Incomplete Cholesky preconditioners that exploit the low-rank property

(theory and practice)

Artem Napov

Service de Métrologie Nucléaire,
Université Libre de Bruxelles

PRECONDITIONING (TU EINDHOVEN)
JUNE 17, 2015

Motivation

The solution of a symmetric positive definite (SPD) system

$$A\mathbf{u} = \mathbf{b}$$

may be obtained with a

- direct method (usually: Cholesky factorization $A = R^T R$)

  $\oplus$ robust
- iterative method (usually: preconditioned conjugate gradient) ;
  efficient if a good SPD preconditioner $B \approx A$ is available, i.e.:
  1. Cheap (to construct, store, "invert", parallelize)
  2. Good approximation of A

  contradictory goals $\Rightarrow$ tradeoff

  Possible black box design:
  - take an exact factorization (here, Cholesky) ;
  - add approximation (to enforce **1.**)

    $\Rightarrow$ incomplete factorization preconditioner

$$B = R^T R \approx A\,.$$

Motivation

Incomplete factorizations

$$B = R^T R \approx A$$

may perform approximations by

- dropping individual entries (mainstream) ;
- using low-rank approximations (emerging) .

Incomplete factorizations:

+ are (almost) black box (approximation threshold required)
+ are (relatively) robust
– may breakdown (but breakdown-free variants exist)
– have no guarantee to converge fast;
  fast convergence if B approximates well A ; that is, if

$$\kappa(R^{-T}AR^{-1}) = \frac{\lambda_{\max}(R^{-T}AR^{-1})}{\lambda_{\min}(R^{-T}AR^{-1})} \quad \text{is small.}$$

Here we present incomplete factorizations that are
breakdown-free and have controllable condition number.

Outline

Theory ...

- One-level variant

  basic approach and underlying analysis

- Multilevel variants

  extensions of analysis to multi-level setting

... and practice

- Sparse solver

  motivation and design choices

- Numerical experiments

  and comparison with other solvers

Outline

Theory ...

- **One-level variant**

  basic approach and underlying analysis

- Multilevel variants

  extensions of analysis to multi-level setting

... and practice

- Sparse solver

  motivation and design choices

- Numerical experiments

  and comparison with other solvers

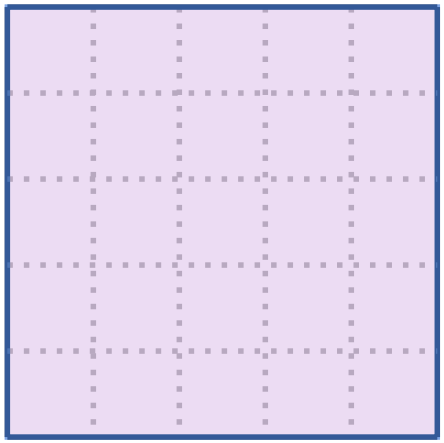
One-level variant : basic ideas

Incomplete (block left-looking) Cholesky

for each block row $k = 1, \ldots, \ell$:

1. factorize
2. solve
3. approximate
4. update (compute new rows)

$(k = 1)$



$\equiv A$

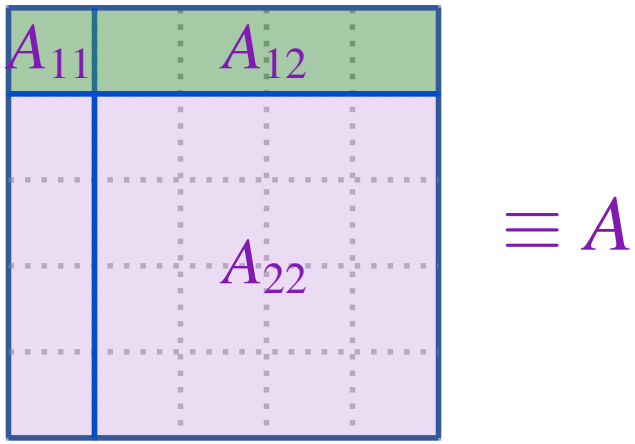
One-level variant : basic ideas

Incomplete (block left-looking) Cholesky

for each block row $k = 1, \ldots, \ell$:

1. factorize
2. solve
3. approximate
4. update (compute new rows)

$(k = 1)$



$\equiv A$

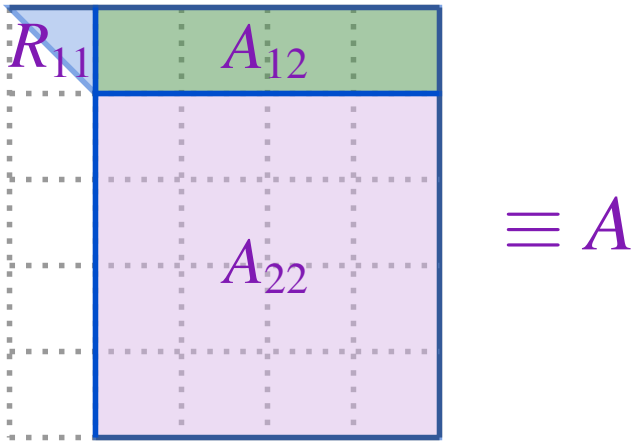
One-level variant : basic ideas

Incomplete (block left-looking) Cholesky

for each block row $k = 1, \ldots, \ell$:

1. factorize
2. solve
3. approximate
4. update (compute new rows)

($k = 1$)



$$A_{11} = R_{11}^T R_{11}$$

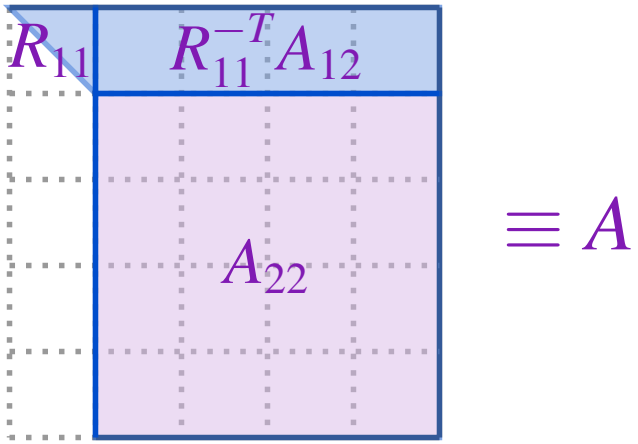
One-level variant : basic ideas

Incomplete (block left-looking) Cholesky

for each block row $k = 1, \ldots, \ell$:

1. factorize
2. solve
3. approximate
4. update (compute new rows)

$(k = 1)$



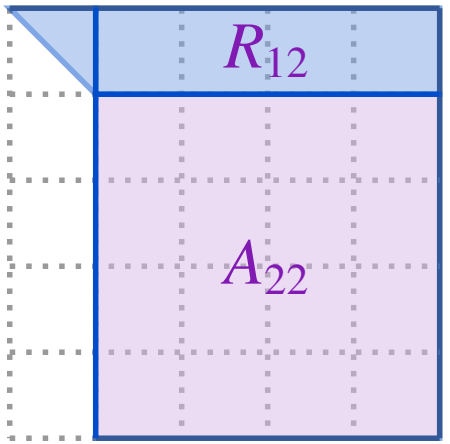
One-level variant : basic ideas

Incomplete (block left-looking) Cholesky

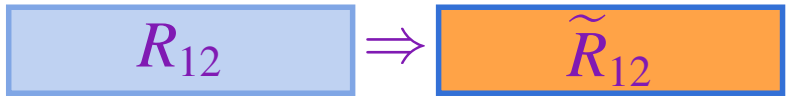
for each block row $k = 1, \ldots, \ell$:

1. factorize
2. solve
3. approximate
4. update (compute new rows)

$(k = 1)$

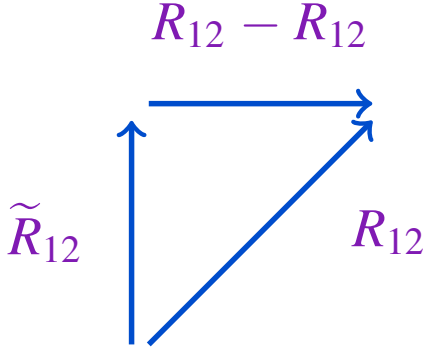


$\equiv A$



The dropping is orthogonal if

$$\widetilde{R}_{12}{}^{T} \left(R_{12} - \widetilde{R}_{12} \right) = O .$$



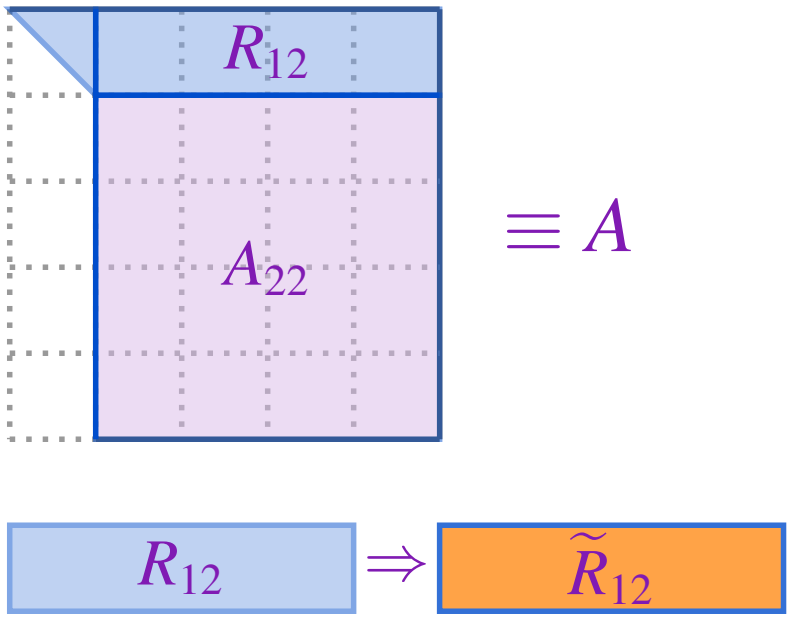
One-level variant : basic ideas

Incomplete (block left-looking) Cholesky

for each block row $k = 1, \ldots, \ell$:

1. factorize
2. solve
3. approximate
4. update (compute new rows)

$(k = 1)$



The dropping is orthogonal if

$$\widetilde{R}_{12}{}^{T} \left(R_{12} - \widetilde{R}_{12} \right) = O \, .$$

This implies monotonicity for Schur complement (for any SPD A!) :

$$\mathbf{v}^T (A_{22} - \widetilde{R}_{12}^T \widetilde{R}_{12}) \mathbf{v} \geq \mathbf{v}^T (A_{22} - R_{12}^T R_{12}) \mathbf{v}$$

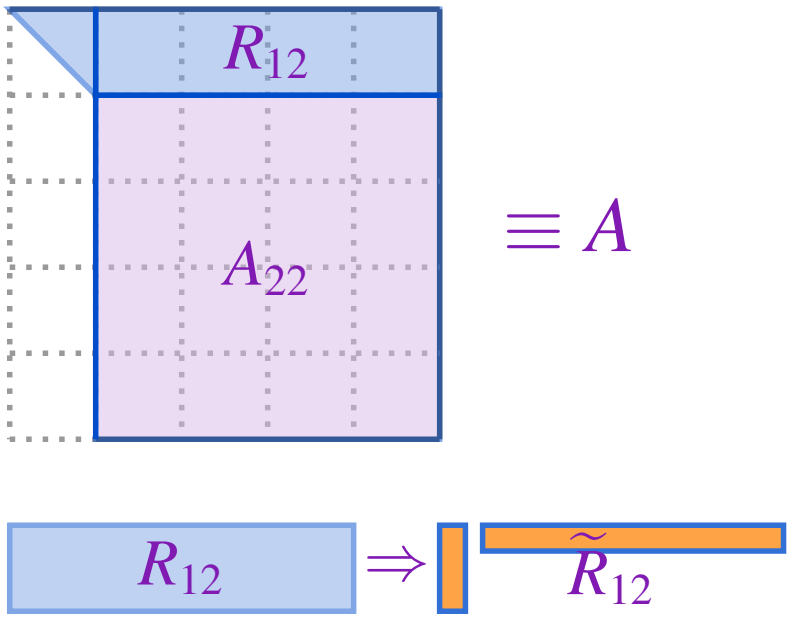
One-level variant : basic ideas

Incomplete (block left-looking) Cholesky

for each block row $k = 1, \ldots, \ell$:

1. factorize
2. solve
3. approximate
4. update (compute new rows)

$(k = 1)$



Example of orthogonal dropping

low-rank approximation via truncated SVD (with absolute threshold tol_a)

$$R_{12} = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 & \\ & \Sigma_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} = \underbrace{U_1(\Sigma_1 V_1^T)}_{\widetilde{R}_{12}} + \underbrace{U_2(\Sigma_2 V_2^T)}_{R_{12}-\widetilde{R}_{12}}$$

If $\| \Sigma_2 \| < tol_a$, then

$$\| R_{12} - \widetilde{R}_{12} \| < tol_a$$

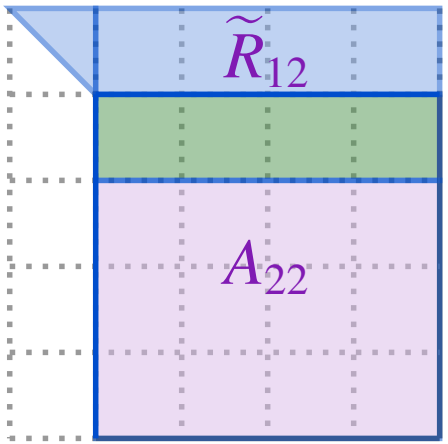
One-level variant : basic ideas

Incomplete (block left-looking) Cholesky

for each block row $k = 1, \ldots, \ell$:

1. factorize
2. solve
3. approximate
4. update (compute new rows)

($k = 1$)



$\equiv B_1$

$$S|_1 = A_{22}|_1 - \widetilde{R}_{12}^T|_1 \widetilde{R}_{12}$$

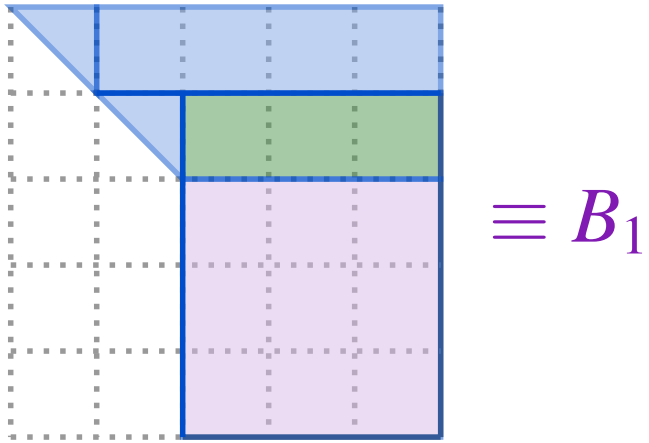
One-level variant : basic ideas

Incomplete (block left-looking) Cholesky

for each block row $k = 1, ..., \ell$:

1. factorize
2. solve
3. approximate
4. update (compute new rows)

$(k = 2)$



$\equiv B_1$

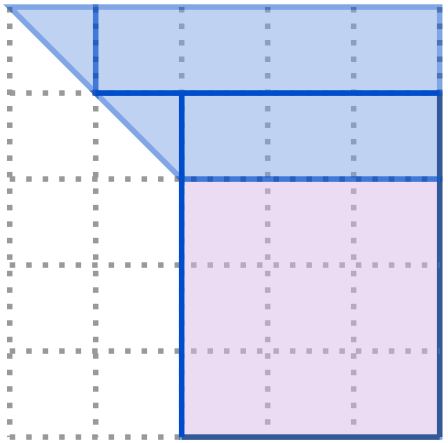
One-level variant : basic ideas

Incomplete (block left-looking) Cholesky

for each block row $k = 1, ..., \ell$:

1. factorize
2. solve
3. approximate
4. update (compute new rows)

$(k = 2)$



$\equiv B_1$

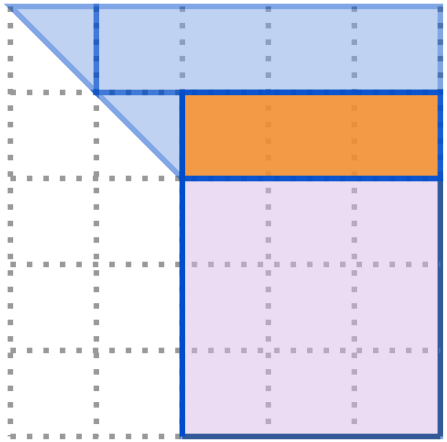
One-level variant : successive approximations

Incomplete (block left-looking) Cholesky

for each block row $k = 1, \ldots, \ell$:

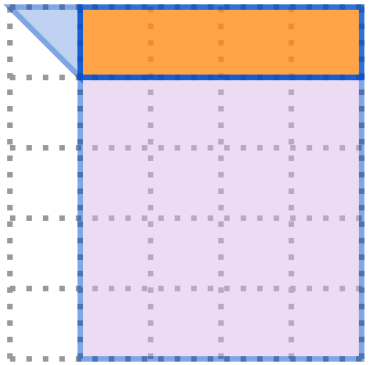
1. factorize
2. solve
3. approximate
4. update (compute new rows)

($k = 2$)

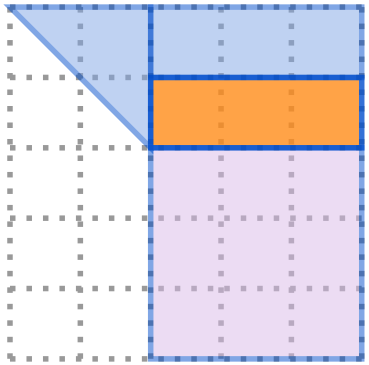


approximate new rows only

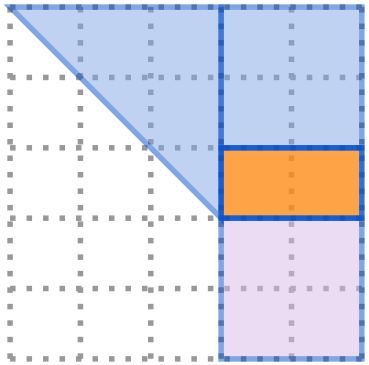
One-level (model) variant



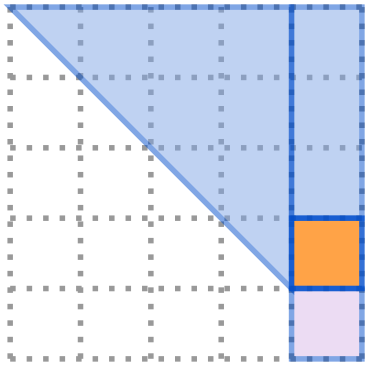
$k = 1$



$k = 2$



$k = 3$



$k = 4$

Accuracy of individual dropping

The orthogonal dropping is assumed, namely

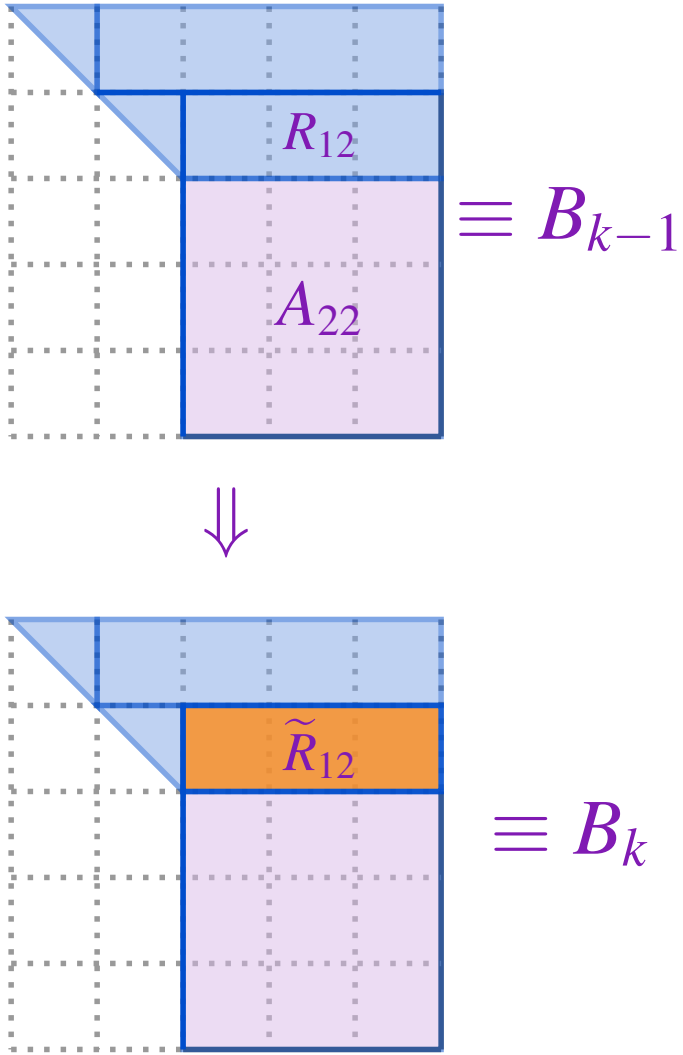
$$\widetilde{R}_{12}{}^{T}\left(R_{12}-\widetilde{R}_{12}\right) \;=\; O$$

The accuracy of individual dropping at step k :

$$\boxed{\gamma_k \;=\; \left\|\left(R_{12}-\widetilde{R}_{12}\right)S_B^{-1/2}\right\| \;<\; 1\,,}$$

with

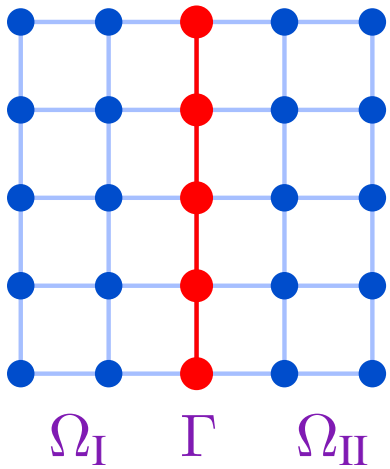
$$S_B \;=\; A_{22}-\widetilde{R}_{12}^{T}\widetilde{R}_{12}\,.$$



Model problem

Model Problem:

$$\begin{cases} -\Delta u + 10^{-4} u = f & \text{in } \Omega = (0,1)^2 \\ \frac{\partial u}{\partial n} = 0 & \text{on } \partial\Omega \end{cases}$$



Full system matrix:

$$A_\Omega = \begin{pmatrix} A_I & & A_{I,\Gamma} \\ & A_{II} & A_{II,\Gamma} \\ A_{I,\Gamma}^T & A_{II,\Gamma}^T & A_\Gamma \end{pmatrix}.$$

Interface system matrix:

$$A = A_\Gamma - A_{I,\Gamma}^T A_I^{-1} A_{\Gamma,I} - A_{II,\Gamma}^T A_{II}^{-1} A_{\Gamma,II}$$

Corresponds to:

- last Schur complement in sparse Cholesky with ND ordering
- Schur complement system in iterative substructuring

Accuracy of individual dropping

The orthogonal dropping is assumed, namely

$$\widetilde{R}_{12}{}^{T}\left(R_{12}-\widetilde{R}_{12}\right) \;=\; O\,.$$

The accuracy of individual dropping at step k:

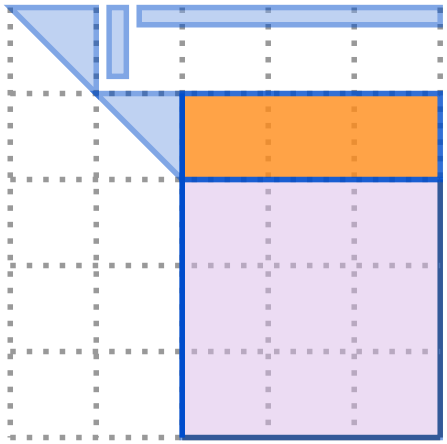
$$\boxed{\gamma_k \;=\; \left\|\left(R_{12}-\widetilde{R}_{12}\right)S_B^{-1/2}\right\| \;<\; 1\,,}$$

with

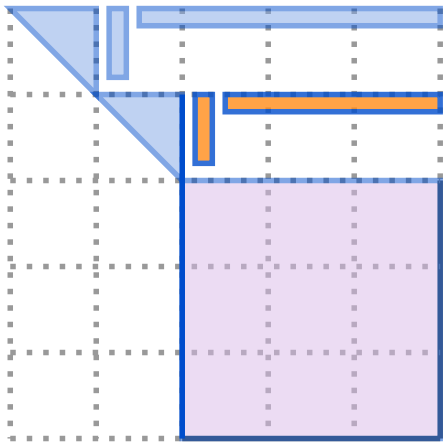
$$S_B \;=\; A_{22}-\widetilde{R}_{12}^{T}\widetilde{R}_{12}\,.$$

Model problem:
($k=1$, block size $=10$, $10^3\times 10^3$ grid,
truncated SVD dropping)

r	0	1	3	5
γ_1	0.99998	0.44	$5.6\,10^{-3}$	$6.2\,10^{-6}$



$\equiv B_{k-1}$



$\equiv B_k$

One-level variant: conditioning analysis (for $\lambda_{\max}$)

- Assume A SPD and that dropping is orthogonal
- Let B_k correspond to the preconditioner at the end of step k, with $A = B_0$
- $B_\ell = R^T R$ is the final preconditioner .
- Only newly computed rows are modified;
- Let

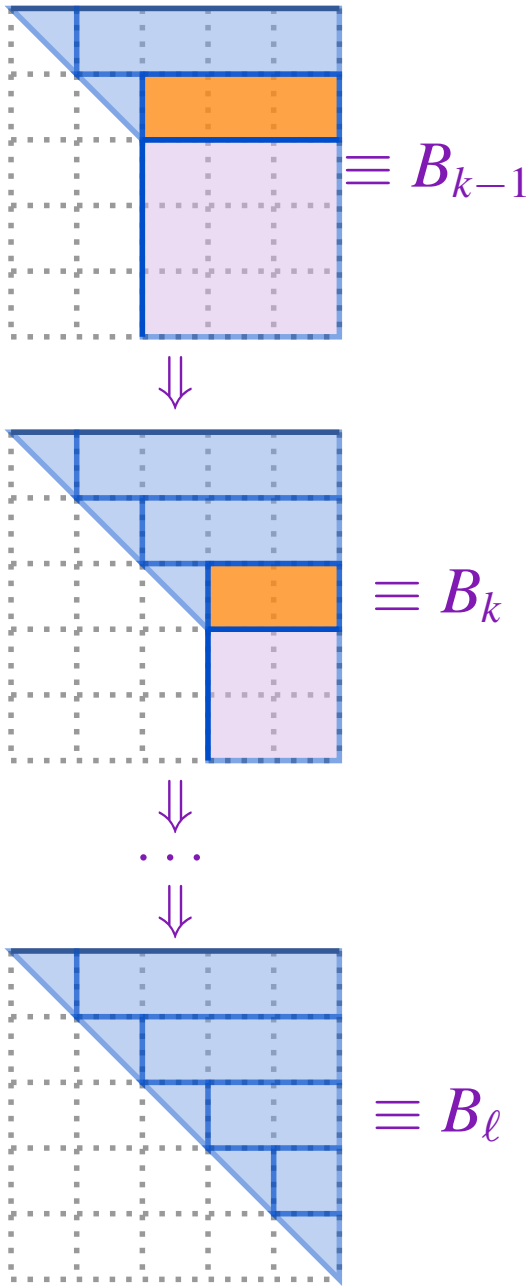
$$\lambda_{\max}^{(k)} = \lambda_{\max}\left(B_\ell^{-1} B_k\right), \quad \lambda_{\max}^{(k-1)} = \lambda_{\max}\left(B_\ell^{-1} B_{k-1}\right)$$

Then

$$\boxed{\lambda_{\max}^{(k)} \leq \lambda_{\max}^{(k-1)} \leq \lambda_{\max}^{(k)} + g\left(\lambda_{\max}^{(k)}, \gamma_k\right)}$$

where

$$g(\lambda, \gamma) = \max_{\beta > 0} \frac{2\gamma\beta - |\lambda - 1|\beta^2}{\beta^2 + \lambda^{-1}}.$$



One-level variant: conditioning analysis (for $\lambda_{\min}$)

- Assume A SPD and that dropping is orthogonal
- Let B_k correspond to the preconditioner at the end of step k, with $A = B_0$
- $B_\ell = R^T R$ is the final preconditioner .
- Only newly computed rows are modified;
- Let

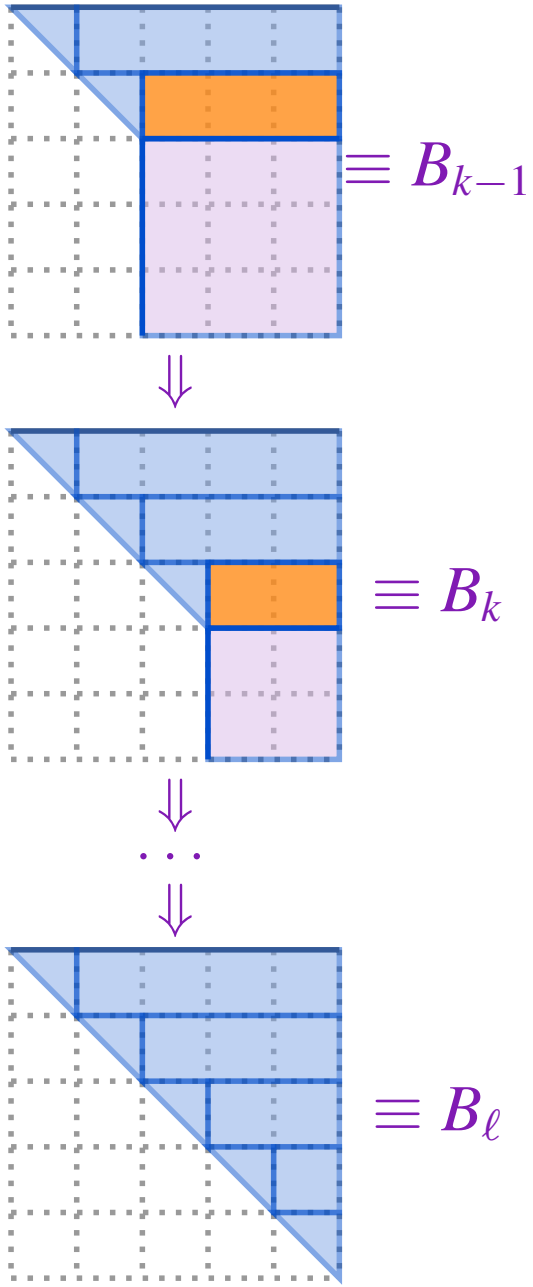
$$\lambda_{\min}^{(k)} = \lambda_{\min}\left(B_\ell^{-1} B_k\right), \quad \lambda_{\min}^{(k)} = \lambda_{\min}\left(B_\ell^{-1} B_{k-1}\right)$$

Then

$$\boxed{\lambda_{\min}^{(k)} \geq \lambda_{\min}^{(k-1)} \geq \lambda_{\min}^{(k)} - g(\lambda_{\min}^{(k)}, \gamma_k)}$$

where

$$g(\lambda, \gamma) = \max_{\beta > 0} \frac{2\gamma\beta - |\lambda - 1|\beta^2}{\beta^2 + \lambda^{-1}}.$$



One-level variant: conditioning analysis

One-level bounds (for $\lambda_{\max}$)

$$\lambda_{\max}^{(k)} \leq \lambda_{\max}^{(k-1)} \leq \lambda_{\max}^{(k)} + g(\lambda_{\max}^{(k)}, \gamma_k)$$

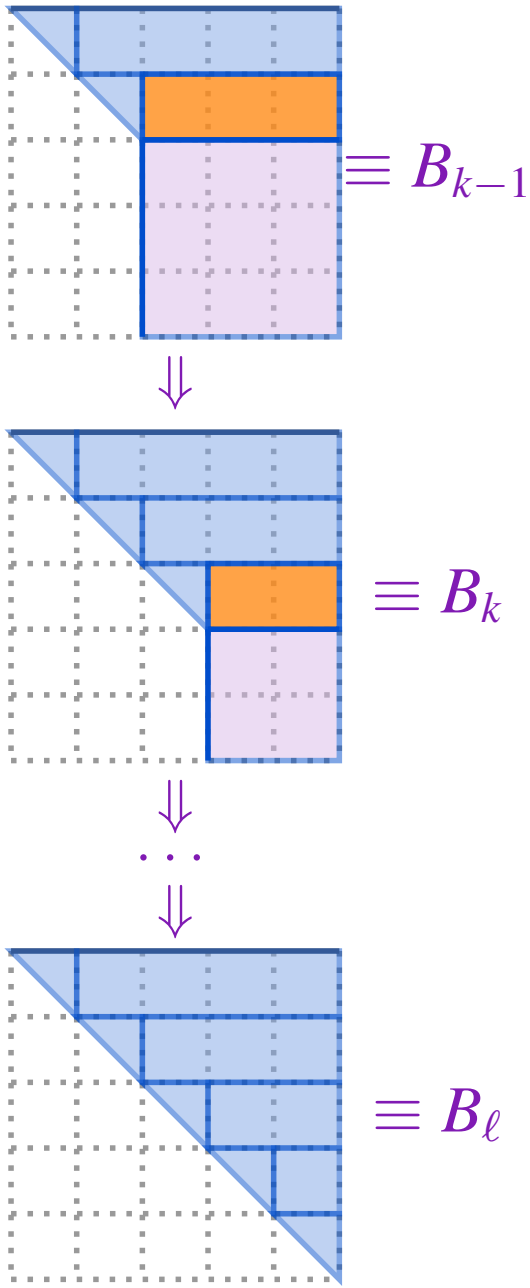
where

$$\lambda_{\max}^{(k)} = \lambda_{\max}\left(B_\ell^{-1} B_k\right), \lambda_{\max}^{(k-1)} = \lambda_{\max}\left(B_\ell^{-1} B_{k-1}\right)$$

- The estimate is sharp:

  For any $\gamma_k < 1$, $k = 1, \ldots, \ell$, there exist
  - a matrix A
  - a sequence of approximations B_k (with $B_0 = A$)

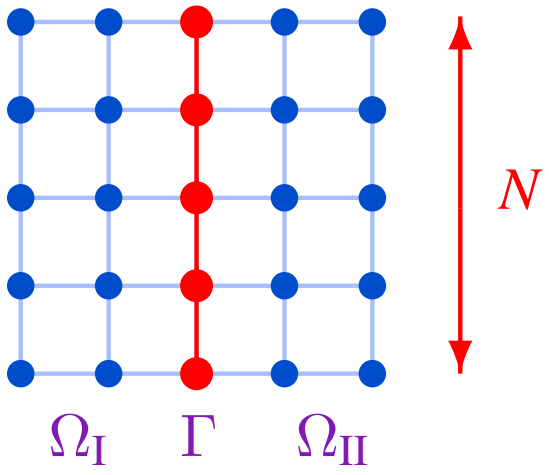
  such that the bounds for $\lambda_{\max}^{(0)}$ and $\lambda_{\min}^{(0)}$ are simultaneously reached.



Model problem: numerical experiments

Model Problem:

$$\begin{cases} -\Delta u + 10^{-4} u = f & \text{in } \Omega = (0,1)^2 \\ \frac{\partial u}{\partial n} = 0 & \text{on } \partial\Omega \end{cases}$$



Full system matrix:

$$A_\Omega = \begin{pmatrix} A_I & & A_{I,\Gamma} \\ & A_{II} & A_{II,\Gamma} \\ A_{I,\Gamma}^T & A_{II,\Gamma}^T & A_\Gamma \end{pmatrix}.$$

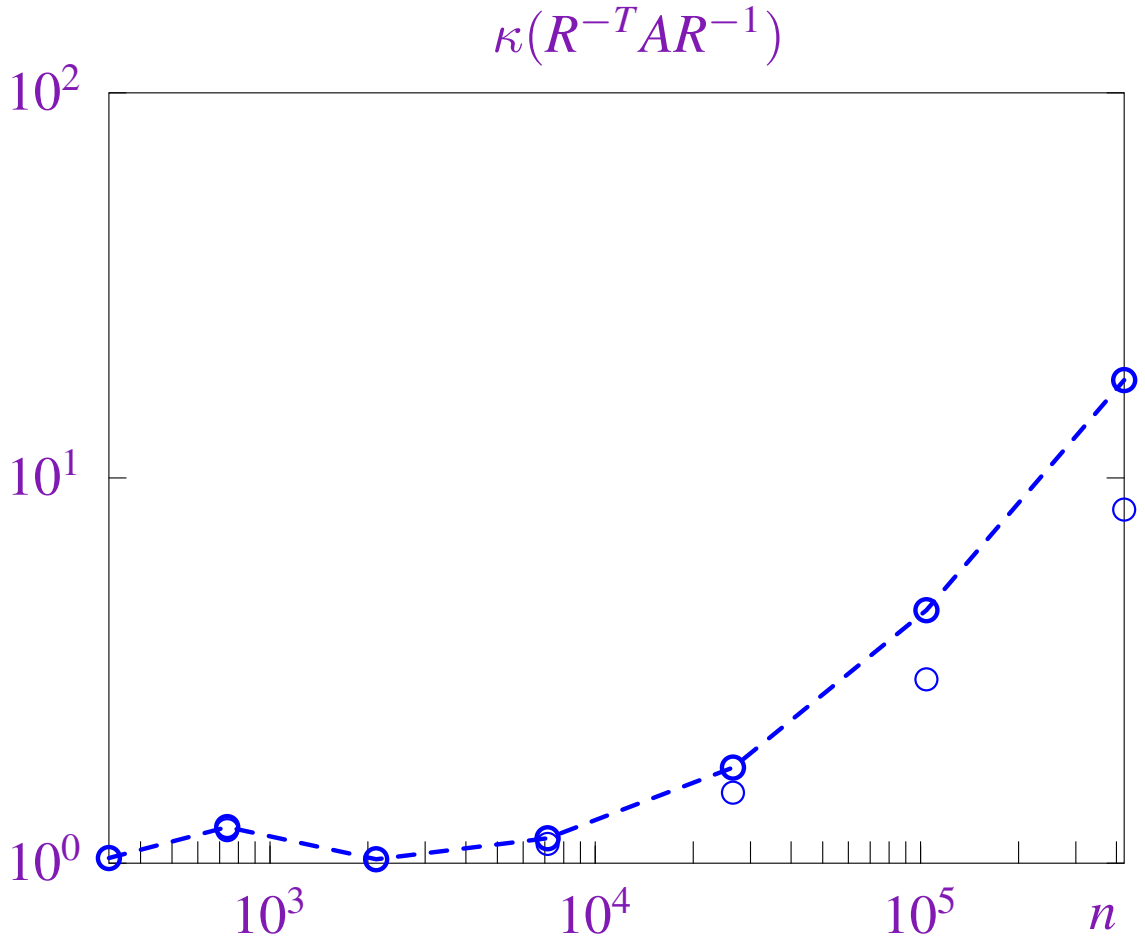
Interface system $N \times N$ matrix:

$$A = A_\Gamma - A_{I,\Gamma}^T A_I^{-1} A_{\Gamma,I} - A_{II,\Gamma}^T A_{II}^{-1} A_{\Gamma,II}$$

Algorithmic details:

- block size $= 10$
- we consider N from 20 to 650 (n from 400 to $4.3\,10^5$, ℓ from 1 to 64)
- maximal ranks remains below 4

One-level variant: numerical experiments ($tol_r = 10^{-3}$)



condition number

- one-level precond.

upper bound

- one-level

Outline

Theory ...

- One-level variant

  basic approach and underlying analysis

- **Multilevel variants**

  extensions of analysis to multi-level setting

... and practice

- Sparse solver

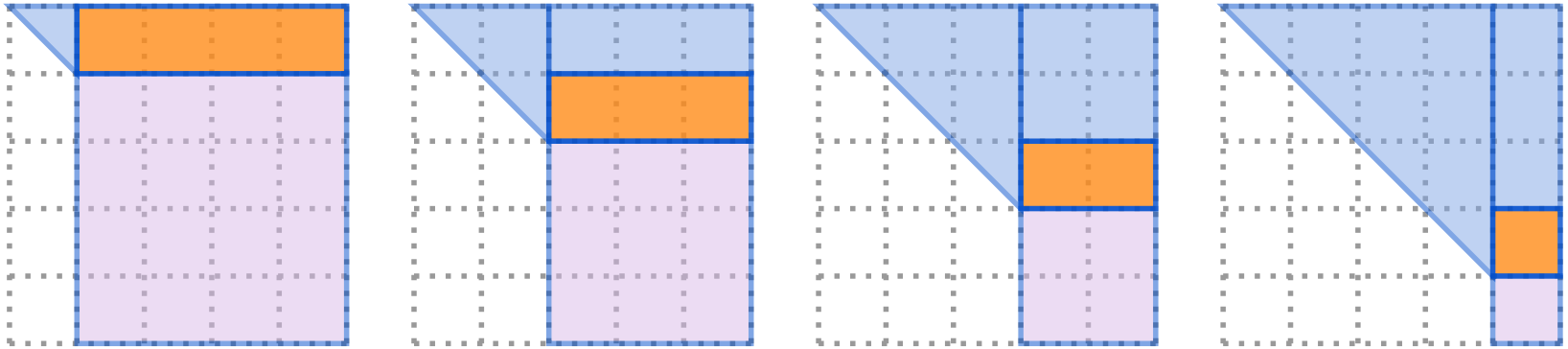
  motivation and design choices

- Numerical experiments

  and comparison with other solvers

Multilevel variants : motivation

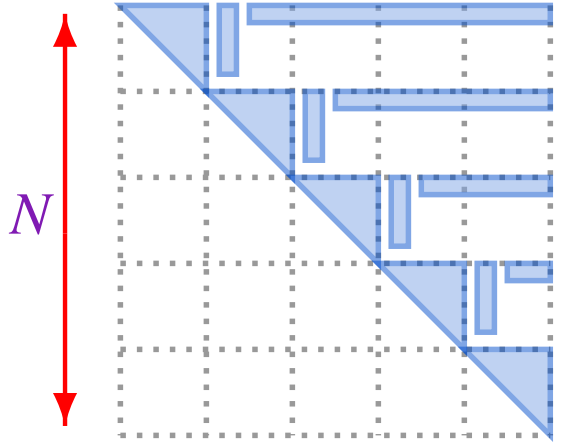
One-level (model) variant



$k = 1$ $k = 2$ $k = 3$ $k = 4$

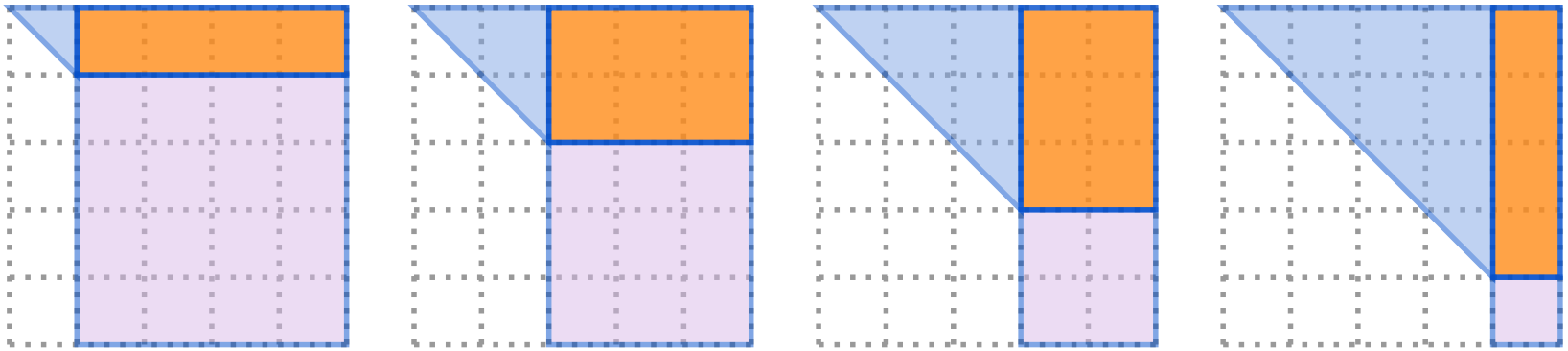
$\Downarrow$

- $\mathcal{O}(N^3)$ cost, $\mathcal{O}(N^2)$ memory (although both cost and memory are improved!)



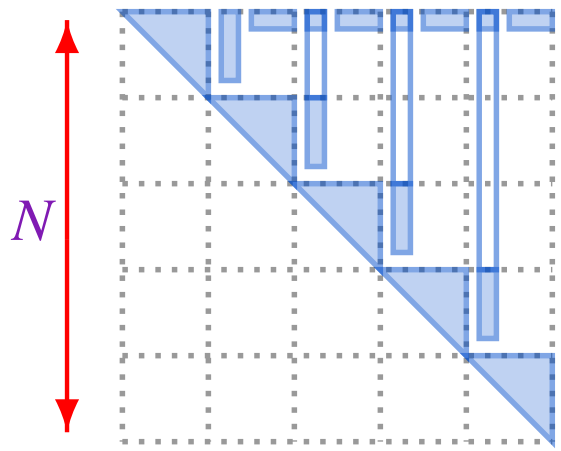
Multilevel variants : motivation

General \ Sequentially Semiseparable variant (SSS) [Gu, Li, Vassilevski 10]



- $\mathcal{O}(rN^2)$ cost, $\mathcal{O}(rN)$ memory
  r - maximal approximation rank

  $\Rightarrow N/r$ improvement!



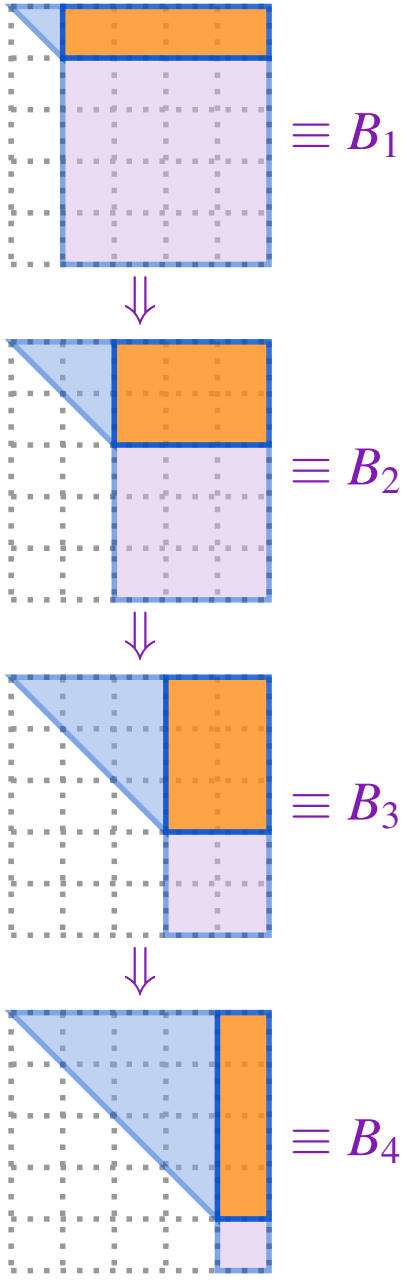
Multilevel variants : general estimate

Assuming that dropping is orthogonal one has

$$\lambda_{\max}\left(B_k^{-1}B_{k-1}\right) = 1+\gamma_k$$
$$\lambda_{\min}\left(B_k^{-1}B_{k-1}\right) = 1-\gamma_k$$

and, hence,

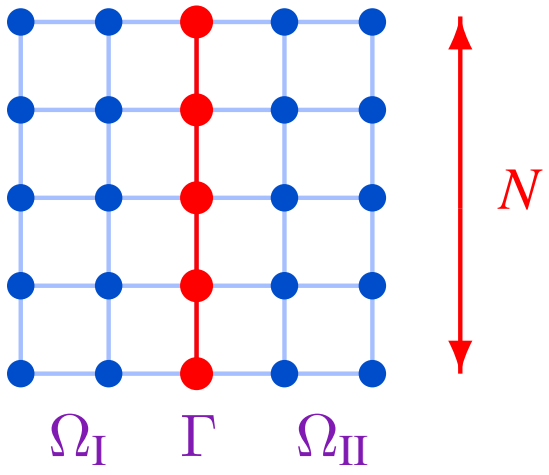
$$\kappa(R^{-T}AR^{-1}) \leq \prod_{k=1}^{\ell}\frac{1+\gamma_k}{1-\gamma_k}\,.$$



Model problem

Model Problem:

$$\begin{cases} -\Delta u + 10^{-4} u = f & \text{in } \Omega = (0,1)^2 \\ \frac{\partial u}{\partial n} = 0 & \text{on } \partial\Omega \end{cases}$$



Full system matrix:

$$A_\Omega = \begin{pmatrix} A_I & & A_{I,\Gamma} \\ & A_{II} & A_{II,\Gamma} \\ A_{I,\Gamma}^T & A_{II,\Gamma}^T & A_\Gamma \end{pmatrix}.$$

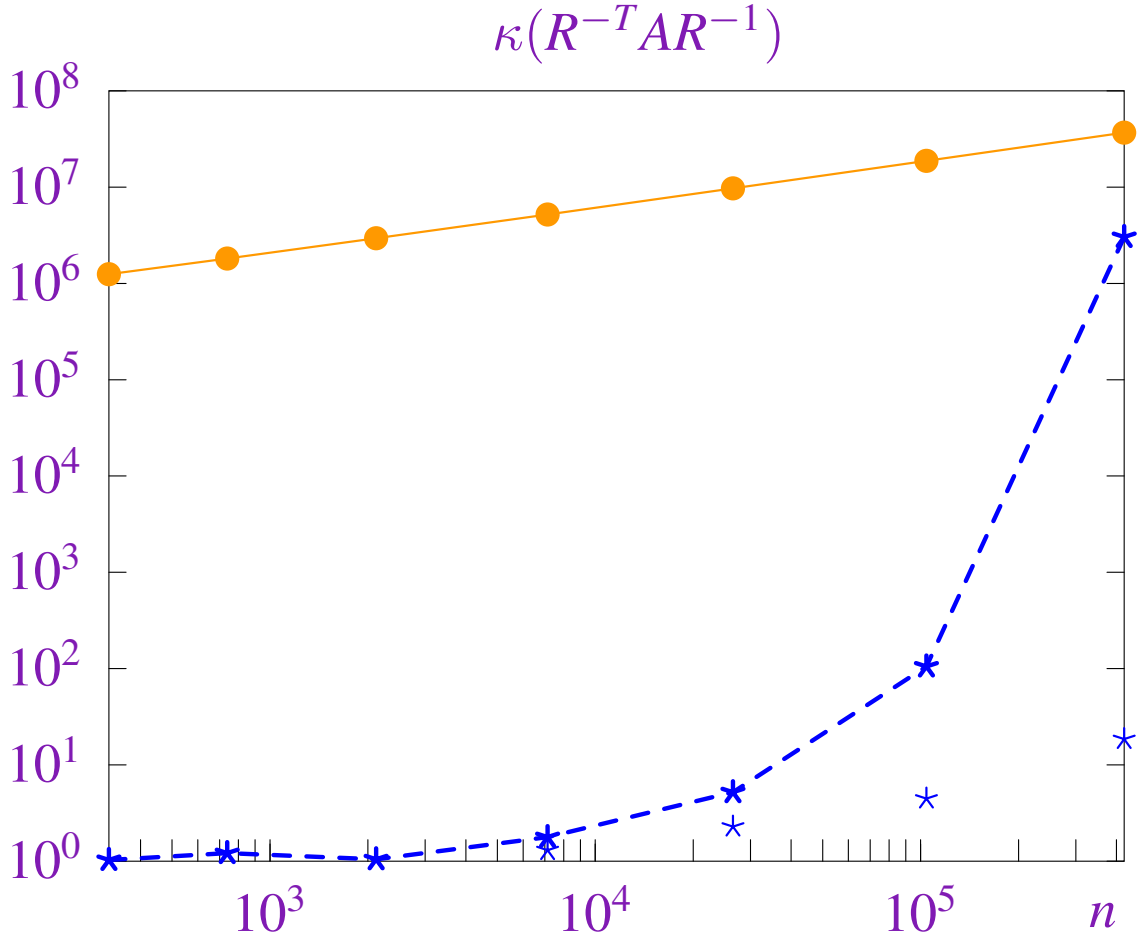
Interface system $N \times N$ matrix:

$$A = A_\Gamma - A_{I,\Gamma}^T A_I^{-1} A_{\Gamma,I} - A_{II,\Gamma}^T A_{II}^{-1} A_{\Gamma,II}$$

Algorithmic details:

- block size $= 10$
- we consider N from 20 to 650 (n from 400 to $4.3\,10^5$, ℓ from 1 to 64)
- maximal ranks remains below 4

Multilevel variants: numerical experiments



condition number

$\kappa(A)$

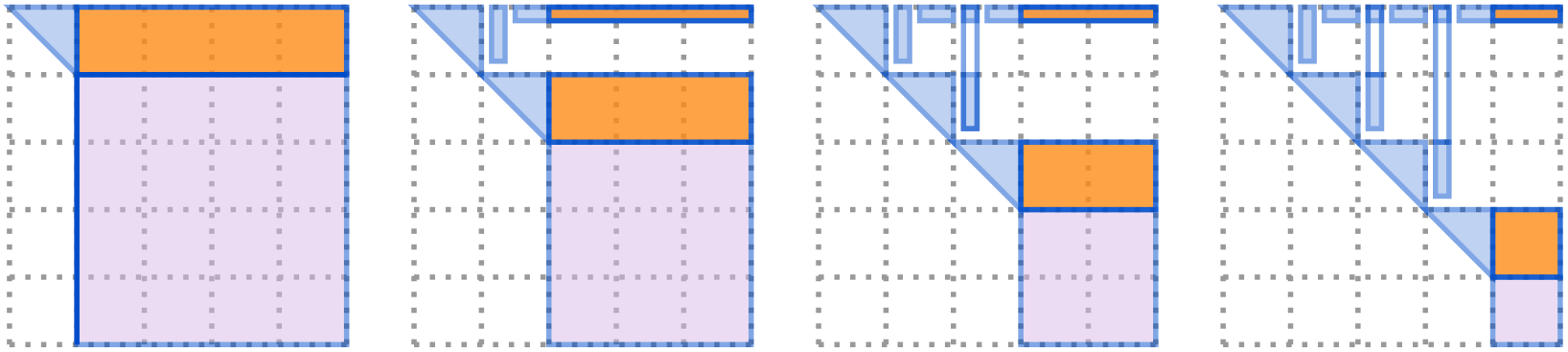
SSS precond. ($tol_r = 10^{-3}$)

upper bound

general (10^{-3})

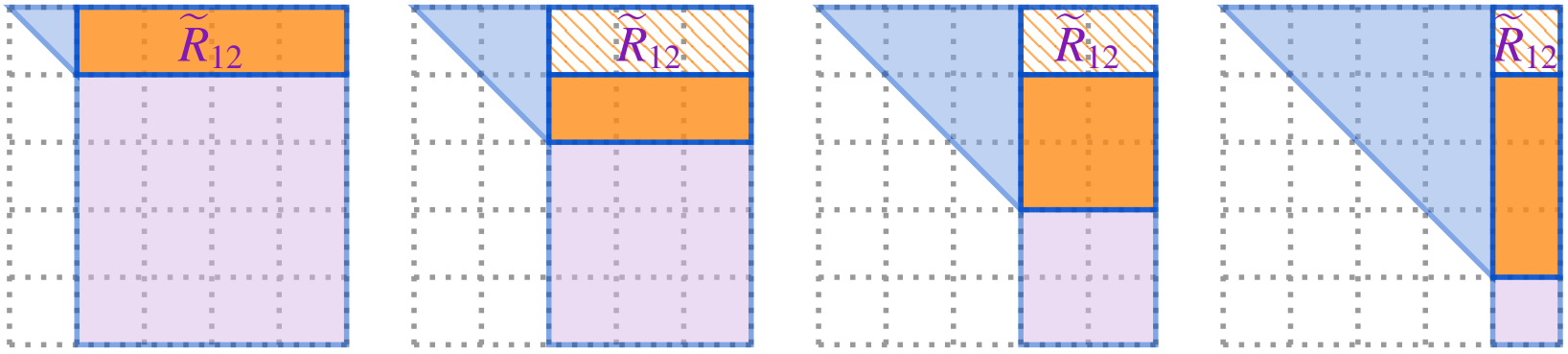
Multilevel variants: nested subspaces

Sequentially Semiseparable variant (SSS) [Gu, Li, Vassilevski 10]



$k = 1$ $k = 2$ $k = 3$ $k = 4$

Nested subspaces assumption

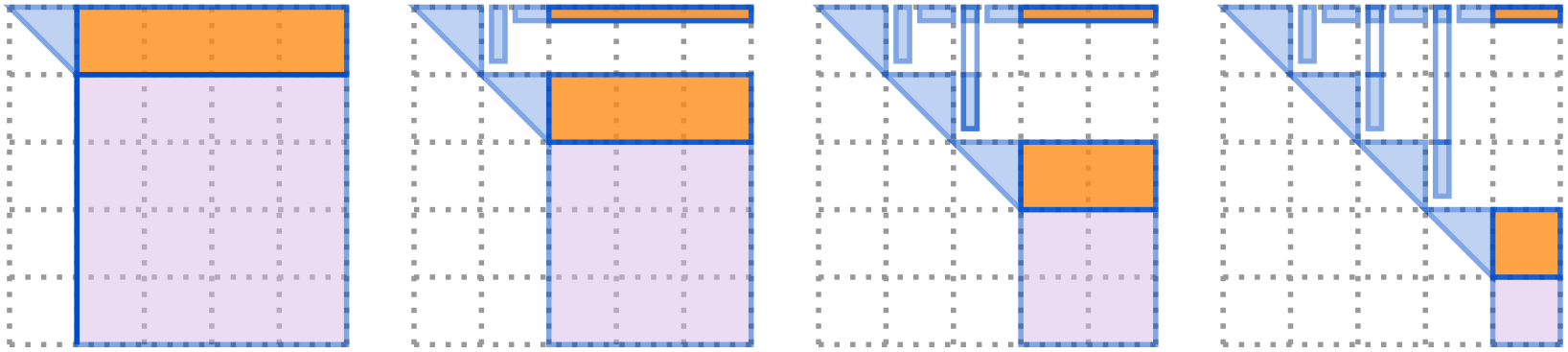


B_1 B_2 B_3 B_4

$\text{span}(\widetilde{R}_{12}) \supset \text{span}(\square) \supset \text{span}(\square) \supset \text{span}(\square)$

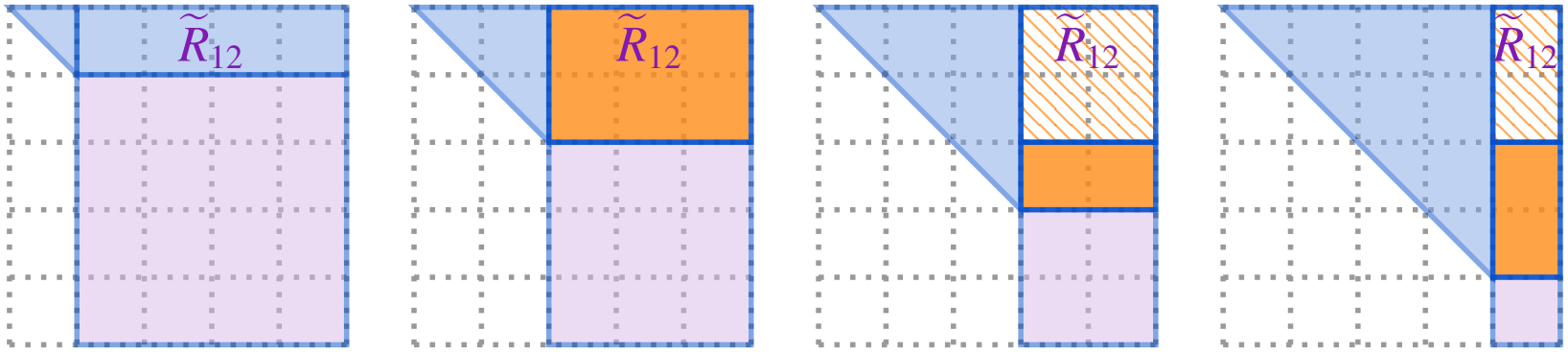
Multilevel variants: nested subspaces

Sequentially Semiseparable variant (SSS) [Gu, Li, Vassilevski 10]



$k = 1$ $k = 2$ $k = 3$ $k = 4$

Nested subspaces assumption

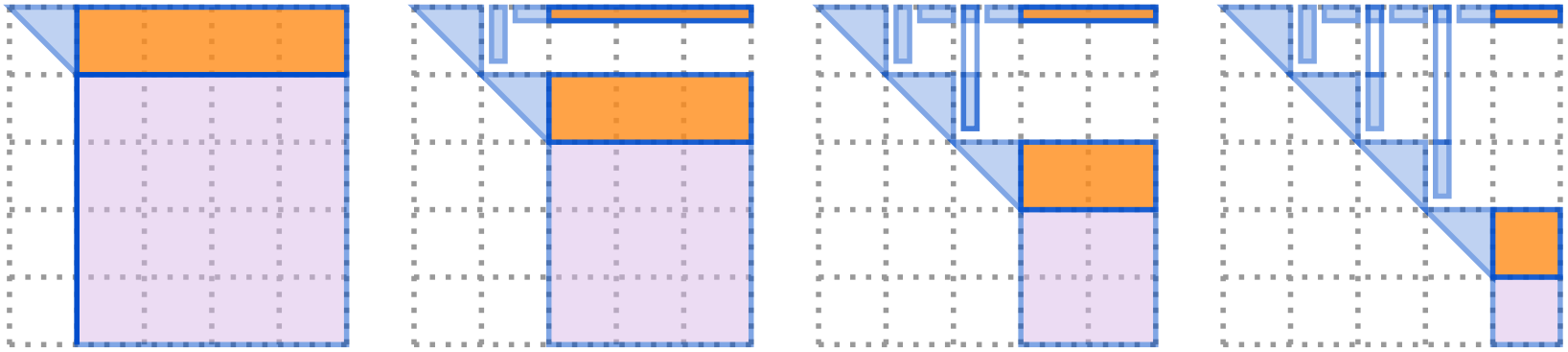


B_1 B_2 B_3 B_4

$$\text{span}(\widetilde{R}_{12}) \quad \supset \quad \text{span}(\square) \quad \supset \quad \text{span}(\square)$$

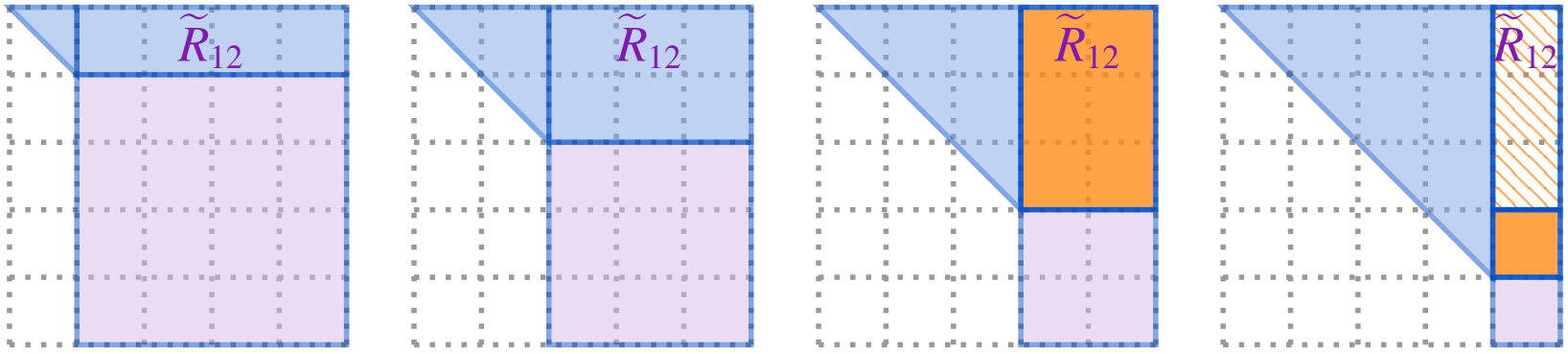
Multilevel variants: nested subspaces

Sequentially Semiseparable variant (SSS) [Gu, Li, Vassilevski 10]



$k = 1$ $k = 2$ $k = 3$ $k = 4$

Nested subspaces assumption



B_1 B_2 B_3 B_4

$$\mathrm{span}(\widetilde{R}_{12}) \quad \supset \quad \mathrm{span}(\boxed{})$$

Multilevel variants: nested subspaces

Same bounds as for one-level case!

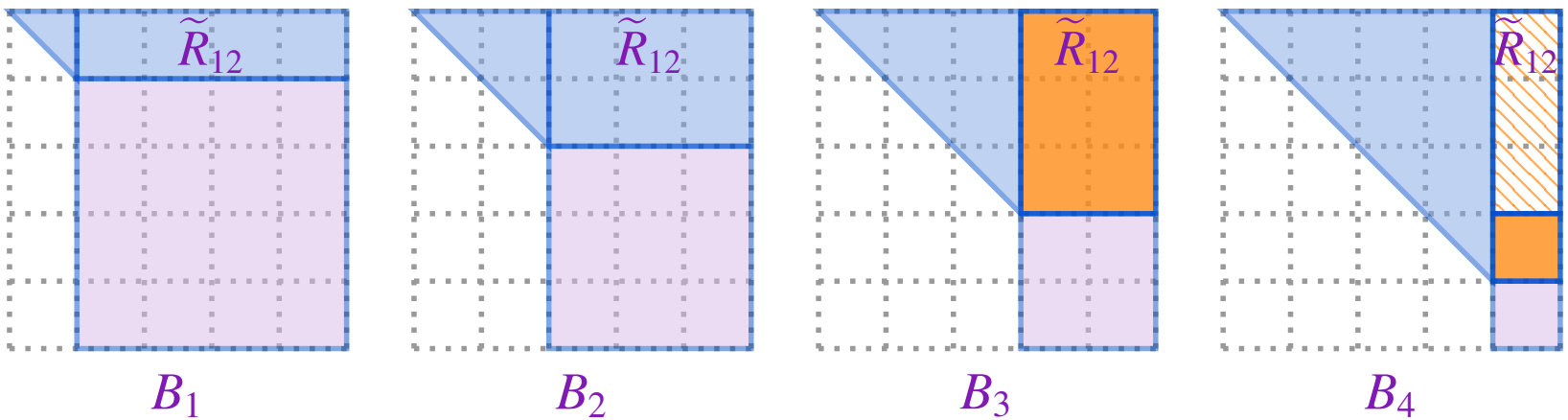
One-level bounds (for $\lambda_{\max}$)

$$\lambda_{\max}^{(k)} \leq \lambda_{\max}^{(k-1)} \leq \lambda_{\max}^{(k)} + g(\lambda_{\max}^{(k)}, \gamma_k)$$

where

$$\lambda_{\max}^{(k)} = \lambda_{\max}\left(B_\ell^{-1} B_k\right), \lambda_{\max}^{(k-1)} = \lambda_{\max}\left(B_\ell^{-1} B_{k-1}\right)$$

Nested subspaces assumption



$$\text{span}(\widetilde{R}_{12}) \supset \text{span}(\square)$$

Multilevel variants: nested subspaces

Same bounds as for one-level case!

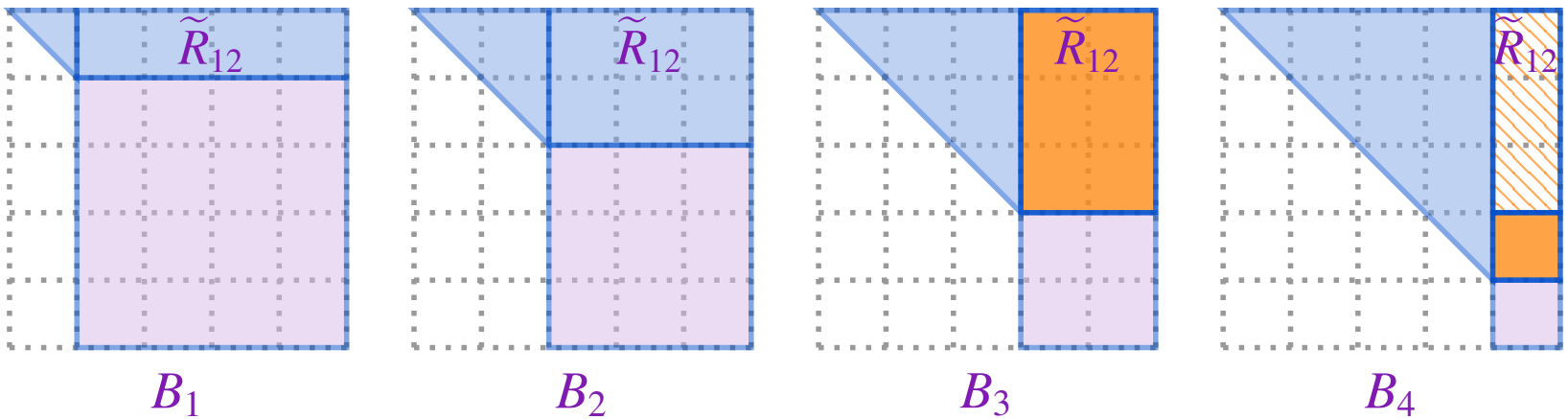
One-level bounds (for $\lambda_{\min}$)

$$\lambda_{\min}^{(k)} \geq \lambda_{\min}^{(k-1)} \geq \lambda_{\min}^{(k)} - g(\lambda_{\min}^{(k)}, \gamma_k)$$

where

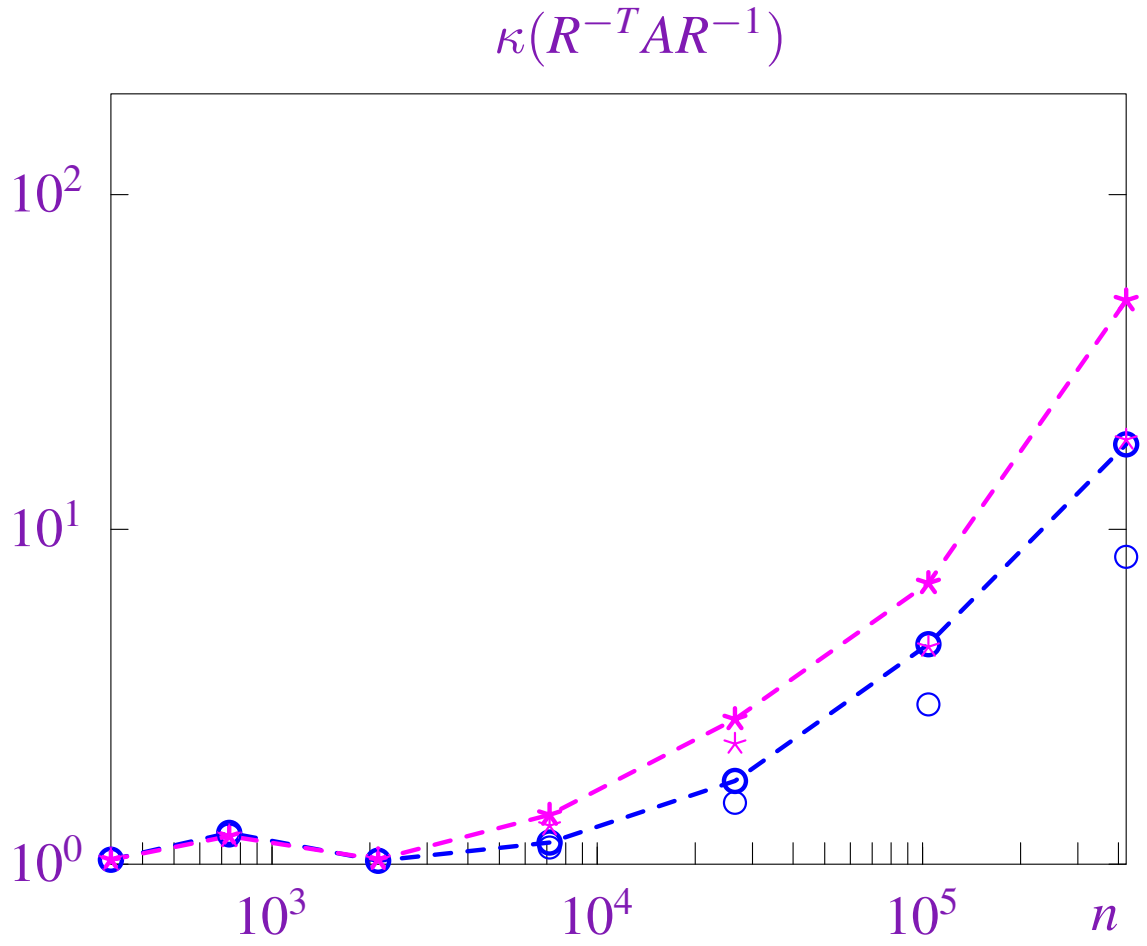
$$\lambda_{\min}^{(k)} = \lambda_{\min}\left(B_\ell^{-1} B_k\right), \lambda_{\min}^{(k-1)} = \lambda_{\min}\left(B_\ell^{-1} B_{k-1}\right)$$

Nested subspaces assumption



$$\text{span}(\widetilde{R}_{12}) \supset \text{span}(\square)$$

Multilevel variants: numerical experiments ($tol_r = 10^{-3}$)



condition number

- $*$ SSS precond.
- $\circ$ one-level precond.

upper bound

- $-\!-\!*\!-\!-$ one-level
- $-\!-\!\circ\!-\!-$ one-level

Outline

Theory ...

- One-level variant

  basic approach and underlying analysis

- Multilevel variants

  extensions of analysis to multi-level setting

... and practice

- **Sparse solver**

  motivation and design choices

- Numerical experiments

  and comparison with other solvers

Sparse solver: factorization

MAIN FEATURES:

- symmetric left-looking block factorization

  accumulate updates before block row computation

- nested dissection (ND) ordering

  induces block structure, enforces sparsity, and reduces operation count

- orthogonal low-rank approximations

  by truncated SVD or rank-revealing QR

Sparse solver: factorization

MAIN FEATURES:

- symmetric left-looking block factorization

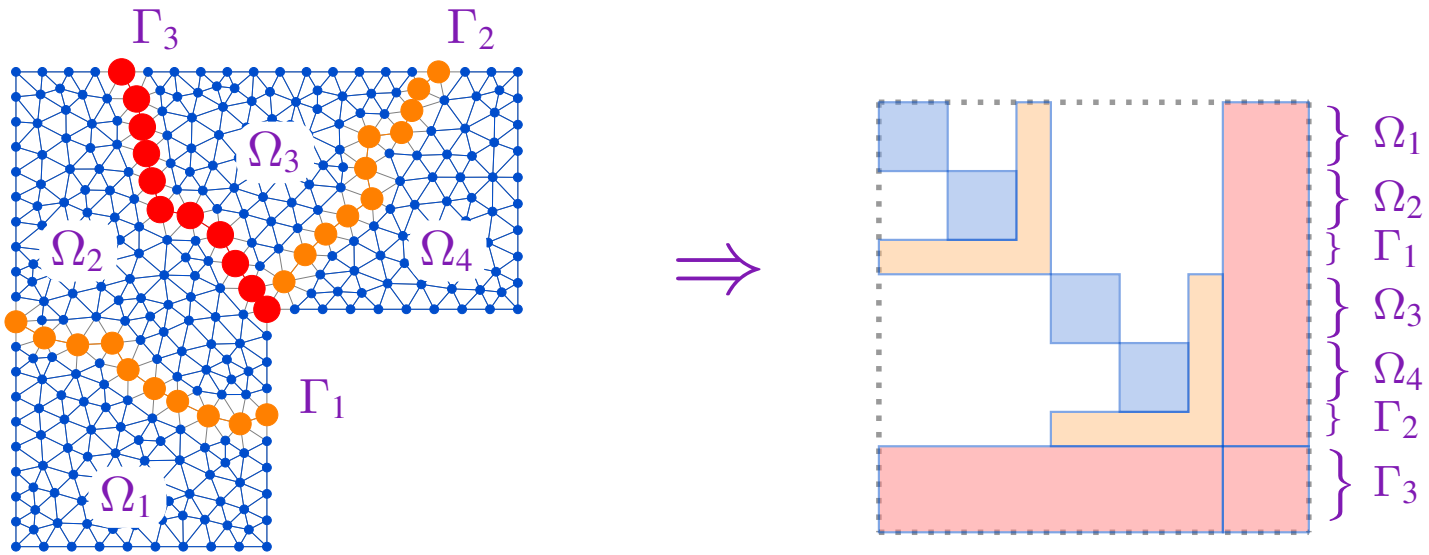
  accumulate updates before block row computation

- nested dissection (ND) ordering

  induces block structure, enforces sparsity, and reduces operation count

- orthogonal low-rank approximations

  by truncated SVD or rank-revealing QR



Sparse solver: factorization

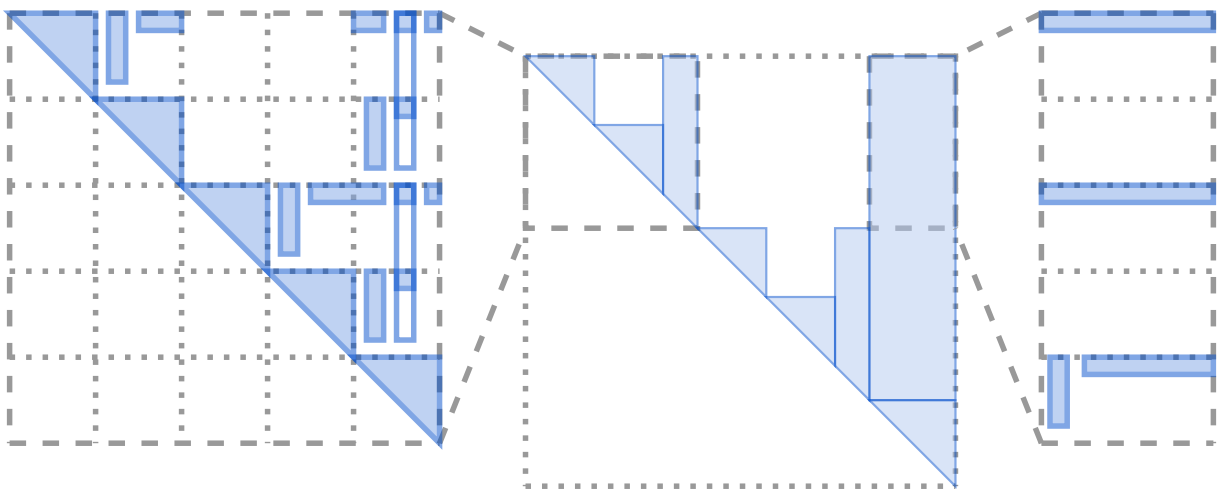
MAIN FEATURES:

- symmetric left-looking block factorization

  accumulate updates before block row computation
- nested dissection (ND) ordering

  induces block structure, enforces sparsity, and reduces operation count
- orthogonal low-rank approximations

  by truncated SVD or rank-revealing QR



Sparse solver: factorization

SPECIAL FEATURES:

- rank-revealing reordering inside ND blocks

  ensures the algebraic character of the solver

- symbolic compression

  ensures that memory usage decreases

- adaptive block size

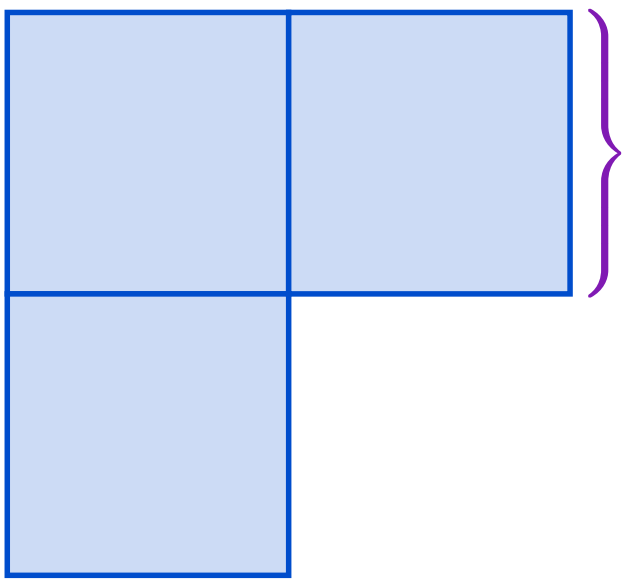
  faster than fixed block size

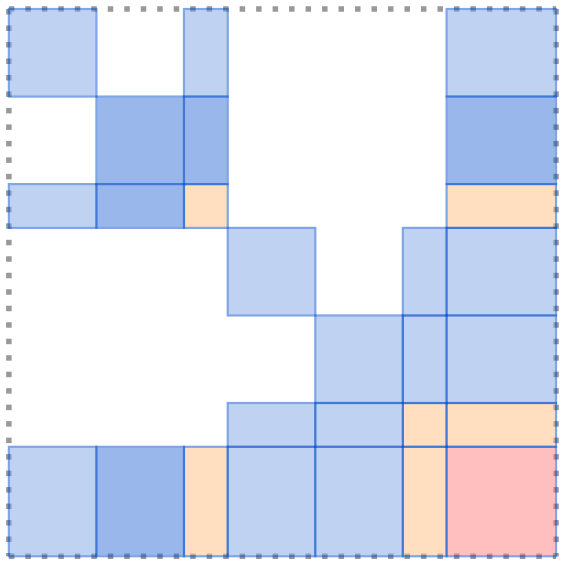
Sparse solver: revealing low-rank

Model problem analysis [Chandrasekaran et al., 10]:
rank is bounded proportionally to the number of connections between the corresp. subset and the remaining nodes of separator

Implemented: recursive edge bisection (via METIS)

2D model separators:



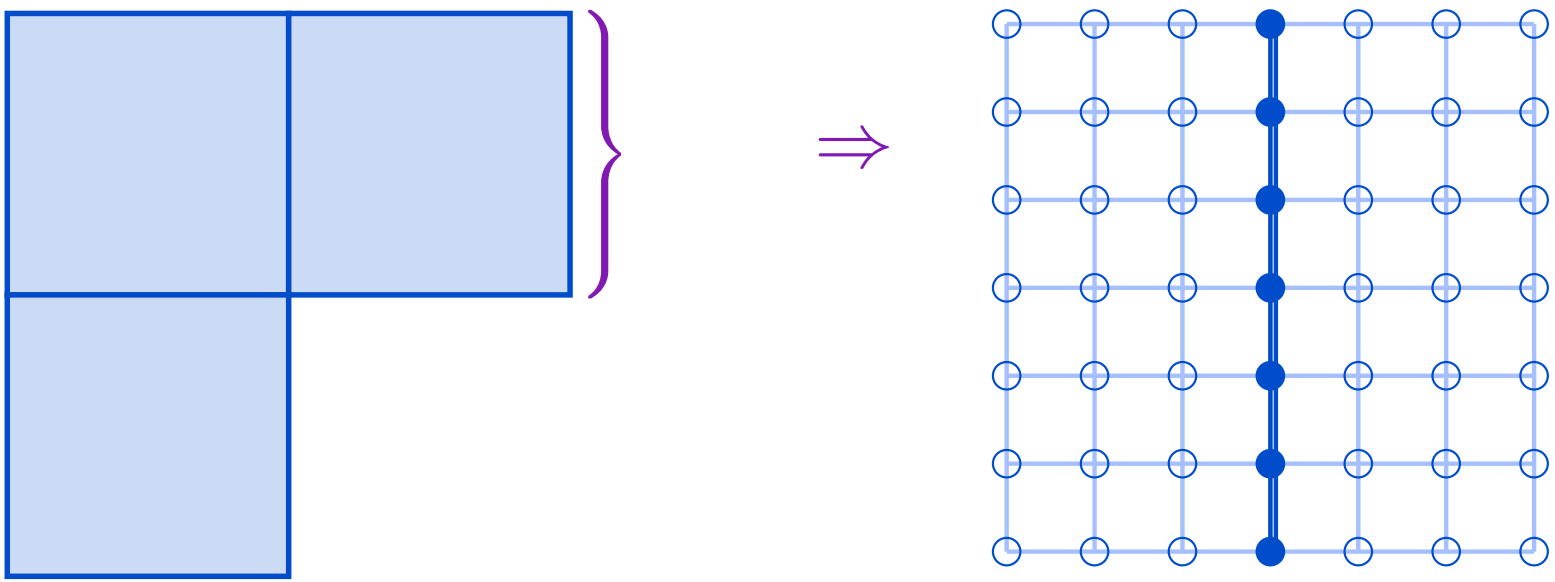


Sparse solver: revealing low-rank

Model problem analysis [Chandrasekaran et al., 10]:
rank is bounded proportionally to the number of connections between the corresp. subset and the remaining nodes of separator

Implemented: recursive edge bisection (via METIS)

2D model separators:

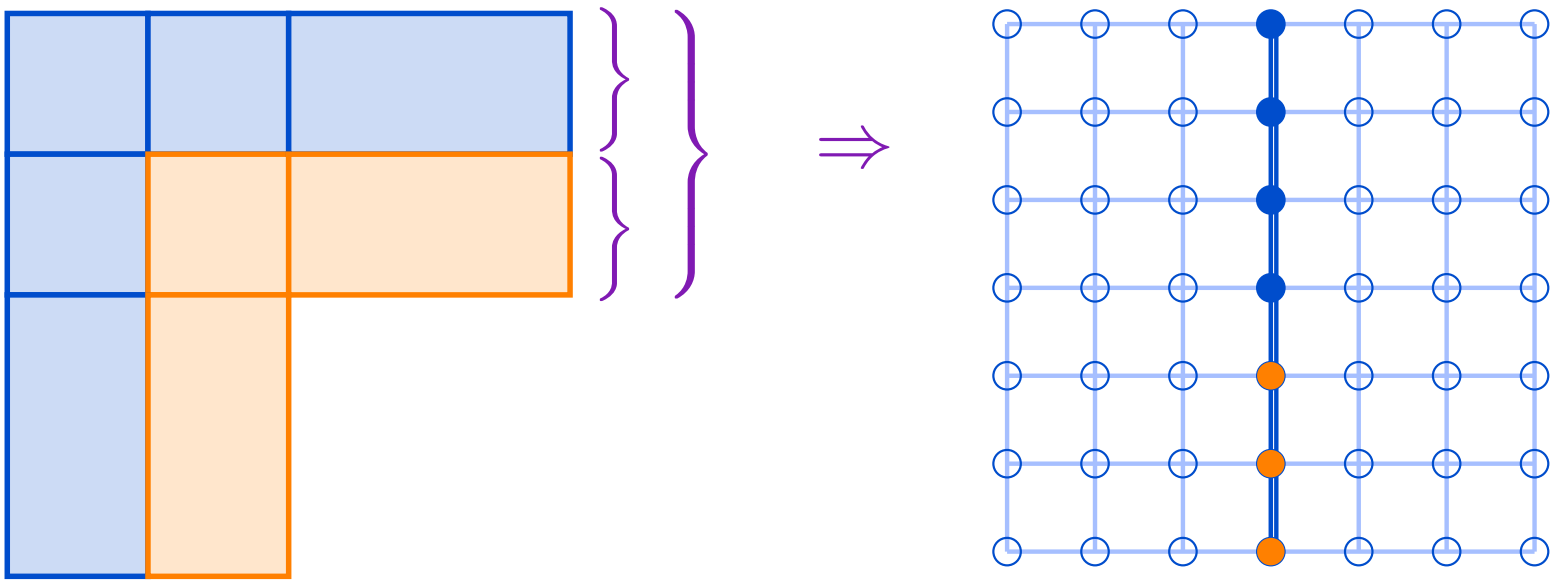


Sparse solver: revealing low-rank

Model problem analysis [Chandrasekaran et al., 10]:
rank is bounded proportionally to the number of connections between the corresp. subset and the remaining nodes of separator

Implemented: recursive edge bisection (via METIS)

2D model separators:

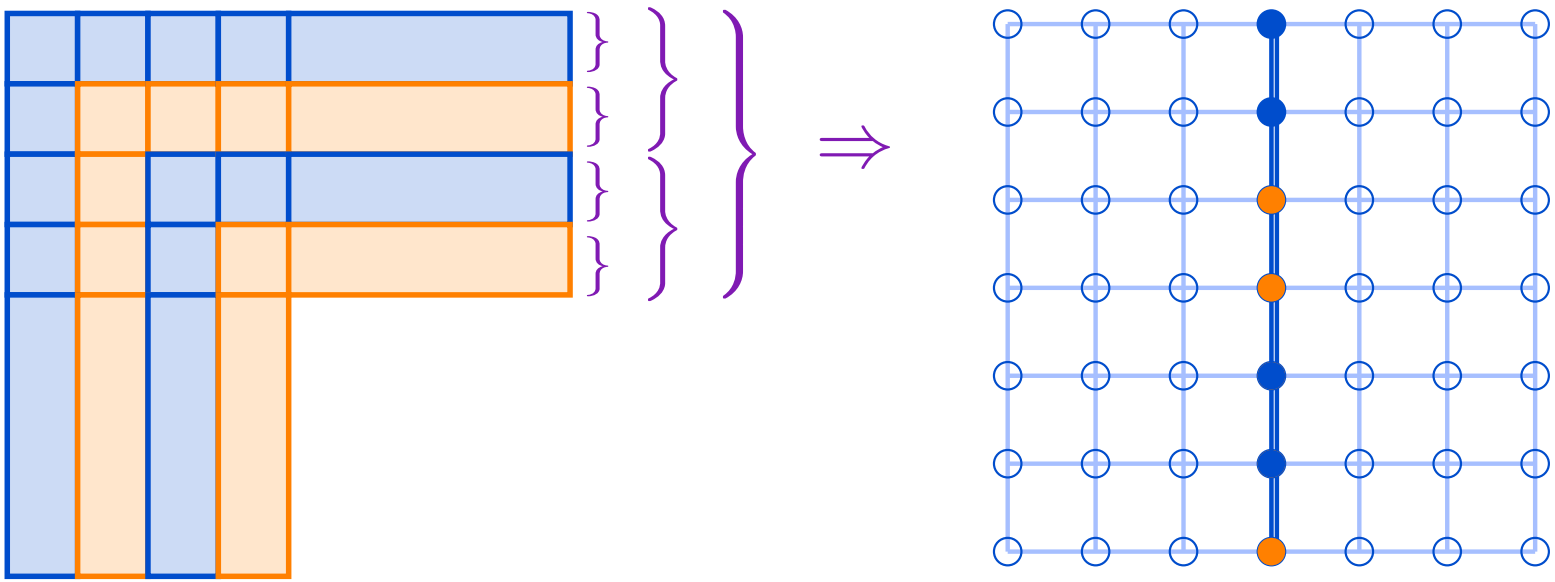


Sparse solver: revealing low-rank

Model problem analysis [Chandrasekaran et al., 10]:
rank is bounded proportionally to the number of connections between the corresp. subset and the remaining nodes of separator

Implemented: recursive edge bisection (via METIS)

2D model separators:

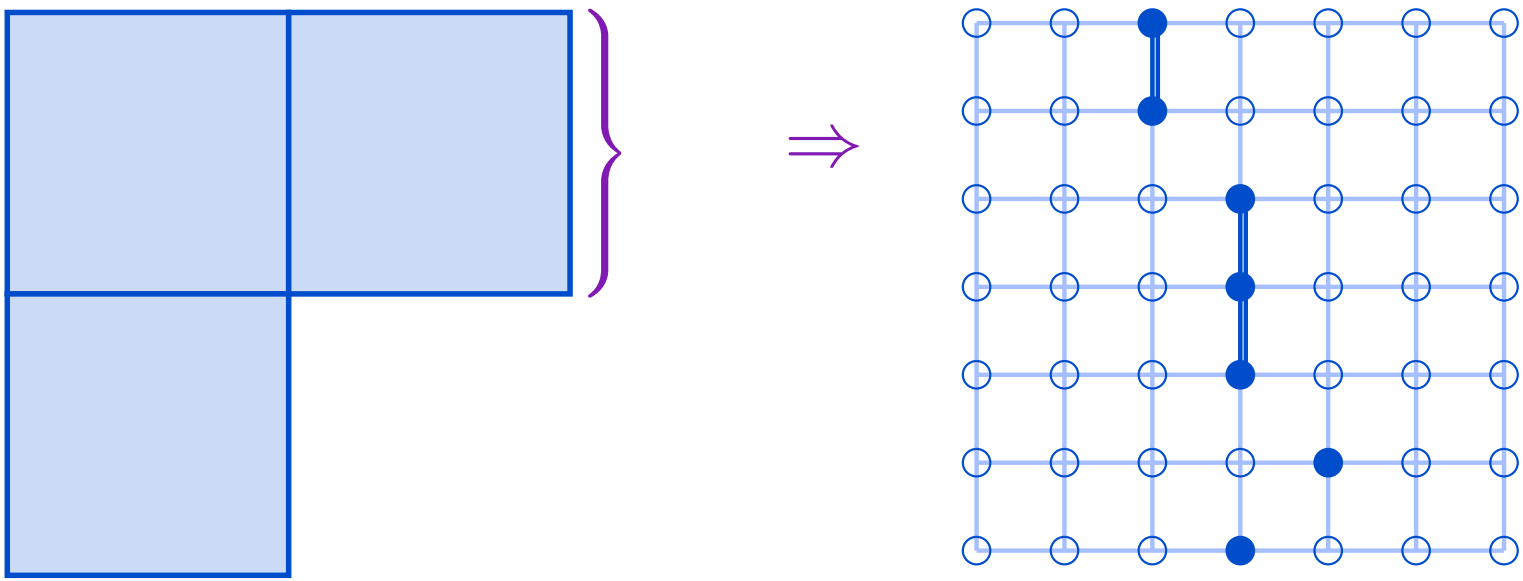


Sparse solver: revealing low-rank

Model problem analysis [Chandrasekaran et al., 10]:
rank is bounded proportionally to the number of connections between the corresp. subset and the remaining nodes of separator

Implemented: recursive edge bisection (via METIS)

2D separators from Scotch:

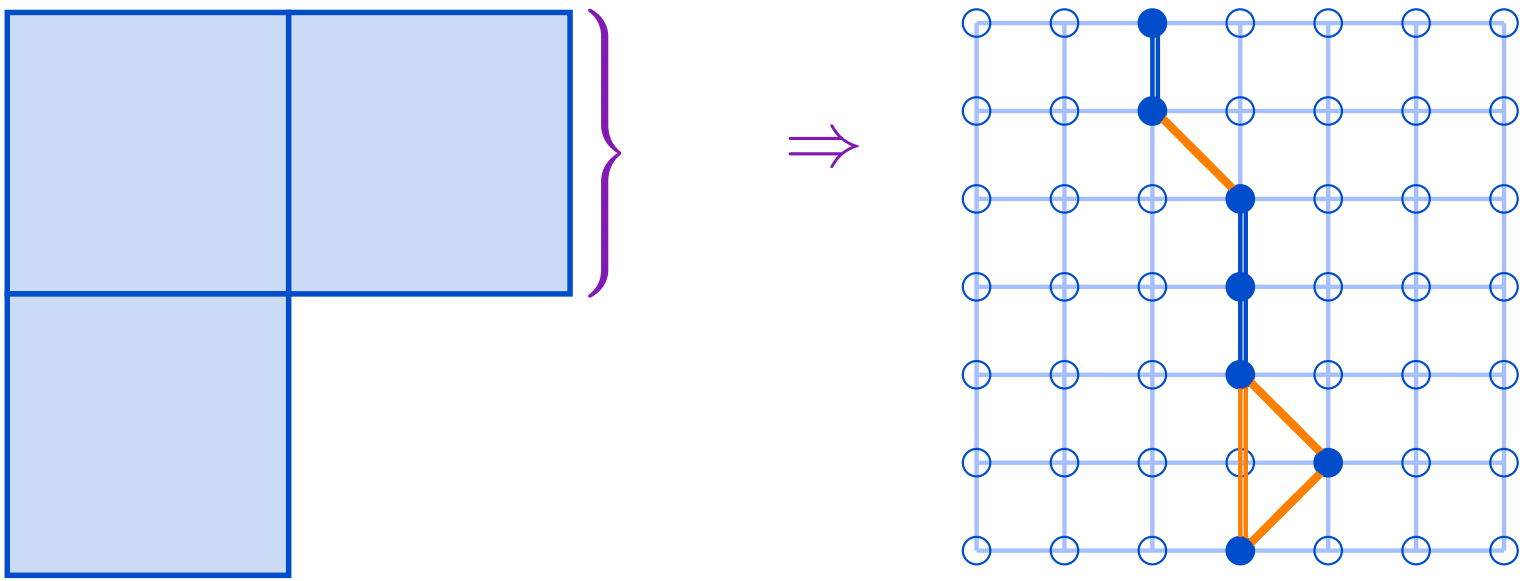


Sparse solver: revealing low-rank

Model problem analysis [Chandrasekaran et al., 10]:
rank is bounded proportionally to the number of connections between the corresp. subset and the remaining nodes of separator

Implemented: recursive edge bisection (via METIS)

2D separators from Scotch: connections of length 2

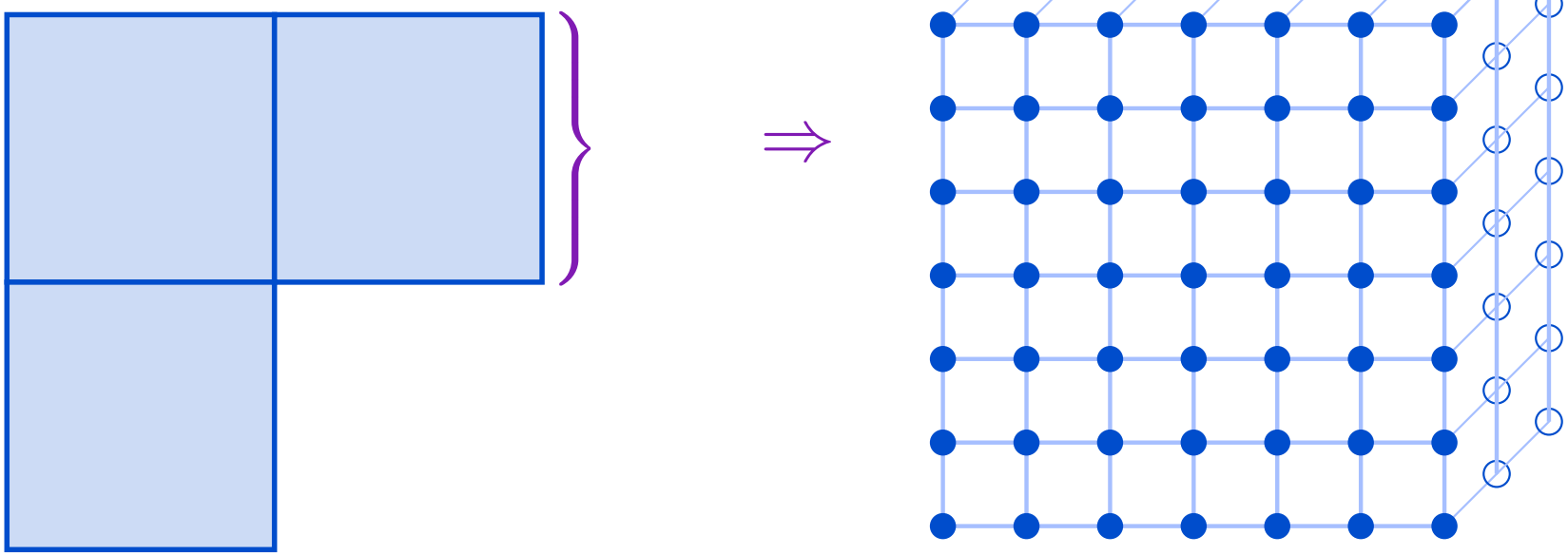


Sparse solver: revealing low-rank

Model problem analysis [Chandrasekaran et al., 10]:
rank is bounded proportionally to the number of connections between the corresp. subset and the remaining nodes of separator

Implemented: recursive edge bisection (via METIS)

3D model separators:

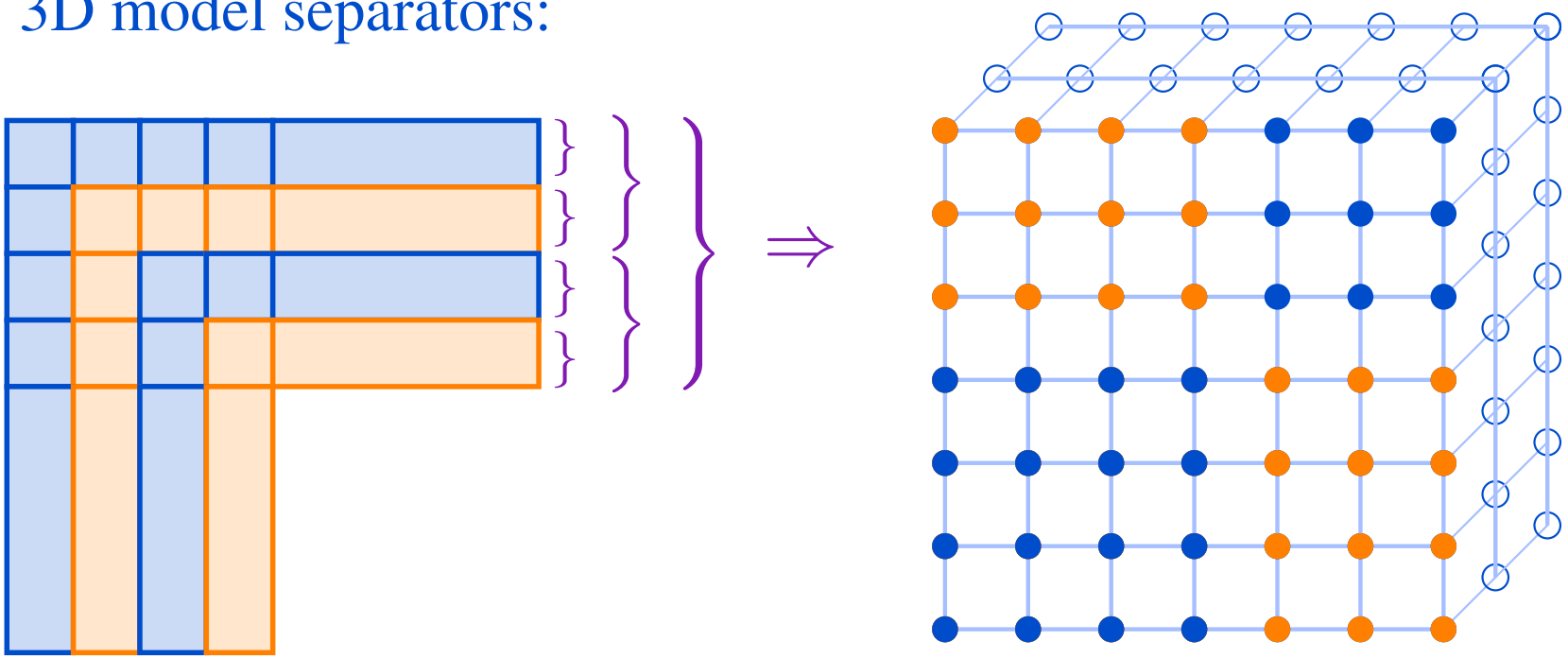


Sparse solver: revealing low-rank

Model problem analysis [Chandrasekaran et al., 10]:
rank is bounded proportionally to the number of connections between the corresp. subset and the remaining nodes of separator

Implemented: recursive edge bisection (via METIS)

3D model separators:



Sparse solver: symbolic compression

If for a low-rank approximation

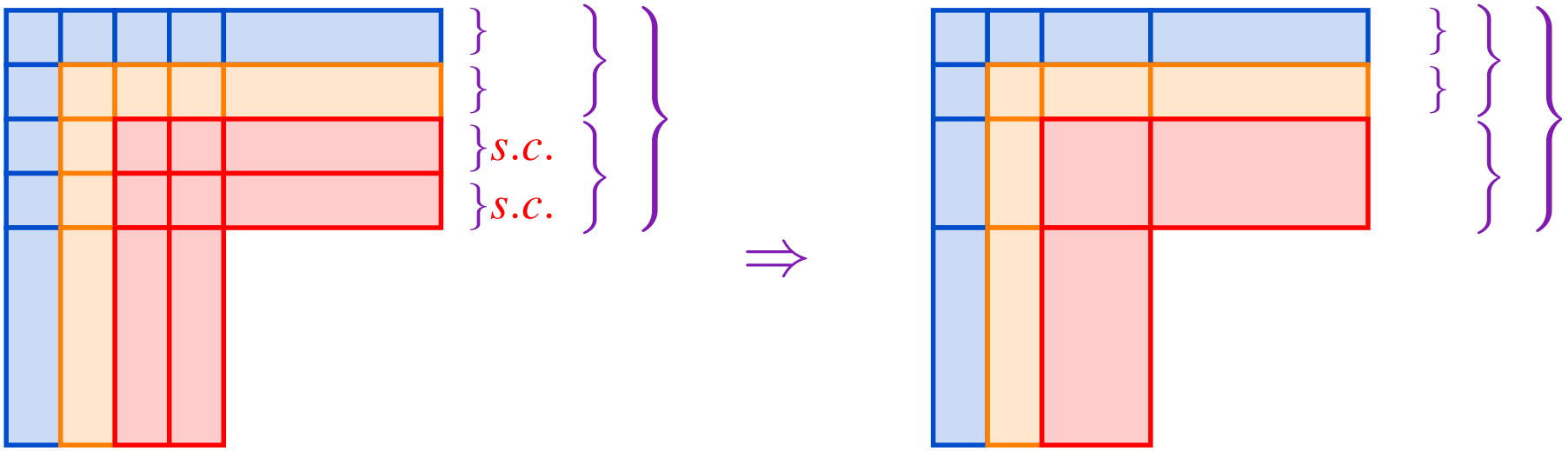
one has

then the low-rank approximation is replaced by

- This "low-rank approximation" is (trivially) orthogonal.
- No need to store I $\Rightarrow$ does not increase memory use.

Sparse solver: adaptive block size

Via symbolic compression (s.c.) , block subdivision can evolve:



We use this evolution:

- by automatically applying s.c. after few occurrences at the same level;
  (more successive s.c. occur, (exponentially) more s.c. are skipped to save computations);
- to adjust the minimal block size for the following separators
  (this also saves computations).

Outline

Theory ...

- One-level variant

  basic approach and underlying analysis

- Multilevel variants

  extensions of analysis to multi-level setting

... and practice

- Sparse solver

  motivation and design choices

- **Numerical experiments**

  and comparison with other solvers

Numerical experiments: setting

Solver parameters: the preconditioner, denoted by `SIC`, is used with

- approximation based on rank-revealing QR with column pivoting and absolute approximation threshold ;
- adaptive block size (initially block size is set to 16);
- PCG as outer iteration.

Test problems: all SPD matrices from University of Florida Sparse Matrix Collection[1] with $n > 10^5$ and random rhs

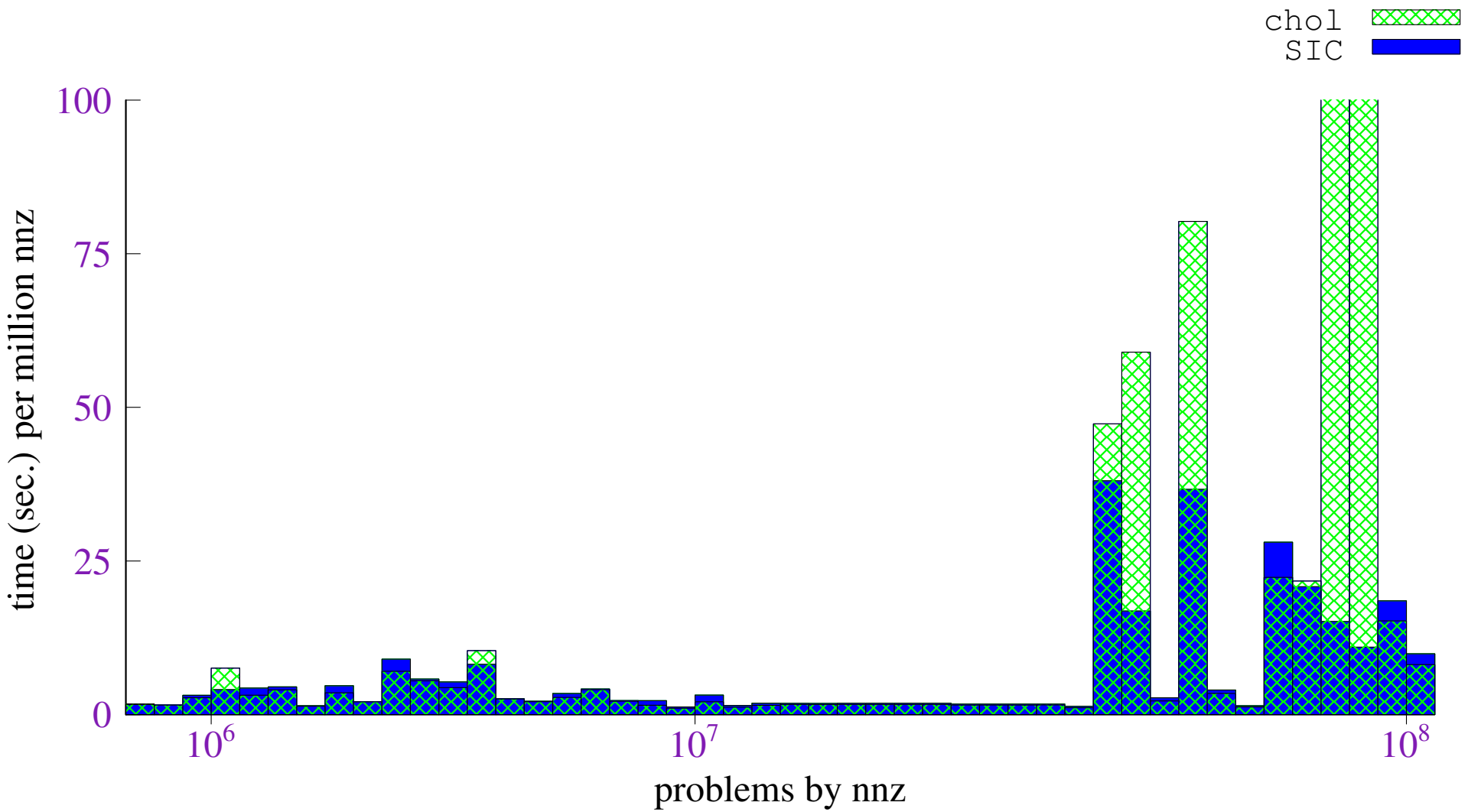
(excluded: `thermomech_TK` and `bmw7st_1`, both have $\kappa(A) \approx 10^{14}$–$10^{15}$) .

Experimental setting:

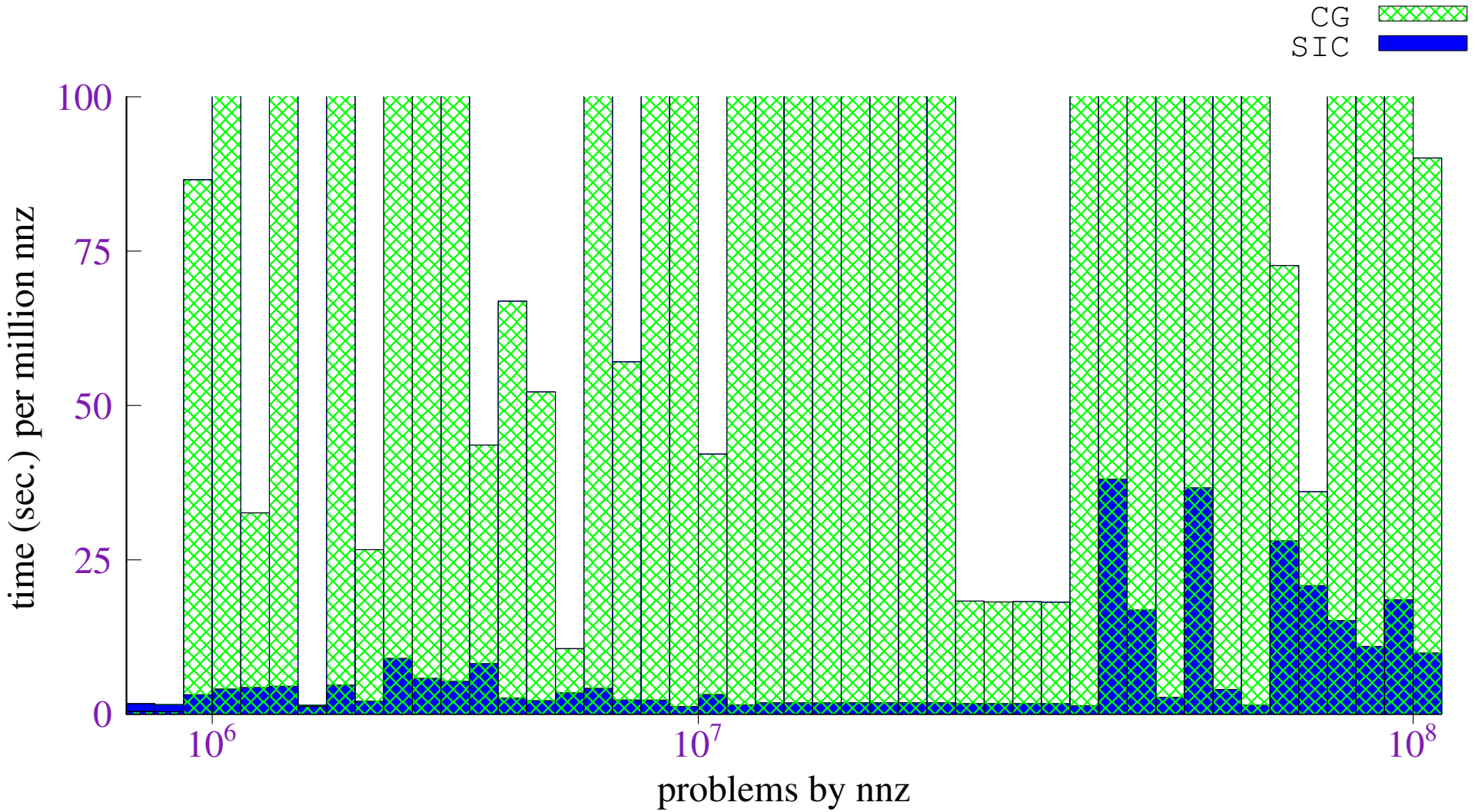
- time reported for best threshold value (chosen among $\{10^{10}, 10^9, \ldots, 10^{-10}\}$)
- stopping criterion : 10^{-6} relative residual decrease;
- hardware : Intel Xeon L5420, 2.5 GHz, 16 GB RAM.

[1] `http://www.cise.ufl.edu/research/sparse/matrices/`

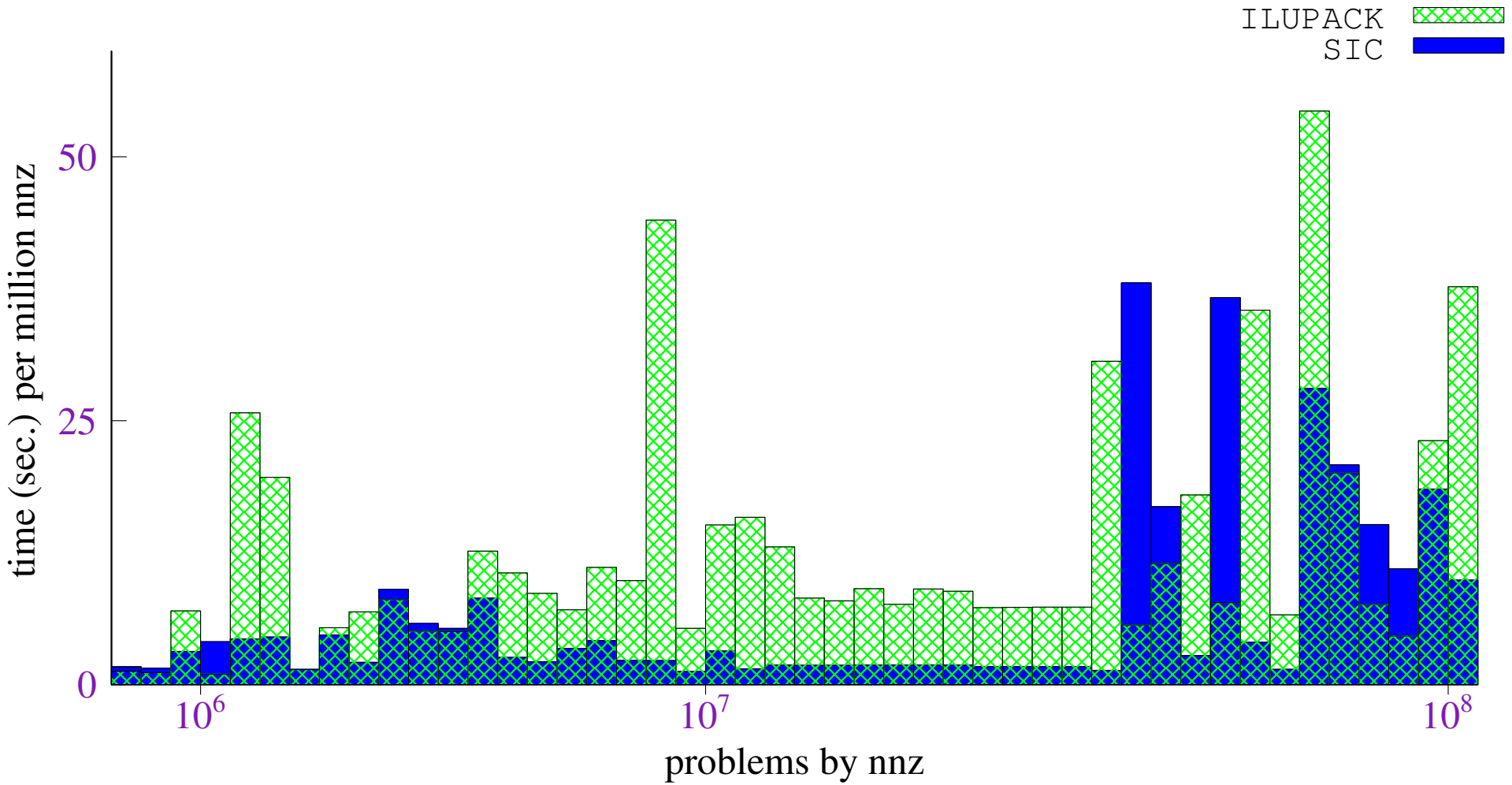
Comparison with (exact) Cholesky



Comparison with (unpreconditioned) CG



Comparison with ILUPACK



Conclusions

In theory ...

- We have presented a conditioning analysis for incomplete Cholesky factorizations preconditioner based on orthogonal dropping
- Only requirement on A: should be SPD
- The analysis relates the condition number of the preconditioned system to the individual dropping accuracies (namely, to γ_k's)
- The analysis is sharp in the one-level case
- One-level bound can be extended to the multilevel setting if an additional nested subspace assumption holds; it naturally holds for the presented preconditioner

Conclusions

... and in practice

- We have presented a preliminary implementation of incomplete Cholesky factorizations preconditioner based on orthogonal dropping
- The solver targets at sparse matrices; it uses sparsity structure to reduce the block rank during the factorization
- Preliminary numerical experiments demonstrate that the solver is competitive

Further details

- Theory:

  A. Napov, Conditioning analysis of incomplete Cholesky factorizations with orthogonal dropping, SIAM J. Matrix Anal. Appl.
  A. Napov, Conditioning analysis of incomplete Cholesky factorizations with orthogonal dropping II: nested subspaces (in preparation)

- Related approaches (for dense matrices):

  M. Gu , X. S. Li , P. Vassilevski, Direction-preserving and Schur-monotonic semiseparable approximations of symmetric positive definite matrices, SIAM J. Matrix Anal. Appl.
  J. Xia, M. Gu, Robust approximate Cholesky factorization of rank-structured symmetric positive definite matrices, SIAM J. Matrix Anal. Appl.

- Rank revealing based on sparsity:

  A. Napov, X. S. Li, An algebraic multifrontal preconditioner that exploits the low-rank property, Numer. Lin. Alg. Appl. (to appear).