Journal of Mathematical Modeling
Vol. 9, No. 3, 2021, pp. 347-359

JMM

A block preconditioner for the Gl-LSMR algorithm

Afsaneh Hasanpour, Maryam Mojarrab*

Department of Mathematics, University of Sistan and Baluchestan, Zahedan, Iran

Email(s): a_hasanpour@pgs.usb.ac.ir, ma_mojarrab@math.usb.ac.ir

Abstract. The global least squares minimal residual (Gl-LSMR) method is an efficient solver for linear systems with multiple right-hand sides. To accelerate the convergence of the Gl-LSMR method, we propose a block preconditioner for the global LSMR method which can be used for solving linear systems with a block partitioned coefficient matrix and multiple right-hand sides. Numerical examples and comparing the preconditioned Gl-LSMR method with the Gl-LSMR method validate the effectiveness of the preconditioner. Numerical results confirm that the Block Preconditioned Gl-LSMR (BPGLSMR) method has a better performance in reducing the number of iterations and CPU time.

Keywords: LSMR method, Gl-LSMR method, preconditioner, block partitioned matrices, multiple right-hand sides.
AMS Subject Classification 2010: 15A06, 65F10, 65F20.

1 Introduction

One of the main topics in matrix theory is matrix equations and play essential roles in many applications. Matrix equations appear in many problems, such as systems in control theory, image processing, and electromagnetic structure computation [3, 7, 31, 32, 39].

Consider the matrix equation of the form

$$AX = B, \tag{1}$$

where $A \in \mathbb{R}^{ms_1 \times ns_2}$ is a full column rank matrix with $ms_1 \geq ns_2$, $m, n, s_1, s_2 \in \mathbb{N}$, $X \in \mathbb{R}^{ns_2 \times l}$, and $B \in \mathbb{R}^{ms_1 \times l}$ are unknown and right-hand side matrices, respectively with usually $l \ll ms_1$. When the coefficient matrix A is large and sparse, we use iterative methods for solving the matrix equations. Especially those methods that are based on the generalization of the classical Krylov subspace.

The first class of iterative methods is the global methods. These methods are based on the use of a global projection onto a matrix Krylov subspace and they are appropriate for the

*Corresponding author.
Received: 15 September 2020 / Revised: 29 October 2020 / Accepted: 7 November 2020
DOI: 10.22124/jmm.2020.17687.1525

sparse multiple linear systems. Jbilou et al. introduced the global Lanczos-based method, the global full orthogonalization (Gl-FOM) and global generalized minimal residual (Gl-GMRES) methods [16,18]. The global Hessenberg (Gl-Hess) method and global changing minimal residual method based on the Hessenberg process (Gl-CMRH) were presented by Heyouni [14]. The global biconjugate gradient (Gl-BiCG) and global biconjugate gradient stabilized (Gl-BiCGStab) methods were obtained by Jbilou et al. for nonsymmetric coefficient matrix [17]. Also, the weighted technique was applied to speed-up the convergence of global methods [15]. The global least squares with QR factorization (GL-LSQR) was proposed by Toutunian and Karimi [34]. See [4, 13, 30, 40] for more global methods.

The second class is the seed methods in which first a single system will be selected as the seed system and developed the corresponding Krylov subspace. Then all the residuals of the other systems will be projected on to the same Krylov subspace for finding new approximations. Seed conjugate gradient method and seed GMRES method are introduced in [20, 31, 37]. Other references on these methods are included in [8, 28].

The third class is the block methods. Similar to the last two classes, the block methods are proposed to solve the linear systems with multiple right-hand sides. They are more appropriate when the coefficient matrix is relatively dense or a preconditioner is used. The first block iterative solver is the block conjugate gradient (Bl-CG) algorithm which is offered by ÓLeary for symmetric problems [27]. The algorithms of this method for parallel computers were presented in [26], and a breakdown-free block conjugate gradient method was presented in [19]. For nonsymmetric problems, the block generalized minimal residuals (Bl-GMRES) method [24, 29, 31, 33, 38], the block quasi minimal residual (Bl-QMR) method [10], the block BiCGStab (Bl-BiCGStab) algorithm [11], the block Lanczos method [12], the block LSQR (Bl-LSQR) method [21], and the block LSMR (Bl-LSMR) methods [35, 36] have been introduced.

The LSMR method was presented for solving the linear system $Ax = b$ and least squares problem $\min\|Ax - b\|_2$, such that A is sparse or a fast linear operator [9]. In this method two sets of vectors $v_1, v_2, \ldots, v_k$ and $u_1, u_2, \ldots, u_k$ which are made by Golub-Kahan bidiagonalization process, can build an orthogonal basis for Krylov subspace $\mathcal{K}_k(A^TA, v_1)$ and $\mathcal{K}_k(AA^T, u_1)$, respectively in which:

$$\mathcal{K}_k(A^TA, v_1) = span\{v_1, A^TAv_1, \ldots, (A^TA)^{k-1}v_1\},$$
$$\mathcal{K}_k(AA^T, u_1) = span\{u_1, AA^Tu_1, \ldots, (AA^T)^{k-1}u_1\}.$$

In Gl-LSMR [25], two sets of matrices $V_1, V_2, \ldots, V_k$ and $U_1, U_2, \ldots, U_k$ can make an F-orthonormal basis for block Krylov subspace $\mathbb{K}_k(A^TA, v_1)$ and $\mathbb{K}_k(AA^T, u_1)$, respectively as follows:

$$\mathbb{K}_k(A^TA, V_1) = span\{V_1, A^TAV_1, \ldots, (A^TA)^{k-1}V_1\},$$
$$\mathbb{K}_k(AA^T, U_1) = span\{U_1, AA^TU_1, \ldots, (AA^T)^{k-1}U_1\}.$$

The performance of Krylov subspace methods can be improved by using a suitable preconditioner or with efficient matrix splitting techniques [1, 2, 19]. Karimi [22] introduced a block preconditioner for the block partitioned matrices wherein the incomplete inverse factor $\hat{R}$ of A^TA is used as a right preconditioner for the GL-LSQR algorithm to solve the partitioned systems.

A block partitioned matrix is a matrix that is interpreted as having been broken into sections called blocks or submatrices. Any matrix may be interpreted as a block matrix in one or more ways, with each interpretation defined by how its rows and columns are partitioned. The matrix $A_{ms_1 \times ns_2}$ can be written as a block matrix as

$$A = \begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1s_2} \\ A_{21} & A_{22} & \cdots & A_{2s_2} \\ \vdots & \vdots & \ddots & \vdots \\ A_{s_1 1} & A_{s_1 2} & \cdots & A_{s_1 s_2} \end{pmatrix}.$$

where $A_{ij} \in \mathbb{R}^{m \times n}$.

To expedite the convergence of the Gl-LSMR method, we propose a block preconditioner for block partitioned matrices using the technique of [22]. This preconditioner is based on block C-orthogonalization where C is a symmetric positive definite matrix. First, the block preconditioned Gl-LSMR (BPGLSMR) algorithm is presented. Also, BPGLSMR and Gl-LSMR are applied to solve the block partitioned linear systems and the results are compared together.

Throughout the paper the following notations are used. The inner product $(X, Y)_F = trace(X^T Y)$ is used for two matrices X and $Y \in \mathbb{R}^{n \times s}$ and the associated norm is the Ferobenius norm defined by $\|X\|_F = \sqrt{(X, X)_F}$. The notation (X, Y) indicates Euclidean inner product in $\mathbb{R}^n$ and the associated norm is denoted by $\|.\|$. I_n demonstrates the identity matrix of order n. Also, e_i denotes the ith column of the identity matrix of a suitable size. The notation $*$ is used for the following products:

$$\mathcal{V}_k * \gamma = \sum_{j=1}^{k} V_j \gamma_j,$$
$$\mathcal{V}_k * T = (\mathcal{V}_k * T_{.,1}, \mathcal{V}_k * T_{.,2}, \ldots, \mathcal{V}_k * T_{.,k}),$$

where $\mathcal{V}_k = (V_1, V_2, \ldots, V_k)$ with $V_i \in \mathbb{R}^{n \times s}, i = 1, 2, \ldots, k$ and $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_k)^T \in \mathbb{R}^k$ is a vector. $T_{.,j}$ is jth column of the matrix $T \in \mathbb{R}^{k \times k}$.

The structure of the paper is as follows. There is a brief description of the LSMR and Gl-LSMR methods in Section 2. In Section 3, a block preconditioner for block partitioned matrices is introduced and the BPGLSMR algorithm is presented. In Section 4, some numerical examples are presented to show the efficiency of the BPGLSMR method. Conclusions are drawn in Section 5.

2 A useful summary of the Gl-LSMR algorithm

This section explains some propertices of Gl-LSMR [25]. In this method, A in (1) can be reduced to the lower bidiagonal form by using global Bidiag 1 [34]. The global Bidiag 1 procedure makes two sets of matrices $V_1, V_2, \ldots$ in $\mathbb{R}^{ns_2 \times l}$ and $U_1, U_2, \ldots$ are in $\mathbb{R}^{ms_1 \times l}$ such that

$$\langle V_i, V_j \rangle_F = 0, \quad \langle U_i, U_j \rangle_F = 0, \quad \text{for } i \neq j,$$

and the scalers $\alpha_i \geqslant 0$ and $\beta_i \geqslant 0$ are chosen so that $\| V_i \|_F = 1$, $\| U_i \|_F = 1$. This process can be described as follows

$$\begin{aligned} \beta_1 U_1 = B, \quad \alpha_1 V_1 = A^T U_1, \\ \beta_{i+1} U_{i+1} = AV_i - \alpha_i U_i, \\ \alpha_{i+1} V_{i+1} = A^T U_{i+1} - \beta_i V_i, \end{aligned} \quad i = 1, 2, \ldots. \tag{2}$$

Now, consider

$$\begin{aligned} \mathcal{U}_k &= (U_1, U_2, \ldots, U_k), \\ \mathcal{V}_k &= (V_1, V_2, \ldots, V_k), \\ T_k &= \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{pmatrix}. \end{aligned}$$

The recurrence relations (2) can be rewritten using the product $*$ as follows

$$\begin{aligned} &\mathcal{U}_{k+1} * (\beta_1 e_1) = B, \\ &A\mathcal{V}_k = \mathcal{U}_{k+1} * T_k, \\ &A^T \mathcal{U}_{k+1} = \mathcal{V}_k * T_k^T + \alpha_{k+1} V_{k+1} * e_{k+1}^T. \end{aligned}$$

Suppose that X_k is an approximate solution of the form

$$X_k = \mathcal{V}_k * y_k,$$

at iteration k, where $y_k \in \mathbb{R}^k$. If $R_k = B - AX_k$, we can write

$$\begin{aligned} A^T R_k &= A^T B - A^T A X_k \\ &= \alpha_1 \beta_1 V_1 - A^T A \mathcal{V}_k * y_k \\ &= \bar{\beta}_1 V_1 - \mathcal{V}_{k+1} * \begin{pmatrix} T_k^T T_k \\ \bar{\beta}_{k+1} e_k^T \end{pmatrix} y_k \\ &= \mathcal{V}_{k+1} * (\bar{\beta}_1 e_1 - \begin{pmatrix} T_k^T T_k \\ \bar{\beta}_{k+1} e_k^T \end{pmatrix} y_k), \end{aligned} \tag{3}$$

where $\bar{\beta}_k = \alpha_k \beta_k$. Therefore, from (3) we have

$$\min_{y \in \mathbb{R}^k} \|A^T R_k\|_F = \min_{y \in \mathbb{R}^k} \|\bar{\beta}_1 e_1 - \begin{pmatrix} T_k^T T_k \\ \bar{\beta}_{k+1} e_k^T \end{pmatrix} y_k\|.$$

The main steps of the Gl-LSMR algorithm can be summarized as follows.

Algorithm 1. Gl-LSMR algorithm

1. Set $X_0 = 0_{ns_2 \times l}$
2. $\beta_1 = \|B\|_F$, $U_1 = B/\beta_1$, $\alpha_1 = \|A^T U_1\|_F$, $V_1 = A^T U_1/\alpha_1$, $\bar{\beta}_1 = \alpha_1\beta_1$
3. Set $\bar{\zeta}_1 = \bar{\beta}_1$, $\hat{\bar{\rho}}_1 = \alpha_1$
4. Set $\bar{\rho}_0 = 1$, $\bar{c}_0 = 1$, $\bar{s}_0 = 0$, $\bar{\theta}_0 = \theta_1 = 0$, $P_{-1} = P_0 = 0_{ns_2 \times l}$
5. For $i = 1, 2, \ldots$ until convergence, Do
6. $\quad W_i = AV_i - \alpha_i U_i$
7. $\quad \beta_{i+1} = \| W_i \|_F$
8. $\quad U_{i+1} = W_i/\beta_{i+1}$
9. $\quad w_i = A^T U_{i+1} - \beta_{i+1} V_i$
10. $\quad \alpha_{i+1} = \| w_i \|_F$
11. $\quad V_{i+1} = w_i/\alpha_{i+1}$
12. $\quad \rho_{i+1} = (\hat{\bar{\rho}}_i^2 + \beta_{i+1}^2)^{1/2}$
13. $\quad c_i = \hat{\bar{\rho}}_i/\rho_i$
14. $\quad s_i = \beta_{i+1}/\rho_i$
15. $\quad \theta_{i+1} = s_i \alpha_{i+1}$
16. $\quad \hat{\bar{\rho}}_{i+1} = c_i \alpha_{i+1}$
17. $\quad \tilde{\bar{\rho}}_i = \bar{c}_{i-1}\rho_i$
18. $\quad \bar{\rho}_i = (\tilde{\bar{\rho}}_i^2 + \theta_{i+1}^2)^{1/2}$
19. $\quad \bar{c}_i = \tilde{\bar{\rho}}_i/\bar{\rho}_i$
20. $\quad \bar{s}_i = \theta_{i+1}/\bar{\rho}_i$
21. $\quad \bar{\theta}_i = -\bar{s}_{i-1}\rho_i$
22. $\quad \zeta_i = \bar{c}_i \bar{\zeta}_i$
23. $\quad \bar{\zeta}_{i+1} = -\bar{s}_i \bar{\zeta}_i$
24. $\quad P_i = (V_i - \bar{\theta}_{i-1}\theta_i P_{i-2} - (\bar{\rho}_{i-1}\theta_i - \bar{\theta}_i\rho_i)P_{i-1})/\rho_i\bar{\rho}_i$
25. $\quad X_i = X_{i-1} - \zeta_i P_i$
26. $\quad$ If $|\ \bar{\zeta}_{i+1}\ |$ is small enough then stop
27. End Do

3 Construction of block preconditioner of Gl-LSMR

Iterative methods that are using for solving a large and sparse linear systems of equations have sufficient accuracy in many cases, but not always. The LSMR and Gl-LSMR methods are in this group and sometimes may fail or have a low convergence rate. In this situation, an acceleration technique should be demanded to remedy this drawback or to speed-up the convergence rate. A way is using preconditioners to help the iterative method.

In this section, we try to present an efficient block preconditioner by using the ideas [5,6,23]. We assume that the coefficient matrix is block partitioned of size $ms_1 \times ns_2$ which each block is an $m \times n$ matrix.

We present a block upper triangular matrix R as a right preconditioner for the Gl-LSMR method which is based on the C-matrix product that is defined as follows:

$$(X, Y)_C = Y^T C X,$$

where C is a symmetric positive definite matrix and $X \in \mathbb{R}^{ns_2 \times k_1}$, and $Y \in \mathbb{R}^{ns_2 \times k_2}$.

Unit basis matrices $E_1, E_2, \ldots, E_{s_2} \in \mathbb{R}^{ns_2 \times n}$ can construct a block set of matrices $Z_1, Z_2, \ldots, Z_{s_2} \in \mathbb{R}^{ns_2 \times n}$ by using block conjugate Gram-Schmidt using the to C-matrix product, where $E_j = e_j \otimes I_n$ and e_j is jth column of I_{s_2}.

The algorithm of block C-orthogonalization starts from $Z_j = E_j$ for $j = 1, 2, \ldots, s_2$. Then the following nested loop will be performed.

$$Z_i \leftarrow Z_i - Z_j[(Z_j, Z_j)_C]^{-1}(Z_i, Z_j)_C. \tag{4}$$

Since A is a full rank matrix, A^TA is a symmetric positive definite matrix. We set $C = A^TA$.

Remark 1. *[22] Assume that X and C are full rank and symmetric positive definite matrices respectively. Then $(X, X)_C$ is nonsingular.*

By using $Z = (Z_1, Z_2, \ldots, Z_{s_2})$ and $D = Z^TCZ$ the inverse upper-lower triangular factorization $(A^TA)^{-1} = ZD^{-1}Z^T$ is obtained. The matrix R can be defined by using SPD block diagonal matrix D as $R = ZD^{-1/2}$. Hence, we obtain the inverse upper-lower triangular factorization $(A^TA)^{-1} = RR^T$. A dropping tolerance $0 < \tau < 1$ can be used for entries of Z_i in (4) such that they are scanned in each update and will be discarded if they are smaller than τ. After this discarding the new sparsified $\hat{Z}_i$ will be defined as follows:

$$\hat{Z} = (\hat{Z}_1, \hat{Z}_2, \ldots, \hat{Z}_{s_2}), \qquad \hat{D} = (\hat{Z}, \hat{Z})_C,$$

and according to this definition, $\hat{R} = \hat{Z}\hat{D}^{-1/2}$ will be the incomplete inverse factor of A^TA.

The LSMR and Gl-LSMR methods search for the approximation of solution in Krylov subspaces $\mathcal{K}_k(A^TA, v_1)$ and $\mathbb{K}_k(A^TA, V_1)$, respectively. Thus the incomplete inverse factor $\hat{R}$ can be the right preconditioner for these methods.

Remark 2. *Assume that $\hat{R} \in \mathbb{R}^{ns_2 \times ns_2}$ is the approximate inverse factor of A^TA. Then $(A\hat{R})^T(A\hat{R}) \approx I_{ns_2}$.*

Proof. According to $(A^TA)^{-1} \approx \hat{R}\hat{R}^T$, we have $A^TA \approx (\hat{R}\hat{R}^T)^{-1}$. So

$$\begin{aligned}(A\hat{R})^T(A\hat{R}) &= \hat{R}^TA^TA\hat{R} \\ &\approx \hat{R}^T\hat{R}^{-T}\hat{R}^{-1}\hat{R} \\ &= I_{ns_2},\end{aligned}$$

which completes the proof. □

Accordingly, the block C-orthogonalization algorithm may be briefed as follows.

Algorithm 2. Block C-orthogonalization

1. Let $Z_j = E_j$, $j = 1, 2, \ldots, s_2$
2. For $j = 1, 2, \ldots, s_2 - 1$
3. For $i = j + 1, j + 2, \ldots, s_2$ Do
4. $Z_i = Z_i - Z_j[(Z_j, Z_j)_C]^{-1}(Z_i, Z_j)_C$
5. Use a dropping strategy for the elements of matrix Z_i
6. EndDo
7. EndDo

Besides, we can summarize the block preconditioned Gl-LSMR algorithm as follows.

Algorithm 3. BPGLSMR algorithm

1. Set $Y_0 = 0_{ns_2 \times l}$
2. $\beta_1 = \|B\|_F$, $U_1 = B/\beta_1$, $Q_1 = A^T U_1$, $\alpha_1 = \|\hat{R}^T Q_1\|_F$, $V_1 = \hat{R}^T Q_1/\alpha_1$, $\bar{\beta}_1 = \alpha_1\beta_1$
3. Set $\bar{\zeta}_1 = \bar{\beta}_1$, $\hat{\rho}_1 = \alpha_1$
4. Set $\bar{\rho}_0 = 1$, $\bar{c}_0 = 1$, $\bar{s}_0 = 0$, $\bar{\theta}_0 = \theta_1 = 0$, $P_{-1} = P_0 = 0_{ns_2 \times l}$
5. For $i = 1, 2, \ldots$ until convergence, Do
6. $\quad D_i = \hat{R}V_i$
7. $\quad W_i = AD_i - \alpha_i U_i$
8. $\quad \beta_{i+1} = \| W_i \|_F$
9. $\quad U_{i+1} = W_i/\beta_{i+1}$
10. $\quad Q_{i+1} = A^T U_{i+1}$
11. $\quad \bar{S}_i = \hat{R}^T Q_{i+1} - \beta_{i+1} V_i$
12. $\quad \alpha_{i+1} = \| \bar{S}_i \|_F$
13. $\quad V_{i+1} = \bar{S}_i/\alpha_{i+1}$
14. $\quad \rho_{i+1} = (\hat{\rho}_i^2 + \beta_{i+1}^2)^{1/2}$
15. $\quad c_i = \hat{\rho}_i/\rho_i$
16. $\quad s_i = \beta_{i+1}/\rho_i$
17. $\quad \theta_{i+1} = s_i\alpha_{i+1}$
18. $\quad \hat{\rho}_{i+1} = c_i\alpha_{i+1}$
19. $\quad \tilde{\bar{\rho}}_i = \bar{c}_{i-1}\rho_i$
20. $\quad \bar{\rho}_i = (\tilde{\bar{\rho}}_i^2 + \theta_{i+1}^2)^{1/2}$
21. $\quad \bar{c}_i = \tilde{\bar{\rho}}_i/\bar{\rho}_i$
22. $\quad \bar{s}_i = \theta_{i+1}/\bar{\rho}_i$
23. $\quad \bar{\theta}_i = -\bar{s}_{i-1}\rho_i$
24. $\quad \zeta_i = \bar{c}_i\bar{\zeta}_i$
25. $\quad \bar{\zeta}_{i+1} = -\bar{s}_i\bar{\zeta}_i$
26. $\quad P_i = (V_i - \bar{\theta}_{i-1}\theta_i P_{i-2} - (\bar{\rho}_{i-1}\theta_i - \bar{\theta}_i\rho_i)P_{i-1})/\rho_i\bar{\rho}_i$
27. $\quad Y_i = Y_{i-1} - \zeta_i P_i$
28. $\quad$ If $|\bar{\zeta}_{i+1}|$ is small enough then $X_i = \hat{R}Y_i$
29. End Do

4 Numerical examples

In this section, we present some numerical experiments to show the effectiveness of the block preconditioned Gl-LSMR method. In all the examples the starting guess is considered as $X_0 = 0$ with suitable size. The right-hand side matrix $B \in \mathbb{R}^{ms_1 \times l}$ is chosen such that the exact solution of (1) will be a matrix of size $ns_2 \times l$ that all its entries are equal to 1, except Example 2. We consider $l = 10$ in all the examples except Example 4. We compare the performance of BPGLSMR with Gl-LSMR. We set $\tau = 10^{-2}$. The stopping criterion $\| R_k \|_F \leq 10^{-8} \| R_0 \|_F$ is used and a maximum of 10000 iterations is allowed and if the methods need more than 10000 iterations it is shown by †. Since the block C-orthogonalization process is independent of the selected value for s_1(the number of partitions in the rows of a matrix) thus in all implementations,

we set $s_1 = 1$. Also, "$bl-size$" demonstrates the number of partitions in the columns of a matrix which will actually be the same amount of s_2. All the examples were executed in double precision in MATLAB R2014a. Also, the CPU time was reported in seconds.

Example 1. The coefficient matrices in this example are given from the Suite Sparse Matrix Collection (https://sparse.tamu.edu/). Table 1 contains the properties of matrices that we use in this example. "Matrix", "row", and "column" denote the matrix name, the number of rows, and the number of columns, respectively. The column "nnz" shows the number of nonzero elements of the matrix. "*cond*" demonstrates the condition number of the matrix, $\text{cond}(A) = \|A\|.\|A^{-1}\|$. The matrix extraction field has also been shown by "Matrix Discipline". In this example, the CPU time for both methods is computed. Table 2 denotes the number of iteration (Iter), CPU time (time), and relative residual norm (Rrn) of the BPGLSMR and Gl-LSMR methods. The results in Table 2 show that BPGLSMR outperforms Gl-LSMR in reducing the number of iterations and CPU time.

Table 1: Test problems information for Example 1.

Matrix	row	column	nnz	cond	Matrix Discipline
add32	4960	4960	23884	2.14e+02	Electronic circuit design
abtaha2	37932	331	137228	Not available	Combinatorial Problem
bfw782a	782	782	7514	4.6e+03	Electrical engineering
cdde6	961	961	4681	5e+02	Computational fluid dynamics
pde2961	2961	2961	14585	9.49e+02	Partial differential equations
sherman1	1000	1000	3750	2.3e+04	Oil reservoir modeling
sherman3	5005	5005	20033	6.9e+16	Oil reservoir modeling
sherman4	1104	1104	3786	7.2e+03	Oil reservoir modeling
sherman5	3312	3312	20793	3.9e+05	Oil reservoir modeling
well1033	1033	320	4732	Not available	Surveying
illc1850	1850	712	4732	Not available	Surveying

Example 2. [29] In this example, the coefficient matrix A of the linear system (1) is obtained from discretization of the Laplace operator

$$\mathcal{L}u = u_{xx} + u_{yy},$$

on the unit square $(0,1)\times(0,1)$ with $u = 0$ on the boundary. The discritization can be performed through the centered difference scheme at the grid points (x_i, y_j) where $x_i = ih$ and $y_j = jh$ for $i, j = 1, 2, \ldots, N+1$ with the mesh size $h = 1/(N+1)$ that yields a block tridiagonal matrix of size $n = N^2$. We consider the right-hand sides matrix B as $B = rand(n, l)$ with elements uniformly distributed in $[0, 1]$. Figure 1 indicates the performance of the Gl-LSMR, BPGLSMR, GL-LSQR, and the preconditioned Gl-LSQR (BPGLSQR) methods in reducing the relative residual norm of $AX = B$. As we observe BPGLSQR and BPGLSMR decrease the relative residual norm in a fewer number of iterations compared to unpreconditioned versions. Figure 1 confirms the effectiveness of the preconditioned methods.

Table 2: Iteration number, CPU time, and relative residual norm of BPGLSMR and Gl-LSMR for Example 1.

		BPGLSMR			Gl-LSMR		
Matrix	bl-size	Iter	Rrn	time	Iter	Rrn	time
add32	496	488	9.05e-09	21.43	633	9.58e-09	27.11
abtaha2	331	30	6.41e-09	0.92	46	7.09e-09	1.23
bfw782a	34	1912	9.81e-09	2.12	2705	9.26e-09	4.01
cdde6	31	30	8.10e-09	0.06	360	9.88e-09	0.74
pde2961	423	752	9.97e-09	11.28	2138	9.70e-09	32.05
sherman1	200	1002	9.94e-09	9.86	3145	9.84e-09	15.04
sherman3	1001	4567	9.71e-09	18.04	9271	9.01e-09	40.01
sherman4	138	394	9.52e-09	0.85	961	9.19e-09	2.17
sherman5	828	897	9.63e-09	11.71	4817	9.14e-09	22.51
well1033	5	130	7.94e-09	0.07	175	6.12e-09	0.21
illc1850	89	2157	9.95e-09	6.16	2175	9.52e-09	9.16

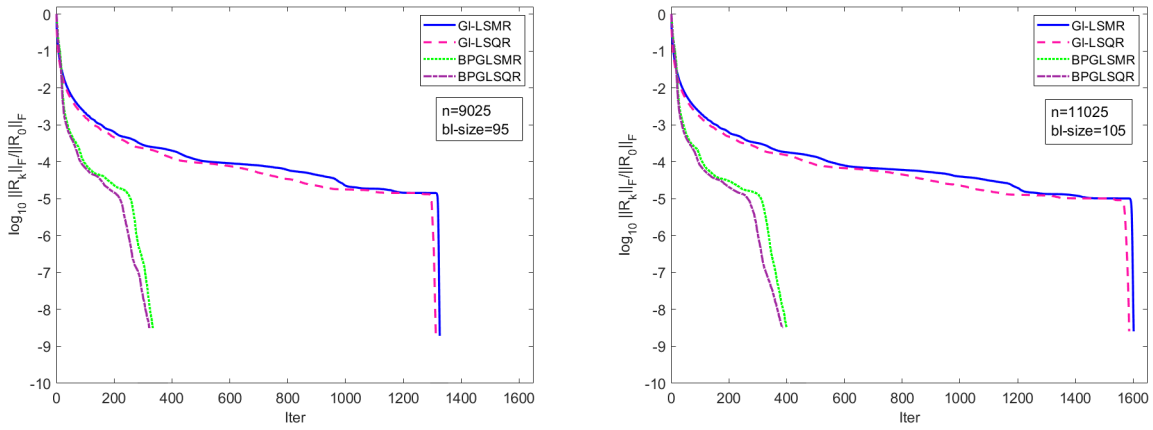

Figure 1: Performance of Gl-LSMR, Gl-LSQR, BPGLSMR, and BPGLSQR in reducing residual norm for Example 2.

Example 3. [1] Consider the three-dimensional operator

$$\mathcal{T}u = -(u_{xx} + u_{yy} + u_{zz}) + q(u_x + u_y + u_z) \tag{5}$$

on the unit cube $\Omega = [0,1] \times [0,1] \times [0,1]$. The coefficient matrix of linear system (1) obtains from the discretization of (5) that subjects to Dirichlet-type boundary conditions. The size of this matrix is $n = N^3$ where the mesh size is $h = 1/(N+1)$. Here q is a constant coefficient and in this example, we consider $q = 0.1$. This operator can be discretized by applying the seven-point finite difference discretizations. The centered difference is applied to the first three terms and the first order upwind approximation is applied to the second three terms. See [1] for more details. Figure 2 shows the behavior of BPGLSMR and Gl-LSMR in reducing $\| R_k \|_F / \| R_0 \|_F$

and $\| A^T R_k \|_F / \| A^T R_0 \|_F$. As we observe that both factors in BPGLSMR are reduced faster than Gl-LSMR.

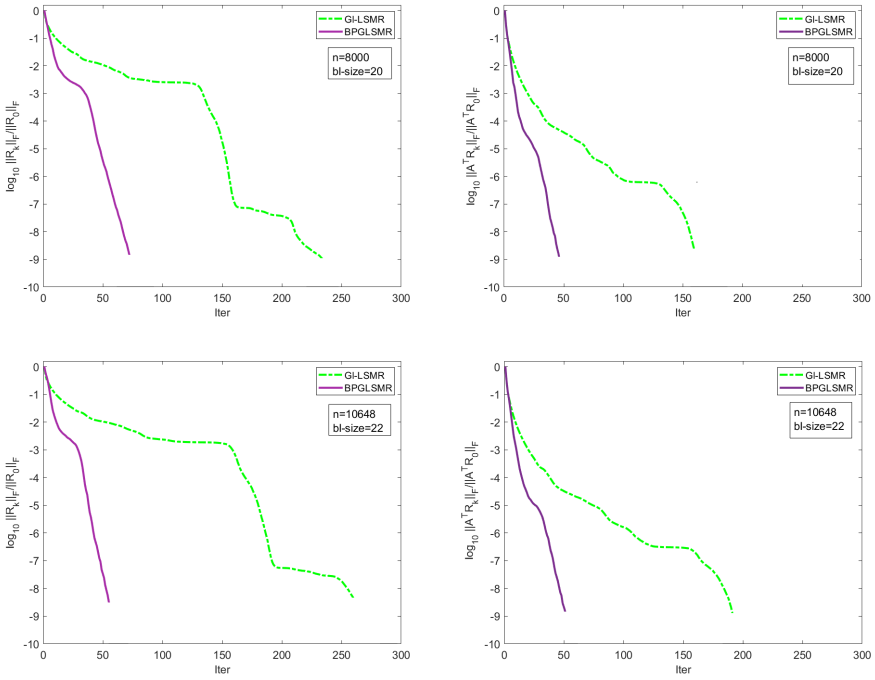

Figure 2: Performance of BPGLSMR and Gl-LSMR in reducing relative residual norm for Example 3 (the left figures show $log_{10}(\| R_k \|_F / \| R_0 \|_F)$ versus the number of iterations and the right figures show $log_{10}(\| A^T R_k \|_F / \| A^T R_0 \|_F)$ versus the number of iterations)

Example 4. In this example the coefficient matrix of (1) is a block partitioned matrix $A \in \mathbb{R}^{ns \times ns}$ which is defined by

$$A = \begin{pmatrix} A_2 & -A_1 & 0 & 0 \\ -A_1 & A_2 & -A_1 & 0 \\ 0 & -A_1 & A_2 & -A_1 \\ 0 & 0 & -A_1 & A_2 \end{pmatrix}, \tag{6}$$

where $A_1 = \text{tridiagonal } (-1, 2, -1) \in \mathbb{R}^{n \times n}$ and $A_2 = \text{tridiagonal } (-2, 3, -2) \in \mathbb{R}^{n \times n}$ and $s = 4$. In this example $B \in \mathbb{R}^{ns \times l}$, where $l = 20$. Table 3 denotes the number of iteration (Iter) and CPU time (time) in seconds for the BPGLSMR and Gl-LSMR methods. The results in Table 3 cinfirm the excellence of BPGLSMR in reducing the number of iterations and CPU time compared to Gl-LSMR.

5 Conclusion

In this study, we proposed a right block preconditioner for the global version of LSMR for solving general linear systems with multiple right-hand sides. This preconditioner is based on

Table 3: The number of iterations and CPU time of BPGLSMR and Gl-LSMR for Example 4.

		BPGLSMR		Gl-LSMR	
order	bl-size	Iter	time	Iter	time
4000	1000	2729	78.39	8315	224.25
8000	2000	4225	547.25	†	-
12000	3000	6252	1706.81	†	-

block C-orthogonalization process such that C is a SPD matrix. For comparison, some numerical examples were implemented by the BPGLSMR, BPGLSQR, Gl-LSMR, and GL-LSQR methods. We have seen that the preconditioned versions were more effective than the standard methods in reducing the number of iterations and CPU time.

References

[1] Z.-Z. Bai, G.H. Golub, M.K. Ng, *Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems*, SIAM. J. Matrix Anal. Appl. **24** (2003) 603-626.

[2] Z.-Z. Bai, G.H. Golub, L.-Z. Lu, J.-F. Yin, *Block triangular and skew-Hermitian splitting methods for positive-definite linear systems*, SIAM J. Sci. Comput. **26** (2005) 844-863.

[3] U. Baur, P. Benner, *Cross-gramian based model reduction for data-sparse systems*, Election. Trans. Numer. Anal. **31** (2008) 256-270.

[4] M. Bellalij, K. Jbilou, H. Sadok, *New convergence results on the global GMRES method for diagonalizable matrices*, J. Comput. Appl. Math. **219** (2008) 350-358.

[5] M. Benzi, R. Kouhia, M. Tuma, *Stabilized and block approximate inverse preconditioners for problems in solid and structural mechanics*, Comput. Methods Appl. Mech. Engrg. **190** (2001) 6533-6554.

[6] M. Benzi, M. Tuma, *Robust preconditioner with low memory requirement for large sparse least squares problems*, SIAM J. Sci. Comput. **25** (2003) 499-512

[7] D. Calvetti, L. Reichel, *Application of ADI iterative methods to the restoration of noisy images*, SIAM J. Matrix Anal. Appl. **17** (1996) 165-186.

[8] T.F. Chan, W.L. Wan, *Analysis of projection methods for solving linear systems with multiple right-hand sides*, SIAM J. Sci. Comput. **18** (1997) 1698-1721.

[9] D.C. Fong, M. Saunders, *LSMR: an iterative algorithm for sparse least-squares problems*, SIAM J. Sci. Comput. **33** (2011) 2950-2971.

[10] R.W. Freund, M. Malhotra, *A block QMR algorithm for non-Hermitian systems with multiple right-hand sides*, Linear Algebra Appl. **254** (1997) 119-157.

[11] A. El Guennouni, K. Jbilou , H. Sadok, *A block version of BICGSTAB for linear systems with multiple right-hand sides*, Electron. Trans. Numer. Anal. **16** (2003) 129-142.

[12] A. El Guennouni, K. Jbilou, H. Sadok, *The block Lanczos method for linear systems with multiple right-hand sides*, Appl. Numer. Math. **51** (2004) 243-256.

[13] C. Gu, Z. Yang, *Global SCD algorithm for real positive definite linear systems with multiple right-hand sides*, Appl. Math. Comput. **189** (2007) 59-67.

[14] M. Heyouni, *The global Hessenberg and global CMRH methods for linear systems with multiple right-hand sides*, Numer. Algorithms **26** (2001) 317-332.

[15] M. Heyouni, A. Essai, *Matrix Krylov subspace methods for linear systems with multiple right-hand sides*, Numer. Algorithms **40** (2005) 137-156.

[16] K. Jbilou, H. Sadok, *Global Lanczos-based methods with applications*, Tech. Rep. LMA 42, Université du Littoral, Calais, France, 1997.

[17] K. Jbilou, H. Sadok, A. Tinzefte, *Oblique projection methods for linear systems with multiple right-hand sides*, Electron. Trans. Numer. Anal. **20** (2005) 119-138.

[18] K. Jbilou, A. Messaoudi, H. Sadok, *Global FOM and GMRES algorithms for matrix equations*, Appl. Numer. Math. **31** (1999) 49-63.

[19] H. Ji, Y. Li, *A breakdown-free block conjugate gradient method*, BIT Numer. Math. **57** (2017) 379-403.

[20] P. Joly, *Résolution de systems linéaires avec plusieurs second members par la methode du gradient conjugue*, Tech. Rep. R-91012, Publications du Laboratire d'Analyse Numérique, Université Pierre et Marie Curie, Paris, 1991.

[21] S. Karimi, F. Toutounian, *The block least squares method for solving nonsymmetric linear systems with multiple right-hand sides*, Appl. Math. Comput. **177** (2006) 852-862.

[22] S. Karimi, B. Zali, *The block preconditioned LSQR and GL-LSQR algorithms for the block partitioned matrices*, Appl. Math. Comput. **227** (2014) 811-820.

[23] S. Karimi, F. Toutounian, D.K. Salkuyeh, *A preconditioner for the LSQR algorithm*, Appl. Math. Comput. **26** (2008) 213-222.

[24] H-L. Liu, B-J. Zhong, *Simpler block GMRES for nonsymmetric systems with multiple right-hand sides*, Electron. Trans. Numer. Anal. **30** (2008) 1-9.

[25] M. Mojarrab, F. Toutounian, *Global LSMR(Gl-LSMR) method for solving general linear systems with several right-hand sides*, J. Comput. Appl. Math. **321** (2017) 78-89.

[26] A.A. Nikishin, A.Y. Yeremin, *Variable block CG algorithms for solving large sparse symmetric positive definite linear systems on parallel computers, I: general iterative scheme*, SIAM J. Matrix Anal. Appl. **16** (1995) 1135-1153.

[27] D. ÓLeary, *The block conjugate gradient algorithm and related methods*, Linear Algebra Appl. **29** (1980) 293-322.

[28] Y. Saad, *On the Lanczos method for solving symmetric linear systems with several right-hand sides*, Math. Comp. **48** (1987) 651-662.

[29] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS press, New York, 1995.

[30] D.K. Salkuyeh, *CG-type algorithms to solve symmetric matrix equations*, Appl. Math. Comput. **172** (2006) 985-999.

[31] V. Simoncini, E. Gallopoulos, *An iterative method for nonsymmetric systems with multiple right-hand sides*, SIAM J. Sci. Comput. **16** (1995) 917-933.

[32] V. Simoncini, E. Gallopoulos, *A hybrid block GMRES method for nonsymmetric systems with multiple right-hand sides*, J. Comput. Appl. Math. **66** (1996) 457-469.

[33] V. Simoncini, E. Gallopoulos, *Convergence properties of block GMRES and matrix polynomials*, Linear Algebra Appl. **247** (1996) 97-119.

[34] F. Toutounian, S. Karimi, *Global least squares method (Gl-LSQR) for solving general linear systems with several right-hand sides*, Appl. Math. Comput. **178** (2006) 452-460.

[35] F. Toutounian, M. Mojarrab, *The block LSMR method :a novel efficient algorithm for solving non-symmetric linear systems with multiple right-hand sides*, Iran. J. Sci. Technol. Trans. A Sci **39** (2015) 69-78.

[36] F. Toutounian, M. Mojarrab, *The block LSMR algorithm for solving linear systems with multiple right-hand sides*. Iran. J. Numer. Anal. Optim. **5** (2015) 11-28.

[37] H. Van Der Vorst, *An iterative solution method for solving $f(A)x = b$, using Krylov subspace information obtained for the symmetric positive definite matrix A*, J. Comput. Appl. Math. **18** (1987) 249-263.

[38] B. Vital, *Etude de quelques mthodes de rsolution de problmes linaires de grande taille sur multiprocesseur,* Ph.D. thesis, University of Rennes, 1990.

[39] Q. Zhang, J. Liu, *Method of solving matrix equation and its applications in economic management*. In: Wang H., Shen Y., Huang T., Zeng Z. (eds) The Sixth International Symposium on Neural Networks (ISNN 2009). Adv. Intell. Syst. Comput. Springer, Berlin, Heidelberg. **56** (2009) 137-141

[40] J. Zhang, H. Dai, *Global CGS algorithm for linear systems with multiple right-hand sides*, Numer. Math. J. Chinese Univ. **30** (2008) 390-399.