Approximate inverse preconditioning with adaptive dropping

Jiří Kopal[1]

joint work with Miroslav Rozložník[2] and Miroslav Tůma[2]

[1]Department of Mathematics, Technical University of Liberec

[2]Institute of Computer Science, Academy of Sciences of the Czech Republic

Sparse Days 2013

17-18 June, 2013
Toulouse

Outline

1. Linear algebraic equations and iterative methods
2. Approximate inverse preconditioning
3. Adaptive dropping technique that produces to robust and sparse preconditioning
4. Test problems

Outline

1. Linear algebraic equations and iterative methods
2. Approximate inverse preconditioning
3. Adaptive dropping technique that produces to robust and sparse preconditioning
4. Test problems

Our approach based on:

- Factorization of the inverse
- Generalized Gram–Schmidt process

Problem and the conjugate gradient method

Consider a system of linear algebraic equations with a *sparse SPD* matrix

$$Ax = b,\ A \in \mathbb{R}^{n,n}$$

- Basic iterative approach
  $\rightarrow$ the conjugate gradient method
- Preconditioners for iterative solvers
  - accelerate their convergence
    (by improving spectral properties of the matrix A)
  - necessary for increasing robustness of iterative methods
- We are interested in approximate inverse preconditioning: $\tilde{Z}\tilde{Z}^T \approx A^{-1}$

The preconditioned linear system we get in the form:

$$\tilde{Z}^T A \tilde{Z} y = \tilde{Z}^T b \text{ , where } x = \tilde{Z} y$$

The preconditioned conjugate gradient method

There are two indicators that predicate the quality of the approximate inverse preconditioning:

- Fill-in in the matrix $\tilde{Z}$ ($nnz(\tilde{Z})$)
  (cost of the preconditioner computation & application)
- Loss of A-orthogonality among column vectors measured by $\|I - \tilde{Z}^T A \tilde{Z}\|$
  (accuracy of the preconditioner)

Their simultaneous minimization is usually in contradiction.

The goal is to find a balance between them.

Generalized Gram–Schmidt and exact arithmetic identities

- We use the inverse decompositions based on the Gram-Schmidt (GS) algorithm with A-inner product
- Column vectors of the matrix $Z^{(0)}$ (here $Z^{(0)} = I$) are A-orthogonalized against previously computed column vectors in Z

Algorithm produces upper triangular matrices Z and U, such that:

- $A = U^T U$ (Cholesky factorization)
- $Z^T A Z = I, \quad UZ = I$
- $ZU = I$

Generalized Gram–Schmidt and exact arithmetic identities

- We use the inverse decompositions based on the Gram-Schmidt (GS) algorithm with A-inner product
- Column vectors of the matrix $Z^{(0)}$ (here $Z^{(0)} = I$) are A-orthogonalized against previously computed column vectors in Z

Algorithm produces upper triangular matrices Z and U, such that:

- $A = U^T U$ (Cholesky factorization)
- $Z^T AZ = I, \quad UZ = I$
- $ZU = I$

Approximate inverse decomposition is based on the incomplete algorithm - the same algorithm, but small entries (in some sense) are discarded $\rightarrow \tilde{Z}, \tilde{U}$

Dropping based on conditioning of $\tilde{U}$ (aposteriori filtering)

The $k-$*th* step of the *incomplete* generalized Gram–Schmidt process provides leading principal submatrices $\tilde{U}_k,\ [\tilde{U}_k]_{ji} = \tilde{u}_{ji}$, $\tilde{Z}_{k-1} = [\tilde{z}_1\ \tilde{z}_2 \ldots \tilde{z}_{k-1}]$, and column vector $\hat{z}_k$.

$$\hat{z}_k = (e_k - \sum_{j=1}^{k-1} \tilde{u}_{jk}\tilde{z}_j)/\|e_k - \sum_{j=1}^{k-1} \tilde{u}_{jk}\tilde{z}_j\|_A$$

For the residuals δe_k we can write $\quad \tilde{U}_k\hat{z}_k - e_k = \delta e_k.$

Assume that $\quad \|\delta e_k\|_\infty \leq \tau_k \|\tilde{U}_k\| \|\hat{z}_k\|_\infty.$

Problem is to find a perturbation δz_k such that $\tilde{z}_k = \hat{z}_k + \delta z_k$ be sparse.

We want to have

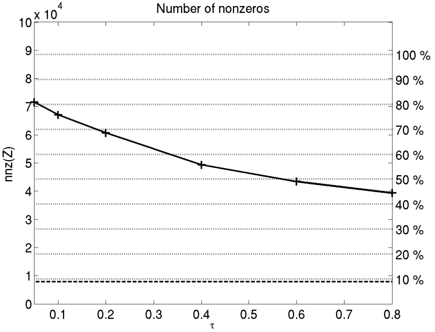
$$\|\tilde{U}_k\| \|\delta z_k\|_\infty \leq \tau_k \|\tilde{U}_k\| \|\hat{z}_k\|_\infty \leq \tau \mathcal{O}(k^{-1/2}) \quad \rightarrow \quad \frac{\|\delta z_k\|_\infty}{\|\hat{z}_k\|_\infty} \leq \tau_k \leq \frac{\tau}{\kappa(\tilde{U}_k)}.$$

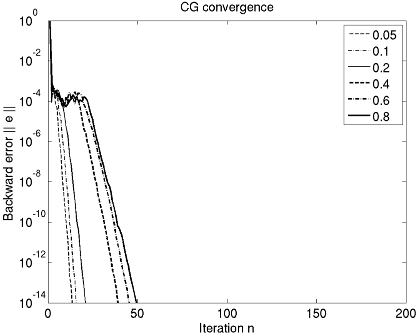
Filtering algorithm:

$$(1) \qquad mask = \frac{|\hat{z}_k|}{\|\hat{z}_k\|_\infty} \geq \tau_k$$

$$(2) \qquad \tilde{z}_k = \hat{z}_k . * mask / \|\hat{z}_k . * mask\|_A$$

Power of our dropping





Pivoting by magnitudes of the diagonal entries of U

- partial pivoting for sparse least squares

  [*S. Bellavia, J. Gondzio and B. Morini: New matrix-free preconditioner for sparse least-squares problems, Technical Report ERGO-11-010, School of Mathematics, The University of Edinburgh, July 8, 2011*]

- pivoting in GS means:
  1. $u_{1,1} \geq u_{2,2} \geq \ldots \geq u_{n,n}$
  2. $u_{j,j} > |u_{j,k}|, \quad \forall j = 1, \ldots, n, \quad \forall k = j+1, \ldots, n$
  3. $u_{i,i}^2 \geq \sum_{k=i}^{j} u_{k,j}^2, \quad \forall j = i+1$

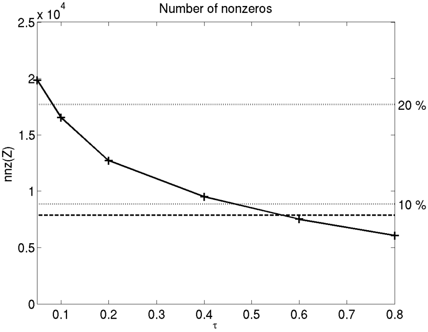
- *pivoting* is easy in right-looking code

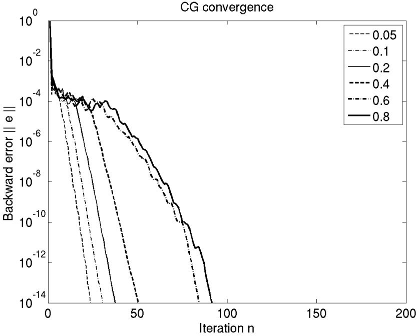
- pivots are not known a priori in left-looking code

- left-looking code needs for *pivoting* additional updates of the diagonal entries

$$(u_{i,i}^{(k)})^2 = a_{i,i}^2 - \sum_{j=1}^{k-1} \langle z_i^{(0)}, z_j \rangle_A^2 \rightarrow \text{left-looking GS with pivoting}$$

Power of our dropping (with approximate pivoting)





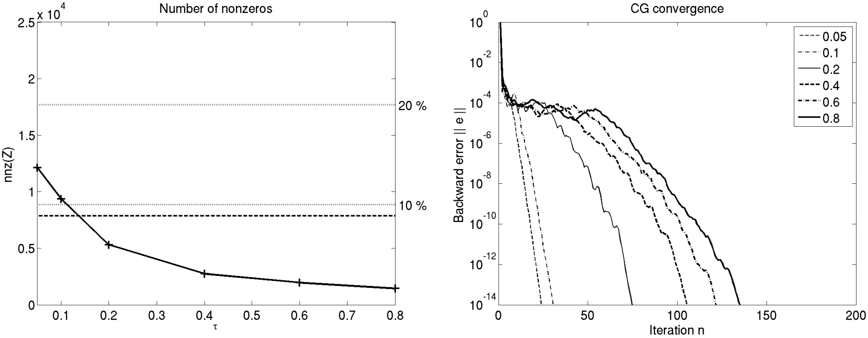
Dropping based on a posteriori filtering on scaled matrix

Consider a diagonal scaling:

$$
\begin{array}{l}
\hat{A}^{(0)} = A = [\hat{a}_1^{(0)} \hat{a}_2^{(0)} \dots \hat{a}_n^{(0)}] \\
D^{(0)} = I_n \\
\text{for} \quad k = 1, 2, \dots \quad \text{until} \quad |\|\hat{a}_i^{(k)}\| - 1| \leq c, \ \forall i \quad \text{do} \\
\qquad D = \operatorname{diag}(\|\hat{a}_i^{(k)}\|^{1/2})_{i=1,2,\dots,n} \\
\qquad \hat{A}^{(k+1)} = D^{-1} A^{(k)} D^{-1} \\
\qquad D^{(k+1)} = D^{(k)} D \\
\ \text{end}
\end{array}
$$

[Chih-Jen Lin and Jorge J. Moré: Incomplete Cholesky Factorizations with Limited Memory (1999)]

Power of our dropping (with approximate pivoting and diagonal scaling)



Comparison of $\tilde{Z}_k^T A_k \tilde{Z}_k$ spektra

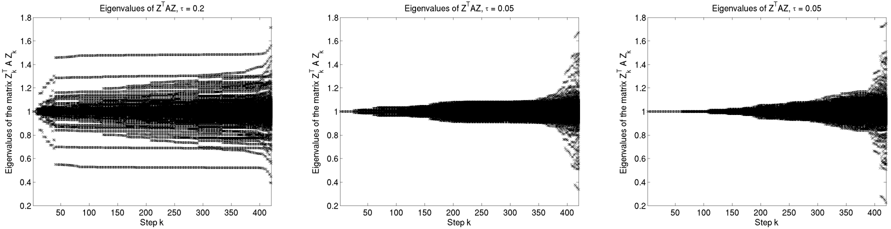
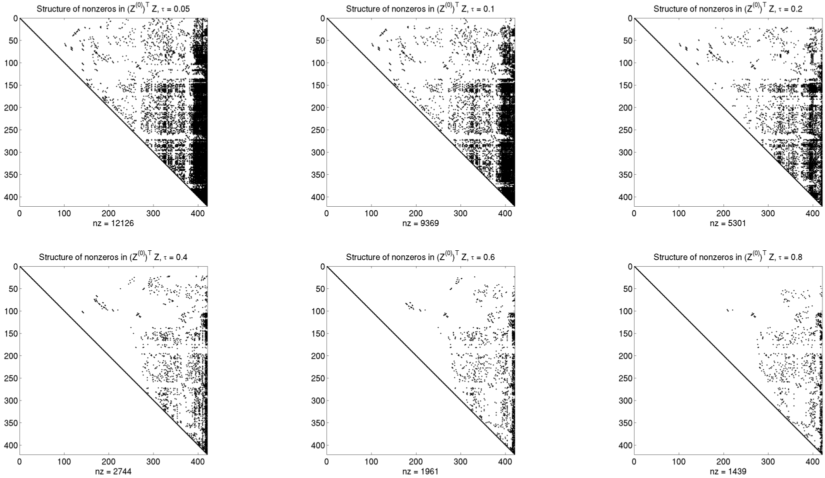


Figure : Group of figures showing the eigenvalues of the matrices $\tilde{Z}_k^T A \tilde{Z}_k$ that with CG converges similarly. On the left we can see results for the basic a posteriori filtering without reordering *(iters = 20, nnz($\tilde{Z}$) = 60659)*, in the middle we can see results for the a posteriori with dynamic pivoting *(iters = 23, nnz($\tilde{Z}$) = 19827)*, and on the right we can see results for the a posteriori with dynamic pivoting and diagonal scaling *(iters = 23, nnz($\tilde{Z}$) =12126)*.

Sparsity of $\tilde{Z}$ (with approximate pivoting and diagonal scaling)



Test problems

matrix	n	$iter.$	$nnz(\tilde{Z})/nnz(A)$	description
bcsstk08	1074	17	0.696	STIFFNESS MATRIX, FRAME BUILDING (TV STUDIO)
bcsstk09	1083	33	1.85	STIFFNESS MATRIX, SQUARE PLATE CLAMPED
msc01050	1050	60	2.4	TEST MATRIX FROM MSC/NASTRAN STARF8.OUT2
msc01440	1440	42	3.41	TEST MATRIX FROM MSC/NASTRAN CYLF8.OUT2
nos2	957	26	5.27	FE APPROXIMATION TO BIHARMONIC OPERATOR ON BEAM
nos3	960	20	5.56	FE APPROXIMATION TO BIHARMONIC OPERATOR ON PLATE
nos5	468	23	1.37	FE APPROXIMATION OF BUILDING
nos6	675	13	1.33	POISSON'S EQUATION IN L SHAPE, MIXED BC
nos7	729	12	2.53	POISSON'S EQUATION IN UNIT CUBE
1138_bus	1138	24	5.02	ADMITTANCE MATRIX 1138 BUS POWER SYSTEM, D.J.TYLAVSKY

Table : Results for some problems for $\tau = 0.01$

Conclusion

- Developed new adaptive dropping strategy
- Theoretically motivated
- Generates sparse and robust preconditioned iterative solvers (in our case)
- Connection to IC $\rightarrow$ new dropping for IC?

Thank you for your attention!!!