NA-T46

42nd Annual Iranian Mathematics Conference

5-8 September 2011, Vali-e-Asr University of Rafsanjan, Iran, pp 1116-1119

DIFFERENT DROPPING STRATEGIES IN THE ILUFF ALGORITHM

FATEMEH SHAHLAEI[1] AND AMIN RAFIEI[2*]

Abstract. In this paper, different ILUFF preconditioners are computed by using different dropping techniques in the ILUFF algorithm [3]. We compare quality of these different preconditioners.

1. Introduction

Consider the linear system of equations

$$AX = b, \tag{1.1}$$

where the coefficient matrix $A \in R^{n\times n}$ is nonsymmetric, nonsingular, large, sparse and $X, b \in R^n$. Suppose $M \approx A$. Linear system

$$M^{-1}AX = M^{-1}b, \tag{1.2}$$

is termed left preconditioned system of system (1.1) and matrix M is called left preconditioner matrix [4]. We solve system (1.2), with Krylov subspace methods [4].
In Algorithm 1 of this paper, $A_{:,j}$ and $A_{j,:}$ refer to j-th column and j-th row of matrix A, respectively.

2. ILUFF algorithm

Suppose that matrix A is nonsymmetric. Also, suppose that $W = [w_1^T, ..., w_n^T]^T$ and $Z = [z_1, ..., z_n]$ are unit lower and upper triangular matrices, respectively and $D = diag(d_1, ..., d_n)$ is a diagonal matrix. FFAPINV algorithm [2] computes matrices W, Z and D such that $WAZ \approx D$. It is possible to obtain an incomplete LU decomposition of matrix A, as by-product of FFAPINV process, such that L is an unit lower triangular and U is an upper

2000 *Mathematics Subject Classification.* Primary 65F10; Secondary 65F50.
Key words and phrases. Incomplete LU decomposition, preconditioning, inverse_based dropping.
* Speaker: Amin Rafiei.

triangular matrix and $A \approx M = LU$. In this case, matrix M is called ILUFF preconditioner [3].

Algorithm 1 (ILUFF algorithm)

1. $w_1 = e_1^T, \quad z_1 = e_1, \quad d_1 = a_{11}.$
2. for $j = 2$ to n do
3. $w_j = e_j^T, \quad z_j = e_j.$
4. for $i = 1$ to $j - 1$ do
5. $L_{ji} = \frac{A_{j,:}z_i}{d_i}, \qquad U_{ij} = \frac{w_i A_{:,j}}{d_i}$
6. apply a dropping rule to L_{ji} and to U_{ij}
7. $z_j = z_j - (\frac{w_i A_{:,j}}{d_i})z_i, \qquad w_j = w_j - (\frac{A_{j,:}z_i}{d_i})w_i$
8. for all $l \leq i$ apply a dropping rule to z_{lj} and to w_{jl}(first format of dropping for W and Z)
9. end for
10. for all $l \leq i$ apply a dropping rule to z_{lj} and to w_{jl}(second format of dropping for W and Z)
11. $d_j = w_j A_{:,j}$(if A is not positive definite)
12. $d_j = w_j A w_j^T$ (if A is positive definite)
13. end for
14. Return $L = (L_{ij})$ and $U = (d_i U_{ij})$

3. Drop entries of Z and W matrices

Suppose that $\varepsilon_Z, \varepsilon_W$ be the drop tolerance parameters of Z and W matrices, respectively. We have used two strategies to drop entries of z_j and w_j vectors in Algorithm 1.

- **First dropping strategy:** Just line 8 of Algorithm 1, will be run and line 10 will not. Then, entries z_{lj} and w_{jl}, for $l \leq i < j$ are dropped when

$$|z_{lj}| \leq \varepsilon_Z, \qquad |w_{jl}| \leq \varepsilon_W. \tag{3.1}$$

- **Second dropping strategy:** Just line 10 of Algorithm 1, will be run and line 8 will not. In this case, the whole vectors z_j and w_j are computed as:

$$z_j = e_j - \sum_{i=1}^{j-1}(\frac{w_i A_{:,j}}{d_i})z_i, \qquad w_j = e_j^T - \sum_{i=1}^{j-1}(\frac{A_{j,:}z_i}{d_i})w_i,$$

and then, entries w_{jl} and z_{lj}, for $l \leq j$, are dropped when criterions (3.1) are satisfied.

4. Drop entries of L and U matrices

- **Inverse_based dropping:** Let $\varepsilon_{L,W}$ be the same drop tolerance parameter for L, W matrices and $\varepsilon_{U,Z}$ be the same drop tolerance parameter for U, Z matrices. Consider $\varepsilon_{L,W}$ as ε_W and $\varepsilon_{U,Z}$ as ε_Z. We drop entries z_{lj} and w_{jl}, for $l \leq i < j$, when criterions (3.1) hold. Then in line 6 of algorithm 1, entries L_{ji} and U_{ij}, for $i < j$, are dropped when $|L_{ji}|\|W_{i,:}\|_1 \leq \varepsilon_{L,W}$ and $|U_{ij}|\|Z_{:,i}\|_\infty \leq \varepsilon_{U,Z}$.
- **Simple dropping:** Let $\varepsilon_L, \varepsilon_U$ be the drop tolerance parameters for L, U matrices. In line 6 of Algorithm 1, entries L_{ji} and U_{ij}, for $i < j$, are dropped when $|L_{ji}| \leq \varepsilon_L$ and $|U_{ij}| \leq \varepsilon_U$.

- **ILUFF1:** First dropping strategy is used for W and Z matrices and simple dropping strategy is used for L and U matrices.
- **ILUFF2:** First dropping strategy is used for W and Z matrices and inverse_based dropping strategy is used for L and U matrices.
- **ILUFF3:** Second dropping strategy is used for W and Z matrices and simple dropping strategy is used for L and U matrices.
- **ILUFF4:** Second dropping strategy is used for W and Z matrices and inverse_based dropping strategy is used for L and U matrices.

5. Numerical results

In this section, we report results of left preconditioned GMRES(16) method. Preconditioners are ILUFF1, ILUFF2, ILUFF3 and ILUFF4. All coefficient matrices are only nonsymmetric and from University of Florida Sparse Matrix Collection [1]. Vector b is Ae in which $e = [1, ..., 1]^T$. All the codes are written in MATLAB and we have run all the experiments on a machine with 1GB of RAM memory. In all the experiments, if the pivot element d_j is less than the machine precision, then we have replaced it by 10^{-4}. Density of Preconditioners is defined as $density = \frac{nnz(L)+nnz(U)}{nnz(A)}$, in which $nnz(L), nnz(U)$ and $nnz(A)$ refer to the number of nonzero entries of L, U and A matrices, respectively. In all the experiments, we have selected $\varepsilon_L, \varepsilon_U, \varepsilon_W, \varepsilon_Z, \varepsilon_{L,W}$ and $\varepsilon_{U,Z}$ equal to 0.1.

Table 1, reports results of GMRES(16) method without preconditioning. In this Table, n indicates the dimension of the matrix and PD column indicates whether or not the matrix is positive definite. *Itime*, indicates the iteration time of GMRES(16) without preconditioning and *it*, is the number of iterations of GMRES(16) method. *Itime* is in second. In this Table, + means that there is no convergence after 10,000 number of iterations. In all the experiments, the stopping criterion is $\frac{\|r_k\|_2}{\|r_0\|_2} \leq 10^{-8}$, in which r_k is the k-th residual vector of the system and r_0 is the initial residual vector. In all the experiments, the initial guess is the zero vector. In Table 2, *Ptime* is the preconditioning time and density is the density of preconditioner. *Ptime* is also in second.

In Table 3, *Ttime* is the total time which is the sum of preconditioning time and iteration time and *it*, is the number of iterations of left preconditioned GMRES(16). In this Table, + indicates that no converge has been obtained in 5000 number of iterations.

Table1: information of GMRES(16) method without preconditioning and matrix properties.

Matrix	n	nnz	PD	$Itime$	it
pde900	900	4380	yes	0.203	10
saylr3	1000	3750	No	0.859	37
cavity06	1182	29675	No	+	+
sherman4	1104	3786	No	0.531	23
epb0	1794	7764	No	+	+
pde2961	2961	14585	yes	0.731	18

Table2: properties of ILUFF1, ILUFF2, ILUFF3 and ILUFF4 preconditioners.

method	ILUFF1		ILUFF2		ILUFF3		ILUFF4	
	density	*Ptime*	*density*	*Ptime*	*density*	*Ptime*	*density*	*Ptime*
pde900	1.273516	125.531	1.424201	262.063	1.283562	172.203	1.437443	223.328
saylr3	1.056533	189.39	1.172000	4550343	1.069333	235.562	1.203733	300.515
cavity06	0.291794	678.782	0.376243	1484.22	0.295636	993	0.407515	817.828
sherman4	1.243001	266.203	1.312467	804.922	1.750386	319.172	1.321447	574.328
epb0	1.575348	943.093	1.981968	2360.44	1.750386	1247.88	2.248583	1777.27
pde2961	1.234763	6996.83	1.327048	10863.1	1.248269	5879.3	1.334248	9262.74

Table3: information of GMRES(16) method for left preconditioned systems.

method	ILUFF1		ILUFF2		ILUFF3		ILUFF4	
	it	*Ttime*	*it*	*Ttime*	*it*	*Ttime*	*it*	*Ttime*
pde900	2	126.953	1	262.829	2	173.953	1	223.953
saylr3	2	191.093	1	456.656	2	237.812	1	301.296
cavity06	+	+	+	+	+	+	95	920.188
sherman4	4	270.328	3	810.734	3	322.687	3	577.485
epb0	14	978.405	8	2385.63	13	1335.89	8	1795.58
pde2961	4	7027.38	4	10894.7	4	5928.36	3	9294.05

6. Conclusion

Results of Tables 1 and 3, show that ILUFF1, ILUFF2, ILUFF3 and ILUFF4 preconditioners are useful tools to decrease the number of iterations of GMRES(16) method.
Comparison of columns 2 and 6 of Table 3, indicates that ILUFF3 preconditioner decreases the number of iterations of GMRES(16) method a little bit more than ILUFF1 preconditioner. Comparison of columns 2 and 4 and columns 6 and 8 of this table, also shows that ILUFF2 preconditioner decreases the number of iterations of GMRES(16) method more than ILUFF1 preconditioner and ILUFF4 preconditioner decreases the number of iterations of GMRES(16) method more than ILUFF3 preconditioner.

References

[1] http://www.cise.ufl.edu/research/sparse/matrices. Accessed 2011.
[2] D. K. Salkuyeh, A Sparse Approximate Inverse Preconditioner for Nonsymetric Positive Definite Matrices, J. Appl. Math and Informatics., 28(2008), 1113-1141.
[3] D. K. Salkuyeh, A. Rafiei and H. Roohani, ILU preconditoning Based on the FFAPINV Algorithm, arXive:1010.2812., (2010).
[4] Y. Saad, Itrative Methods for Sparse Linear Systems. PWS publishing, New York., (1996).

[1] Department of Mathematics, Sabzevar Tarbiat Moallem University, P. O. Box 9617976487, Sabzevar, Iran.
E-mail address: fateme_shahlaei@yahoo.com

[2] Department of Mathematics, Sabzevar Tarbiat Moallem University, P. O. Box 9617976487, Sabzevar, Iran.
E-mail address: rafiei.am@gmail.com