Global Iterative Methods for Sparse Approximate Inverses of Symmetric Positive-Definite Matrices

Workshop on
Applied and Numerical Linear Algebra
University of Bologna

Nicolas Venkovic & Prof. Hartwig Anzt

Group of Computational Mathematics
School of Computation, Information and Technology
Technical University of Munich

October 23, 2025

Outline I

Introduction

The "What?", the "Why?" and the "How?"

- Given a sparse matrix $A \in \mathbb{R}^{n \times n}$, we seek a sparse matrix $M \in \mathbb{R}^{n \times n}$ such that $I_n - AM$ is small in some sense:

  We say that M is a right sparse approximate inverse (SPAI) of A.
- In this talk, we aim at minimizing the Frobenius residual norm:

$$\boxed{\text{Find a sparse } M \in \mathbb{R}^{n \times n} \text{ s.t. } f(M) := \|I_n - AM\|_F^2 \text{ is minimized}}$$

- SPAIs are good candidates to precondition the iterative solve of linear systems.
- Existing methods to find SPAIs include:
  - Method of Hotelling and Bodewig,
  - Steepest descent (SD) method,
  - Minimal residual (MR) method.

  To which non-zero dropping strategies are added.
- Our experiments show that, even without dropping strategy deployed, some of these methods struggle to achieve symmetric positive definite (SPD) spectra for M when A is SPD.

One-dimensional descent methods

One-dimensional descent methods

- For this problem, it is standard to consider the Frobenius inner product $(X, Y)_F := \mathrm{tr}(X^T Y)$ with induced norm $\|X\|_F := (X, X)_F^{1/2}$.
- In this work:
  - We look into descent methods as projections with carefully defined orthogonality constraints.
  - Orthogonality: $\boxed{\mathbb{R}^{m\times n} \ni X \perp \mathcal{S} \subset \mathbb{R}^{m\times n} \iff (X, Y)_F = 0 \; \forall Y \in \mathcal{S}}$.

Definition (One dimensional descent methods)

- Given $A \in \mathbb{R}^{n\times n}$ and $M_0 \in \mathbb{R}^{n\times n}$ a sequence of one-dimensional descent iterates with search directions $P_i \in \mathbb{R}^{n\times n}$ is defined by:

$$M_{i+1} \in M_i + \mathrm{span}\{P_i\} \;\text{ s.t. }\; R_{i+1} := I_n - AM_{i+1} \perp A\,\mathrm{span}\{P_i\}$$

  for $i = 0, 1, \ldots$.
- This leads to the update formula:

$$M_{i+1} = M_i + \alpha_i P_i \;\text{ where }\; \alpha_i = \frac{(R_i, AP_i)_F}{\|AP_i\|_F^2} \;\text{ for }\; i = 0, 1, \ldots.$$

Steepest descent method

- Of particular interest for the definition of a search direction in a descent method is the gradient given by:

$$\nabla_M f(M) = -2AR = -2(I_n - AM)$$

- The steepest descent (SD) method is obtained by setting the search direction opposed to the gradient direction:

$$P_i := AR_i \quad \text{for} \quad i = 0, 1, \ldots.$$

The resulting algorithm is given by:

Algorithm 1 SD(A, M_0)

1: $R_0 := I_n - AM_0$
2: $P_0 := AR_0$
3: **for** $i = 0, 1, \ldots$ **do**
4: $\quad \alpha_i := (R_i, AP_i)_F / \|AP_i\|_F^2$
5: $\quad M_{i+1} := M_i + \alpha_i P_i$
6: $\quad R_{i+1} := R_i - \alpha_i AP_i$
7: $\quad P_{i+1} := AR_{i+1}$

Minimal residual method

- Alternatively, the minimal residual (MR) method (Chow and Saad, 1998) is formed by setting the search direction to the residual:

$$P_i := R_i \quad \text{for} \quad i = 0, 1, \ldots.$$

The resulting algorithm is given by:

Algorithm 2 MR(A, M_0)

1: $R_0 := I_n - AM_0$
2: **for** $i = 0, 1, \ldots$ **do**
3: $\quad \alpha_i := (R_i, AR_i)_F / \|AR_i\|_F^2$
4: $\quad M_{i+1} := M_i + \alpha_i R_i$
5: $\quad R_{i+1} := R_i - \alpha_i AR_i$

- Both the SD and MR methods are optimal in the sense that:

$$\|I_n - AM_{i+1}\|_F = \min_{M \in M_i + \text{span}\{P_i\}} \|I_n - AM\|_F \quad \text{for} \quad i = 0, 1, \ldots.$$

Chow, E., & Saad, Y. (1998). Approximate inverse preconditioners via sparse-sparse iterations. SIAM Journal on Scientific Computing, 19(3), 995-1023.

Locally optimal variants

Local optimality

- The term "locally optimal" was coined by Andrew Knyazev (2001) to refer to the enrichment with previous search directions of the search space used for the subspace optimization of Rayleigh quotients of symmetric matrices.
- To apply local optimality to the subspace minimization of $f(M)$, we stress that the MR search directions constitute a trivial case of:

$$P_i \in \text{span}\{P_{i-1}, R_i\}. \tag{1}$$

Theorem (Local optimality)

Given $A \in \mathbb{R}^{n\times n}$ and $M_0 \in \mathbb{R}^{n\times n}$ with search directions satisfying Eq. (1) and $P_{-1} := 0_{n\times n}$, we have:

$$\min_{M\in M_i+\textit{span}\{P_{i-1},R_i\}} f(M) \leq \min_{M\in M_i+\textit{span}\{R_i\}} f(M)$$

- Similarly, local optimality can be stated with respect to SD-like iterates.

Andrew Knyazev (2001). Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. SIAM journal on scientific computing, 23(2):517–541.

Local optimal minimal residual method

- Upon deploying a locally optimal iteration in the context of the MR approach, we obtain the following locally optimal minimal residual (LOMR) method:

Definition (LOMR)

- Given $A \in \mathbb{R}^{n\times n}$ and $M_0 \in \mathbb{R}^{n\times n}$, a sequence of LOMR iterates is defined by:

$$M_{i+1} := \arg \min_{M \in M_i + \mathsf{span}\{P_{i-1}, R_i\}} \|I_n - AM\|_F \quad \text{for} \quad i = 0, 1, \ldots.$$

- The main iterates are given by:

$$M_{i+1} := M_i + \delta_i R_i + \gamma_i P_{i-1} \quad \text{for} \quad i = 0, 1, \ldots$$

where the optimal step sizes δ_i and γ_i depend on:

$$(R_i, AR_i)_F, \|AR_i\|_F^2, \|AP_{i-1}\|_F^2, (AR_i, AP_i)_F, (R_i, AP_i)_F.$$

Conjugate gradient methods

Conjugate gradient method #1

- We introduce a first conjugate gradient (CG1) method as follows:

Definition (CG1)

- Given $A \in \mathbb{R}^{n\times n}$ and $M_0 \in \mathbb{R}^{n\times n}$, a sequence of CG1 iterates is defined by:

$$M_{i+1} \in M_i + \mathsf{span}\{P_i\} \;\text{ s.t. }\; R_{i+1} := I_n - AM_{i+1} \perp \mathsf{span}\{P_i\}$$

for $i = 0, 1, \ldots,$ with search direction $P_i \in \mathbb{R}^{n\times n}$ defined as

$$P_i \in -G_i + \mathsf{span}\{P_{i-1}\} \;\text{ s.t. }\; P_i \perp A\,\mathsf{span}\{P_{i-1}\} \;\text{ for }\; i = 1, 2, \ldots$$

with $P_0 := -G_0$ and $G_i := -AR_i$, where G_i denotes the gradient direction of $f(M)$ at M_i.

Conjugate gradient method #1, con'd

- We introduce a first conjugate gradient (CG1) method as follows:

Definition (CG1, cont'd)

- The iterates of the CG1 method are given by:

$$M_{i+1} := M_i + \alpha_i P_i \quad \text{where} \quad \alpha_i := -\frac{(R_i, G_i)_F}{(P_i, AP_i)_F} \quad \text{and} \quad G_i := -AR_i$$

for $i = 0, 1, \ldots$, in which the search direction is updated as follows:

$$P_i := -G_i + \beta_i P_{i-1} \quad \text{where} \quad \beta_i := \frac{(R_i, G_i)_F}{(R_{i-1}, G_{i-1})_F} \quad \text{for} \quad i = 1, 2, \ldots.$$

Conjugate gradient method #1, con'd

- The CG1 method can be seen as a Krylov method:

Theorem (Optimality of CG1 iterates)

The CG iterates are given by

$$M_i \in M_0 + \mathcal{K}_i(A^2, G_0) \;\; \textit{s.t.} \;\; R_i := I_n - AM_i \perp \mathcal{K}(A^2, G_0) \;\; \textit{for} \;\; i = 1, 2, \ldots$$

and the right-approximate inverse M_i^{-1} is optimal in the sense that

$$\|A^{-1} - M_i\|_{F,A} = \min_{M \in M_0 + \mathcal{K}_i(A^2, G_0)} \|A^{-1} - M\|_{F,A} \;\; \textit{for} \;\; i = 1, 2, \ldots.$$

That is, the CG iterate minimizes the Frobenius A-norm of the error $E := A^{-1} - M$ over the affine Krylov subspace of A^2 generated by the initial gradient $G_0 := -AR_0$.

Conjugate gradient method #2

- Lastly, we implement a second conjugate gradient (CG2) method which corresponds to the standard conjugate gradient algorithm with matrix iterates and Frobenius inner products:

Definition (CG2)

- Given $A \in \mathbb{R}^{n\times n}$ and $M_0 \in \mathbb{R}^{n\times n}$, a sequence of CG2 iterates is defined by:

$$M_{i+1} \in M_i + \mathsf{span}\{P_i\} \;\; \text{s.t.} \;\; R_{i+1} := I_n - AM_{i+1} \perp \mathsf{span}\{P_i\}$$

for $i = 0, 1, \ldots,$ with search direction $P_i \in \mathbb{R}^{n\times n}$ defined as

$$P_i \in R_i + \mathsf{span}\{P_{i-1}\} \;\; \text{s.t.} \;\; P_i \perp A\,\mathsf{span}\{P_{i-1}\} \;\; \text{for} \;\; i = 1, 2, \ldots$$

with $P_0 := R_0$.

Non-zero dropping strategies

Non-zero dropping strategy — main iterate

- Non-zero values are dropped in the main iterate, $M \mapsto \widehat{M}$, to try and decrease the residual norm achieved after dropping, i.e., $\|\widehat{R}\|_F < \|R\|_F$:
  1. Symmetrize M: $\widehat{M} := \left(M + M^T\right)/2$.
  2. Drop insignificant non-diagonal components:
     $\widehat{m}_{k\ell} := 0 \; \forall \; |\widehat{m}_{k\ell}| < u$ with $k \neq \ell$, where u denotes the unit round-off.
  3. If $|\mathcal{NNZ}(\widehat{M})| > nnz$:

For $(k,\ell) \in \mathcal{D} = \underset{\substack{\mathcal{S} \subset \mathcal{NNZ}(\widehat{M}) \setminus \{(k,k),\, k=1,\ldots,n\} \\ |\mathcal{NNZ}(\widehat{M})| - |\mathcal{S}| = nnz}}{\arg} \min \left\{ \sum_{(k,\ell)\in\mathcal{S}} \widehat{m}_{k\ell}^2 \|Ae_k\|_2^2 \right.$

$$\left. +2 \sum_{(k,\ell)\in\mathcal{S}} \widehat{m}_{k\ell} \cdot (AR)_{k\ell} \right\},$$

set $\widehat{m}_{k\ell} := 0$.

- The matrix AR is already formed at each iteration.
- The dot products $\|Ae_1\|_2^2, \ldots, \|Ae_n\|_2^2$ need be computed only once, at the start of the algorithm.

Non-zero dropping strategy — search direction

- Similarly, we drop non-zero values in the search direction iterate, $P \mapsto \widehat{P}$, although without sophistication:

1. If $|\mathcal{NNZ}(P)| > nnz$:

$$\text{For } (k,\ell) \in \mathcal{D}_P = \underset{\substack{\mathcal{S}\subset\mathcal{NNZ}(P)\\ |\mathcal{NNZ}(P)|-|\mathcal{S}|=m}}{\arg} \min \left\{ \sum_{(k,\ell)\in\mathcal{S}} |p_{k\ell}| \right\},$$

   set $\widehat{p}_{k\ell} := 0$, otherwise $\widehat{p}_{k\ell} := p_{k\ell}$.

2. Otherwise, $\widehat{P} := P$.

Summary of methods

Summary of computing cost per iteration

- The main computing effort of the methods presented is decomposed into:
  - Multiplication between sparse matrices (SpGEMM),
  - Frobenius inner products of sparse matrices.
- In detail, the operation count per iteration of the methods is as follows:

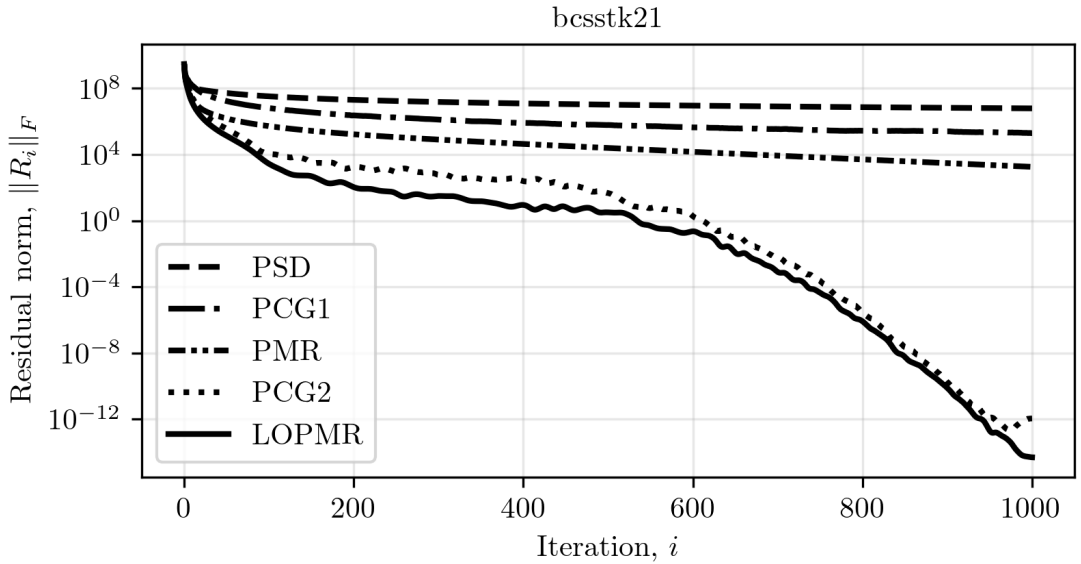
Method	Operation count per iteration
MR	1 SpGEMM + 2 sparse inner products
SD	2 SpGEMMs + 2 sparse inner products
LOMR	2 SpGEMMs + 5 sparse inner products
CG1	2 SpGEMMs + 4 sparse inner products
CG2	1 SpGEMM + 4 sparse inner products

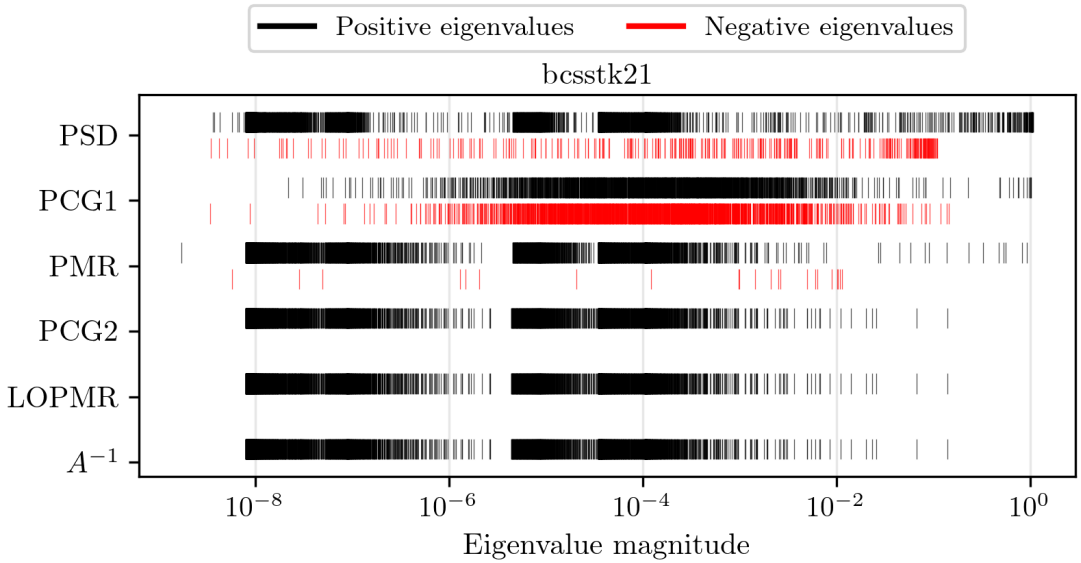
- A substantial added cost is that of dropping zeros and changing the data structures of the sparse iterates.

Numerical experiments

Dropping-free experiments

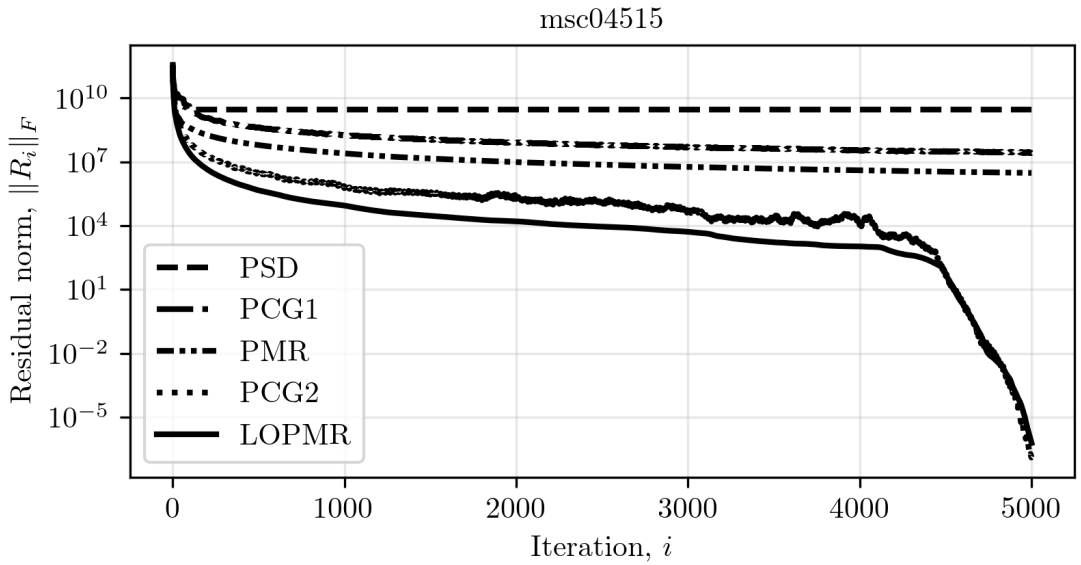
- bcsstk21 matrix (from SuiteSparse Collection), with Jacobi preconditioner:

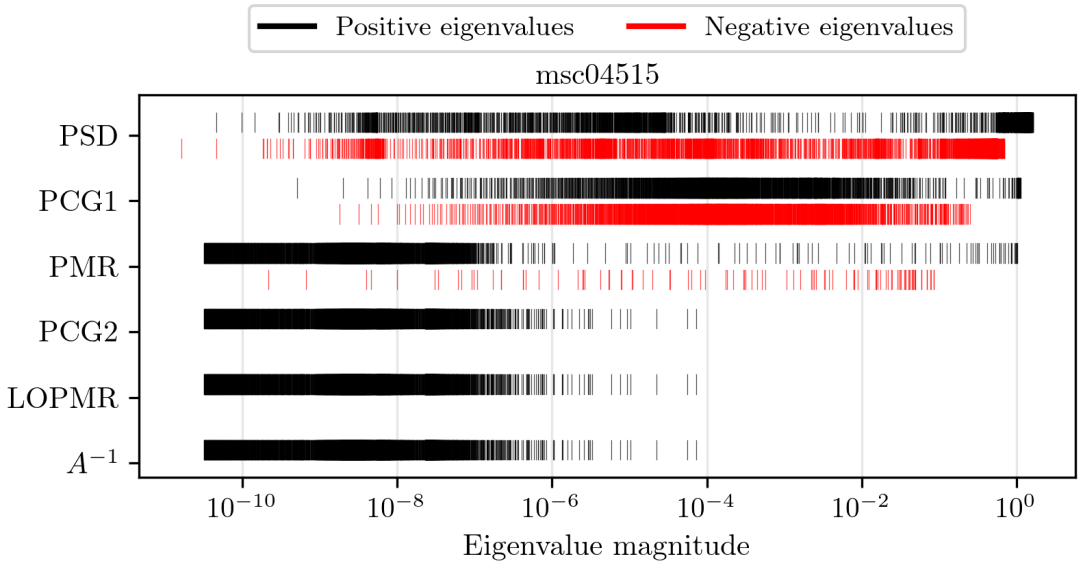




Dropping-free experiments, cont'd

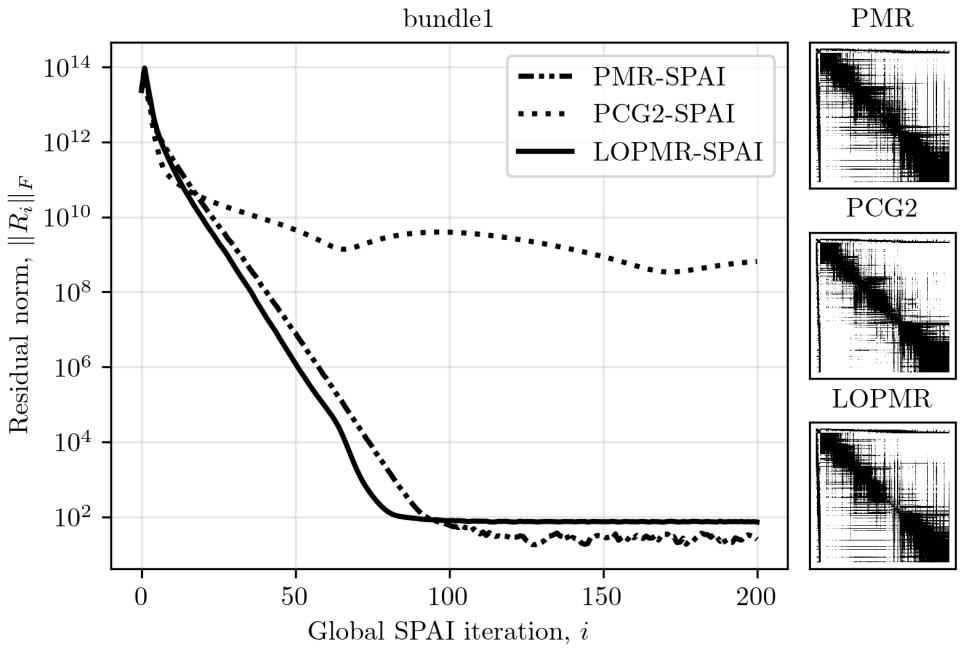
- msc04515 matrix (from SuiteSparse Collection), with Jacobi preconditioner:

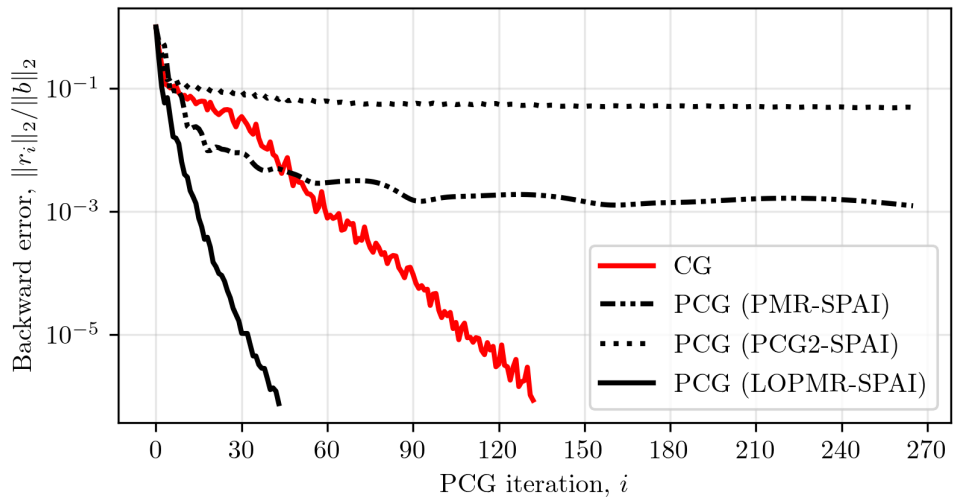




With dropping experiments

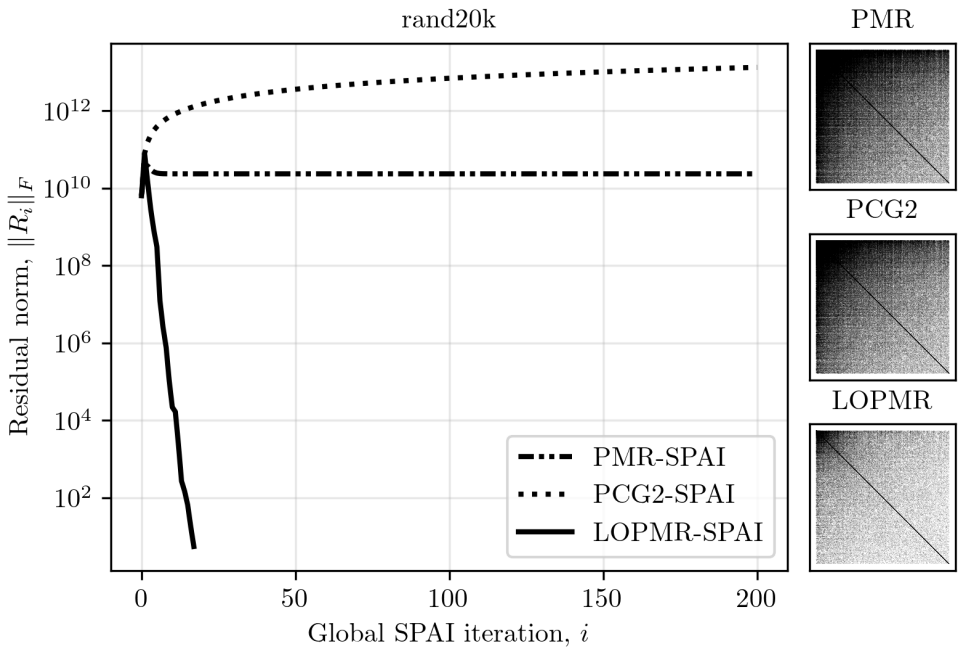
- bundle1 matrix (from SuiteSparse Collection) with Jacobi preconditioner, and 3% density:

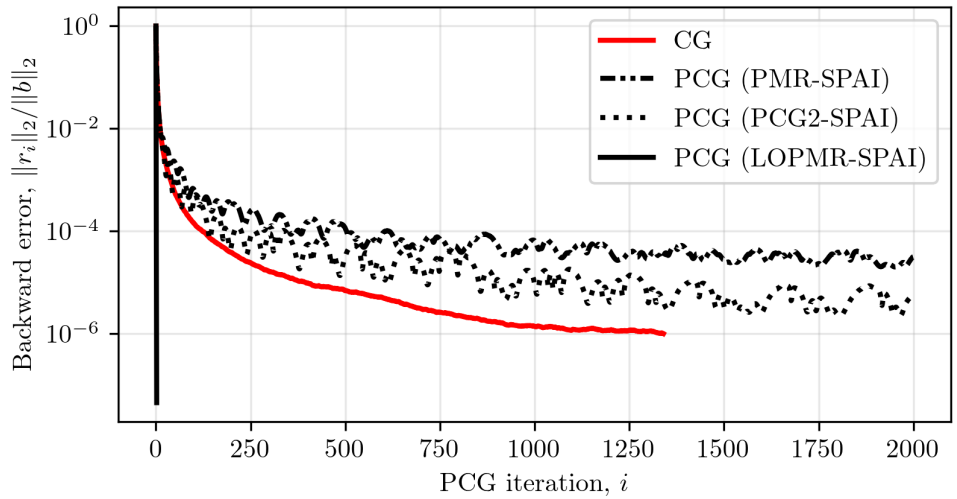




With dropping experiments, cont'd

- rand20k matrix (github.com/venkovic/matrix-market) with Jacobi preconditioner,
  and 3% density:





Closing remarks

Conclusion

- Findings:

  Global iterative methods were introduced for the approximation of SPAIs of SPD matrices:

  - SD, MR and CG1 all fail to yield SPAIs with SPD spectra for SPD matrices, even without dropping.
  - LOMR and CG2 both consistently yield SPAIs with SPD spectra for SPD matrices, without dropping.
  - LOMR achieves better SPAIs, with dropping, than CG2.

- Dissemination:

  - Preprint:

    Venkovic & Anzt (2025). Global iterative methods for sparse approximate inverses of symmetric positive-definite matrices.

  - Repository allowing reproducible experiments:

    github.com/venkovic/julia-global-spd-spai

  - Find this presentation at:

    venkovic.github.io/research

Related ongoing and future work

- Global iterative methods for the approximation of SPAIs of general matrices:

  Venkovic & Anzt (2025). Global iteration methods for sparse approximate inverses of general matrices.

  github.com/venkovic/julia-global-general-spai

- Randomized short-recurrence iterative methods for approximate low-rank matrix factorizations:

  Venkovic & Anzt (2025). Randomized first-order short-recurrence subspace iterative methods for approximate low-rank matrix factorizations.

  github.com/venkovic/julia-iterative-low-rank

- Related future works:
  - SPAIs:
    - Parallelization.
  - Low-rank approximation:
    - Application to matrix recovery (completion and sensing problems).
    - Non-negative matrix factorizations.
    - Tensor factorizations.