Right-looking approximate

Amin Rafiei & Leila Kakhki Beydokhti*

Department of applied mathematics, Hakim Sabzevari University, Sabzevar, Iran

rafiei.am@gmail.com, l.kakhki2971@gmail.com

Article Code: A-10-550-3

Abstract

In this paper, we present a block format of right-looking $AINV$ preconditioner. In the numerical tests, we compare the results of the plain and the block version of this preconditioner.

1 Introduction

The right-looking approximate inverse preconditioner M for the nonsymmetric linear system of equations

$$Ax = b, \tag{1}$$

is in the following form

$$A^{-1} \approx M = ZD^{-1}W^T, \tag{2}$$

where Z and W are unit upper triangular matrices and D is a diagonal matrix. This preconditioner was first studied in [1]. In this reference, the authors used the notation $AINV$ for this preconditioner. For a symmetric matrix A, a block format of $AINV$ preconditioner has been presented in [2]. In this case, the $AINV$ preconditioner has only two Z and D factors. Z is a block unit upper triangular and D is a block diagonal matrix.

In this paper, we introduce a new block right-looking $AINV$ preconditioner. In this block format, Z and W are again unit upper triangular while D is a block diagonal matrix. The diagonal blocks of D are of order 1×1 or 2×2.

2 Algorithms of plain and block right-looking $AINV$ preconditioner

Algorithm 1 computes the block right-looking $AINV$ preconditioner. In each step i of this algorithm we check in lines 16 and 23 to know whether we have a 1×1 pivot entry or a 2×2 block pivot. If at step i of Algorithm 1 the pivot is selected to be a 1×1 entry, then $status(i)$ is set to be 1 and we call Algorithm 2 in line 19. Inside Algorithm 2 and based on the value of $status(i)$, only the lines 2-5 are run. In these lines the column i of matrix Z will update the columns $(i+1)$st to n of this matrix. Calling Algorithm 2 in line 20 of Algorithm 1 will update the columns $(i+1)$st to n of matrix W. If at step i of Algorithm 1 the pivot is selected to be a 2×2 block, then $status(i)$ is set equal to 2 in line 27 and we should call Algorithm 2 two more times in lines 28 and 29. In this case, the columns i and $i+1$ of matrix Z (W) will update the columns $i+2$ to n of this matrix.

Algorithm 1 (Block right-looking AINV preconditioner)

Input: $A \in \mathbb{R}^{n\times n}$ a nonsymmetric matrix, $\tau_w, \tau_z \in (0,1)$ are the drop tolerance parameters for W and Z .
Output: $A^{-1} \approx ZD^{-1}W^T$ where Z and W are unit upper triangular and D is a block diagonal matrix .

1. $status(i) = 0, 1 \le i \le n$
2. $i = 1$
3. **while** $i < n$ **do**
4.   **for** $j = i$ to n **do**
5. $S_{ij}^{(i-1)} = (w_i^{(i-1)})^T A z_j^{(i-1)}$
6. $S_{i+1,j}^{(i-1)} = (w_{i+1}^{(i-1)})^T A z_j^{(i-1)}$
7.   **end for**
8.   **for** $j = i+2$ to n **do**
9. $S_{ji}^{(i-1)} = (w_j^{(i-1)})^T A z_i^{(i-1)}$
10. $S_{j,i+1}^{(i-1)} = (w_j^{(i-1)})^T A z_{i+1}^{(i-1)}$
11.   **end for**
12. $v_i = max\{\frac{1}{|S_{ii}^{(i-1)}|}\sum_{j=i+1}^n |S_{ij}^{(i-1)}|, \frac{1}{|S_{ii}^{(i-1)}|}\sum_{j=i+1}^n |S_{ij}^{(i-1)}|\}$
13. $w_i^1 = \sum_{j=i+2}^n \|\begin{bmatrix} S_{ii}^{(i-1)} & S_{i,i+1}^{(i-1)} \\ S_{i+1,i}^{(i-1)} & S_{i+1,i+1}^{(i-1)} \end{bmatrix}^{-1} \begin{pmatrix} S_{ij}^{(i-1)} \\ S_{i+1,j}^{(i-1)} \end{pmatrix}\|_\infty$
14. $w_i^2 = \sum_{j=i+2}^n \|\begin{pmatrix} S_{ji}^{(i-1)} & S_{j,i+1}^{(i-1)} \end{pmatrix} \begin{bmatrix} S_{ii}^{(i-1)} & S_{i,i+1}^{(i-1)} \\ S_{i+1,i}^{(i-1)} & S_{i+1,i+1}^{(i-1)} \end{bmatrix}^{-1}\|_\infty$
15. $w_i = \max\{w_i^1, w_i^2\}$
16.   **if** $v_i < w_i$ **then**
17. $D_{ii} = S_{ii}^{(i-1)}$
18. $status(i) = 1$
19.   Column_Const (Z , τ_z , A , D , i , $status$)
20.   Column_Const (W , τ_w , A^T , D^T , i , $status$)
21. $i = i + 1$
22.   **end if**
23.   **if** $v_i \ge w_i$ **then**
24. $D_{i:i+1,i:i+1} = \begin{bmatrix} S_{ii}^{(i-1)} & S_{i,i+1}^{(i-1)} \\ S_{i+1,i}^{(i-1)} & S_{i+1,i+1}^{(i-1)} \end{bmatrix}$
25. $z_{i+1}^{(i)} = z_{i+1}^{(i-1)}$
26. $w_{i+1}^{(i)} = w_{i+1}^{(i-1)}$
27. $status(i) = 2$
28.   Column_Const (Z , τ_z , A , D , i , $status$)
29.   Column_Const (W ,τ_w , A^T , D^T , i , $status$)
30. $i = i + 2$
31.   **end if**
32. **end while**
33. **if** $status(n-1) = 0 \ || \ status(n-1) = 1$ **then**
34. $D_{nn} = (w_n^{(n-1)})^T A z_n^{(n-1)}$
35. **end if**
36. Return $Z = [z_1^{(0)}, z_2^{(1)}, \cdots, z_n^{(n-1)}]$, $W = [w_1^{(0)}, w_2^{(1)}, \cdots, w_n^{(n-1)}]$ and D

Algorithm 2 (Column construction of a matrix)

$Column\_Const(Z, \tau_z, A, D, i, status)$

1. **if** $status(i) = 1$ **then**
2.   **for** $j = i+1$ to n **do**
3. $z_j^{(i)} = z_j^{(i-1)} - z_i^{(i-1)} \frac{1}{D_{ii}} A_{i,:} z_j^{(i-1)}$
4.   suppose that $z_j^{(i)} = (z_{lj}^{(i)})$. For all $l \le i$, if $|z_{lj}^{(i)}| < \tau_z$, then $z_{lj}^{(i)} = 0$
5.   **end for**
6. **end if**
7. **if** $status(i) = 2$ **then**
8.   **for** $j = i+2$ to n **do**
9. $k = i + status(i) - 1$
10. $z_j^{(k)} = z_j^{(i-1)} - [z_i^{(i-1)} \quad z_{i+1}^{(i)}] D_{i:i+1,i:i+1}^{-1} A_{i:i+1} z_j^{(i-1)}$
11.   suppose that $z_j^{(i)} = (z_{lj}^{(i)})$. For all $l \le i$, if $|z_{lj}^{(i)}| < \tau_z$, then $z_{lj}^{(i)} = 0$
12.   **end for**
13. **end if**

Algorithm 3 gives the plain right-looking $AINV$ preconditioner. At the end of this algorithm, the approximate inverse factorization (2) will be computed. At step i of this algorithm and in lines 4-8, the ith column of matrix Z (W) will update the columns $(i+1)$st to n of this matrix. In line 9 of this algorithm, the pivot element D_{ii} is introduced.

Algorithm 3 (Right-looking $AINV$ preconditioner)

Input: $A \in \mathbb{R}^{n\times n}$ a nonsymmetric matrix, $\tau_w, \tau_z \in (0,1)$ are the drop tolerance parameters for W and Z .
Output: $A^{-1} \approx ZD^{-1}W^T$

1. $w_i^{(0)} = e_i, \ z_i^{(0)} = e_i, \ 1 \le i \le n.$
2. **for** $i = 1$ to n **do**
3. $p_i^{(i-1)} = (w_i^{(i-1)})^T A z_i^{(i-1)}, \ q_i^{(i-1)} = (w_i^{(i-1)})^T A z_i^{(i-1)}.$
4.   **for** $j = i+1$ to n **do**
5. $p_j^{(i-1)} = \frac{(w_j^{(i-1)})^T A z_i^{(i-1)}}{p_i^{(i-1)}}, \ q_j^{(i-1)} = \frac{(w_i^{(i-1)})^T A z_j^{(i-1)}}{q_i^{(i-1)}}.$
6. $w_j^{(i)} = w_j^{(i-1)} - p_j^{(i-1)} w_i^{(i-1)}, \ z_j^{(i)} = z_j^{(i-1)} - q_j^{(i-1)} z_i^{(i-1)}.$
7.   for all , $l \le i$, drop entries $z_{lj}^{(i)}, w_{lj}^{(i)}$ if their absolute values are less than $\tau_z and \tau_w$, respectively.
8.   **end for**
9. $D_{ii} = p_i^{(i-1)}.$
10. **end for**
11. Return $Z = [z_1^{(0)}, z_2^{(1)}, \cdots, z_n^{(n-1)}]$, $W = [w_1^{(0)}, w_2^{(1)}, \cdots, w_n^{(n-1)}]$ and $D = diag(D_{ii})_{1 \le i \le n}$.

3 Numerical tests

In this section, we have downloaded 4 nonsymmetric matrices from [3]. Then, we have generated artificial linear systems (1) where $b = Ae$ and $e = [1, \cdots, 1]^T$. After that, we have used the $GMRES(15)$ Krylov subspace method [4] to compute an approximate solution for these systems. The initial solution for this method is selected to be the zero vector and the stopping criterion is satisfied when the relative residual is less than 10^{-8}. In Table 1, n and nnz are the dimension and the number of nonzero entries of the matrix. $Iter(1)$ and $Iter(2)$ are the inner and outer iterations of the $GMRES(15)$ method. We have implemented Algorithms 1, 2 and 3 in MATLAB. We have also applied the command $gmres$ in this software.

Table 1: matrix properties and results of $GMRES(15)$

Matrix properties			GMRES(15)		
name	n	nnz	iter(1)	iter(2)	Time
orsirr-2	886	5970	449	2	2.0898
pde900	900	4380	13	1	0.1456
fs-680-3	680	2471	1000	15	4.0480
pores-3	532	3474	1000	15	3.8389

For all the four artificial linear systems we have set $\tau_w = \tau_z = 0.1$ and computed the plain and the block right-looking $AINV$ preconditioner. Then, we have used these two preconditioners as the right preconditioner and have solved the right preconditioners systems by the $GMRES(15)$ method. In Table 2, the notation $RLAINV(0.1) + GMRES(15)$ is used for the case when the preconditioner is the plain right-looking $AINV$ and it is mixed by the $GMRES(15)$. In this table, the notation $BRLAINV(0.1) + GMRES(15)$ shows that the preconditioner is the block right-looking $AINV$ and $GMRES(15)$ has been used to solve the right preconditioned systems. In Table 2, $ptime$ is the preconditioning time which is in seconds and $density$ is defined as

$$density = \frac{nnz(Z) + nnz(W)}{nnz(A)},$$

where $nnz(Z)$, $nnz(W)$ and $nnz(A)$ are the number of nonzero entries of matrices Z, W and A, respectively. In this table, $Ttime$ is the total time which is the preconditioning time plus the iteration time of $GMRES(15)$. $Iter(1)$ and $Iter(2)$ have the same definition as in Table 1. The results of Table 2 indicate that $BRLAINV(0.1) + GMRES(15)$ is a better solver than $RLAINV(0.1) + GMRES(15)$ since it gives a less number of iterations.

Table 2: properties of the preconditioners and results of $GMRES(15)$ to solve the right preconditioned systems

	RLAINV(0.1)+GMRES(15)					BRLAINV(0.1)+GMRES(15)				
matrix	ptime	density	iter(1)	iter(2)	Ttime	ptime	density	iter(1)	iter(2)	Ttime
orsirr-2	17.7469	10.1365	114	15	34.6323	69.7356	1.1028	4	8	70.2005
pde900	18.7464	2.4301	3	15	18.8261	68.0779	2.1694	3	10	68.5125
fs-680-3	8.6196	1.8672	1	7	8.8288	31.6749	1.2460	1	5	31.6884
pores-3	4.45718	24.5046	714	15	12.9983	19.3002	2.6914	4	8	19.4104

References

[1] M. Benzi and M. Tuma, A sparse approximate inverse preconditioner for nonsymmetric linear systems, SIAM J. Sci. comput. 19(3), (1998) 968-994.

[2] M. Benzi, R. Kouhia and M. Tuma, Stabilized and block approximate inverse preconditioners for problems in solid and structural mechanics, Computer Meth. Appl. Mech. Eng., 190, 6533-6554 (2001)

[3] T. Davis, *The SuiteSparse Matrix Collection*, http://www.cise.ufl.edu/research/sparse/matrices.

[4] Y. Saad, *Iterative methods for sparse linear systems*, SIAM, Philadelphia. 2nd edition (2003).

48th Annual Iran MATHEMATICS Conference

Bu-Ali Sina University, August 22-25, 2017

Hamedan - Iran

Address:

Department of Mathematics, Faculty-

of Sciences, Bu-Ali Sina University

Tel & Fax: +98 81 3838 1426

zip code: 65178-38695

نشانی دبیرخانه:

دانشگاه بوعلی سینا

دانشکده علوم پایه-گروه ریاضی

تلفکس: ۰۸۱-۳۸۳۸۱۴۲۶

کدپستی: ۶۵۱۷۸-۳۸۶۹۵

aimc48@ims.ir

www.conf.ims.ir/aimc48

باسمه تعالی

تاریخ:
شماره:
پیوست:

48th Annual Iranian MATHEMATICS Conference
Bu-Ali Sina University, August 22-25, 2017
Hamedan- Iran

چهل و هشتمین کنفرانس ریاضی ایران
۳۱ مرداد تا ۳ شهریور ۱۳۹۶
دانشگاه بوعلی سینا- همدان- ایران

بدینوسیله گواهی می شود

سرکار خانم لیلا کاخکی بیدختی

در چهل و هشتمین کنفرانس ریاضی ایران که از تاریخ ۳۱ مرداد ماه الی ۳ شهریور ماه سال ۱۳۹۶ در دانشگاه بوعلی سینا برگزار گردید شرکت کرده و پوستر تحت عنوان

Right-looking approximate

ارائه نموده اند.

دکتر اشرف دانشخواه
دبیر علمی کنفرانس

چهل و هشتمین کنفرانس ریاضی ایران
48 Annual Iranian MATHEMATICS Conference
Bu-Ali Sina University

A-10-550-3

نشانی دبیرخانه:
دانشگاه بوعلی سینا- دانشکده علوم پایه- گروه ریاضی
تلفکس: ۰۸۱-۳۸۳۸۱۴۲۶ کدپستی: ۶۵۱۷۸-۳۸۶۹۵

Department of Mathematics- Faculty of Sciences- Bu-Ali Sina University
Tel & Fax: +98 81 3838 1426 zip code: 65178-38695 Http://conf.ims.ir/aimc48 Email: aimc48@ims.ir