The 9th Seminar on Linear Algebra and its Applications
July 5-6, 2017, University of Tabriz, Tabriz, Iran

LEFT-LOOKING APPROXIMATE INVERSE PRECONDITIONER IN BLOCK FORM

AMIN RAFIEI AND SAMANEH HOSSEINI SANI

Abstract. In this paper we present a block version of left-looking AINV preconditioner. In the numerical tests, we compare the quality of the plain and block versions of this preconditioner.

Keywords: AINV preconditioner; GMRES method.

1. Introduction

Consider the linear system

$$Ax = b$$

where $A \in \mathbb{R}^{n\times n}$ is real, nonsymmetric and invertible matrix and $x, b \in \mathbb{R}^{n\times 1}$. The left-looking A-biconjugation algorithm can be used to factorize A^{-1} in the following form

$$A^{-1} = ZD^{-1}W^T.$$

In this factorization Z and W are unit upper triangular matrices and D is a diagonal matrix [1]. If we apply dropping in this algorithm, then the left-looking $AINV$ will be obtained. In [2], we have presented a block format of left-looking A-biconjugation algorithm. In this case, Z and W factors are agin unit upper triangular while D is a block diagonal matrix. The diagonal blocks of D are of order 1×1 or 2×2. Applying the dropping strategy in this block algorithm will give us a block version of left-looking $AINV$ preconditioner. In this paper, we will compare the quality of the block and plain left-looking $AINV$ preconditioners at reducing the number of iterations of the $GMRES$ [4] Krylov subspace method.

2. Algorithms of plain and block left-looking $AINV$ preconditioners

In this section, we have presented three algorithms. Algorithm 1 and 2 are used to compute the block version of left-looking $AINV$ preconditioner. We call Algorithm 2 inside Algorithm 1 to construct a column of matrices Z and W.

2010 *Mathematics Subject Classification.* 65F10, 65F08.
Speaker: Samaneh Hosseini Sani.

Algorithm 1 (A block format of left-looking AINV preconditioner)

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

$logic\_z = logic\_w = true$
$status(i) = 0, \quad 1 \le i \le n$
$Z = [z_1^{(0)}, z_2^{(0)}, ..., z_n^{(0)}] = I_{n\times n}, W = [w_1^{(0)}, w_2^{(0)}, ..., w_n^{(0)}] = I_{n\times n}, D = 0 \in \mathbb{R}^{n\times n}$
$i = 1$
while $i < n$ **do**
  if $logic\_z$ **then**
    $call\ column - const(Z, \tau_z, A, D, i, status)$
  else
    $logic\_z = true$
  end if
  call $column - const(Z, \tau_z, A, D, i+1, status)$
  $S_{ii}^{(i-1)} = e_i^T A z_i^{(i-1)}$
  for $j = i+1$ **to** n **do**
    $S_{ji}^{(i-1)} = e_j^T A z_i^{(i-1)}, \quad S_{j,i+1}^{(i-1)} = e_j^T A z_{i+1}^{(i-1)}$
  end for
  if $logic\_w$ **then**
    column-const$(W, \tau_w, A^T, D^T, i, status)$
  else
    $logic\_w = true$
  end if
  $call\ column - const(W, \tau_w, A^T, D^T, i+1, status)$
  $S_{ij}^{(i-1)} = (w_i^{(i-1)})^T A e_j, \quad j \ge i+1$
  $S_{i+1,j}^{(i-1)} = (w_{i+1}^{(i-1)})^T A e_j, \quad j \ge i+2$
  $v_i = max\{\frac{1}{|s_{ij}^{(i-1)}|} \sum_{j=i+1}^n |S_{ii}^{(i-1)}|, \ \frac{1}{|s_{ii}^{(i-1)}|} \sum_{j=i+1}^n |S_{ji}^{(i-1)}|$
  $w_i^1 = \sum_{j=i+2}^n || \begin{bmatrix} S_{ii}^{(i-1)} & S_{i,i+1}^{(i-1)} \\ S_{i+1,i}^{(i-1)} & S_{i+1,i+1}^{(i-1)} \end{bmatrix}^{-1} \begin{pmatrix} S_{ij}^{(i-1)} \\ S_{i+1,j}^{(i-1)} \end{pmatrix} ||_\infty$
  $w_i^2 = \sum_{j=i+2}^n || \begin{pmatrix} S_{ij}^{(i-1)} & S_{i+1,j}^{(i-1)} \end{pmatrix} \begin{bmatrix} S_{ii}^{(i-1)} & S_{i,i+1}^{(i-1)} \\ S_{i+1,i}^{(i-1)} & S_{i+1,i+1}^{(i-1)} \end{bmatrix}^{-1} ||_\infty$
  $w_i = max\{w_i^1, w_i^2\}$
  if $v_i < w_i$ **then**
    $D_{ii} = S_{ii}^{(i-1)}$
    $z_{i+1}^{(i)} = z_{i+1}^{(i-1)} - (\frac{e_i^T A z_{(i+1)}^{(i-1)}}{D_{ii}}) z_i^{(i-1)}$. For all $l \le i$ apply dropping rule to $z_{l,i+1}^{(i)}$ if its absolute value is less than τ_z
    $w_{i+1}^{(i)} = w_{i+1}^{(i-1)} - (\frac{(w_{i+1}^{(i-1)})^T A e_i}{D_{ii}}) w_i^{(i-1)}$. For all $l \le i$ apply dropping rule to $w_{l,i+1}^{(i)}$ if its absolute value is less than τ_w
    $logic\_z = false, \quad logic\_w = false$
    $status(i) = 1$
    $i = i+1$
  else
    $D_{i:i+1,i:i+1} = \begin{bmatrix} S_{ii}^{(i-1)} & S_{ii+1}^{(i-1)} \\ S_{i+1i}^{(i-1)} & S_{i+1i+1}^{(i-1)} \end{bmatrix}$
    $z_{i+1}^{(i)} = z_{i+1}^{(i-1)}, \quad w_{i+1}^{(i)} = w_{i+1}^{(i-1)}$
    $status(i) = 2$
    $i = i+2$
  end if
end while
if $status(n-1) = 0$ **then**
  call $column - const(Z, \tau_z, A, D, n, status)$
  $call\ column - const(W, \tau_w, A^T, D^T, n, status)$
  $D_{n,n} = e_n^T A z_n^{(n-1)}$
end if
if $status(n-1) = 1$ **then**
  $D_{n,n} = e_n^T A z_n^{(n-1)}$
end if
$Return\ Z = [z_1^{(0)}, z_2^{(1)}, ..., z_n^{(n-1)}], W = [w_1^{(0)}, w_2^{(1)}, ..., w_n^{(n-1)}]\ and\ D$

Algorithm 3 gives the plain left-looking $AINV$ preconditioner. This algorithm was first introduced by Benzi and Tuma in reference [1].

Algorithm 2 (**Column construction of a matrix**)

Column_ Const$(Z, \tau_z, A, D, i, status)$

Input: $Z = [z_1^{(0)}, z_2^{(1)}, \cdots, z_{i-1}^{(i-2)}, z_i^{(0)}, \cdots, z_n^{(0)}] \in \mathbb{R}^{n\times n}$, $\tau_z \in (0,1)$ is the drop tolerance for Z, $A \in \mathbb{R}^{n\times n}$ a nonsymmetric matrix, $D \in \mathbb{R}^{n\times n}$ a block diagonal matrix, i is an integer, $status$ is an integer array

Output: updated Z

$j = 1$
while $j \leq i-1$ **do**
  $k = j + status(j) - 1$
  if $status(j) = 1$ **then**
    $z_i^{(k)} = z_i^{(j-1)} - z_j^{(j-1)} \times \frac{1}{D_{jj}} \times A_{j,:} \times z_i^{(j-1)}$
    $j = j + 1$
  else if $status(j) = 2$ **then**
    $z_i^{(k)} = z_i^{(j-1)} - [z_j^{(j-1)}\ z_{j+1}^{(j)}] \times [D(j:j+1, j:j+1)]^{-1} \times A_{j:j+1,:} \times z_i^{(j-1)}$
    $j = j + 2$
  end if
  Consider $z_i^{(k)} = (z_{li}^{(k)})$. For all l, apply dropping rule to $z_{li}^{(k)}$ if its absolute value is less than τ_z
end while
Return Z

Algorithm 3 (**Left-looking** $AINV$ **preconditioner**)

Input: $A \in \mathbb{R}^{n\times n}$ a nonsymmetric matrix, $\tau_w, \tau_z \in (0,1)$ the drop tolerances for W and Z.

Output: $A^{-1} \approx ZD^{-1}W^T$.

$D = 0 \in \mathbb{R}^{n\times n}$
for $i = 1$ to n **do**
  $w_i^{(0)} = e_i$, $z_i^{(0)} = e_i$
  for $j = 1$ to $i-1$ **do**
    $w_i^{(j)} = w_i^{(j-1)} - (\frac{(w_i^{(j-1)})^T A e_j}{D_{jj}}) w_j^{(j-1)}$, $z_i^{(j)} = z_i^{(j-1)} - (\frac{e_j^T A z_i^{(j-1)}}{D_{jj}}) z_j^{(j-1)}$
    for all $l \leq j$ *apply a dropping rule to* $w_{li}^{(j)}$ *and to* $z_{li}^{(j)}$ *if their absolute values are less than* τ_w *and* τ_z.
  end for
  $D_{ii} = e_i^T A z_i^{(i-1)}$
end for
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 $D = diag(D_{ii})_{1\leq i\leq n}$.

3. Numerical tests

In this section, we have reported the results of numerical experiments.We have implemented 3 algorithms in Matlab. We have selected 4 matrices from [3]. Then, we constructed the artificial linear systems $A[1, \cdots, 1]^T = b$. These systems have been solved by the $GMRES(15)$ Krylov subspace method. The command $GMRES$ in Matlab provides us this method. The matrix information and the convergence results of the $GMRES(15)$ can be found in Table 1. In this table, n and nnz are the dimension and number of nonzero entries of the matrix. $iter(1)$ and $iter(2)$ are the number of external and internal iterations of $GMRES(15)$, respectively. $Time$ is the iteration time which is in seconds.

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

Matrix properties			GMRES(15)		
name	n	nnz	iter(1)	iter(2)	Time
sherman4	1104	3786	37	12	0.305
orsirr-2	886	5970	397	9	1.629
sherman1	1000	2375	132	15	0.559
cdde1	961	4681	9	2	0.035

In Table 2, we have set $\tau_z = \tau_w = 0.1$ and computed both the plain and the block left-looking $AINV$ preconditioners. The notations $LLAINV(0.1)$ and $BLLAINV(0.1)$ refer to these two cases. Then, these two preconditioners have been used as the right preconditioner for linear systems. After that, the preconditioned systems have been solved by the $GMRES(15)$ method. The resuls of these tests can be found in Table 2. In this table, $ptime$ 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, $iter(1)$ and $iter(2)$ have the same definition as in Table 1 and $Ttime$ is the summation of preconditioning time and the iteration time of the $GMRES(15)$ method.

TABLE 2. properties of the preconditioners and results of $GMRES(15)$ to solve the preconditioned systems

	LLAINV(0.1)+GMRES(15)					BLLAINV(0.1)+GMRES(15)				
matrix	ptime	density	iter(1)	iter(2)	Ttime	ptime	density	iter(1)	iter(2)	Ttime
sherman4	40.84	1.331	7	13	41.0025	5641.11	4.079	7	3	5641.44
orsirr-2	26.05	0.919	4	10	26.1109	3321.73	3.458	4	9	3321.79
sherman1	29.61	1.350	1	6	29.621	3589.79	1.551	1	6	3589.8
cdde1	34.41	1.558	20	14	34.7793	4550.61	15.341	9	1	4550.76

The results in Table 2 indicate that for matrices sherman4, orsirr-2 and cdde1, the block left-looking AINV preconditioner is more effective than the plain left-looking $AINV$ at reducing the number of iterations of $GMRES(15)$ method. For matrix sherman1, both preconditioners make the $GMRES(15)$ method convergent in the same number of iterations. Comparing the preconditioning time of both preconditioners show that the block left-looking $AINV$ needs more time to be constructed. By analyzing the number of iterations in Tables 1 and 2 one may come to conclusion that both preconditioners are effective tools at reducing the number of iterations of $GMRES(15)$ method.

REFERENCES

[1] M.Benzi and M.Tuma, A sparse approximate inverse preconditioner for nonsymmetric linear systems, SIAM J.Sci. Comput. 19(3), (1988) 968-994

[2] A.Rafiei, M.Bollhofer and T.Huckle, A block factorization format for the inverse of a nonsymmetric matrix , 47^{th} Annual Iranian Mathematics conference (AIMC47), Kharazmi University, (2016).

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

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

DEPARTMENT OF APPLIED MATHEMATICS, HAKIM SABZEVARI UNIVERSITY, IRAN
E-mail address: rafiei.am@gmail.com, a.rafiei@hsu.ac.ir

DEPARTMENT OF APPLIED MATHEMATICS, HAKIM SABZEVARI UNIVERSITY, IRAN
E-mail address: samanehhoseini@yahoo.com