Balancing Incomplete Factorizations for Preconditioning (Do we know standard matrix decompositions?)

Miroslav Tůma

Institute of Computer Science

Academy of Sciences of the Czech Republic

joint work with

Rafael Bru, José Marín, José Mas

Universidad Politécnica de Valencia.

Householder Symposium XVII.

June 1-6, 2008, Zeuthen, Germany

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
- but also: sufficiently robust

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
- but also: sufficiently robust
- 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
- but also: sufficiently robust
- sparse enough
- providing just sufficient approximation of the algebraic problem, and not more if this makes computations faster.

Structure of this talk

Structure of this talk

1. Very schematic description of a couple of ideas for algebraic preconditioning. Showing how easily they can fail.

Structure of this talk

1. Very schematic description of a couple of ideas for algebraic preconditioning. Showing how easily they can fail.
2. Drawing attention to some approaches which exploit info on matrix inverse.

Structure of this talk

1. Very schematic description of a couple of ideas for algebraic preconditioning. Showing how easily they can fail.
2. Drawing attention to some approaches which exploit info on matrix inverse.
3. Presenting an approach based on a new way to decompose the input matrix and not on preprocessings, postprocessings, additional frameworks or modifications

Outline

Incomplete decompositions

Trivial paterns

Incompleteness based on pattern or on values?

Incomplete decompositions

Trivial paterns

Incompleteness based on pattern or on values?

A) Very simple patterns for cheap / cache-efficient preconditioners?

Incomplete decompositions

Trivial paterns

Incompleteness based on pattern or on values?

A) Very simple patterns for cheap / cache-efficient preconditioners?

Example: banded pattern: BCSSTK38, $n = 8032$, $nz = 181,746$

bandwidth (full)	iterations
1	426
3	821
5	648
9	1638
15	792
1011	105
1311	56
1511	†
3111	35
4111	18

Incomplete decompositions

Matrix-based patterns

B) Matrix-based patterns for preconditioners?

Incomplete decompositions

Matrix-based patterns

B) Matrix-based patterns for preconditioners?

Example: pattern of $A = \text{pattern } (L + L^T), \;\; L = tril(A)$

Well known: error R of the decomposition $A = LL^T - R$ satisfies:

$$r_{ij} = 0 \;\; \text{if} \;\; (i,j) \in \text{pattern}$$

Incomplete decompositions

Matrix-based patterns

B) Matrix-based patterns for preconditioners?

Example: pattern of $A = \text{pattern } (L + L^T), \;\; L = tril(A)$

Well known: error R of the decomposition $A = LL^T - R$ satisfies:

$$r_{ij} = 0 \;\; \text{if} \;\; (i,j) \in \text{pattern}$$

As above, error outside the prescribed pattern can be arbitrary, if (linear system, PDE, etc.) model allows this.

Incomplete decompositions

Enhanced matrix-based patterns

C) Enhancing matrix-based patterns for preconditioners?

Incomplete decompositions

Enhanced matrix-based patterns

C) Enhancing matrix-based patterns for preconditioners?

Example: Pattern of powers of A

Incomplete decompositions

Enhanced matrix-based patterns

C) Enhancing matrix-based patterns for preconditioners?

Example: Pattern of powers of A

- Motivated by matrix inverse

Incomplete decompositions

Enhanced matrix-based patterns

C) Enhancing matrix-based patterns for preconditioners?

Example: Pattern of powers of A

- Motivated by matrix inverse
- Well-known how powers of A are related to the decomposition

Incomplete decompositions

Enhanced matrix-based patterns

C) Enhancing matrix-based patterns for preconditioners?

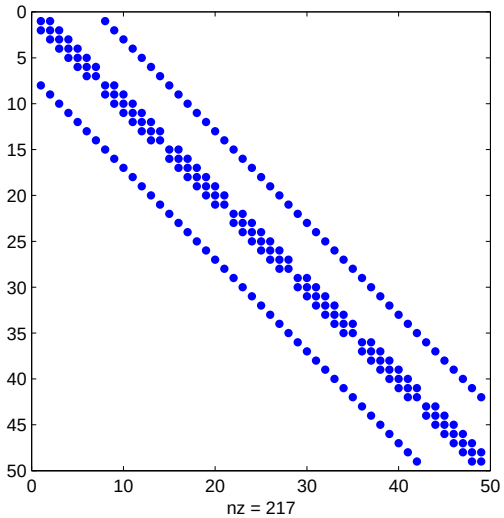
Example: Pattern of powers of A

- Motivated by matrix inverse
- Well-known how powers of A are related to the decomposition
- Sometimes called dropping by levels

C) Enhancing matrix-based patterns for preconditioners?

Example: Pattern of powers of A

- Motivated by matrix inverse
- Well-known how powers of A are related to the decomposition
- Sometimes called dropping by levels

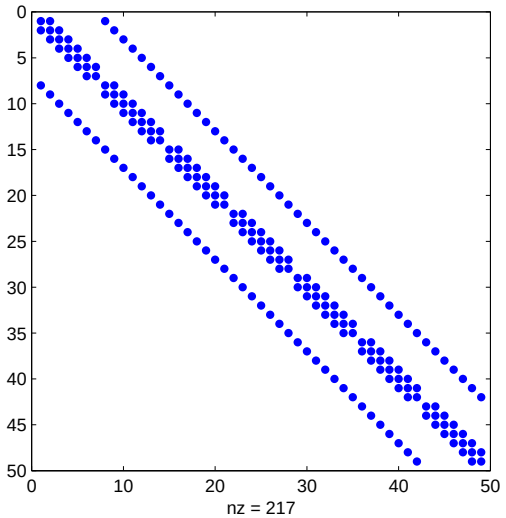


ILU(0)

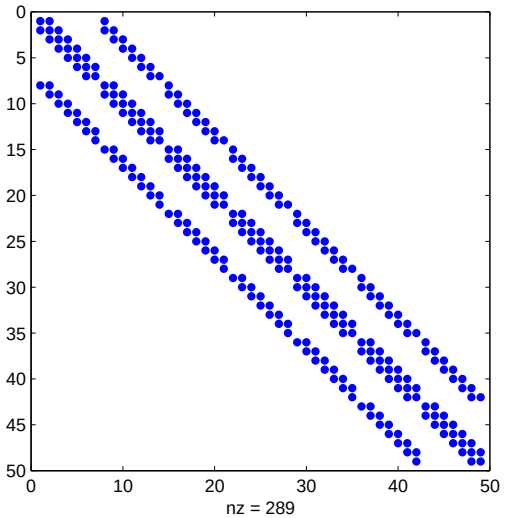
C) Enhancing matrix-based patterns for preconditioners?

Example: Pattern of powers of A

- Motivated by matrix inverse
- Well-known how powers of A are related to the decomposition
- Sometimes called dropping by levels



ILU(0)

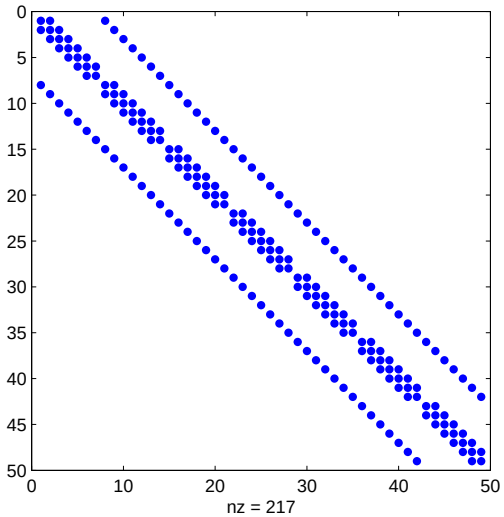


ILU(1)

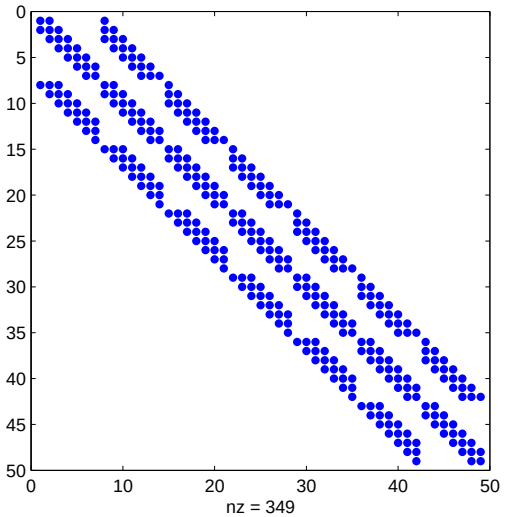
C) Enhancing matrix-based patterns for preconditioners?

Example: Pattern of powers of A

- Motivated by matrix inverse
- Well-known how powers of A are related to the decomposition
- Sometimes called dropping by levels



ILU(0)

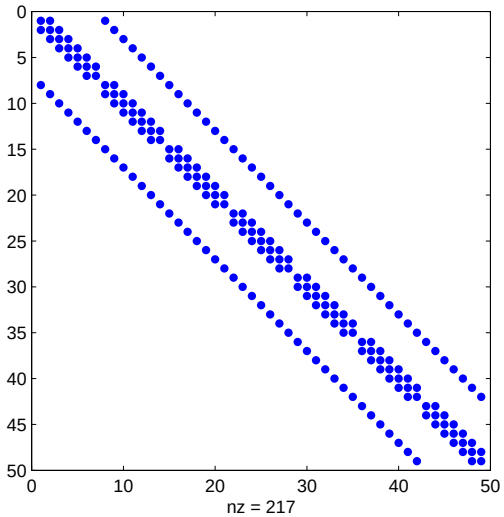


ILU(2)

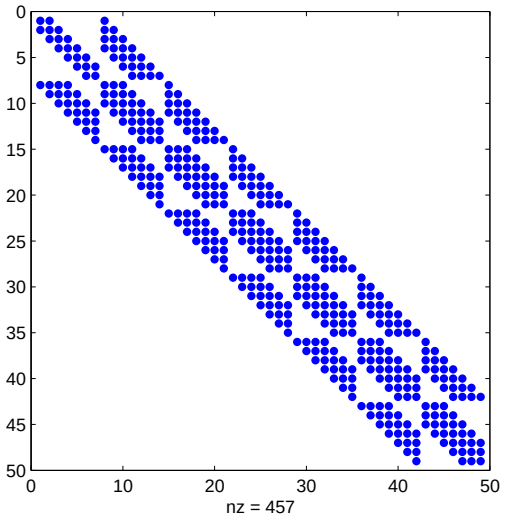
C) Enhancing matrix-based patterns for preconditioners?

Example: Pattern of powers of A

- Motivated by matrix inverse
- Well-known how powers of A are related to the decomposition
- Sometimes called dropping by levels



ILU(0)

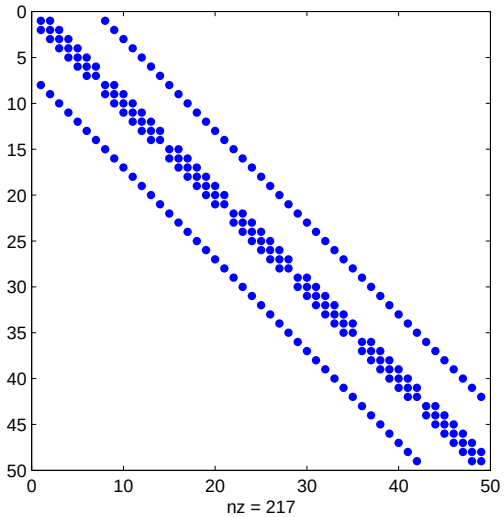


ILU(3)

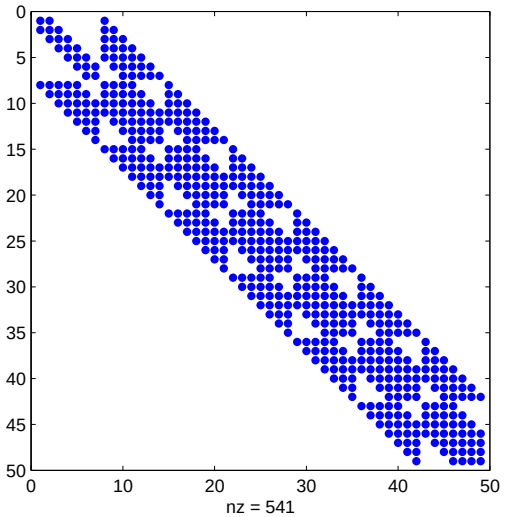
C) Enhancing matrix-based patterns for preconditioners?

Example: Pattern of powers of A

- Motivated by matrix inverse
- Well-known how powers of A are related to the decomposition
- Sometimes called dropping by levels



ILU(0)

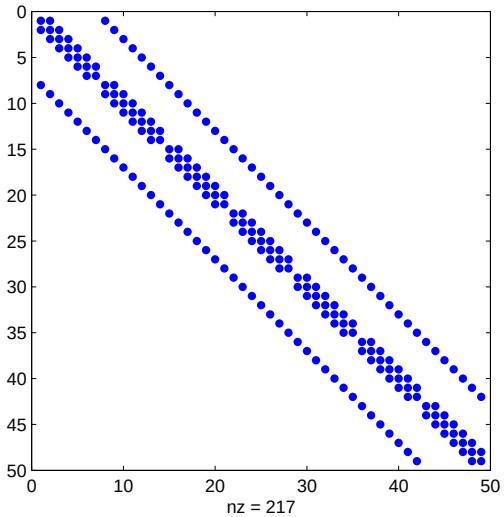


ILU(4)

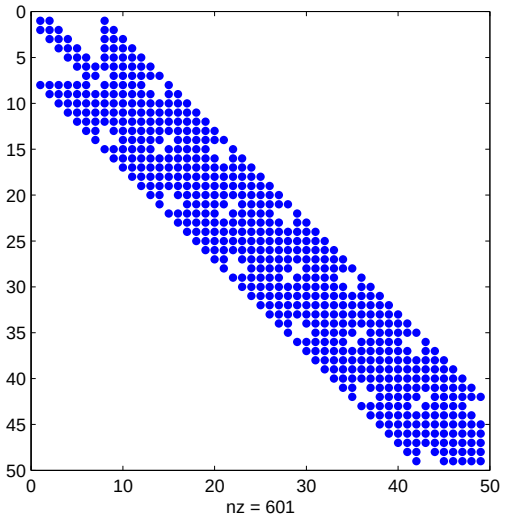
C) Enhancing matrix-based patterns for preconditioners?

Example: Pattern of powers of A

- Motivated by matrix inverse
- Well-known how powers of A are related to the decomposition
- Sometimes called dropping by levels



ILU(0)

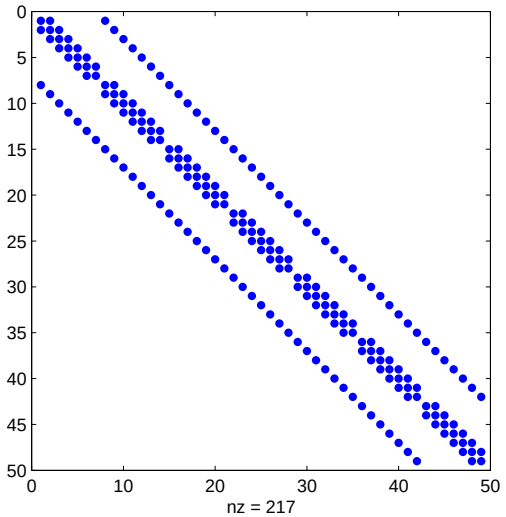


ILU(5)

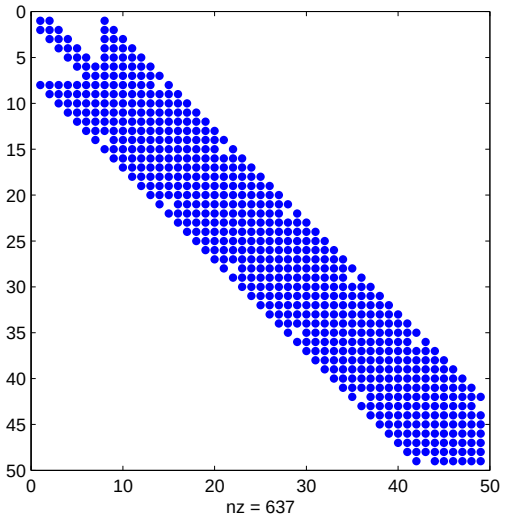
C) Enhancing matrix-based patterns for preconditioners?

Example: Pattern of powers of A

- Motivated by matrix inverse
- Well-known how powers of A are related to the decomposition
- Sometimes called dropping by levels



ILU(0)

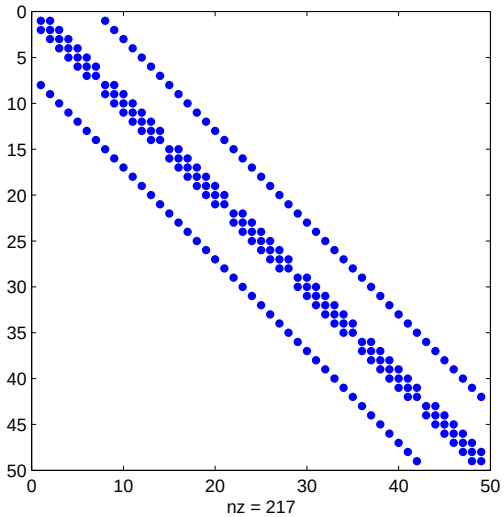


ILU(6)

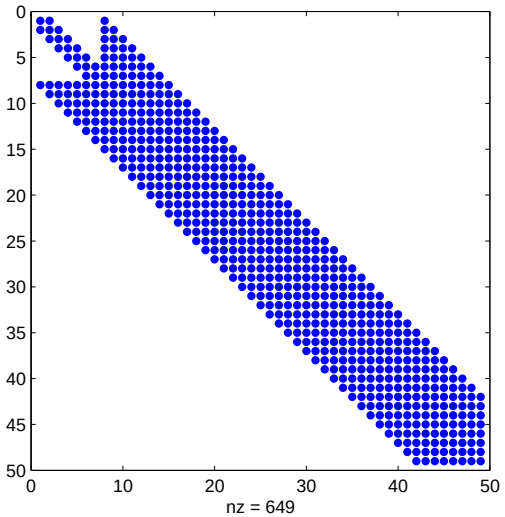
C) Enhancing matrix-based patterns for preconditioners?

Example: Pattern of powers of A

- Motivated by matrix inverse
- Well-known how powers of A are related to the decomposition
- Sometimes called dropping by levels



ILU(0)



ILU(7)

Incomplete decompositions

Enhanced matrix-based patterns: using levels

- Fast computation (Hysom, Pothen, 2001)
- Typically expensive to apply for modest number of levels (powers of A)

Incomplete decompositions

Enhanced matrix-based patterns: using levels

- Fast computation (Hysom, Pothen, 2001)
- Typically expensive to apply for modest number of levels (powers of A)

Example: Matrix ENGINE, $n = 143,571$, $nz = 2,424,822$

levels	size prec	iterations
0	2,424,822	523
1	4,458,588	300
2	7,595,466	199
3	12,128,289	115
4	18,078,603	87
5	25,474,380	54
6	34,153,746	45
7	43,861,328	46
8	54,276,063	36

Incomplete decompositions

External enhancements

D) Preprocessing, postprocessing, modifications

preprocessing: reorderings, prefiltration, scalings

Incomplete decompositions

External enhancements

D) Preprocessing, postprocessing, modifications

preprocessing: reorderings, prefiltration, scalings

Example: Matrix ENGINE, $n = 143,571$, $nz = 2,424,822$, MMD

Incomplete decompositions

External enhancements

D) Preprocessing, postprocessing, modifications

preprocessing: reorderings, prefiltration, scalings

Example: Matrix ENGINE, $n = 143,571$, $nz = 2,424,822$, MMD

levels	size	its	size	its
0	2,424,822	523	2,424,822	439
1	4,458,588	300	4,394,040	214
2	7,595,466	199	6,509,826	159
3	12,128,289	115	8,859,522	96
4	18,078,603	87	11,292,927	66
5	25,474,380	54	13,664,157	49
6	34,153,746	45	15,891,321	34
7	43,861,328	46	†	†
8	54,276,063	36	19,590,303	18

Incomplete decompositions

External enhancements

D) Preprocessing, postprocessing, modifications

preprocessing: reorderings, prefiltration, scalings

Example: Matrix ENGINE, $n = 143,571$, $nz = 2,424,822$, MMD

levels	size	its	size	its
0	2,424,822	523	2,424,822	439
1	4,458,588	300	4,394,040	214
2	7,595,466	199	6,509,826	159
3	12,128,289	115	8,859,522	96
4	18,078,603	87	11,292,927	66
5	25,474,380	54	13,664,157	49
6	34,153,746	45	15,891,321	34
7	43,861,328	46	†	†
8	54,276,063	36	19,590,303	18

Similarly: postprocessings, diagonal/offdiagonal modifications based on sizes of entries

Incomplete decompositions

Values

E) Values should be considered throughout

Incomplete decompositions

Values

E) Values should be considered throughout

- again: model can provide useful info (decay, etc.)

Incomplete decompositions

Values

E) Values should be considered throughout

- again: model can provide useful info (decay, etc.)
- if only magnitudes of entries are used - structural information may be lost

Incomplete decompositions

Values

E) Values should be considered throughout

- again: model can provide useful info (decay, etc.)
- if only magnitudes of entries are used - structural information may be lost
- more complicated schemes may strongly restrict implementation (e.g., if both row and column access for intermediate quantities is needed)

Incomplete decompositions

Values

E) Values should be considered throughout

- again: model can provide useful info (decay, etc.)
- if only magnitudes of entries are used - structural information may be lost
- more complicated schemes may strongly restrict implementation (e.g., if both row and column access for intermediate quantities is needed)

Example: Matrix LDOOR, $n = 952,203$, $nz = 23,737,339$ (mostly various SPD variants of ILUT (Saad, 1994))

Incomplete decompositions

Values

E) Values should be considered throughout

- again: model can provide useful info (decay, etc.)
- if only magnitudes of entries are used - structural information may be lost
- more complicated schemes may strongly restrict implementation (e.g., if both row and column access for intermediate quantities is needed)

Example: Matrix LDOOR, $n = 952,203$, $nz = 23,737,339$ (mostly various SPD variants of ILUT (Saad, 1994))

precond / precond. size	its
Jacobi	810
IC(0)	> 1000
23,838,704	> 1000
30,047,027	> 1000
37,809,756	> 1000

Incomplete decompositions

Values (continued)

E) Values should be considered throughout (continued)

Incomplete decompositions

Values (continued)

E) Values should be considered throughout (continued)

Example: Matrix ENGINE, $n = 143571$, $nz = 2424822$:

Incomplete decompositions

Values (continued)

E) Values should be considered throughout (continued)

Example: Matrix ENGINE, $n = 143571$, $nz = 2424822$:

- fast convergence with IC(0)

Incomplete decompositions

Values (continued)

E) Values should be considered throughout (continued)

Example: Matrix ENGINE, $n = 143571$, $nz = 2424822$:

- fast convergence with IC(0)
- very bad results for all tested cases of IC by value

Incomplete decompositions

Values (continued)

E) Values should be considered throughout (continued)

Example: Matrix ENGINE, $n = 143571$, $nz = 2424822$:

- fast convergence with IC(0)
- very bad results for all tested cases of IC by value
- Consequently: Still very far from any predictable behavior: total lack of robustness

Incomplete decompositions

Values (continued)

E) Values should be considered throughout (continued)

Example: Matrix ENGINE, $n = 143571$, $nz = 2424822$:

- fast convergence with IC(0)
- very bad results for all tested cases of IC by value
- Consequently: Still very far from any predictable behavior: total lack of robustness
- Any idea?: Use inverse of A during the construction

Incomplete decompositions

Values (continued)

E) Values should be considered throughout (continued)

Example: Matrix ENGINE, $n = 143571$, $nz = 2424822$:

- fast convergence with IC(0)
- very bad results for all tested cases of IC by value
- Consequently: Still very far from any predictable behavior: total lack of robustness
- Any idea?: Use inverse of A during the construction
- Next: IF via/with inverses. But, see also work of Saad and Bollhöfer, 2002.

Outline

IF via approximate inverses

RIF motivation

RIF (Robust incomplete factorization)

IF via approximate inverses

RIF motivation

RIF (Robust incomplete factorization)

- Based on factorized approximate inverses, Benzi, T., 2003.

IF via approximate inverses

RIF motivation

RIF (Robust incomplete factorization)

- Based on factorized approximate inverses, Benzi, T., 2003.
- Consider the triangular decomposition $A^{-1} \sim \widehat{L^{-T}} D^{-1} \widehat{L^{-1}}$.

IF via approximate inverses

RIF motivation

RIF (Robust incomplete factorization)

- Based on factorized approximate inverses, Benzi, T., 2003.
- Consider the triangular decomposition $A^{-1} \sim \widehat{L^{-T}} D^{-1} \widehat{L^{-1}}$.
- Notation: $L, \hat{L} : (l_{ij})$, $L^{-1}, \widehat{L^{-1}} : (\ell_{ij}) \equiv (\ell_j)$

IF via approximate inverses

RIF motivation

RIF (Robust incomplete factorization)

- Based on factorized approximate inverses, Benzi, T., 2003.
- Consider the triangular decomposition $A^{-1} \sim \widehat{L^{-T}} D^{-1} \widehat{L^{-1}}$.
- Notation: $L, \hat{L} : (l_{ij})$, $L^{-1}, \widehat{L^{-1}} : (\ell_{ij}) \equiv (\ell_j)$
- Compare with the (exact) LDL^T decomposition of A:

IF via approximate inverses

RIF motivation

RIF (Robust incomplete factorization)

- Based on factorized approximate inverses, Benzi, T., 2003.
- Consider the triangular decomposition $A^{-1} \sim \widehat{L^{-T}} D^{-1} \widehat{L^{-1}}$.
- Notation: $L, \hat{L} : (l_{ij})$, $L^{-1}, \widehat{L^{-1}} : (\ell_{ij}) \equiv (\ell_j)$
- Compare with the (exact) LDL^T decomposition of A:
  Factor L of $A = LDL^T$ is $L = AL^{-T}D^{-1}$
- It can be easily retrieved from this inverse factorization

$$\Downarrow$$

$$AL^{-1} = LD, \text{ lower triangular}$$

IF via approximate inverses

RIF motivation

RIF (Robust incomplete factorization)

- Based on factorized approximate inverses, Benzi, T., 2003.
- Consider the triangular decomposition $A^{-1} \sim \widehat{L^{-T}} D^{-1} \widehat{L^{-1}}$.
- Notation: $L, \hat{L} : (l_{ij})$, $L^{-1}, \widehat{L^{-1}} : (\ell_{ij}) \equiv (\ell_j)$
- Compare with the (exact) LDL^T decomposition of A:
  Factor L of $A = LDL^T$ is $L = AL^{-T}D^{-1}$
- It can be easily retrieved from this inverse factorization

$$\Downarrow$$

$$AL^{-1} = LD, \text{ lower triangular}$$

$$\frac{\langle e_k, A\ell_j \rangle}{d_k} = l_{kj} \text{ for } k \geq j$$

IF via approximate inverses

RIF motivation

RIF (Robust incomplete factorization)

- Based on factorized approximate inverses, Benzi, T., 2003.
- Consider the triangular decomposition $A^{-1} \sim \widehat{L^{-T}} D^{-1} \widehat{L^{-1}}$.
- Notation: $L, \hat{L} : (l_{ij})$, $L^{-1}, \widehat{L^{-1}} : (\ell_{ij}) \equiv (\ell_j)$
- Compare with the (exact) LDL^T decomposition of A:
  Factor L of $A = LDL^T$ is $L = AL^{-T}D^{-1}$
- It can be easily retrieved from this inverse factorization

$$\Downarrow$$

$$AL^{-1} = LD, \text{ lower triangular}$$

$$\frac{\langle e_k, A\ell_j \rangle}{d_k} = l_{kj} \text{ for } k \geq j$$

From L^{-1} we can get L (from $\widehat{L^{-1}}$ get $\hat{L}$)

IF via approximate inverses

RIF implementation

Note: $l_{kj} = \dfrac{\langle e_k, A\ell_j \rangle}{d_k} \equiv \dfrac{\langle \ell_k, A\ell_j \rangle}{d_k}$ for $k \geq j$

IF via approximate inverses

RIF implementation

Note: $l_{kj} = \dfrac{\langle e_k, A\ell_j \rangle}{d_k} \equiv \dfrac{\langle \ell_k, A\ell_j \rangle}{d_k}$ for $k \geq j$

$$\Downarrow$$

IF via approximate inverses

RIF implementation

$$\text{Note: } l_{kj} = \frac{\langle e_k, A\ell_j \rangle}{d_k} \equiv \frac{\langle \ell_k, A\ell_j \rangle}{d_k} \text{ for } k \geq j$$

$$\Downarrow$$

- The latter equivalence provides a breakdown-free implementation (Benzi, T., 2003).

IF via approximate inverses

RIF implementation

$$\text{Note: } l_{kj} = \frac{\langle e_k, A\ell_j \rangle}{d_k} \equiv \frac{\langle \ell_k, A\ell_j \rangle}{d_k} \text{ for } k \geq j$$

$$\Downarrow$$

- The latter equivalence provides a breakdown-free implementation (Benzi, T., 2003).
- Experimentally, it is often more space efficient for the same iteration counts.

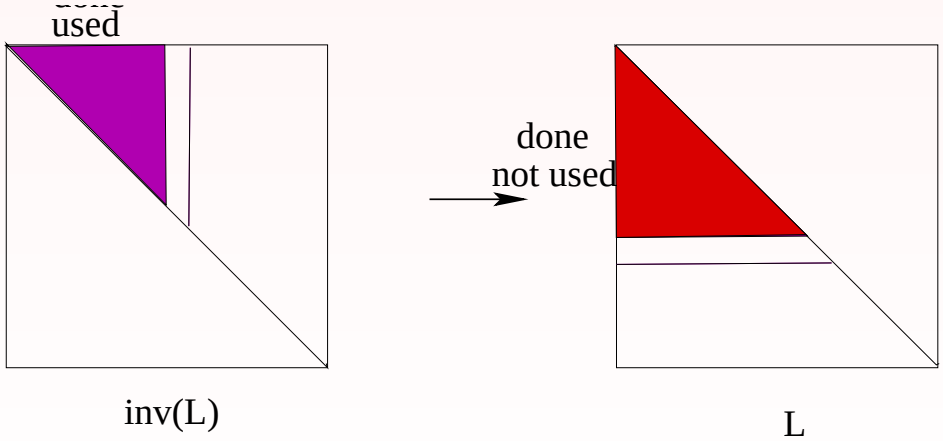
IF via approximate inverses

RIF implementation

$$\text{Note: } l_{kj} = \frac{\langle e_k, A\ell_j \rangle}{d_k} \equiv \frac{\langle \ell_k, A\ell_j \rangle}{d_k} \text{ for } k \geq j$$

$$\Downarrow$$

- The latter equivalence provides a breakdown-free implementation (Benzi, T., 2003).
- Experimentally, it is often more space efficient for the same iteration counts.



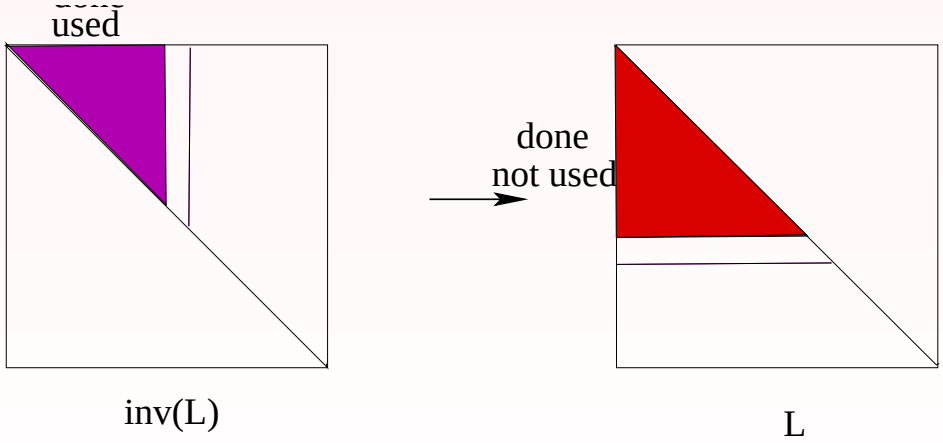
IF via approximate inverses

RIF implementation

$$\text{Note: } l_{kj} = \frac{\langle e_k, A\ell_j \rangle}{d_k} \equiv \frac{\langle \ell_k, A\ell_j \rangle}{d_k} \text{ for } k \geq j$$

$$\Downarrow$$

- The latter equivalence provides a breakdown-free implementation (Benzi, T., 2003).
- Experimentally, it is often more space efficient for the same iteration counts.



One way tranfer of information

Outline

IF with approximate inverses

$(I - A^{-1})^{-1}$ biconjugation

- Consider

$$A = I + \sum_{k=1}^{n} e_k (a_k - e_k)^T$$

IF with approximate inverses

$(I - A^{-1})^{-1}$ biconjugation

- Consider

$$A = I + \sum_{k=1}^{n} e_k (a_k - e_k)^T$$

- Apply n Sherman-Morrison updates to get A^{-1}. (Bru, Cerdán, Marín, Mas, 2003)

IF with approximate inverses

$(I - A^{-1})^{-1}$ biconjugation

- Consider

$$A = I + \sum_{k=1}^{n} e_k (a_k - e_k)^T$$

- Apply n Sherman-Morrison updates to get A^{-1}.
  (Bru, Cerdán, Marín, Mas, 2003)
- The process for $R = (r_k)$, $V = (v_k)$, $D = \mathrm{diag}(d_1, \ldots, d_n)$ for $k = 1, 2, \ldots, n$:

$$r_k = e_k - \sum_{i=1}^{k-1} \frac{v_i^T e_k}{s r_i} r_i \quad , \; v_k = (a_k - e_k)_k - \sum_{i=1}^{k-1} \frac{(a_k - e_k)_k^T r_i}{s r_i} v_i,$$

$$d_k = 1 + (a_k - e_k)_k^T r_k = 1 + v_k^T e_k.$$

IF with approximate inverses

$(I - A^{-1})^{-1}$ biconjugation

- Consider

$$A = I + \sum_{k=1}^{n} e_k (a_k - e_k)^T$$

- Apply n Sherman-Morrison updates to get A^{-1}.
  (Bru, Cerdán, Marín, Mas, 2003)
- The process for $R = (r_k)$, $V = (v_k)$, $D = \mathrm{diag}(d_1, \ldots, d_n)$ for $k = 1, 2, \ldots, n$:

$$r_k = e_k - \sum_{i=1}^{k-1} \frac{v_i^T e_k}{s r_i} r_i \quad , \; v_k = (a_k - e_k)_k - \sum_{i=1}^{k-1} \frac{(a_k - e_k)_k^T r_i}{s r_i} v_i,$$

$$d_k = 1 + (a_k - e_k)_k^T r_k = 1 + v_k^T e_k.$$

- $I - A^{-1} = R D^{-1} V^T$, R unit upper triangular.

IF with approximate inverses

balancing L and L^{-1}

Theorem

(Bru, Mas, Marín, T. 2007) For an SPD A, let there exist the decomposition from above

$$A^{-1} = I - RDV^T \tag{1}$$

and let $A = L\bar{D}L^T$ be the LDL^T decomposition of A. Then

$$V = L\bar{D} - L^{-T},\ R = L^{-1},\ \bar{D} = D.$$

IF with approximate inverses

balancing L and L^{-1}

Theorem

(Bru, Mas, Marín, T. 2007) For an SPD A, let there exist the decomposition from above

$$A^{-1} = I - RDV^T \tag{1}$$

and let $A = L\bar{D}L^T$ be the LDL^T decomposition of A. Then

$$V = L\bar{D} - L^{-T},\ R = L^{-1},\ \bar{D} = D.$$

Pictorially:

IF with approximate inverses

balancing L and L^{-1}

Theorem

(Bru, Mas, Marín, T. 2007) For an SPD A, let there exist the decomposition from above

$$A^{-1} = I - RDV^T \tag{1}$$

and let $A = L\bar{D}L^T$ be the LDL^T decomposition of A. Then

$$V = L\bar{D} - L^{-T},\ R = L^{-1},\ \bar{D} = D.$$

Pictorially:

$$V = \begin{bmatrix} \ddots & & -L^{-T} \\ & \ddots & \\ LD & & \ddots \end{bmatrix}, \qquad \operatorname{diag}(V) = D - I. \tag{2}$$

IF with approximate inverses

balancing L and L^{-1}

$$V = \begin{bmatrix} \ddots & & -L^{-T} \\ & \ddots & \\ LD & & \ddots \end{bmatrix}, \qquad \operatorname{diag}(V) = D - I. \tag{3}$$

IF with approximate inverses

balancing L and L^{-1}

$$V = \begin{bmatrix} \ddots & & -L^{-T} \\ & \ddots & \\ LD & & \ddots \end{bmatrix}, \qquad \operatorname{diag}(V) = D - I. \tag{3}$$

- That is, we compute L and L^{-1} at the same time, by columns. To get L, only V is necessary.

IF with approximate inverses

balancing L and L^{-1}

$$V = \begin{bmatrix} \ddots & & -L^{-T} \\ & \ddots & \\ LD & & \ddots \end{bmatrix}, \qquad \operatorname{diag}(V) = D - I. \tag{3}$$

- That is, we compute L and L^{-1} at the same time, by columns. To get L, only V is necessary.
- Nonsymmetric extension is clear. Further improvements of the algorithm are possible as well.

IF with approximate inverses

balancing L and L^{-1}

$$V = \begin{bmatrix} \ddots & & -L^{-T} \\ & \ddots & \\ LD & & \ddots \end{bmatrix}, \qquad \operatorname{diag}(V) = D - I. \tag{3}$$

- That is, we compute L and L^{-1} at the same time, by columns. To get L, only V is necessary.
- Nonsymmetric extension is clear. Further improvements of the algorithm are possible as well.
- Sparse case used for preconditioning: The factors L and L^{-1} influence (balance) each other during the computation and can be connected via dropping (Bru, Mas, Marín, T. 2007)

IF with approximate inverses

balancing L and L^{-1}

$$V = \begin{bmatrix} \ddots & & -L^{-T} \\ & \ddots & \\ LD & & \ddots \end{bmatrix}, \qquad \operatorname{diag}(V) = D - I. \tag{3}$$

- That is, we compute L and L^{-1} at the same time, by columns. To get L, only V is necessary.
- Nonsymmetric extension is clear. Further improvements of the algorithm are possible as well.
- Sparse case used for preconditioning: The factors L and L^{-1} influence (balance) each other during the computation and can be connected via dropping (Bru, Mas, Marín, T. 2007)
- They can influence each other even in the exact case, purely by the decomposition (Bru, Mas, Marín, T. 2008).

IF with approximate inverses

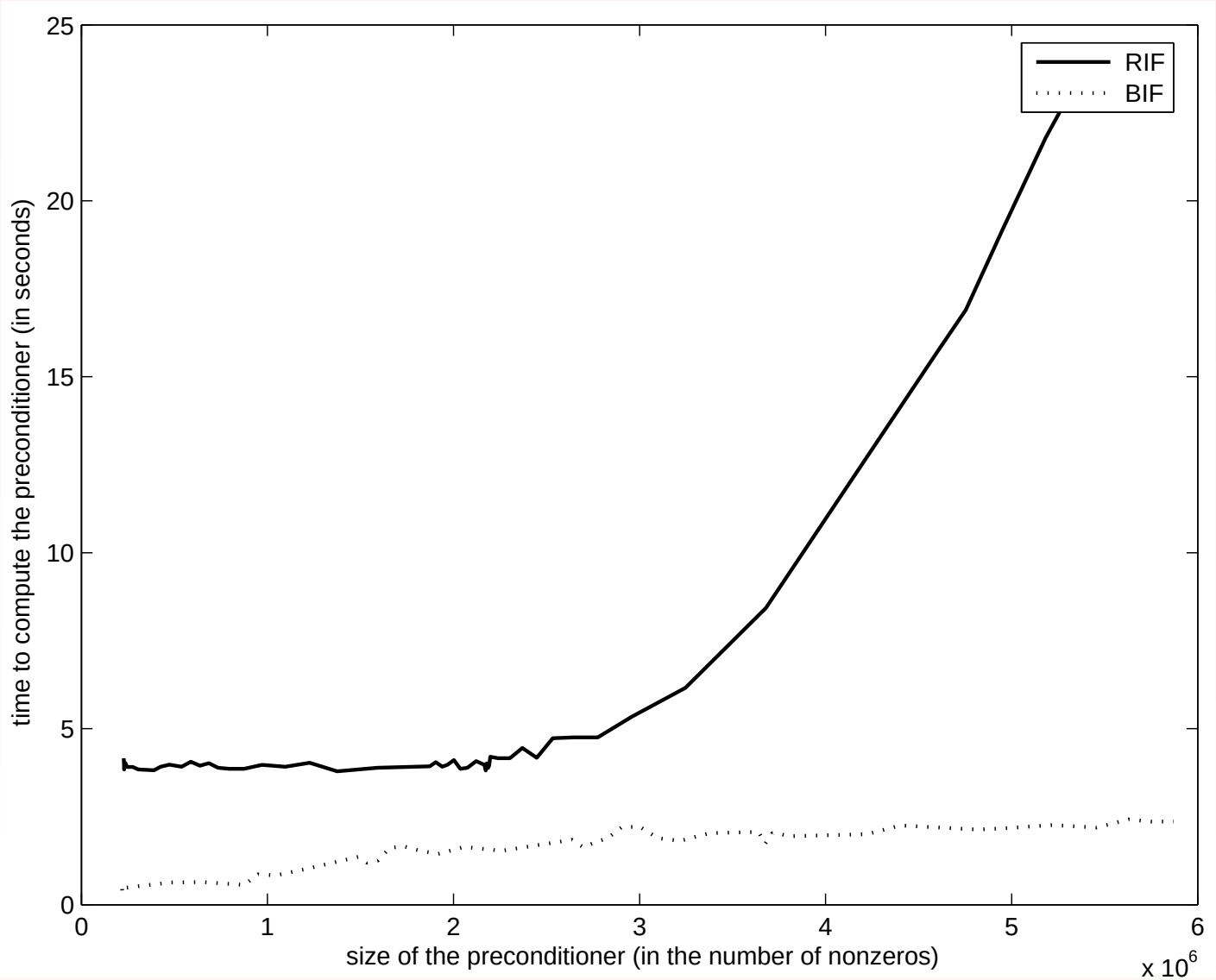
BIF experiments

Example: matrix PWTK, n=217,918, nnz=5,926,171

IF with approximate inverses

BIF experiments

Example: matrix PWTK, n=217,918, nnz=5,926,171



IF with approximate inverses

Of course: not only pros; cons as well

- Taking approximate inverses into account, dropping must be always strong. Prefiltration of entries of A seems to be standard strategy.

IF with approximate inverses

Of course: not only pros; cons as well

- Taking approximate inverses into account, dropping must be always strong. Prefiltration of entries of A seems to be standard strategy.
- We used the inverse-based dropping rules based on Saad, Bollhöfer, 2002. They need to be further investigated. They often seem to influence entries of the factors nonuniformly. Also, the dropping often forces skipping a lot of updates in the decomposition. Is this really the right way to go?

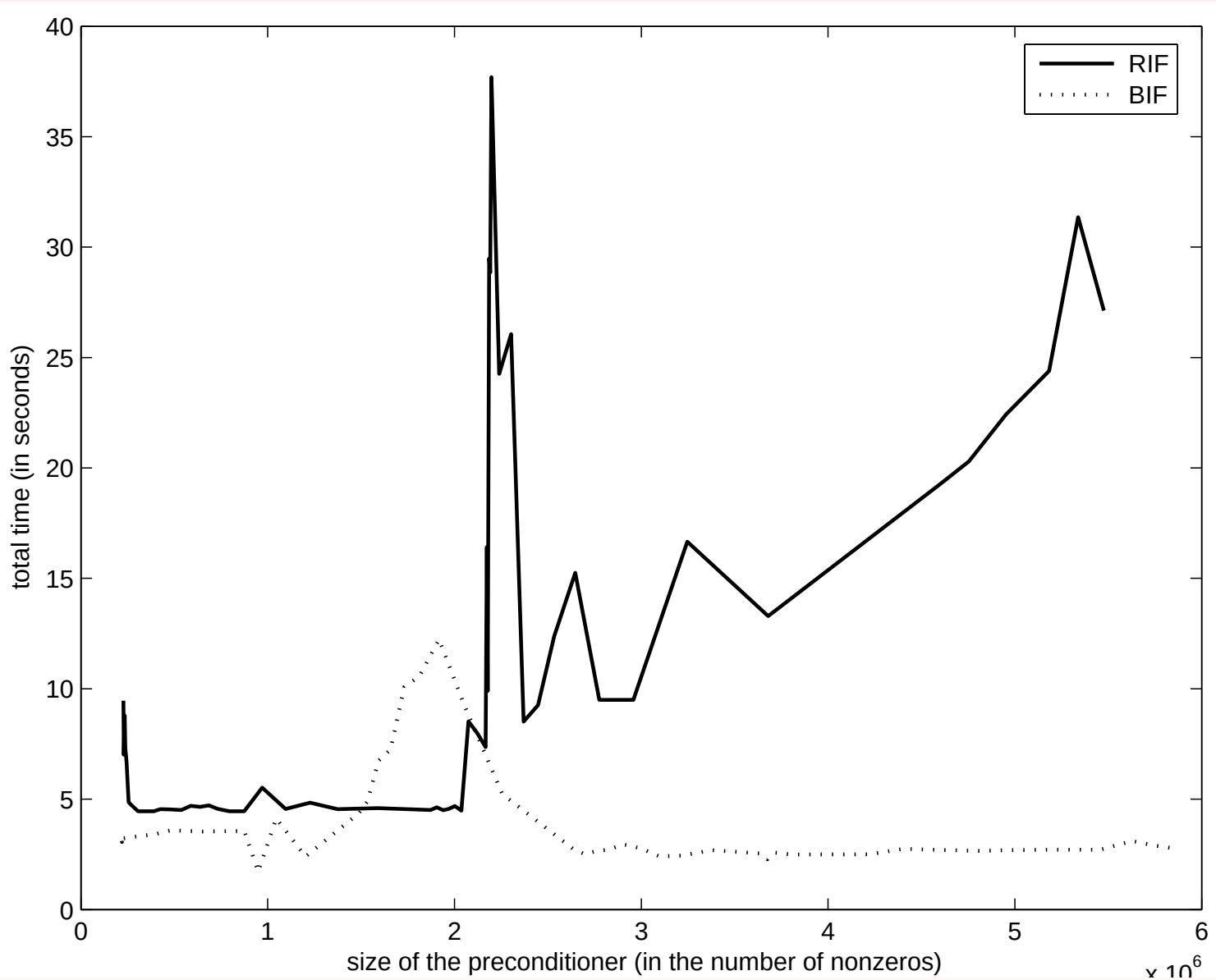
IF with approximate inverses

Of course: not only pros; cons as well

- Taking approximate inverses into account, dropping must be always strong. Prefiltration of entries of A seems to be standard strategy.
- We used the inverse-based dropping rules based on Saad, Bollhöfer, 2002. They need to be further investigated. They often seem to influence entries of the factors nonuniformly. Also, the dropping often forces skipping a lot of updates in the decomposition. Is this really the right way to go?
- Convergence curve is later often flat if we run many iterations. Is the accuracy sufficient for solving sequences from nonlinear solvers?

IF with approximate inverses

BIF experiments



Outline

Conclusions

- Progress in rethinking decompositions still possible.

Conclusions

- Progress in rethinking decompositions still possible.
- Algebraic preconditionings can profit from this.

Conclusions

- Progress in rethinking decompositions still possible.
- Algebraic preconditionings can profit from this.
- Do we understand basic decompositions?

Conclusions

- Progress in rethinking decompositions still possible.
- Algebraic preconditionings can profit from this.
- Do we understand basic decompositions?

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!