Available online at www.sciencedirect.com
SCIENCE DIRECT®
Applied Mathematics and Computation 178 (2006) 452–460
APPLIED MATHEMATICS AND COMPUTATION
www.elsevier.com/locate/amc

Global least squares method (Gl-LSQR) for solving general linear systems with several right-hand sides

F. Toutounian, S. Karimi *

Department of Mathematics, Ferdowsi University of Mashhad, P.O. Box 1159-91775, Mashhad, Iran

Abstract

In this paper, we propose a new method for solving general linear systems with several right-hand sides. This method is based on global least squares method and reduces the original matrix to the lower bidiagonal form. We derive a simple recurrence formula for generating the sequence of approximate solutions $\{X_k\}$. Some theoretical properties of the new method are discussed and we also show that how this method can be implemented for the sylvester equation. Finally, some numerical experiments on test matrices are presented to show the efficiency of the new method.

Keywords: LSQR method; Bidiagonalization; Global methods; Iterative methods; Multiple right-hand sides

1. Introduction

Many applications require the solution of several sparse systems of equations

$$Ax^{(i)} = b^{(i)}, \quad i = 1, 2, \ldots, s \tag{1}$$

with the same coefficient matrix and different right-hand sides. When all the $b^{(i)}$'s are available simultaneously, Eq. (1) can be written as

$$AX = B, \tag{2}$$

where A is an $n \times n$ nonsingular and nonsymmetric real matrix, B and X are $n \times s$ rectangular matrices whose columns are $b^{(1)}, b^{(2)}, \ldots, b^{(s)}$ and $x^{(1)}, x^{(2)}, \ldots, x^{(s)}$, respectively.

In practice, s is of moderate size $s \ll n$. Instead of applying an iterative method to each linear system, it is more efficient to use a method for all the systems simultaneously. In the last years, generalizations of the classical Krylov subspace methods have been developed. The first class of these methods contains the block solvers such as the BL-BCG algorithm [2,11], the BL-GMRES algorithm introduced in [17], the BL-QMR algorithm [5], and recently the BL-BiCGSTAB algorithm [4]. The second class contains the global GMRES

* Corresponding author.
E-mail addresses: Toutounian@math.um.ac.ir (F. Toutounian), Karimi@math.um.ac.ir (S. Karimi).

0096-3003/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.amc.2005.11.065

[8] algorithm and global Lanczos-based methods [9]. A third class of methods use a single linear system named the seed system and then consider the corresponding Krylov subspace. The initial residuals of the other linear systems are projected onto this Krylov subspace. The process is repeated until convergence; see [13,1,10,15,16].

In this paper, we present a global version of least squares (LSQR) algorithm [12] for solving the problem (2). The algorithm will be derived from the LSQR algorithm in the same way as global FOM and GMRES methods were derived from FOM and GMRES algorithms, respectively [8]. The global FOM and GMRES algorithms reduce the coefficient matrix A to an upper Hesenberge matrix form. In new method, the coefficient matrix A will be reduced to a lower bidiagonal matrix form. This leads to a simple recurrence formula for generating the sequence of approximate solutions $\{X_k\}$. Our algorithm has certain advantages over global FOM and GMRES algorithms; namely we do not need to store the basis vectors, we do not need to predetermine a subspace dimension, and the approximate solutions and residuals are cheaply computed at every stage of the algorithm because they are updated with short-term recurrences.

Throughout this paper, we use the following notations. For two $n \times s$ matrices X and Y, we define the following inner product: $\langle X, Y\rangle_F = \mathrm{tr}(X^{\mathrm{T}}Y)$, where $\mathrm{tr}(Z)$ denotes the trace of the square matrix Z. The associated norm is the Frobenius norm defined by $\|X\|_F = (\langle X, X\rangle_F)^{1/2}$. We will use the notation $\langle\cdot\rangle_2$ for the usual inner product in $\mathbb{R}^n$ and the associated norm denoted by $\|\cdot\|_2$.

The outline of this paper is as follows. In Section 2, we give a quick overview of LSQR method and its properties. In Section 3, we present the global version of the LSQR algorithm and some properties. In Section 3.2, we will show that how this method can be implemented for the sylvester equation. In Section 4, some numerical experiments on test matrices from application problems and Harwell–Boeing collection are presented to show the efficiency of the method. Finally, we make some concluding remarks in Section 5.

2. The LSQR algorithm

In this section, we recall some fundamental properties of LSQR algorithm [12], which is an iterative method for solving real linear systems of the form

$$Ax = b,$$

where A is a nonsymmetric matrix of order n and $x, b \in \mathbb{R}^n$.

LSQR algorithm uses an algorithm of Golub and Kahan [6], which stated as procedure Bidiag 1 in [12], to reduce A to the lower bidiagonal form. The procedure Bidiag 1 can be described as follows.

Bidiag 1 (*starting vector b*; *reduction to lower bidiagonal form*)

$$\left.\begin{aligned} &\beta_1 u_1 = b, \quad \alpha_1 v_1 = A^{\mathrm{T}} u_1,\\ &\beta_{i+1} u_{i+1} = A v_i - \alpha_i u_i,\\ &\alpha_{i+1} v_{i+1} = A^{\mathrm{T}} u_{i+1} - \beta_{i+1} v_i, \end{aligned}\right\} \quad i = 1, 2, \ldots \tag{3}$$

The scalars $\alpha_i \geqslant 0$ and $\beta_i \geqslant 0$ are chosen so that $\|u_i\|_2 = \|v_i\|_2 = 1$. With the definitions

$$U_k \equiv [u_1, u_2, \ldots, u_k], \quad V_k \equiv [v_1, v_2, \ldots, v_k], \quad B_k \equiv \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{bmatrix}$$

the recurrence relations (3) may be rewritten as

$$\begin{aligned} &U_{k+1}(\beta_1 e_1) = b,\\ &AV_k = U_{k+1} B_k,\\ &A^{\mathrm{T}} U_{k+1} = V_k B_k^{\mathrm{T}} + \alpha_{k+1} v_{k+1} e_{k+1}^{\mathrm{T}}. \end{aligned}$$

As we observe the procedure Bidiag 1 will be stop if $Av_i - \alpha_i u_i = 0$ or $A^{\mathrm{T}} u_{i+1} - \beta_{i+1} v_i = 0$, for some i.

For the procedure Bidiag 1, we have the following propositions.

Proposition 1. *Suppose that k step of the procedure Bidiag 1 have been taken, then the vectors* $v_1, v_2, \ldots, v_k$, *and* $u_1, u_2, \ldots, u_k, u_{k+1}$ *are orthonormal basis of the Krylov subspaces* $\mathcal{K}_k(A^{\mathrm{T}}A, v_1)$ *and* $\mathcal{K}_{k+1}(AA^{\mathrm{T}}, u_1)$, *respectively.*

Proposition 2. *The procedure Bidiag 1 will stop at step m if and only if* $\min\{\mu, \lambda\}$ *is m, where* μ *is the grade of* v_1 *with respect to* $A^{\mathrm{T}}A$ *and* λ *is the grade of* u_1 *with respect to* AA^{T}.

The proof of these propositions are similar to those given in [14] for the classical Arnoldi process.

By using the procedure Bidiag 1, the LSQR method constructs an approximation solution of the form $x_k = V_k y_k$ which solves the least-squares problem, $\min\|b - Ax\|_2$. The main steps of the LSQR algorithm can be summarized as follows.

Algorithm 1 (*LSQR algorithm*)

Set $x_0 = 0$
Compute $\beta_1 = \|b\|_2$, $u_1 = b/\beta_1$, $\alpha_1 = \|A^{\mathrm{T}}u_1\|_2$, $v_1 = A^{\mathrm{T}}u_1/\alpha_1$
Set $w_1 = v_1$, $\bar{\phi}_1 = \beta_1$, $\bar{\rho}_1 = \alpha_1$
For $i = 1, 2, \ldots$, until convergence Do:
$w = Av_i - \alpha_i u_i$
$\beta_{i+1} = \|w\|_2$
$u_{i+1} = w/\beta_{i+1}$
$z_i = A^{\mathrm{T}}u_{i+1} - \beta_{i+1}v_i$
$\alpha_{i+1} = \|z_i\|_2$
$v_{i+1} = z_i/\alpha_{i+1}$
$\rho_i = (\bar{\rho}_i^2 + \beta_{i+1}^2)^{1/2}$
$c_i = \bar{\rho}_i/\rho_i$
$s_i = \beta_{i+1}/\rho_i$
$\theta_{i+1} = s_i\alpha_{i+1}$
$\bar{\rho}_{i+1} = -c_i\alpha_{i+1}$
$\phi_i = c_i\bar{\phi}_i$
$\bar{\phi}_{i+1} = s_i\bar{\phi}_i$
$x_i = x_{i-1} + (\phi_i/\rho_i)w_i$
$w_{i+1} = v_{i+1} - (\theta_{i+1}/\rho_i)w_i$
If $|\phi_{i+1}|$ is small enough then stop
EndDo.

More details about the LSQR algorithm can be found in [11].

3. Global least squares method

In this section, we propose a new procedure, based on Bidiag 1, for reducing A to the lower bidiagonal form, and we introduce the notations of block subspaces and global LSQR method.

The global Bidiag 1 procedure constructs the sets of the $n \times s$ block vectors $V_1, V_2, \ldots$ and $U_1, U_2, \ldots$ such that $\langle V_i, V_j\rangle_F = 0$, $\langle U_i, U_j\rangle_F = 0$, for $i \neq j$, and $\|V_i\|_F = 1$, $\|U_i\|_F = 1$; and they form two F-orthonormal basis of $\mathbb{R}^{n\times ks}$.

Global Bidiag 1 Algorithm 1 (*starting matrix B*; *reduction to lower bidiagonal form*)

$$\left.\begin{aligned} &\beta_1 U_1 = B, \quad \alpha_1 V_1 = A^{\mathrm{T}}U_1, \\ &\beta_{i+1}U_{i+1} = AV_i - \alpha_i U_i, \\ &\alpha_{i+1}V_{i+1} = A^{\mathrm{T}}U_{i+1} - \beta_{i+1}V_i, \end{aligned}\right\} \quad i = 1, 2, \ldots \tag{4}$$

The scalars $\alpha_i \geqslant 0$ and $\beta_i \geqslant 0$ are chosen so that $\|U_i\|_F = \|V_i\|_F = 1$.

We define

$$\mathscr{U}_k \equiv [U_1, U_2, \ldots, U_k], \qquad \mathscr{V}_k \equiv [V_1, V_2, \ldots, V_k], \qquad T_k \equiv \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{bmatrix}$$

and as [8], we use the notation $*$ for the following product:

$$\mathscr{V} * \gamma = \sum_{j=1}^{k} V_j \gamma_j,$$

where $\gamma = [\gamma_1, \gamma_2, \ldots, \gamma_k]^{\mathrm{T}}$ is a vector of $\mathbb{R}^k$ and, by the same way, we set

$$\mathscr{V} * T_k = [\mathscr{V} * T_{.,1}, \mathscr{V} * T_{.,2}, \ldots, \mathscr{V} * T_{.,k}],$$

where $T_{.,j}$ is jth column of the matrix T_k. It is easy to see that the following relations are satisfied:

$$\mathscr{V} * (\gamma + \eta) = \mathscr{V} * \gamma + \mathscr{V} * \eta \quad \text{and} \quad (\mathscr{V} * T_k) * \gamma = \mathscr{V} * (T_k \gamma),$$

where γ and η are two vectors of $\mathbb{R}^k$. Now, according to the notation $*$, the recurrence relations (4) may be rewritten as

$$\mathscr{U}_{k+1} * (\beta_1 e_1) = B,$$
$$A\mathscr{V}_k = \mathscr{U}_{k+1} * T_k,$$
$$A^{\mathrm{T}}\mathscr{U}_{k+1} = \mathscr{V}_k * T_k^{\mathrm{T}} + \alpha_{k+1} V_{k+1} * e_{k+1}^{\mathrm{T}},$$

where e_i is the ith column of identity matrix.

For the Global Bidiag 1, we have the following proposition. It is similar to Proposition 1.

Proposition 3. *Suppose that k step of the procedure Global Bidiag 1 have been taken, then the block vectors $V_1, V_2, \ldots, V_k$, and $U_1, U_2, \ldots, U_{k+1}$ are F-orthonormal basis of the Krylov subspaces $\mathscr{K}_k(A^{\mathrm{T}}A, V_1)$ and $\mathscr{K}_{k+1}(AA^{\mathrm{T}}, U_1)$, respectively.*

The proof of this proposition is also similar to that given in [8] for the classical Arnoldi process.

Proposition 2 can be also restated with the block vectors $V_1, V_2, \ldots, V_k$; $U_1, U_2, \ldots, U_k$ rather than the vectors $v_1, v_2, \ldots, v_k$; $u_1, u_2, \ldots, u_k$.

Using the above notations, we have also the following result which has been proved in [8].

Proposition 4. *Let $\mathscr{U}_k$ be the matrix defined by $\mathscr{U}_k = [U_1, U_2, \ldots, U_k]$, where the $n \times s$ matrices U_i, $i = 1, .., k$, are defined by the global Bidiag 1 process. Then we have $\|\mathscr{U}_k * \eta\|_F = \|\eta\|_2$, where η is a vector of $\mathbb{R}^k$.*

3.1. Global LSQR (Gl-LSQR) algorithm

In this section we describe a Gl-LSQR algorithm for solving linear systems (2). At iteration k we seek an approximate solution X_k of the form

$$X_k = \mathscr{V}_k * y_k, \tag{5}$$

where y_k is in $\mathbb{R}^k$. The residual matrix for this approximate solution is given by

$$R_k = B - AX_k = B - A\mathscr{V}_k * y_k = \beta_1 U_1 - \mathscr{U}_{k+1} * T_k * y_k = \mathscr{U}_{k+1} * (\beta_1 e_1 - T_k * y_k), \tag{6}$$

where $e_1 \in \mathbb{R}^{k+1}$. The global LSQR algorithm chooses the vector y_k which minimizes the Frobenius norm of the matrix residual R_k. From (6), the residual R_k can be written as

$$\|R_k\|_F = \|\mathscr{U}_{k+1} * (\beta_1 e_1 - T_k * y_k)\|_F.$$

According to Proposition 4, we have

$$\|R_k\|_F = \|\beta_1 e_1 - T_k y_k\|_2.$$

Thus

$$\min \|R_k\|_F = \min_{y\in\mathbb{R}^k} \|\beta_1 e_1 - T_k y\|_2.$$

This minimization problem is accomplished by using the QR decomposition [7], where a unitary matrix Q_k is determined so that

$$Q_k[\,T_k \quad \beta_1 e_1\,] = \begin{bmatrix} \mathscr{R}_k & f_k \\ 0 & \bar{\phi}_{k+1} \end{bmatrix} = \begin{bmatrix} \rho_1 & \theta_2 & & & & \phi_1 \\ & \rho_2 & \theta_3 & & & \phi_2 \\ & & \ddots & \ddots & & \vdots \\ & & & \rho_{k-1} & \theta_k & \phi_{k-1} \\ & & & & \rho_k & \phi_k \\ & & & & 0 & \bar{\phi}_{k+1} \end{bmatrix},$$

where ρ_l and θ_l are scalars. The above QR factorization is determined by constructing the kth plane rotation $Q_{k,k+1}$ to operate on rows k and $k+1$ of the transformed $[\,T_k \quad \beta_1 e_1\,]$ to annihilate β_{k+1}. This gives the following simple recurrence relation:

$$\begin{bmatrix} c_k & s_k \\ -s_k & c_k \end{bmatrix} \begin{bmatrix} \bar{\rho}_k & 0 & \bar{\phi}_k \\ \beta_{k+1} & \alpha_{k+1} & 0 \end{bmatrix} = \begin{bmatrix} \rho_k & \theta_{k+1} & \phi_k \\ 0 & \bar{\rho}_{k+1} & \bar{\phi}_{k+1} \end{bmatrix},$$

where $\bar{\rho}_1 \equiv \alpha_1$, $\bar{\phi}_1 \equiv \beta_1$, and the scalars c_k and s_k are the nontrivial elements of $Q_{k,k+1}$. The quantities $\bar{\rho}_k, \bar{\phi}_k$ are intermediate scalars that are subsequently replaced by ρ_k, ϕ_k.

With setting

$$y_k = \mathscr{R}_k^{-1} * f_k$$

the approximate solution is given by

$$X_k = \mathscr{V}_k * (\mathscr{R}_k^{-1} * f_k) = (\mathscr{V}_k * \mathscr{R}_k^{-1}) * f_k.$$

Letting

$$\mathscr{P}_k \equiv \mathscr{V}_k * \mathscr{R}_k^{-1} \equiv [P_1 P_2 \ldots P_k]$$

then

$$X_k = \mathscr{P}_k * f_k.$$

The $n \times s$ matrix P_k, the last block column of $\mathscr{P}_k$, can be computed from the previous P_{k-1} and V_k, by the simple update

$$P_k = (V_k - P_{k-1}\theta_k)\rho_k^{-1}, \tag{7}$$

also note that,

$$f_k = \begin{bmatrix} f_{k-1} \\ \phi_k \end{bmatrix},$$

in which

$$\phi_k = c_k \bar{\phi}_k.$$

Thus, X_k can be updated at each step, via the relation

$$X_k = X_{k-1} + P_k \phi_k.$$

The matrix residual norm $\|R_k\|_F$ is computed directly from the quantity $\bar{\phi}_{k+1}$ as

$$\|R_k\|_F = |\bar{\phi}_{k+1}|.$$

Some of the work in (7) can be eliminated by using matrices $W_k \equiv P_k\rho_k$ in place of P_k. The main steps of Gl-LSQR algorithm can be summarized as follows.

Algorithm 2 (*Gl-LSQR algorithm*)

Set $X_0 = 0_{n\times s}$
$\beta_1 = \|B\|_F$, $U_1 = B/\beta_1$, $\alpha_1 = \|A^{\mathrm{T}}U_1\|_F$, $V_1 = A^{\mathrm{T}}U_1/\alpha_1$.
Set $W_1 = V_1$, $\bar{\phi}_1 = \beta_1$, $\bar{\rho}_1 = \alpha_1$
For $i = 1, 2, \ldots$ until convergence Do:
$\mathscr{W}_i = AV_i - \alpha_i U_i$
$\beta_{i+1} = \|\mathscr{W}_i\|_F$
$U_{i+1} = \mathscr{W}_i/\beta_{i+1}$
$\mathscr{S}_i = A^{\mathrm{T}}U_{i+1} - \beta_{i+1}V_i$
$\alpha_{i+1} = \|\mathscr{S}_i\|_F$
$V_{i+1} = \mathscr{S}_i/\alpha_{i+1}$
$\rho_i = (\bar{\rho}_i^2 + \beta_{i+1}^2)^{1/2}$
$c_i = \bar{\rho}_i/\rho_i$
$s_i = \beta_{i+1}/\rho_i$
$\theta_{i+1} = s_i\alpha_{i+1}$
$\bar{\rho}_{i+1} = c_i\alpha_{i+1}$
$\phi_i = c_i\bar{\phi}_i$
$\bar{\phi}_{i+1} = -s_i\bar{\phi}_i$
$X_i = X_{i-1} + (\phi_i/\rho_i)W_i$
$W_{i+1} = V_{i+1} - (\theta_{i+1}/\rho_i)W_i$
If $|\bar{\phi}_{i+1}|$ is small enough then stop
EndDo.

As we observe, our algorithm has certain advantages over global FOM and GMRES algorithms; namely, we do not need to store the basis vectors, we do not predetermine a subspace dimension, and the approximate solution and the Frobenius norm of residuals are cheaply computed at every stage of the algorithm because they are updated with short-term recurrences. We will give now some convergence results on the global LSQR algorithm.

Proposition 5. *Assume that* $AA^{\mathrm{T}} = U\Lambda U^{\mathrm{T}}$, *where* U *is an orthogonal matrix*, $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_k)$ *and* λ_i*'s are eigenvalues of* AA^{T}. *Then, at step* k, *the residual norms produced by the Gl-LSQR satisfy*

$$\|R_k\|_F \leqslant n\sqrt{n}\|B\|_F \min_{p\in\mathscr{P}_k, p(0)=1} (\max_{i=1,..,n} |p(\lambda_i)|),$$

where $\mathscr{P}_k$ *is the set of polynomials of degree less or equal than* k.

The proof of this proposition is similar to Proposition 4 in [8].

3.2. Application to the Sylvester matrix equation

Sylvester equations $AX - XB = C$ play an important role in numerical linear algebra. For example, they arise in the computation of invariant subspaces, in control problems, as linearizations of algebraic Riccati equations, and in the discretization of partial differential equations. For small equations, direct methods are feasible. For large systems, iterative solution methods are available, like Krylov subspace methods.

Now we want to show how to apply the Gl-LSQR algorithm for solving the Sylvester matrix equation. We consider the following Sylvester matrix equation:

$$AX - XB = C, \tag{8}$$

where A and B are $n \times n$ and $s \times s$ matrices, respectively, and X, C are $n \times s$ matrices.

Let $\mathscr{A}$ be the linear operator defined as follows:

$$\mathscr{A} : \mathbb{R}^{n\times s} \to \mathbb{R}^{n\times s},$$
$$X \to \mathscr{A}(X) = AX - XB,$$
$$\mathscr{A}^{\mathrm{T}}(X) = A^{\mathrm{T}}X - XB^{\mathrm{T}}.$$

Hence the problem (8) can be expressed as

$$\mathscr{A}(X) = C. \tag{9}$$

The matrix equation (9) can be solved by Gl-LSQR algorithm by replacing in Algorithm 2 the matrix B by C and the matrix products AX and $A^{\mathrm{T}}X$ by $AX - XB$ and $A^{\mathrm{T}}X - XB^{\mathrm{T}}$, respectively.

4. Numerical examples

In this section, we give some experimental results. Our examples have been coded in Matlab with double precision and have been executed on a PIV/1.8 GHz/Full workstation. For all the experiments, the initial guess was $X_0 = 0$ and $B = \mathrm{rand}(n, s)$, where function rand creates an $n \times s$ random matrix with coefficients uniformly distributed in $[0, 1]$. In all test problems, we have not used any preconditioning. All the tests were stopped as soon as, $\max_{1\leqslant j\leqslant s}(\|\mathrm{col}_j(R_k)\|_2/\|\mathrm{col}_j(R_0)\|_2) \leqslant 10^{-7}$.

As [8], the first matrix test A_1 represents the 5-point discretization of the operator

$$L(u) = -u_{xx} - u_{yy} + \delta u_x \tag{10}$$

on the unit square $[0, 1] \times [0, 1]$ with homogeneous Dirichlet boundary conditions. The discretization was performed using a grid size of $h = 1/61$ which yields a matrix of dimension $n = 3600$; we chose $\delta = 0.5$.

Also, we use some matrices from Harwell–Boeing collection. These matrices with their properties are shown in Table 1.

In Table 2, we give the ratio $t(s)/t(1)$, where $t(s)$ is the CPU time for Gl-LSQR algorithm and $t(1)$ is the CPU time obtained when applying LSQR for one right-hand side linear system. Note that the time obtained by LSQR for one right-hand side depends on which right-hand was used. So, as [8], $t(1)$ is the average of the times needed for the s right-hand sides using LSQR. We note that Gl-LSQR is effective if the indicator $t(s)/t(1)$ is less than s. In Table 2, we list the ratio $t(s)/t(1)$, for $s = 5$, 10, 15, 20, and 25, for the Gl-LSQR. As shown in

Table 1
Test problems information

Matrix\Property	order	sym	nnz	cond
PDE900	900	No	4380	2.9e+02
PDE2961	2961	No	14,585	9.49e+02
SHERMAN4	1104	No	3786	7.2e+03
SHERMAN5	3312	No	20,793	3.9e+05
GR − 30 − 30	900	Yes	4322	35.8e+02

Table 2
Effectiveness of Gl-LSQR algorithm measured by $t(s)/t(1)$

Matrix\S	5	10	15	20	25
A_1	2.30	4.12	6.65	9.64	12.19
PDE900	2.60	6.07	9.48	12.61	15.69
PDE2961	1.78	3.44	4.54	6.97	9.15
SHERMAN4	2.39	5.35	8.35	11.43	14.58
GR − 30 − 30	1.64	2.59	3.91	5.28	6.37

Table 3
Performance of Gl-LSQR algorithm for Sylvester equation $n = 4000$

$v\backslash s$	2	5	8	10	cond(A)
10	24 (0.093)	83 (0.766)	169 (2.329)	246 (4.532)	1.8e+06
50	8 (0.047)	65 (0.609)	67 (0.906)	84 (1.484)	3.9e+05

Table 2 the Gl-LSQR algorithm is effective and less expensive than the LSQR algorithm applied to each right-hand side.

For test problem of Sylvester equation, we consider the elliptic operator

$$L(u) = -\triangle u + 2vu_x + 2vu_y$$

with Dirichlet boundary conditions which was described in [3]. The operator was discretized using central finite differences on $[0,1]\times[0,1]$, with mesh sizes $h = 1/(n+1)$ in the "x" direction and $k = 1/(s+1)$ in the "y" direction. This yields a linear system of algebraic equations that can be written as a Sylvester equation (8), in which A and B are tridiagonal matrices of the form

$$A = \text{tridiag}(-1 - vh, 2, -1 + vh) \quad \text{and} \quad B = -\text{tridiag}(-1 - vk, 2, -1 + vk).$$

The entries of the matrix C were random values uniformly distributed on $[0,1]$ and the initial guess was $X_0 = 0_{n,s}$. The tests were stopped as soon as $\|R_k\|_F/\|R_0\|_F \leqslant 10^{-8}$, where $R_k = C - AX_k + X_kB$. The results obtained for $v = 10$, 50, $n = 4000$ and different values of s are presented in Table 3. Table 3 contains the number of iterations and, in parenthesis, the CPU time for Gl-LSQR to converge. The last column of this Table contains the condition number of matrix A. The results presented in Table 3 show the effectiveness of Gl-LSQR for the Sylvester equations. As we observe for $v = 10$ the number of iterations and the CPU time are more than those for $v = 50$. This is due to the condition number of matrix A. The efficiency and robustness of Gl-LSQR method can be improved by using a good preconditioner for the symmetric positive definite matrix $A^{\mathrm{T}}A$.

5. Conclusion

In this paper we have presented a global version of least squares (LSQR) algorithm for solving general linear systems with several right-hand sides. As we observed, the new method reduces the coefficient matrix A to a lower bidiagonal matrix form. This leads to a simple recurrence formula for generating the sequence of approximate solutions $\{X_k\}$. Our algorithm has certain advantages over global FOM and GMRES algorithms; namely we do not need to store the basis vectors, we do not need to predetermine a subspace dimension, and the approximate solutions and residuals are cheaply computed at every stage of the algorithm because they are updated with short-term recurrences. Experimental results show that the proposed method are effective and less expensive than the LSQR algorithm applied to each right-hand side. We have also noted how this algorithm could be applied to other matrix equations such as the Sylvester equation. Finally, we mention that the efficiency and robustness of this new method can be improved by using a good preconditioner for the symmetric positive definite matrix $A^{\mathrm{T}}A$.

References

[1] T. Chan, W. Wang, Analysis of projection methods for solving linear systems with multiple right-hand sides, SIAM J. Sci. Comput. 18 (1997) 1698–1721.
[2] A. EL Guennouni, K. Jbilou, H. Sadok, The block Lanczos method for linear systems with multiple right-hand sides, ano.univ-lille 1.fr/pub/1999/ano 396.html-2k.
[3] A. EL Guennouni, K. Jgilon, A.J. Riquet, Block Krylov subspace methods for solving large Sylvester equations, Numer. Algorithms 29 (1–3) (2002) 75–96.
[4] A. EL Guennouni, K. Jgilon, H. Sadok, A block BiCGSTAB algorithm for multiple linear systems, Electron. Trans. Numer. Anal. 16 (2003) 129–142.
[5] R. Freund, M. Malhotra, A block-QMR algorithm for non-hermitian linear systems with multiple right-hand sides, Linear Algebra Appl. 254 (1997) 119–157.

[6] G.H. Golub, W. Kahan, Algorithm LSQR is based on the Lanczos process and bidiagonalization procedure, SIAM J. Numer. Anal. 2 (1965) 205–224.
[7] G.H. Golub, C.F. Van Loan, Matrix Computations, Johns Hopkins University Press, Baltimore, MD, 1983.
[8] K. Jbilou, A. Messaoudi, H. Sadok, Global FOM and GMRES algorithms for matrix equations, Appl. Numer. Math. 31 (1) (1999) 49–63.
[9] K. Jbilou, H. Sadok, Global Lanczos-based methods with applications, Technical Report LMA 42, Univérsité du Littora, Calais, France, 1997.
[10] P. Joly, Résolution de Systéms Linéaires Avec Plusieurs Second Members par la Méthode du Gradient Conjugué, Technical Report R-91012, Publications du Laboratire d'Analyse Numérique, Univérsité Pierre et Marie Curie, Paris, 1991.
[11] D. O'Leary, The block conjugate gradient algorithm and related methods, Linear Algebra Appl. 29 (1980) 293–322.
[12] Christopher C. Paige, Michael A. Saunders, LSQR: an algorithm for sparse linear equations and sparse least squares, ACM Trans. Math. Software 8 (1) (1982) 43–71.
[13] Y. Saad, On the Lanczos method for solving symmetric linear systems with several right-hand sides, Math. Comput. 48 (1987) 651–662.
[14] Y. Saad, Iterative Methods for Sparse Linear Systems, second ed. with corrections, January 3RD, 2000.
[15] V. Simoncini, E. Gallopolous, An iterative method for nonsymmetric systems with multiple right-hand sides, SIAM J. Sci. Comput. 16 (1995) 917–933.
[16] H. Van Der Vorst, An iterative solution method for solving $f(A) = b$, using Krylov subspace information obtained for the symmetric positive definite matrix A, J. Comput. Appl. 18 (1987) 249–263.
[17] B. Vital, Etude de Quelques Méthodes de Résolution de Problèmes Linéares de Grande Taille sur Multiprocesseur, Ph.D thesis, Univérsité de Rennes, Rennes, France, 1990.