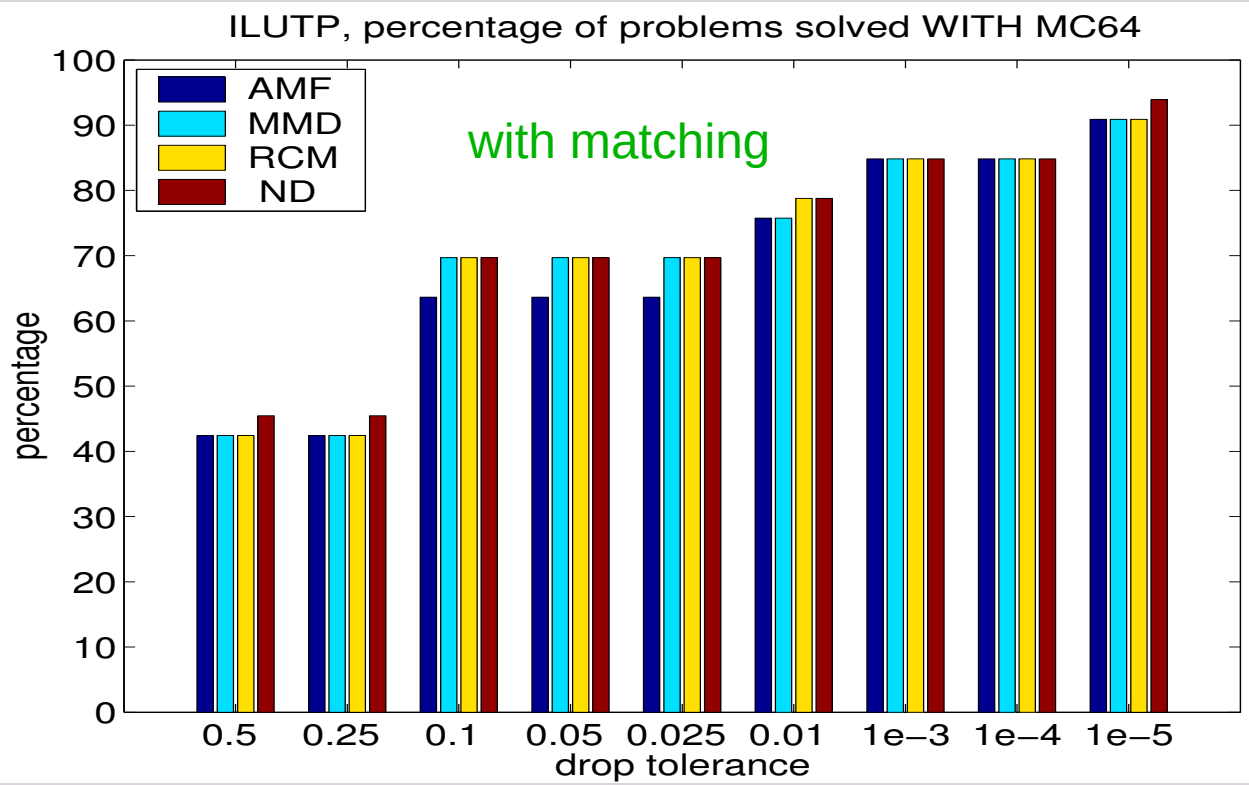
- 33 large sparse unstructured systems from chemical engineering
- Simple dual threshold ILU with pivoting from SPARSKIT [Saad'94]
- Restarted GMRES(30), check convergence after at most 500 steps



How matchings improve iterative solvers

Example

- 33 large sparse unstructured systems from chemical engineering
- Simple dual threshold ILU with pivoting from SPARSKIT [Saad'94]
- Restarted GMRES(30), check convergence after at most 500 steps



How matchings improve iterative solvers

- Minimum weight matchings significantly improve preconditioning methods [Benzi,Haws,Tuma'00]

  also applicable to factored sparse approximate inverses

- High potential also as part direct solvers (PARDISO, SuperLU,. . .)
- Matchings allow
  - Level 3 BLAS with static pivoting
  - use of symmetric reordering techniques

How matchings improve iterative solvers

- Minimum weight matchings significantly improve preconditioning methods [Benzi,Haws,Tuma'00]

  also applicable to factored sparse approximate inverses

- High potential also as part direct solvers (PARDISO, SuperLU,. . .)
- Matchings allow
  - Level 3 BLAS with static pivoting
  - use of symmetric reordering techniques

BUT

- matching destroys symmetry structures
- leads to unsymmetric scaling
- in this way not directly applicable to symmetrically structured systems

Outline

Basic Idea of Symmetric Matchings

Symmetric matchings [Duff,Gilbert'02], [Duff,Pralet'04]

Example

$$\begin{pmatrix} * & & * & & * \\ & & * & * & \\ * & * & & * & \\ & * & * & & \\ * & & & & \end{pmatrix} \qquad \pi = (5,3,4,2,1)$$

original A

Basic Idea of Symmetric Matchings

Symmetric matchings [Duff,Gilbert'02], [Duff,Pralet'04]

- Decompose permutation as product of cycles

Example

$$\begin{pmatrix} * & & * & & * \\ & & * & * & \\ * & * & & * & \\ & * & * & & \\ * & & & & \end{pmatrix}$$

original A

$$\begin{aligned} \pi &= (5,3,4,2,1) \\ &= (5,1)(3,4,2) \end{aligned}$$

Basic Idea of Symmetric Matchings

Symmetric matchings [Duff,Gilbert'02], [Duff,Pralet'04]

- Decompose permutation as product of cycles
- each cycle is associated with a diagonal block, pivots stay there

Example

$$\begin{pmatrix} * & & * & & * \\ & & * & * & \\ * & * & & * & \\ & * & * & & \\ * & & & & \end{pmatrix} \quad \pi = (5,3,4,2,1) = (5,1)(3,4,2) \quad \begin{pmatrix} * & & & & \\ * & * & & * & \\ & * & * & & \\ & & * & * & \\ * & & * & & * \end{pmatrix}$$

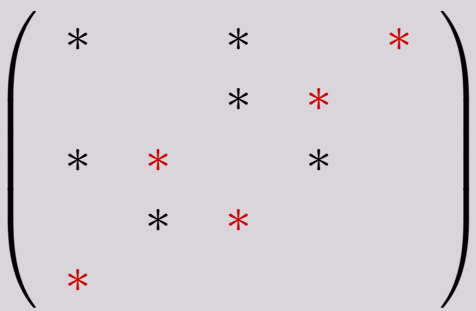
original A permuted $\Pi^\top A$

Basic Idea of Symmetric Matchings

Symmetric matchings [Duff,Gilbert'02], [Duff,Pralet'04]

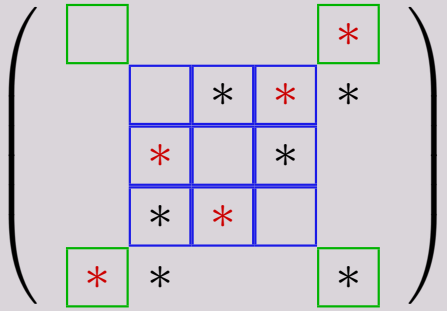
- Decompose permutation as product of cycles
- each cycle is associated with a diagonal block, pivots stay there
- pivots stay inside if related SYMMETRIC permutation is applied

Example



original A

$$\pi = (5,3,4,2,1) = (5,1)(3,4,2)$$

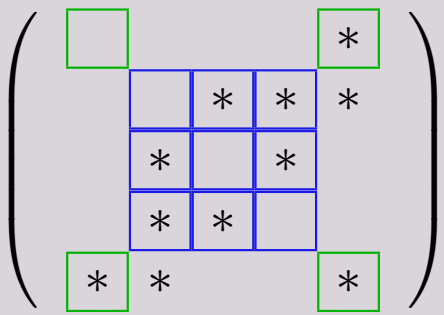


permuted $\Pi^\top A \Pi$

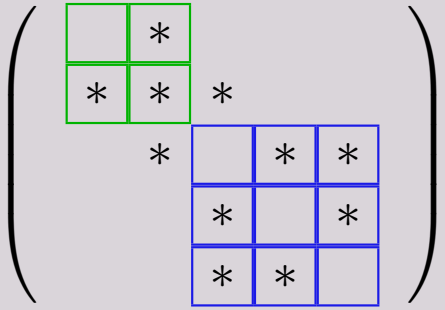
Basic Idea of Symmetric Matchings

- break up cycles into pairs of 2–cycles and group them together
  - various strategies of breaking up the cycles [Duff,Gilbert'02],[Duff,Pralet'04],[Schenk,Hagemann'04]
  - long cycles rarely show up in practice [Duff,Pralet'04]
- Scaling replaced by symmetric scaling $D = (D_c D_r)^{1/2}$ ensures that entries are still at most 1 in modulus

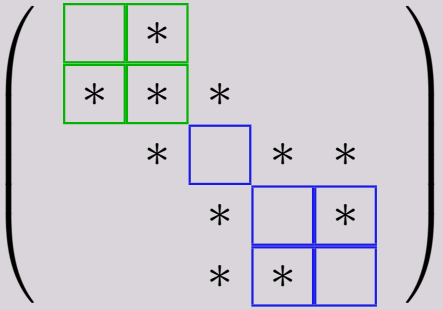
Example



$\Pi^\top A \Pi$



rearranged $\Pi_1^\top A \Pi_1$



final $\Pi_2^\top A \Pi_2$

Outline

Symbolic reorderings techniques

- multilevel nested dissection

  $\rightarrow$ MeTiS [Karypis,Kumar'95]

- Approximate minimum degree [Amestoy,Davis,Duff,Gilbert,Larimore,Ng]

  $\rightarrow$ AMD, UMFPACK

- further old–fashioned orderings (Reverse Cuthill–McKee, Minimum Degree)

- Different approach: diagonal dominance + sparsity [Saad'03]

  alternative to matching $+$ reordering

Outline

Algebraic Multilevel Approach

$$\text{Reordering } (+ \text{ rescaling}) \text{ the system} \longrightarrow \begin{cases} \mathcal{F} & \text{“fine grid points”} \\ \mathcal{C} & \text{“coarse grid points”} \end{cases}$$

$$A \longrightarrow \Pi^{\top} A \Pi = \begin{pmatrix} A_{\mathcal{F}\mathcal{F}} & A_{\mathcal{F}\mathcal{C}} \\ A_{\mathcal{C}\mathcal{F}} & A_{\mathcal{C}\mathcal{C}} \end{pmatrix}$$

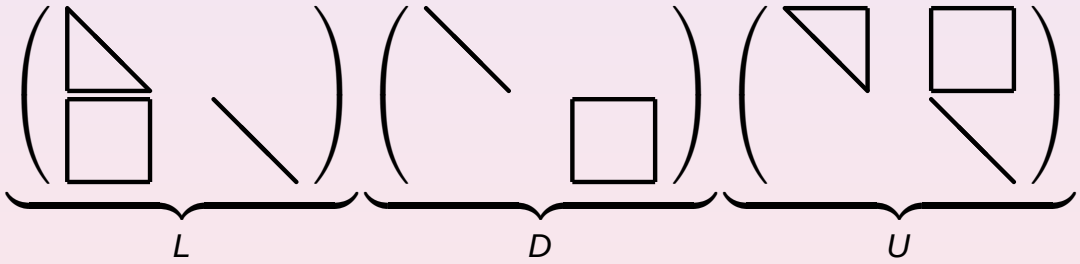
Algebraic Multilevel Approach

Reordering (+ rescaling) the system $\rightarrow \begin{cases} \mathcal{F} & \text{“fine grid points”} \\ \mathcal{C} & \text{“coarse grid points”} \end{cases}$

$$A \rightarrow \Pi^\top A \Pi = \begin{pmatrix} A_{\mathcal{F}\mathcal{F}} & A_{\mathcal{F}\mathcal{C}} \\ A_{\mathcal{C}\mathcal{F}} & A_{\mathcal{C}\mathcal{C}} \end{pmatrix}$$

Approximate block decomposition

$$\begin{pmatrix} A_{\mathcal{F}\mathcal{F}} & A_{\mathcal{F}\mathcal{C}} \\ A_{\mathcal{C}\mathcal{F}} & A_{\mathcal{C}\mathcal{C}} \end{pmatrix} = \begin{pmatrix} L_{\mathcal{F}\mathcal{F}} & 0 \\ L_{\mathcal{C}\mathcal{F}} & I \end{pmatrix} \begin{pmatrix} D_{\mathcal{F}\mathcal{F}} & 0 \\ 0 & S_{\mathcal{C}\mathcal{C}} \end{pmatrix} \begin{pmatrix} U_{\mathcal{F}\mathcal{F}} & U_{\mathcal{F}\mathcal{C}} \\ 0 & I \end{pmatrix} + E$$



Algebraic Multilevel Approach

Reordering ($+$ rescaling) the system $\rightarrow \begin{cases} \mathcal{F} & \text{“fine grid points”} \\ \mathcal{C} & \text{“coarse grid points”} \end{cases}$

$$A \rightarrow \Pi^\top A \Pi = \begin{pmatrix} A_{\mathcal{F}\mathcal{F}} & A_{\mathcal{F}\mathcal{C}} \\ A_{\mathcal{C}\mathcal{F}} & A_{\mathcal{C}\mathcal{C}} \end{pmatrix}$$

Approximate block decomposition

$$\begin{pmatrix} A_{\mathcal{F}\mathcal{F}} & A_{\mathcal{F}\mathcal{C}} \\ A_{\mathcal{C}\mathcal{F}} & A_{\mathcal{C}\mathcal{C}} \end{pmatrix} = \underbrace{\begin{pmatrix} L_{\mathcal{F}\mathcal{F}} & 0 \\ L_{\mathcal{C}\mathcal{F}} & I \end{pmatrix}}_{L} \underbrace{\begin{pmatrix} D_{\mathcal{F}\mathcal{F}} & 0 \\ 0 & S_{\mathcal{C}\mathcal{C}} \end{pmatrix}}_{D} \underbrace{\begin{pmatrix} U_{\mathcal{F}\mathcal{F}} & U_{\mathcal{F}\mathcal{C}} \\ 0 & I \end{pmatrix}}_{U} + E$$

- $S_{\mathcal{C}\mathcal{C}}$ coarse grid system, E error matrix
- E represents the entries being discarded in L, U

solution operator

- Approximation $B_{\mathcal{FF}} \approx A^{-1}_{\mathcal{FF}}$, e.g. via solving with $L_{\mathcal{FF}} D_{\mathcal{FF}} U_{\mathcal{FF}}$
- $B_{\mathcal{FC}} \approx -A^{-1}_{\mathcal{FF}} A_{\mathcal{FC}}$, e.g. via $-L^{-1}_{\mathcal{FF}} L_{\mathcal{FC}}$
- $B_{\mathcal{CF}} \approx -A_{\mathcal{CF}} A^{-1}_{\mathcal{FF}}$, e.g. via $-U_{\mathcal{CF}} U^{-1}_{\mathcal{FF}}$

solution operator

- Approximation $B_{\mathcal{FF}} \approx A_{\mathcal{FF}}^{-1}$, e.g. via solving with $L_{\mathcal{FF}} D_{\mathcal{FF}} U_{\mathcal{FF}}$
- $B_{\mathcal{FC}} \approx -A_{\mathcal{FF}}^{-1} A_{\mathcal{FC}}$, e.g. via $-L_{\mathcal{FF}}^{-1} L_{\mathcal{FC}}$
- $B_{\mathcal{CF}} \approx -A_{\mathcal{CF}} A_{\mathcal{FF}}^{-1}$, e.g. via $-U_{\mathcal{CF}} U_{\mathcal{FF}}^{-1}$

$$\begin{pmatrix} A_{\mathcal{FF}} & A_{\mathcal{FC}} \\ A_{\mathcal{CF}} & A_{\mathcal{CC}} \end{pmatrix}^{-1} \approx \begin{pmatrix} B_{\mathcal{FF}} & 0 \\ 0 & 0 \end{pmatrix} + \underbrace{\begin{pmatrix} B_{\mathcal{FC}} \\ I \end{pmatrix}}_{P} S_{\mathcal{CC}}^{-1} \underbrace{\begin{pmatrix} B_{\mathcal{CF}} & I \end{pmatrix}}_{R^\top}$$

- P “interpolation”
- $R^\top$ “restriction”

solution operator

- Approximation $B_{\mathcal{FF}} \approx A_{\mathcal{FF}}^{-1}$, e.g. via solving with $L_{\mathcal{FF}} D_{\mathcal{FF}} U_{\mathcal{FF}}$
- $B_{\mathcal{FC}} \approx -A_{\mathcal{FF}}^{-1} A_{\mathcal{FC}}$, e.g. via $-L_{\mathcal{FF}}^{-1} L_{\mathcal{FC}}$
- $B_{\mathcal{CF}} \approx -A_{\mathcal{CF}} A_{\mathcal{FF}}^{-1}$, e.g. via $-U_{\mathcal{CF}} U_{\mathcal{FF}}^{-1}$

$$\begin{pmatrix} A_{\mathcal{FF}} & A_{\mathcal{FC}} \\ A_{\mathcal{CF}} & A_{\mathcal{CC}} \end{pmatrix}^{-1} \approx \begin{pmatrix} B_{\mathcal{FF}} & 0 \\ 0 & 0 \end{pmatrix} + \underbrace{\begin{pmatrix} B_{\mathcal{FC}} \\ I \end{pmatrix}}_{P} S_{\mathcal{CC}}^{-1} \underbrace{\begin{pmatrix} B_{\mathcal{CF}} & I \end{pmatrix}}_{R^\top}$$

- P “interpolation”
- $R^\top$ “restriction”

Multilevel approach: same approach recursively applied to $S_{\mathcal{CC}}$

Solution Operator — Refinement

- Supplement by smoothing steps G_1, G_2 (e.g. Jacobi, Gauss–Seidel)
  Iterations matrix for the error $e = x - \tilde{x}$

$$e \to (I - \left\{ \begin{pmatrix} B_{\mathcal{FF}} & 0 \\ 0 & 0 \end{pmatrix} + P S_{\mathcal{CC}}^{-1} R^\top \right\} A) e$$

$$\downarrow$$

$$e \to (I - G_2 A)(I - P S_{\mathcal{CC}}^{-1} R^\top A)(I - G_1 A) e$$

Solution Operator — Refinement

- Supplement by smoothing steps G_1, G_2 (e.g. Jacobi, Gauss–Seidel)
  Iterations matrix for the error $e = x - \tilde{x}$

$$e \to (I - \left\{ \begin{pmatrix} B_{\mathcal{FF}} & 0 \\ 0 & 0 \end{pmatrix} + PS_{\mathcal{CC}}^{-1} R^\top \right\} A)e$$

$$\downarrow$$

$$e \to (I - G_2 A)(I - PS_{\mathcal{CC}}^{-1} R^\top A)(I - G_1 A)e$$

- V–cycle ($\mu = 1$), W–cycle ($\mu = 2$)

$$(I - G_2 A)(I - PS_{\mathcal{CC}}^{-1} R^\top A)^{\mu}(I - G_1 A)$$

What means inverse–based decomposition

Prescribed uniform bound κ for the inverse transformations L^{-1}, U^{-1}

$$\|L^{-1}\| = \| \begin{pmatrix} L_{\mathcal{F}\mathcal{F}}^{-1} & 0 \\ -L_{\mathcal{C}\mathcal{F}} L_{\mathcal{F}\mathcal{F}}^{-1} & I \end{pmatrix} \| \leqslant \kappa$$

$$\|U^{-1}\| = \| \begin{pmatrix} U_{\mathcal{F}\mathcal{F}}^{-1} & -U_{\mathcal{F}\mathcal{F}}^{-1} U_{\mathcal{F}\mathcal{C}} \\ 0 & I \end{pmatrix} \| \leqslant \kappa$$

What means inverse–based decomposition

Prescribed uniform bound κ for the inverse transformations L^{-1}, U^{-1}

$$\|L^{-1}\| = \|\begin{pmatrix} L_{\mathcal{F}\mathcal{F}}^{-1} & 0 \\ -L_{\mathcal{C}\mathcal{F}}L_{\mathcal{F}\mathcal{F}}^{-1} & I \end{pmatrix}\| \sim \|\underbrace{\begin{pmatrix} L_{\mathcal{F}\mathcal{F}}^{-1} & 0 \\ \boxed{-B_{\mathcal{C}\mathcal{F}}} & I \end{pmatrix}}_{R^\top}\| \leqslant \kappa$$

$$\|U^{-1}\| = \|\begin{pmatrix} U_{\mathcal{F}\mathcal{F}}^{-1} & -U_{\mathcal{F}\mathcal{F}}^{-1}U_{\mathcal{F}\mathcal{C}} \\ 0 & I \end{pmatrix}\| \sim \|\underbrace{\begin{pmatrix} U_{\mathcal{F}\mathcal{F}}^{-1} & \boxed{B_{\mathcal{F}\mathcal{C}}} \\ 0 & I \end{pmatrix}}_{P}\| \leqslant \kappa$$

Why inverse–based Decompositions

Generally speaking:

- Norm of the inverse factors drive the approximation error
- "Inverse error" $F = L^{-1}EU^{-1}$ is amplified

If the norms of inverse factors are even kept bounded:

- Absolute error of the coarse grid system $S_{\mathcal{C}\mathcal{C}}$ can be predicted
- Tight κ forces approximately sparse factors L^{-1}, U^{-1}

Approximation Error

Why Inverse–Based Decompositions

- Approximation $A = LDU + E$
- For solving $Ax = b$ we have to apply L^{-1}, U^{-1}

Approximation Error

Why Inverse–Based Decompositions

- Approximation $L^{-1}AU^{-1} = \quad D \quad + F$
- Can be used directly to construct an approximate inverse decomposition

Approximation Error
Why Inverse–Based Decompositions

- Approximation $L^{-1}AU^{-1} = \quad D \quad + F$
- Can be used directly to construct an approximate inverse decomposition

"AINV" [Benzi, Tuma '98]

- compute approximate inverses $W \approx L^{-1}$, $Z \approx U^{-1}$ directly without L, U

$$A \rightarrow \begin{pmatrix} * & \\ & \square \end{pmatrix} + F_1 \rightarrow \begin{pmatrix} * & \\ * & \square \end{pmatrix} + F_2 \rightarrow \cdots \rightarrow WAZ = \begin{pmatrix} \diagdown & \\ & \square \end{pmatrix} + F$$

- Numerical usually more expensive than ILU
- Significantly more robust if small entries in W, Z are discarded.
- ILUs as by-product from AINV inherit robustness [Benzi,Tuma'03]

Approximation Error
Why Inverse–Based Decompositions

$$\left.\begin{array}{l} |w_{ij}| \\ |z_{kl}| \end{array}\right\} \leqslant \varepsilon \rightarrow \left\{\begin{array}{lcl} w_{ij} & := & 0 \\ z_{kl} & := & 0 \end{array}\right.$$

Theorem [B.,Saad'02]

$$\left.\begin{array}{l} |l_{im}| \cdot \|e_m^\top W\| \\ |u_{mj}| \cdot \|Ze_m\| \end{array}\right\} \leqslant \varepsilon \rightarrow \left\{\begin{array}{lcl} l_{im} & := & 0 \\ u_{mj} & := & 0 \end{array}\right.$$

$$\Rightarrow e_i^\top |I - LW| e_m \leqslant (m-i)\varepsilon, \quad e_m^\top |I - ZU| e_j \leqslant (m-j)\varepsilon$$

Approximation Error
Why Inverse–Based Decompositions

$$\left.\begin{array}{l} |w_{ij}| \\ |z_{kl}| \end{array}\right\} \leqslant \varepsilon \rightarrow \left\{\begin{array}{lcl} w_{ij} & := & 0 \\ z_{kl} & := & 0 \end{array}\right.$$

Theorem [B.,Saad'02]

$$\left.\begin{array}{l} |l_{im}| \cdot \|e_m^\top W\| \\ |u_{mj}| \cdot \|Ze_m\| \end{array}\right\} \leqslant \varepsilon \rightarrow \left\{\begin{array}{lcl} l_{im} & := & 0 \\ u_{mj} & := & 0 \end{array}\right.$$

$$\Rightarrow e_i^\top |I - LW| e_m \leqslant (m-i)\varepsilon, \quad e_m^\top |I - ZU| e_j \leqslant (m-j)\varepsilon$$

Observation

Norm of the inverse factors drive the approximation error

The Inverse Error

Why Inverse–Based Decompositions

$$\begin{pmatrix} L_{\mathcal{F}\mathcal{F}} & 0 \\ L_{\mathcal{C}\mathcal{F}} & I \end{pmatrix}^{-1} A \begin{pmatrix} U_{\mathcal{F}\mathcal{F}} & U_{\mathcal{F}\mathcal{C}} \\ 0 & I \end{pmatrix}^{-1} = \begin{pmatrix} D_{\mathcal{F}\mathcal{F}} & 0 \\ 0 & S_{\mathcal{C}\mathcal{C}} \end{pmatrix} + \underbrace{\begin{pmatrix} * & F_{\mathcal{F}\mathcal{C}} \\ F_{\mathcal{C}\mathcal{F}} & * \end{pmatrix}}_{F}$$

The Inverse Error

Why Inverse–Based Decompositions

$$\begin{pmatrix} L_{\mathcal{FF}} & 0 \\ L_{\mathcal{CF}} & I \end{pmatrix}^{-1} A \begin{pmatrix} U_{\mathcal{FF}} & U_{\mathcal{FC}} \\ 0 & I \end{pmatrix}^{-1} = \begin{pmatrix} D_{\mathcal{FF}} & 0 \\ 0 & S_{\mathcal{CC}} \end{pmatrix} + \underbrace{\begin{pmatrix} * & F_{\mathcal{FC}} \\ F_{\mathcal{CF}} & * \end{pmatrix}}_{F}$$

Lemma [B.,Saad'04]

Denote by $E_{L,\mathcal{FF}}$, $E_{U,\mathcal{FF}}$ the entries being dropped from $L_{\mathcal{FF}}$, $U_{\mathcal{FF}}$.

- *coarse grid system $\tilde{S}_{\mathcal{CC}}$ from ILU*

$$\Rightarrow F_{\mathcal{FC}} = -L_{\mathcal{FF}}^{-1} \left(E_{L,\mathcal{FF}} D_{\mathcal{FF}} + D_{\mathcal{FF}} E_{U,\mathcal{FF}}\right) U_{\mathcal{FF}}^{-1} U_{\mathcal{FC}}$$

- *Coarse grid system $\tilde{S}_{\mathcal{CC}} = R^{\top} AP$ (successively obtained via Galerkin)*

$$\Rightarrow F_{\mathcal{FC}} = \qquad\qquad - D_{\mathcal{FF}} E_{U,\mathcal{FF}}\, U_{\mathcal{FF}}^{-1} U_{\mathcal{FC}}$$

Error Propagation

Why Inverse–Based Decompositoins

Suppose that the diagonal entries occuring during the decomposition $L_{\mathcal{F}\mathcal{F}} D_{\mathcal{F}\mathcal{F}} U_{\mathcal{F}\mathcal{F}} = A_{\mathcal{F}\mathcal{F}} + E_{\mathcal{F}\mathcal{F}}$ are uniformly bounded.

Error Propagation

Why Inverse–Based Decompositoins

Suppose that the diagonal entries occuring during the decomposition $L_{\mathcal{F}\mathcal{F}}D_{\mathcal{F}\mathcal{F}}U_{\mathcal{F}\mathcal{F}} = A_{\mathcal{F}\mathcal{F}} + E_{\mathcal{F}\mathcal{F}}$ are uniformly bounded.

Theorem [B.,Saad '04]

- Coarse grid system $\tilde{S}_{\mathcal{C}\mathcal{C}}$ from ILU

  If l_{im}, u_{mj} are dropped, whenever $|l_{im}|, |u_{mj}| \leqslant \varepsilon/\kappa^2$,

  $\Rightarrow$ There exists a constant K such that $|\tilde{s}_{ij} - s_{ij}| \leqslant K(\kappa\varepsilon)^2$

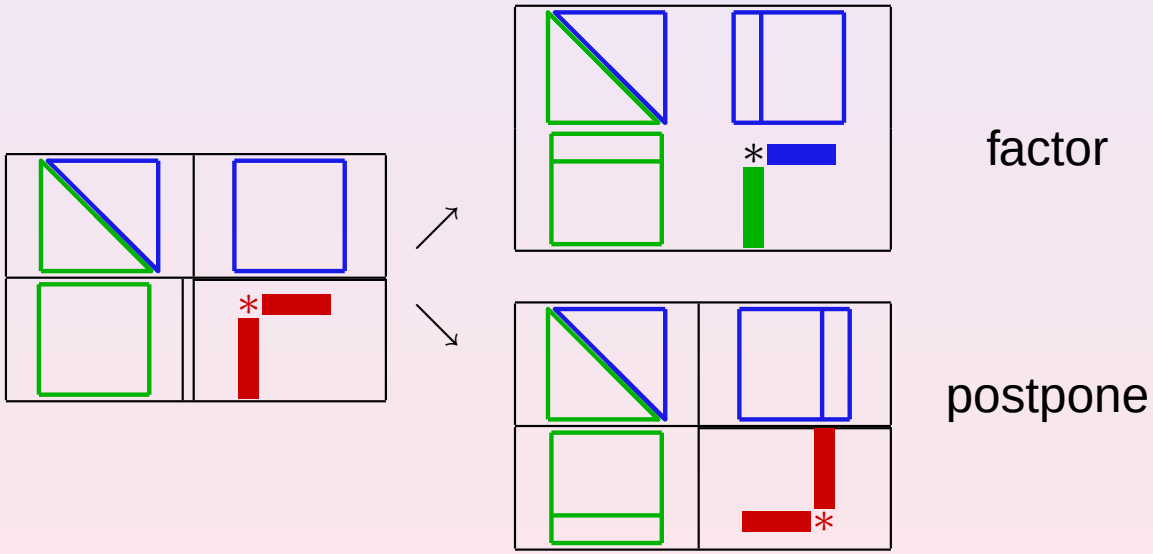
- Coarse grid system $\tilde{S}_{\mathcal{C}\mathcal{C}} = R^\top AP$ (successively from Galerkin)

  If l_{im}, u_{mj} are dropped, whenever $|l_{im}|, |u_{mj}| \leqslant \varepsilon$,

  $\Rightarrow$ There exists a constant K such that $|\tilde{s}_{ij} - s_{ij}| \leqslant K(\kappa\varepsilon)^2$

Consequences for inverse–based multilevel methods

- Estimate $\|L^{-1}\|$, $\|U^{-1}\|$ efficiently [Cline,Moler,Stewart,Wilkinson'77]
- Construct a well–suited initial system by a priori permutation and scaling ($\rightarrow A_{\mathcal{FF}}$)
- Keep $\|L^{-1}\|$, $\|U^{-1}\|$ below κ by inverse-based pivoting



- Tight κ desired, since $\|L^{-1}\|_\infty \leqslant \kappa \Rightarrow \sum_{j<i} |(L^{-1})_{ij}| \leqslant \kappa - 1$

Outline

Aggressive Dropping

Problem

- for practical problems we do not precisely know the optimal ε
- to be save we prefer a smaller tolerance

Consequences

- as $\varepsilon \longrightarrow 0$, the fill–in significantly increases
- as $\varepsilon \longrightarrow 0$, number of iteration steps decreases to a few number of steps
- memory requirement dramatically increases
- even for the iterative part, efficiency does not increase since the fill increases

Aggressive Dropping

Solution

- We do not necessarily need a small number of iteration steps
- instead: $L^{-1}AU^{-1} = D - F$ should lead to small $\|F\|$ and small perturbations in L^{-1}, U^{-1} could be tolerated

Lemma

Denote by μ_k, ν_k the number of nonzeros in column k of L (resp. row k of U). Let $\tilde{L}, \tilde{U}$ be constructed from L, U by dropping entries l_{ik}, u_{kj} satisfying

$$\|L^{-1}e_i\| \cdot |l_{ik}| \leqslant \frac{\tau}{\mu_k}, \quad |u_{kj}| \cdot \|e_j^\top U^{-1}\| \leqslant \frac{\tau}{\nu_k},$$

then

$$D - F = (I + E_L)(\tilde{D} - \tilde{F})(I + E_U)$$

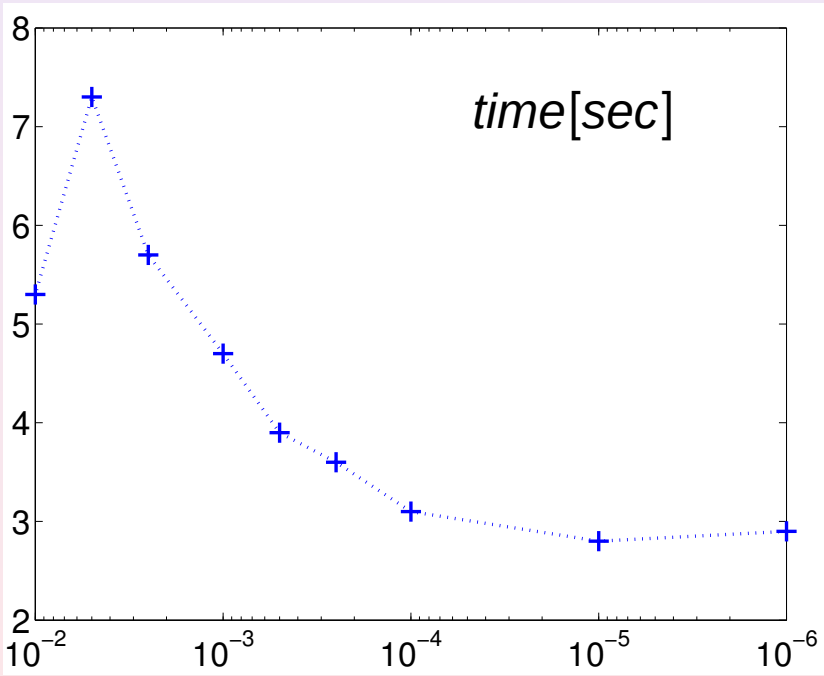
where

$$\|E_L\|_1 \leqslant \tau, \|E_U\|_\infty \leqslant \tau.$$

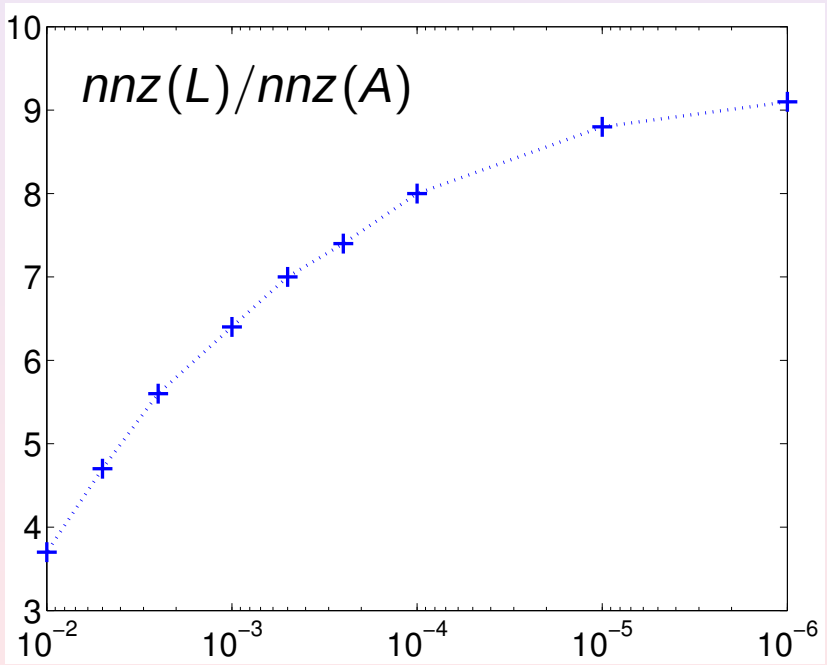
Aggressive Dropping

Example. BCSSTK25.

- Compare time and memory WITHOUT and WITH aggressive dropping
- drop tolerance ε is decreased
- threshold τ for aggressive dropping is kept at 10^{-1}



drop tolerance

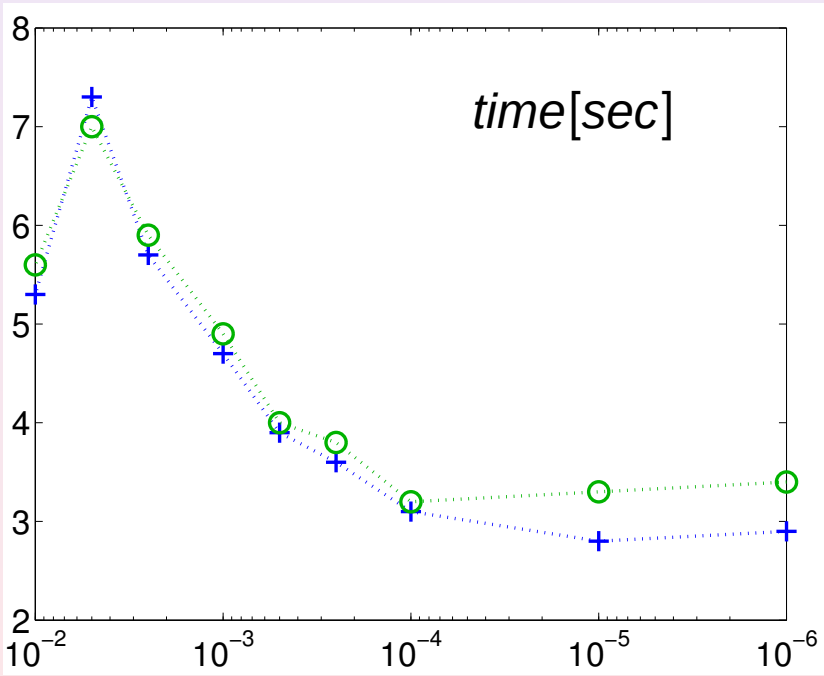


drop tolerance

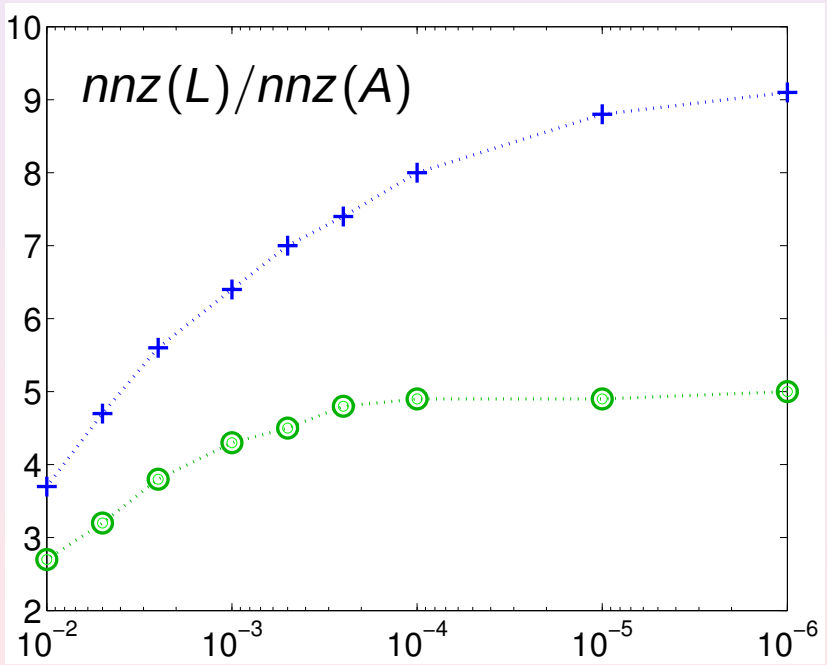
Aggressive Dropping

Example. BCSSTK25.

- Compare time and memory WITHOUT and WITH aggressive dropping
- drop tolerance ε is decreased
- threshold τ for aggressive dropping is kept at 10^{-1}



drop tolerance



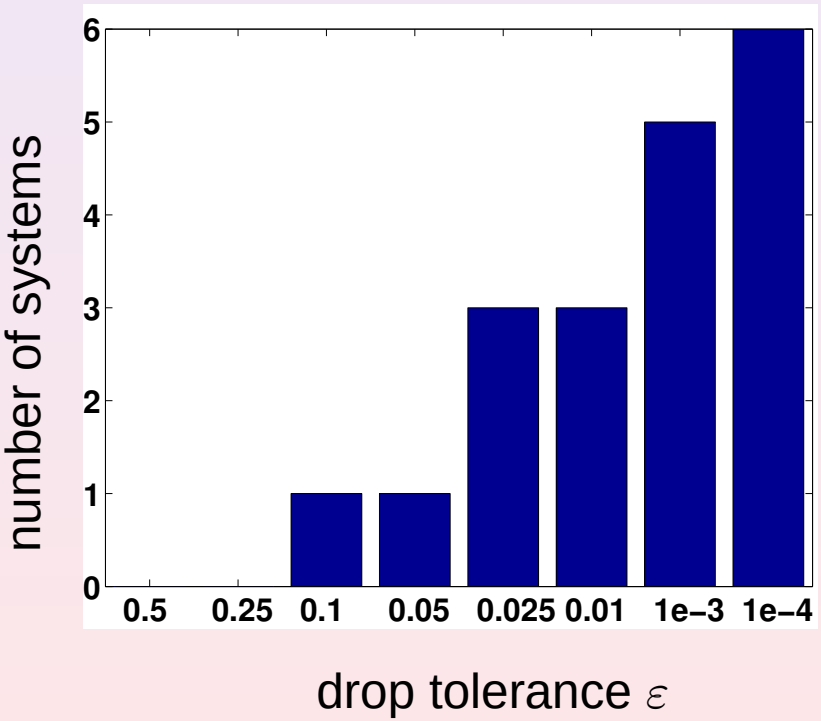
drop tolerance

Outline

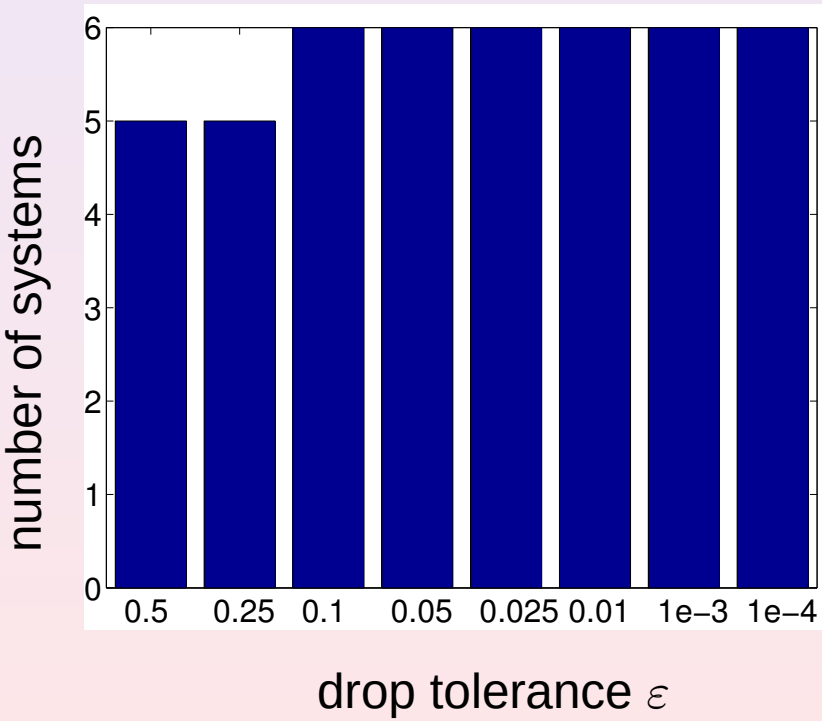
Circuit Simulation (benchmark collection from Infineon)

- compare regular and inverse–based ILU
- Dependence of the decomposition on drop tolerance ε (Convergence of GMRES(30) after at most 500 steps)

simple ILU



inverse–based ILU



Device Simulation

- direct solver (PARDISO, [Schenk,Gärner'04] excellent, but causes a lot of fill
- inverse–based ILU (ILUPACK [B.,Saad'04]): fix drop tolerance at $\varepsilon = 10^{-3}$ and use $\kappa = 10$.
- "regular" ILU fails until $1e-7$ for most problems
- both use minimum weight matching and MeTiS.

	direct solver		inverse–based ILU	
	$\frac{nnz(L+U)}{nnz(A)}$	time[sec]	$\frac{nnz(L+U)}{nnz(A)}$	time[sec]
barrier2-1	32.3	4.7e3	0.9	7.9e1
barrier2-2	32.3	4.6e3	0.9	6.1e1
barrier2-3	32.3	4.6e3	0.9	—
barrier2-4	32.3	4.6e3	0.9	1.1e2
barrier2-9	32.8	4.6e3	0.9	3.2e1
barrier2-10	32.8	4.6e3	0.9	7.5e1
barrier2-11	32.8	4.7e3	0.9	7.4e1
barrier2-12	32.8	4.7e3	0.9	6.5e1

Outline

Challenges

- Symmetry requires structure preserving solver
- Block–oriented variant (1×1, 2×2), since highly indefinite
- Matching–based preprocessing [Duff,Koster'98], [Gilbert,Duff'02], [Duff,Pralet'04], [Schenk,Hagemann'04]
- classical approach: Cullum–Willoughby algorithm ['85]
- iterative methods ARPACK [Lehoucq,Sorensen,Yang'98], Jacobi–Davidson [Sleijpen,Van der Vorst'96],[Geus'02]
- direct solver PARDISO [Schenk,Gärtner], preconditioner ILUPACK [B.,Saad'04],[B.,Schenk'05]

Numerical results [Schenk,B.,Römer'05]

$n = m^3$, $m = 100, 130, 160, 190$. $\omega = \omega_c = 16.5$.
(SGI Altix/BX2, Intel Itanium2, 1.6GHz)

	Time			
system size $n = m^3$	10^6	$2 \cdot 10^6$	$4 \cdot 10^6$	$7 \cdot 10^6$
Cullum–Willoughby	71 : 04	—	—	—

Numerical results [Schenk,B.,Römer'05]

$n = m^3$, $m = 100, 130, 160, 190$. $\omega = \omega_c = 16.5$.
(SGI Altix/BX2, Intel Itanium2, 1.6GHz)

	Time			
system size $n = m^3$	10^6	$2 \cdot 10^6$	$4 \cdot 10^6$	$7 \cdot 10^6$
Cullum–Willoughby	71 : 04	—	—	—
ARPACK + PARDISO	5 : 37	—	—	—

Numerical results [Schenk,B.,Römer'05]

$n = m^3$, $m = 100, 130, 160, 190$. $\omega = \omega_c = 16.5$.
(SGI Altix/BX2, Intel Itanium2, 1.6GHz)

	Time			
system size $n = m^3$	10^6	$2 \cdot 10^6$	$4 \cdot 10^6$	$7 \cdot 10^6$
Cullum–Willoughby	71 : 04	—	—	—
ARPACK + PARDISO	5 : 37	—	—	—
ARPACK + ILUPACK	0 : 45	5 : 46	13 : 58	—

Numerical results [Schenk,B.,Römer'05]

$n = m^3$, $m = 100, 130, 160, 190$. $\omega = \omega_c = 16.5$.
(SGI Altix/BX2, Intel Itanium2, 1.6GHz)

	Time			
system size $n = m^3$	10^6	$2 \cdot 10^6$	$4 \cdot 10^6$	$7 \cdot 10^6$
Cullum–Willoughby	71 : 04	—	—	—
ARPACK + PARDISO	5 : 37	—	—	—
ARPACK + ILUPACK	0 : 45	5 : 46	13 : 58	—
Jacobi–Davidson + ILUPACK	0 : 18	1 : 01	5 : 03	9 : 06

Numerical results [Schenk,B.,Römer,'05]

$n = m^3$, $m = 100, 130, 160, 190$. $\omega = \omega_c = 16.5$.

	Memory [GB]			
system size $n = m^3$	10^6	$2 \cdot 10^6$	$4 \cdot 10^6$	$7 \cdot 10^6$
ARPACK + PARDISO	14.3	—	—	—
ARPACK + ILUPACK	1.4	3.0	5.8	—
Jacobi–Davidson + ILUPACK	1.4	3.0	5.8	9.6

Inverse–based decompositions: the key to success

- clear: modern iterative eigenvalue solver (Jacobi-Davidson) essential
- Matching quite helpful
- key role: bound κ for $\|L^{-1}\|$

Numerical example $m = 70, 100$. Vary bound κ inside ILUPACK

	$\kappa(m=70)$			$\kappa(m=100)$		
	5	10	20	5	10	20
fill–in $\frac{nnz(L)}{nnz(A)}$	11.7	18.6	31.7	11.6	18.6	—
Time [sec]	1.7e2	3.7e2	7.8e2	5.2e2	1.1e3	—

Outline

ILUPACK V2.0

- multilevel ILU preconditioning software package
- inverse–based decompositions
- single/double, real and complex arithmetic supported
- classes of matrices that are supported
  - general
  - symmetric (Hermitian) positive definite
- new in V2.0
  - real/complex symmetric, complex Hermitian indefinite
  - interfaces to incorporate matchings is provided (PARDISO, MC64)
- `http://www.math.tu-berlin.de/ilupack/`

  to be released soon . . .

Conclusions

- development in recent years has significantly changed preconditioning methods as well as direct solvers
- matchings dramatically stabilize preconditioning methods
- symmetric matchings nowadays allow to efficiently treat symmetrically structured indefinite systems
- inverse–based decompositions are complementary approach
- analysis partially gives an explanation for this effect