The importance of Structure in Algebraic Preconditioners

Jennifer Scott

Rutherford Appleton Laboratory

Miroslav Tůma

Institute of Computer Science

Academy of Sciences of the Czech Republic

August 14, 2009,

Hong Kong Baptist University

Outline

Outline

Preconditioned iterative methods

Solving large, sparse SPD systems by iterative methods

$$Ax = b$$

Preconditioned iterative methods

Solving large, sparse SPD systems by iterative methods

$$Ax = b$$

Algebraic preconditioning as a transformation

$$M^{-1}Ax = M^{-1}b$$

.

Preconditioned iterative methods

Solving large, sparse SPD systems by iterative methods

$$Ax = b$$

Algebraic preconditioning as a transformation

$$M^{-1}Ax = M^{-1}b$$

In particular: Incomplete decompositions

Preconditioned iterative methods

Solving large, sparse SPD systems by iterative methods

$$Ax = b$$

Algebraic preconditioning as a transformation

$$M^{-1}Ax = M^{-1}b$$

In particular: Incomplete decompositions

- As usual, should be cheap, fast to compute, implying fast converging preconditioned iterative method

Preconditioned iterative methods

Solving large, sparse SPD systems by iterative methods

$$Ax = b$$

Algebraic preconditioning as a transformation

$$M^{-1}Ax = M^{-1}b$$

In particular: Incomplete decompositions

- As usual, should be cheap, fast to compute, implying fast converging preconditioned iterative method
- sparse enough

Preconditioned iterative methods

Solving large, sparse SPD systems by iterative methods

$$Ax = b$$

Algebraic preconditioning as a transformation

$$M^{-1}Ax = M^{-1}b$$

In particular: Incomplete decompositions

- As usual, should be cheap, fast to compute, implying fast converging preconditioned iterative method
- sparse enough
- providing just sufficient approximation of the algebraic problem if this makes computations faster

Preconditioned iterative methods

Solving large, sparse SPD systems by iterative methods

$$Ax = b$$

Algebraic preconditioning as a transformation

$$M^{-1}Ax = M^{-1}b$$

In particular: Incomplete decompositions

- As usual, should be cheap, fast to compute, implying fast converging preconditioned iterative method
- sparse enough
- providing just sufficient approximation of the algebraic problem if this makes computations faster
- Our target is robustness, not a fragile power

Outline

Goal of this talk

Search of more robust algebraic preconditioners

Goal of this talk

Search of more robust algebraic preconditioners

1. Show the importance of structure of the matrix and its decomposition in algebraic preconditioners.

Goal of this talk

Search of more robust algebraic preconditioners

1. Show the importance of structure of the matrix and its decomposition in algebraic preconditioners.
2. Present the effect separately from the other possible improvements (no compensations).

Goal of this talk

Search of more robust algebraic preconditioners

1. Show the importance of structure of the matrix and its decomposition in algebraic preconditioners.
2. Present the effect separately from the other possible improvements (no compensations).
3. Propose the new way to of level-based strategy in the incomplete decomposition.

Goal of this talk

Search of more robust algebraic preconditioners

1. Show the importance of structure of the matrix and its decomposition in algebraic preconditioners.
2. Present the effect separately from the other possible improvements (no compensations).
3. Propose the new way to of level-based strategy in the incomplete decomposition.
4. The techniques are basis of the new HSL code MI22.

Goal of this talk

Search of more robust algebraic preconditioners

1. Show the importance of structure of the matrix and its decomposition in algebraic preconditioners.
2. Present the effect separately from the other possible improvements (no compensations).
3. Propose the new way to of level-based strategy in the incomplete decomposition.
4. The techniques are basis of the new HSL code MI22.

Structure of this talk

- Some notes on the history of the structure-based preconditioners

Goal of this talk

Search of more robust algebraic preconditioners

1. Show the importance of structure of the matrix and its decomposition in algebraic preconditioners.
2. Present the effect separately from the other possible improvements (no compensations).
3. Propose the new way to of level-based strategy in the incomplete decomposition.
4. The techniques are basis of the new HSL code MI22.

Structure of this talk

- Some notes on the history of the structure-based preconditioners
- Basic ways of improvements

Goal of this talk

Search of more robust algebraic preconditioners

1. Show the importance of structure of the matrix and its decomposition in algebraic preconditioners.
2. Present the effect separately from the other possible improvements (no compensations).
3. Propose the new way to of level-based strategy in the incomplete decomposition.
4. The techniques are basis of the new HSL code MI22.

Structure of this talk

- Some notes on the history of the structure-based preconditioners
- Basic ways of improvements
- Experimental results showing some structure-based effects

Outline

Stencil-based advent

- First algebraic preconditioners M combined with A into $M^{-1}A$ and derived directly via interpolations from the grid stencil (Buleev, 1959, 1960).

Stencil-based advent

- First algebraic preconditioners M combined with A into $M^{-1}A$ and derived directly via interpolations from the grid stencil (Buleev, 1959, 1960).
- Later co-invented and interpreted as incomplete decompositions (Varga, 1960)

Stencil-based advent

- First algebraic preconditioners M combined with A into $M^{-1}A$ and derived directly via interpolations from the grid stencil (Buleev, 1959, 1960).
- Later co-invented and interpreted as incomplete decompositions (Varga, 1960)
- Additional corrections started to be heavily parametrized, added purely algebraic relaxations (Baker, Oliphant (1960), Il'in (1970), Woznicki (1989)); changing compensations dynamically (Sabinin, 1981, 1985).

Stencil-based advent

- First algebraic preconditioners M combined with A into $M^{-1}A$ and derived directly via interpolations from the grid stencil (Buleev, 1959, 1960).
- Later co-invented and interpreted as incomplete decompositions (Varga, 1960)
- Additional corrections started to be heavily parametrized, added purely algebraic relaxations (Baker, Oliphant (1960), Il'in (1970), Woznicki (1989)); changing compensations dynamically (Sabinin, 1981, 1985).
- stencils $\leftrightarrow$ local interpolation $\leftrightarrow$ elimination

Stencil-based advent

- First algebraic preconditioners M combined with A into $M^{-1}A$ and derived directly via interpolations from the grid stencil (Buleev, 1959, 1960).
- Later co-invented and interpreted as incomplete decompositions (Varga, 1960)
- Additional corrections started to be heavily parametrized, added purely algebraic relaxations (Baker, Oliphant (1960), Il'in (1970), Woznicki (1989)); changing compensations dynamically (Sabinin, 1981, 1985).
- stencils $\leftrightarrow$ local interpolation $\leftrightarrow$ elimination
- starting with first order factorizations $N = M - A = O(h)$

Stencil-based advent (continued)

Stencil-based advent (continued)

- Later, second order factorizations (SIP - Stone, 1968). Difficulties to symmetrize them (Saylor, 1974).

Stencil-based advent (continued)

- Later, second order factorizations (SIP - Stone, 1968). Difficulties to symmetrize them (Saylor, 1974).
- Nice results related to conditioning of $M^{-1}A$, subsequent diagonal modifications and their "algebraizations" (Dupont, Kendall, Rachford (1968), Gustafsson (1978), Axelsson, Lindsgog (1986), van der Vorst (1989) and a lot later work!.

Stencil-based advent (continued)

- Later, second order factorizations (SIP - Stone, 1968). Difficulties to symmetrize them (Saylor, 1974).
- Nice results related to conditioning of $M^{-1}A$, subsequent diagonal modifications and their "algebraizations" (Dupont, Kendall, Rachford (1968), Gustafsson (1978), Axelsson, Lindsgog (1986), van der Vorst (1989) and a lot later work!.

More general patterns

Stencil-based advent (continued)

- Later, second order factorizations (SIP - Stone, 1968). Difficulties to symmetrize them (Saylor, 1974).
- Nice results related to conditioning of $M^{-1}A$, subsequent diagonal modifications and their "algebraizations" (Dupont, Kendall, Rachford (1968), Gustafsson (1978), Axelsson, Lindsgog (1986), van der Vorst (1989) and a lot later work!.

More general patterns

- Crucial moment: paper by Meijerink and van der Vorst (1977) recognizing the potential of incomplete decompositions for preconditioning.

Incomplete decompositions

First encounter (continued)

Stencil-based advent (continued)

- Later, second order factorizations (SIP - Stone, 1968). Difficulties to symmetrize them (Saylor, 1974).
- Nice results related to conditioning of $M^{-1}A$, subsequent diagonal modifications and their "algebraizations" (Dupont, Kendall, Rachford (1968), Gustafsson (1978), Axelsson, Lindsgog (1986), van der Vorst (1989) and a lot later work!.

More general patterns

- Crucial moment: paper by Meijerink and van der Vorst (1977) recognizing the potential of incomplete decompositions for preconditioning.
- Incomplete decompositions classified by adding (ℓ) after the name. Starting to denote them by number of additional diagonals in simple problems $\rightarrow IC(\ell)$.

Matrix-based approach

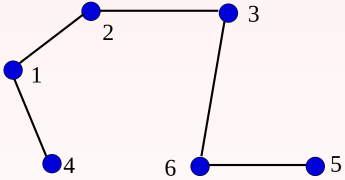
Matrix-based approach

- Matrix $\longrightarrow$ graph

Matrix-based approach

- Matrix $\rightarrow$ graph

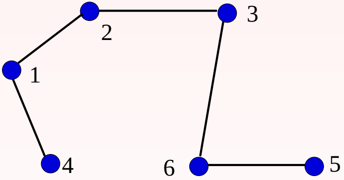
$$A = \begin{bmatrix} 1 & 1 & & 1 & & \\ 1 & 1 & 1 & & & \\ & 1 & 1 & & & 1 \\ 1 & & & 1 & & \\ & & & & 1 & 1 \\ & & 1 & & 1 & 1 \end{bmatrix}$$



Matrix-based approach

- Matrix $\rightarrow$ graph

$$A = \begin{bmatrix} 1 & 1 & & 1 & & \\ 1 & 1 & 1 & & & \\ & 1 & 1 & & & 1 \\ 1 & & & 1 & & \\ & & & & 1 & 1 \\ & & 1 & & 1 & 1 \end{bmatrix}$$



- Fill-path is a path in G joining nodes i and i via nodes with labels lower than both i and j.

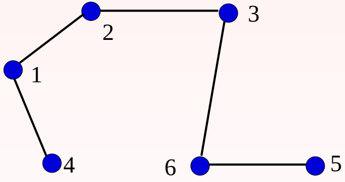
Incomplete decompositions

General patterns

Matrix-based approach

- Matrix $\rightarrow$ graph

$$A = \begin{bmatrix} 1 & 1 & & 1 & & \\ 1 & 1 & 1 & & & \\ & 1 & 1 & & & 1 \\ 1 & & & 1 & & \\ & & & & 1 & 1 \\ & & 1 & & 1 & 1 \end{bmatrix}$$



- Fill-path is a path in G joining nodes i and i via nodes with labels lower than both i and j.
- Entries of the Cholesky factor $l_{ij}, i > j$ are nonzero if and only if there is a fill path joining i and j in G.

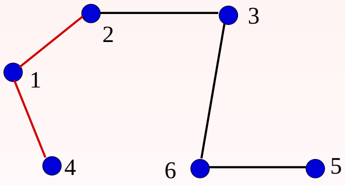
Incomplete decompositions

General patterns

Matrix-based approach

- Matrix $\rightarrow$ graph

$$A = \begin{bmatrix} 1 & 1 & & 1 & & \\ 1 & 1 & 1 & & & \\ & 1 & 1 & & & 1 \\ 1 & & & 1 & & \\ & & & & 1 & 1 \\ & & 1 & & 1 & 1 \end{bmatrix}$$



- Fill-path is a path in G joining nodes i and i via nodes with labels lower than both i and j.
- Entries of the Cholesky factor $l_{ij}, i > j$ are nonzero if and only if there is a fill path joining i and j in G.

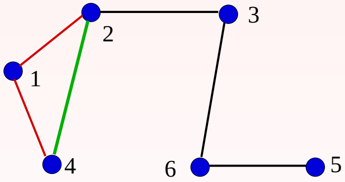
Incomplete decompositions

General patterns

Matrix-based approach

- Matrix $\rightarrow$ graph

$$A = \begin{bmatrix} 1 & 1 & & 1 & & \\ 1 & 1 & 1 & {\color{red}1} & & \\ & 1 & 1 & & & 1 \\ 1 & {\color{red}1} & & 1 & & \\ & & & & 1 & 1 \\ & & 1 & & 1 & 1 \end{bmatrix}$$



- Fill-path is a path in G joining nodes i and i via nodes with labels lower than both i and j.
- Entries of the Cholesky factor $l_{ij}, i > j$ are nonzero if and only if there is a fill path joining i and j in G.

Matrix-based approach

- Allowing fill up to a maximum length ℓ of any fill path (Watts III, (1981)).

Matrix-based approach

- Allowing fill up to a maximum length ℓ of any fill path (Watts III, (1981)).
- Practically: A fill entry is permitted provided $level(i,j) \le \ell$.

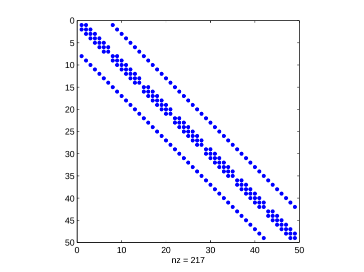
$$level(i,j) = \min_{1 \le l \le \min\{i,j\}} \{level(i,l) + level(l,j) + 1\}$$

(one of more slightly different definitions)

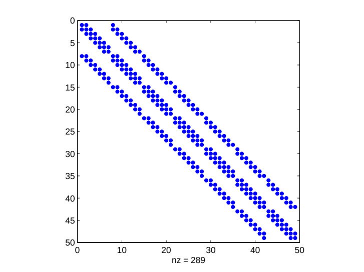
Example: Structure-based preconditioners

Incomplete decompositions

General patterns: an example



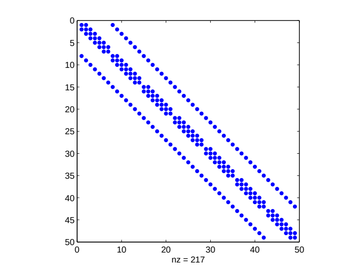
IC(0)



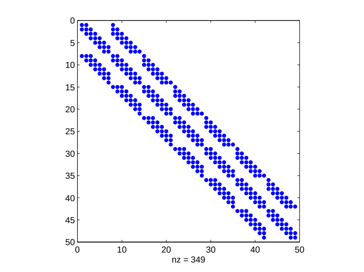
IC(1)

Incomplete decompositions

General patterns: an example



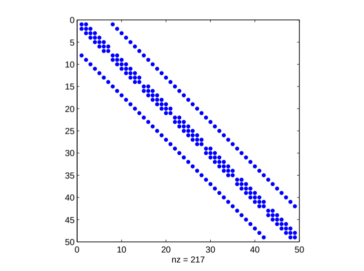
IC(0)



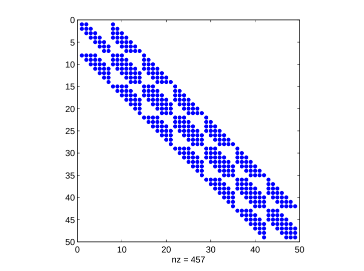
IC(2)

Incomplete decompositions

General patterns: an example



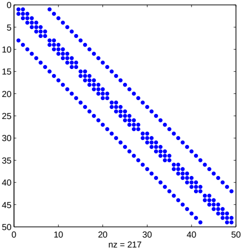
IC(0)



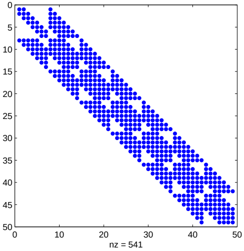
IC(3)

Incomplete decompositions

General patterns: an example



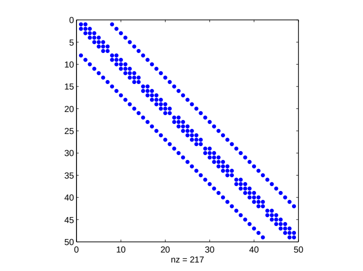
IC(0)



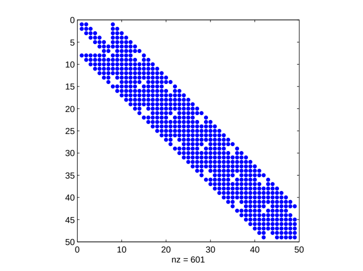
IC(4)

Incomplete decompositions

General patterns: an example



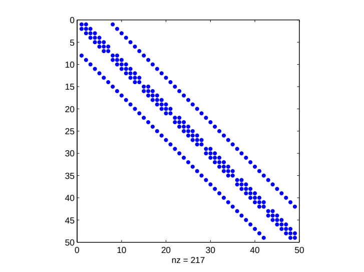
IC(0)



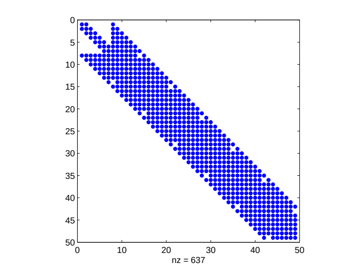
IC(5)

Incomplete decompositions

General patterns: an example



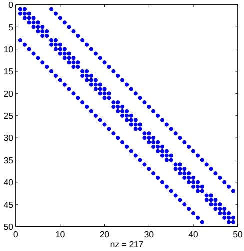
IC(0)



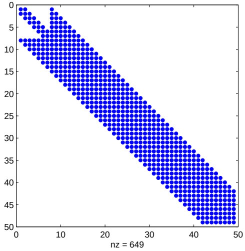
IC(6)

Incomplete decompositions

General patterns: an example



IC(0)



IC(7)

Matrix-based approach

- Often found that fill in L grows too quickly with ℓ.

Matrix-based approach

- Often found that fill in L grows too quickly with ℓ.
- While the error $R = A - LL^T$ inside the pattern is zero, outside can be large.

Matrix-based approach

- Often found that fill in L grows too quickly with ℓ.
- While the error $R = A - LL^T$ inside the pattern is zero, outside can be large.
- But: Decay of entries away from diagonal may help a lot.

Incomplete decompositions

Natural competitor of level-based methods: considering values

Incomplete decompositions

Natural competitor of level-based methods: considering values

- Dropping entries with “smaller magnitudes” (absolutely/relatively) (Zlatev et al. (1978), Munksgaard (1980), Axelsson (1972, 1983 et al. etc.)

Incomplete decompositions

Natural competitor of level-based methods: considering values

- Dropping entries with “smaller magnitudes” (absolutely/relatively) (Zlatev et al. (1978), Munksgaard (1980), Axelsson (1972, 1983 et al. etc.)
- But: if only magnitudes of entries are used - structural information may be lost

Incomplete decompositions

Natural competitor of level-based methods: considering values

- Dropping entries with “smaller magnitudes” (absolutely/relatively) (Zlatev et al. (1978), Munksgaard (1980), Axelsson (1972, 1983 et al. etc.)
- But: if only magnitudes of entries are used - structural information may be lost
- More complicated schemes may strongly influence implementations (e.g., if both row and column access for intermediate quantities is needed)

Incomplete decompositions

Natural competitor of level-based methods: considering values

- Dropping entries with "smaller magnitudes" (absolutely/relatively) (Zlatev et al. (1978), Munksgaard (1980), Axelsson (1972, 1983 et al. etc.)
- But: if only magnitudes of entries are used - structural information may be lost
- More complicated schemes may strongly influence implementations (e.g., if both row and column access for intermediate quantities is needed)
- Plassman, Jones (1995): no structure, just the memory predictability, see also Freund, Nachtigal, (1990). Similarly Lin, Moré with extended memory. ILUT by Saad, (1994).

Incomplete decompositions

Natural competitor of level-based methods: considering values

- Dropping entries with "smaller magnitudes" (absolutely/relatively) (Zlatev et al. (1978), Munksgaard (1980), Axelsson (1972, 1983 et al. etc.)
- But: if only magnitudes of entries are used - structural information may be lost
- More complicated schemes may strongly influence implementations (e.g., if both row and column access for intermediate quantities is needed)
- Plassman, Jones (1995): no structure, just the memory predictability, see also Freund, Nachtigal, (1990). Similarly Lin, Moré with extended memory. ILUT by Saad, (1994).
- The importance of error matrix $E = A - LL^T$ understood (Duff, Meurant, (1989)) and exploited (D'Azevedo, Forsyth, Tang, 1992)

- First simple combination of level-based approaches with dropping: D'Azevedo, Forsyth, Tang, (1992a).

Incomplete decompositions

Natural competitor of level-based methods: considering values

- First simple combination of level-based approaches with dropping: D'Azevedo, Forsyth, Tang, (1992a).
- Generally, convergence behavior can be far from predictable

Incomplete decompositions

Natural competitor of level-based methods: considering values

- First simple combination of level-based approaches with dropping: D'Azevedo, Forsyth, Tang, (1992a).
- Generally, convergence behavior can be far from predictable
- The real breakthrough in level-based approaches: cheap predictions by Hysom, Pothen, (2002)

Incomplete decompositions

Natural competitor of level-based methods: considering values

- First simple combination of level-based approaches with dropping: D'Azevedo, Forsyth, Tang, (1992a).
- Generally, convergence behavior can be far from predictable
- The real breakthrough in level-based approaches: cheap predictions by Hysom, Pothen, (2002)
- Our MI22 preconditioner is a new way to use level-based information, memory prediction and dropping at the same time.

Outline

First component of our approach: new setting of levels

Preassign levels to the entries individually

First component of our approach: new setting of levels

Preassign levels to the entries individually

- Computing the absolute values of the smallest and largest entries of A: $msmall$ and $mbig$.

First component of our approach: new setting of levels

Preassign levels to the entries individually

- Computing the absolute values of the smallest and largest entries of A: $msmall$ and $mbig$.
- Distribute nonzero entries uniformly by $\log |a_{ij}|$ into $ngroup0 = [log(mbig) - log(msmall)] + 1$ groups. Shrink zero groups to get $ngroup$ of them.

First component of our approach: new setting of levels

Preassign levels to the entries individually

- Computing the absolute values of the smallest and largest entries of A: $msmall$ and $mbig$.
- Distribute nonzero entries uniformly by $\log|a_{ij}|$ into $ngroup0 = [log(mbig) - log(msmall)] + 1$ groups. Shrink zero groups to get $ngroup$ of them.
- Set $level(i,j)$ for individual entries: For $\ell < ngroup$: $level(i,j) = (\ell - 1) * (l/ngroup) + 1$ where l $(1 \leq l \leq ngroup0)$ is the index of the group a_{ij} belongs to, and slightly differently otherwise.

First component of our approach: new setting of levels

Preassign levels to the entries individually

- Computing the absolute values of the smallest and largest entries of A: $msmall$ and $mbig$.
- Distribute nonzero entries uniformly by $\log |a_{ij}|$ into $ngroup0 = [log(mbig) - log(msmall)] + 1$ groups. Shrink zero groups to get $ngroup$ of them.
- Set $level(i, j)$ for individual entries: For $\ell < ngroup$: $level(i, j) = (\ell - 1) * (l/ngroup) + 1$ where l ($1 \leq l \leq ngroup0$) is the index of the group a_{ij} belongs to, and slightly differently otherwise.
- During the $IC(\ell)$ decomposition, entries of the factor L that correspond to nonzero entries of A are assigned the level $level(i, j)$.

First component of our approach: new setting of levels

Preassign levels to the entries individually

- Computing the absolute values of the smallest and largest entries of A: $msmall$ and $mbig$.
- Distribute nonzero entries uniformly by $\log |a_{ij}|$ into $ngroup0 = [log(mbig) - log(msmall)] + 1$ groups. Shrink zero groups to get $ngroup$ of them.
- Set $level(i,j)$ for individual entries: For $\ell < ngroup$: $level(i,j) = (\ell - 1) * (l/ngroup) + 1$ where l ($1 \leq l \leq ngroup0$) is the index of the group a_{ij} belongs to, and slightly differently otherwise.
- During the $IC(\ell)$ decomposition, entries of the factor L that correspond to nonzero entries of A are assigned the level $level(i,j)$.
- Each potential fill entry l_{ij} is assigned a level

$$level(i,j) = \min_{1 \leq l \leq \min\{i,j\}} \{level(i,l) + level(l,j) + 1\}.$$

A fill entry is permitted provided $level(i,j) \leq k$.

First component of our approach: new setting of levels

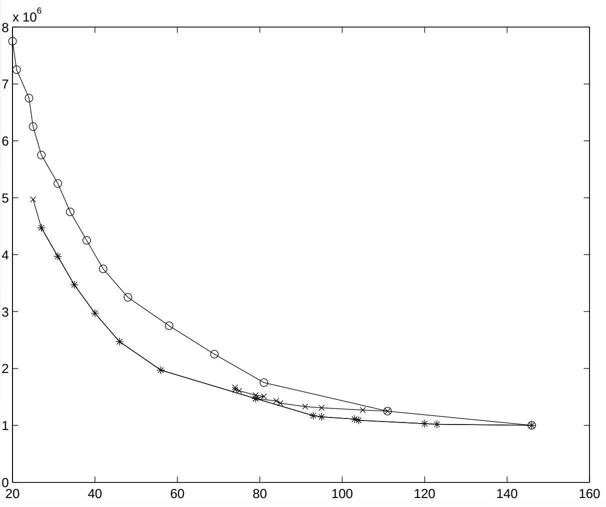
Experiments: Kohn-Sham equation, n=250500

Effects individual level preassignments

First component of our approach: new setting of levels

Experiments: Kohn-Sham equation, n=250500

Effects individual level preassignments



First component of our approach: new setting of levels

Experience from the experiments

Notes on the experiments

Notes on the experiments

- (+) Settings do not increase timings significantly.

First component of our approach: new setting of levels

Experience from the experiments

Notes on the experiments

- (+) Settings do not increase timings significantly.
- (-)The improvements are often small. We intend to construct a robust strategy which should be used as a default value.

First component of our approach: new setting of levels

Experience from the experiments

Notes on the experiments

- (+) Settings do not increase timings significantly.
- (-)The improvements are often small. We intend to construct a robust strategy which should be used as a default value.
- Open problem: determine more sophisticated rules to preassign levels.

Second component of our approach: keeping structure

Integrate the predefined factor structure with dropping

Second component of our approach: keeping structure

Integrate the predefined factor structure with dropping

- Symbolic part of out modified (MI22) $IC(\ell)$ predefines the structure.

Second component of our approach: keeping structure

Integrate the predefined factor structure with dropping

- Symbolic part of out modified (MI22) $IC(\ell)$ predefines the structure.
- Only very small entries from the structure are not kept. The space is then freed and can be further used.
- The final size parametrized by memory multiplier $0 \leq \theta$.

Second component of our approach: keeping structure

Integrate the predefined factor structure with dropping

- Symbolic part of out modified (MI22) $IC(\ell)$ predefines the structure.
- Only very small entries from the structure are not kept. The space is then freed and can be further used.
- The final size parametrized by memory multiplier $0 \leq \theta$.
- Additional space distributed (1) uniformly or (2) nonuniformly.

Second component of our approach: keeping structure

Integrate the predefined factor structure with dropping

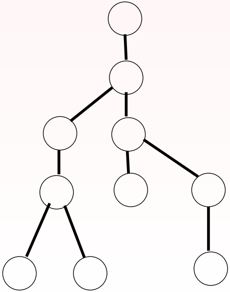
- Symbolic part of out modified (MI22) $IC(\ell)$ predefines the structure.
- Only very small entries from the structure are not kept. The space is then freed and can be further used.
- The final size parametrized by memory multiplier $0 \leq \theta$.
- Additional space distributed (1) uniformly or (2) nonuniformly.

- Nonuniform distribution based on the elimination tree and its row subtrees

Second component of our approach: keeping structure

Integrate the predefined factor structure with dropping

- Symbolic part of out modified (MI22) $IC(\ell)$ predefines the structure.
- Only very small entries from the structure are not kept. The space is then freed and can be further used.
- The final size parametrized by memory multiplier $0 \leq \theta$.
- Additional space distributed (1) uniformly or (2) nonuniformly.



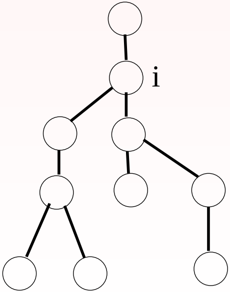
- Nonuniform distribution based on the elimination tree and its row subtrees

Second component of our approach: keeping structure

Integrate the predefined factor structure with dropping

- Symbolic part of out modified (MI22) $IC(\ell)$ predefines the structure.
- Only very small entries from the structure are not kept. The space is then freed and can be further used.
- The final size parametrized by memory multiplier $0 \leq \theta$.
- Additional space distributed (1) uniformly or (2) nonuniformly.

- Nonuniform distribution based on the elimination tree and its row subtrees

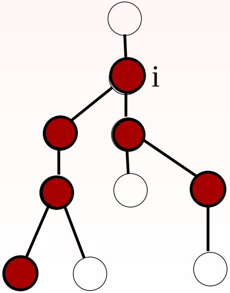


Second component of our approach: keeping structure

Integrate the predefined factor structure with dropping

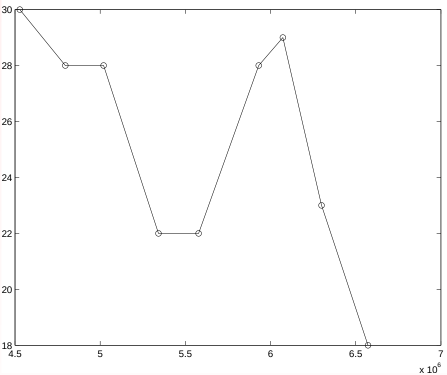
- Symbolic part of out modified (MI22) $IC(\ell)$ predefines the structure.
- Only very small entries from the structure are not kept. The space is then freed and can be further used.
- The final size parametrized by memory multiplier $0 \leq \theta$.
- Additional space distributed (1) uniformly or (2) nonuniformly.

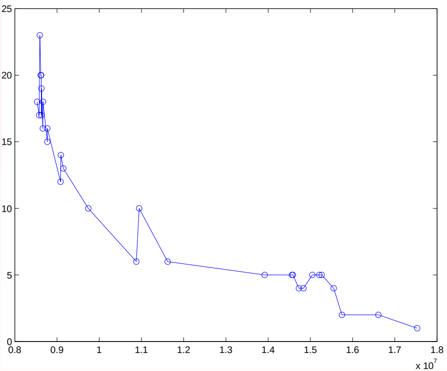
- Nonuniform distribution based on the elimination tree and its row subtrees



Second component of our approach: keeping structure

Experiments with memory multiplier $\theta = 1$: more difficult problem



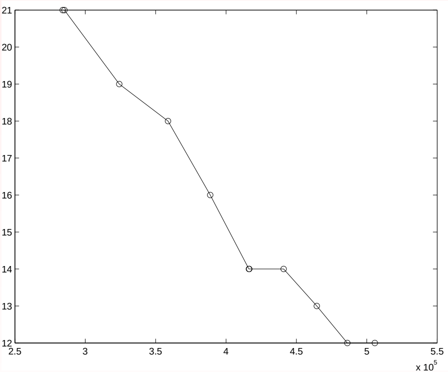


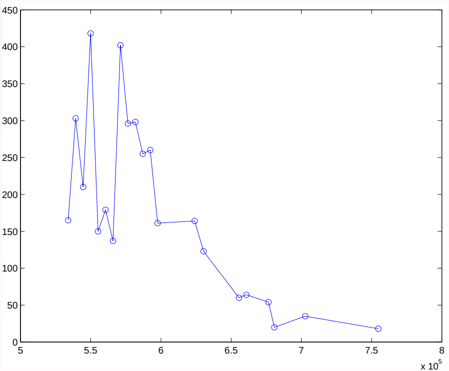
MI22 ILUT (ICT)

NASASRB, structural mechanics, n=54870

Second component of our approach: keeping structure

Experiments with memory multiplier $\theta = 1$: simpler problem





MI22 ILUT (ICT)

S1RMT3M1, cylindrical shell problem, n=5489

MI22 with levels versus $IC(\tau)$(also via MI22)

TUBE1, cylindrical shell, n=21498

struc	drop=0.0		drop=10^{-7}	
level	size	its	size	its
5	1250952	†	1227570	†
6	1660827	429	1618808	423
7	1807337	405	1756733	408
8	2178312	272	2104496	281
9	2368289	260	2280081	267
10	3026431	184	2873613	185
11	3968731	426	3656826	335
12	4874629	†	4398086	†
13	5849563	†	5178688	†
14	6840871	664	5938543	647
15	7838623	262	6680235	215

IC(τ)	size	its
55	280626	†
50	1458024	†
45	2076970	†
40	2252687	†
1e-3	16139618	†
1e-4	9001342	†
5e-5	9649083	471
2e-5	9610841	87
1e-5	10050227	18
5e-6	10741254	6
1e-6	12451396	2
0	21802746	1

MI22 with levels versus $IC(\tau)$(also via MI22)

TUBE1, cylindrical shell, n=21498

struc	drop=0.0		drop=10^{-7}	
level	size	its	size	its
5	1250952	†	1227570	†
6	1660827	429	1618808	423
7	1807337	405	1756733	408
8	2178312	272	2104496	281
9	2368289	260	2280081	267
10	3026431	184	2873613	185
11	3968731	426	3656826	335
12	4874629	†	4398086	†
13	5849563	†	5178688	†
14	6840871	664	5938543	647
15	7838623	262	6680235	215

IC(τ)	size	its
55	280626	†
50	1458024	†
45	2076970	†
40	2252687	†
1e-3	16139618	†
1e-4	9001342	†
5e-5	9649083	471
2e-5	9610841	87
1e-5	10050227	18
5e-6	10741254	6
1e-6	12451396	2
0	21802746	1

But: Reorderings may minimize the effect.

MI22 and memory multiplier $\theta < 1$

simple problem, 2D POISSON on a square, n=10000

memory	drop=0.0		drop=10^{-4}	
0.2	10000	160	10000	160
0.3	11880	226	11880	226
0.4	15840	205	15840	205
0.5	19800	155	19800	155
0.6	27729	141	27721	142
0.7	35597	111	35524	111
0.8	39583	63	39583	63
0.9	39584	58	39584	63
1	39601	41	39601	41
1.5	59401	42	59401	43
2	79202	42	79202	42
3	118803	42	118803	40
4	158404	42	158404	42
5	198005	44	198005	39
8	316808	42	316808	32
10	396010	42	396010	22
15	594015	37	471092	6

Outline

Conclusions

- Preserving the structure may play significant role in incomplete decompositions.

Conclusions

- Preserving the structure may play significant role in incomplete decompositions.
- Codes may be reasonably fast and robust.

Conclusions

- Preserving the structure may play significant role in incomplete decompositions.
- Codes may be reasonably fast and robust.
- MI22 code of Harwell Subroutine Library offers a way to implement this.

Conclusions

- Preserving the structure may play significant role in incomplete decompositions.
- Codes may be reasonably fast and robust.
- MI22 code of Harwell Subroutine Library offers a way to implement this.

Thank you for your attention!

Last but not least

Thank you for your attention!

Last but not least

Thank you for your attention!

Last but not least

Thank you for your attention!

Last but not least

Thank you for your attention!