Japan J. Indust. Appl. Math., 25 (2008), 185–207 Area ⟨2⟩

Preconditioned GMRES Methods for Least Squares Problems

Tokushi Ito* and Ken Hayami†,‡

*Business Design Laboratory Co., Ltd.
Design Lab, Design Center Bldg., NADYA Park
3–18–1, Sakae, Naka-ku, Nagoya-shi, Aichi 460–0008, Japan
E-mail: ito@business-design.co.jp

†National Institute of Informatics
2–1–2, Hitotsubashi, Chiyoda-ku, Tokyo 101–8430, Japan and
Department of Informatics
Graduate University for Advanced Studies (Sokendai)
E-mail: hayami@nii.ac.jp

Received June 8, 2005
Revised October 12, 2007

For least squares problems of minimizing $\|\boldsymbol{b} - A\boldsymbol{x}\|_2$ where A is a large sparse $m \times n$ $(m \geq n)$ matrix, the common method is to apply the conjugate gradient method to the normal equation $A^{\mathrm{T}}A\boldsymbol{x} = A^{\mathrm{T}}\boldsymbol{b}$. However, the condition number of $A^{\mathrm{T}}A$ is square of that of A, and convergence becomes problematic for severely ill-conditioned problems even with preconditioning. In this paper, we propose two methods for applying the GMRES method to the least squares problem by using an $n \times m$ matrix B. We give the necessary and sufficient condition that B should satisfy in order that the proposed methods give a least squares solution. Then, for implementations for B, we propose an incomplete QR decomposition IMGS(l). Numerical experiments showed that the simplest case $l = 0$ gives the best results, and converges faster than previous methods for severely ill-conditioned problems. The preconditioner IMGS(0) is equivalent to the case $B = (\mathrm{diag}(A^{\mathrm{T}}A))^{-1}A^{\mathrm{T}}$, so $(\mathrm{diag}(A^{\mathrm{T}}A))^{-1}A^{\mathrm{T}}$ was the best preconditioner among IMGS(l) and Jennings' IMGS(τ). On the other hand, CG-IMGS(0) was the fastest for well-conditioned problems.

Key words: least squares problems, GMRES, preconditioning, incomplete QR decomposition, singular systems

1. Introduction

In this paper, we consider the linear least squares problem

$$\min_{\boldsymbol{x} \in \mathbf{R}^n} \|\boldsymbol{b} - A\boldsymbol{x}\|_2, \tag{1.1}$$

where A is an $m \times n$ $(m \geq n)$ large sparse matrix with full column rank.

The popular methods to solve the problem (1.1) are, the direct method [2] and the iterative method applying the Conjugate Gradient (CG) method [5] to the normal equation

$$A^{\mathrm{T}}A\boldsymbol{x} = A^{\mathrm{T}}\boldsymbol{b}. \tag{1.2}$$

This iterative method is called the CGLS method [2].

‡The research of this author was supported by the Grant-in-Aid for Scientific Research of the Ministry of Education, Culture Sports, Science and Technology, Japan.

The direct method is reliable, but computation time and storage become prohibitive for large problems. On the other hand, the iterative method generally requires small storage, and can be fast in execution time. However, this applies only if the convergence proceeds fast enough.

Here, let $A^\dagger$ be the generalized inverse of $A \in \mathbf{R}^{m\times n}$, $\operatorname{rank} A = r$, and let σ_1 and σ_r be the largest and smallest (nonzero) singular value of A, respectively. Then, the condition number of A is

$$\kappa(A) := \|A\|_2 \, \|A^\dagger\|_2 = \frac{\sigma_1}{\sigma_r},$$

and that of $A^{\mathrm{T}}A$ is

$$\kappa(A^{\mathrm{T}}A) = \left(\frac{\sigma_1}{\sigma_r}\right)^2 = \kappa(A)^2.$$

The convergence speed of the CGLS method is known to depend on $\kappa(A^{\mathrm{T}}A) = \kappa(A)^2$, including the case when A is rank deficient [2]. Hence, the convergence of the CGLS method may be slow for ill-conditioned problems, so that preconditioning becomes necessary.

Instead of using the normal equation, Zhang and Oyanagi [10], [11], [12] introduced mapping the least squares problem (1.1) to a system of linear equations with a large $m \times m$ square coefficient matrix AB by an $n \times m$ matrix B and then applying the Orthomin(k) method. By selecting the mapping matrix B properly, the condition number becomes small, and the convergence can be improved.

Following their work, in this paper, we will first consider applying the Generalized Minimal Residual (GMRES(k)) method to the system of linear equations with the (large) $m \times m$ coefficient matrix AB. Next, we will propose using the (small) $n \times n$ matrix BA, which is equivalent to applying the GMRES(k) method to the system of linear equations $BA\boldsymbol{x} = B\boldsymbol{b}$. Numerical experiments show that this latter approach requires less computation. We will also derive the necessary and sufficient condition that the mapping matrix B should satisfy in order that both methods give a least squares solution. As the mapping matrix B, we propose using an incomplete QR decomposition. Numerical experiments show that the proposed method is faster and robust compared to previous methods, for severely ill-conditioned problems.

2. Previous methods for the linear least squares problems

2.1. Direct methods

The common direct method to solve (1.1) is to use QR decomposition [2].

The QR decomposition of an $m \times n$ rectangular matrix A is given by

$$A = QR,$$

where Q is an $m \times n$ rectangular matrix whose column vectors are orthonormal, i.e. $Q^{\mathrm{T}}Q = \mathrm{I}_n$, where I_n is the $n \times n$ identity matrix, and R is an $n \times n$ upper

triangular matrix. If this decomposition can be carried out, the normal equation (1.2) becomes

$$R^{\mathrm{T}}R\boldsymbol{x} = R^{\mathrm{T}}Q^{\mathrm{T}}\boldsymbol{b}, \tag{2.1}$$

and when A is full column rank, R is nonsingular, and (2.1) becomes

$$R\boldsymbol{x} = Q^{\mathrm{T}}\boldsymbol{b}. \tag{2.2}$$

Since R is an upper triangular matrix, (2.2) can be solved by backward substitution. The QR decomposition can be carried out in a finite number of steps. There are various algorithms for implementing the QR decomposition, among which the Householder method and the (modified) Gram–Schmidt method is commonly used. The direct method is reliable, but when the problem is large, the required computational work and memory become huge.

2.2. Iterative methods using the normal equation

By transforming to the normal equation (1.2), the problem becomes a system of linear equations whose coefficient matrix is square and symmetric positive definite. The method of applying the conjugate gradient (CG) method to the normal equation is called the CGLS method [2]. The algorithm of the CGLS method is as follows.

Algorithm 2.1. CGLS method

Choose $\boldsymbol{x}_0$.
$\tilde{\boldsymbol{r}}_0 = A^{\mathrm{T}}(\boldsymbol{b} - A\boldsymbol{x}_0)$
$\boldsymbol{p}_0 = \tilde{\boldsymbol{r}}_0$
for $i = 0, 1, 2, \ldots$ **until convergence**
  $\alpha_i = \frac{(\tilde{\boldsymbol{r}}_i, \tilde{\boldsymbol{r}}_i)}{(\boldsymbol{p}_i, A^{\mathrm{T}}A\boldsymbol{p}_i)}$
  $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \boldsymbol{p}_i$
  $\tilde{\boldsymbol{r}}_{i+1} = \tilde{\boldsymbol{r}}_i - \alpha_i A^{\mathrm{T}}A\boldsymbol{p}_i$
  $\beta_i = \frac{(\tilde{\boldsymbol{r}}_i, \tilde{\boldsymbol{r}}_i)}{(\tilde{\boldsymbol{r}}_{i-1}, \tilde{\boldsymbol{r}}_{i-1})}$
  $\boldsymbol{p}_{i+1} = \tilde{\boldsymbol{r}}_{i+1} + \beta_i \boldsymbol{p}_i$
end

Iterative methods generally require relatively small memory and may be computationally efficient. However, this is true only when the convergence is sufficiently fast. The condition number of $A^{\mathrm{T}}A$ is square of the condition number of A. Hence, the convergence may be slow for ill-conditioned problems so that preconditioning becomes necessary. As preconditioners, diagonal scaling, incomplete Cholesky decomposition, and incomplete QR decomposition are commonly used [2]. Recently, a robust preconditioner was proposed in [1]. In this paper, we will look at preconditoning methods based on incomplete QR decomposition, in comparing the use of the GMRES method to the use of the CG method for least squares problems.

2.3. Incomplete QR decomposition

The incomplete QR decomposition as a preconditioner was proposed by Jennings [6], and it can be classified roughly into two ways. One is the IMGS (Incomplete Modified Gram–Schmidt decomposition) method in which the incomplete decomposition is realized by an incomplete modified Gram–Schmidt method. The other is the IG (Incomplete Givens) method in which the incomplete decomposition is realized by Givens rotations [6].

The IMGS method uses a threshold for the elements in R, where the magnitude of each off-diagonal element r_{ij} is compared against a threshold τ times the norm of the corresponding column norm $d_i \equiv \|\boldsymbol{a}_i\|_2$, i.e., elements which satisfy $|r_{ij}| < \tau d_i$ are dropped, where $A = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n]$ [2]. In the IMGS method, A is approximated by QR, where R is an upper triangular matrix with positive diagonal elements, and

$$\langle \boldsymbol{q}_1, \ldots, \boldsymbol{q}_n \rangle = \langle \boldsymbol{a}_1, \ldots, \boldsymbol{a}_n \rangle,$$

where $\langle \boldsymbol{a}_1, \ldots, \boldsymbol{a}_n \rangle$ denotes the subspace spanned by $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n$. The algorithm of the IMGS method is as follows, where $\boldsymbol{a}_i^{(1)} = \boldsymbol{a}_i$ $(i = 1, \ldots, n)$.

Algorithm 2.2. IMGS method (Jennings' version)

for $i = 1, 2, \ldots, n$
  $r_{ii} = \left\|\boldsymbol{a}_i^{(i)}\right\|_2$, $\boldsymbol{q}_i = \frac{\boldsymbol{a}_i^{(i)}}{r_{ii}}$
  for $j = i+1, \ldots, n$
    $r_{ij} = \boldsymbol{q}_i^{\mathrm{T}} \boldsymbol{a}_j^{(i)}$
    If $|r_{ij}| < \tau d_i$, then $r_{ij} = 0$, end.
    $\boldsymbol{a}_j^{(i+1)} = \boldsymbol{a}_j^{(i)} - r_{ij} \boldsymbol{q}_i$
  end
end

If A is full column rank, this algorithm never breaks down.

Saad proposed an alternative algorithm in which the p_Q largest (in absolute value) elements in a column of Q and the p_R largest elements in a row of R are kept, where p_Q and p_R are two chosen parameters [8]. This algorithm is given as follows.

Algorithm 2.3. IMGS method (Saad's version)

for $i = 1, 2, \ldots, n$
  $r_{ii} = \left\|\boldsymbol{a}_i^{(i)}\right\|_2$, $\boldsymbol{q}_i = \frac{\boldsymbol{a}_i^{(i)}}{r_{ii}}$
  Determine the p_Q largest elements of $\boldsymbol{q}_i$ and assign 0 to the other elements.
  for $j = i+1, \ldots, n$
    $r_{ij} = \boldsymbol{q}_i^{\mathrm{T}} \boldsymbol{a}_j^{(i)}$
    Determine the p_R largest r_{ij}'s for $i+1 \le j \le n$, and assign 0 to the others.
    $\boldsymbol{a}_j^{(i+1)} = \boldsymbol{a}_j^{(i)} - r_{ij} \boldsymbol{q}_i$
  end
end

When these IMGS preconditioners are applied to the normal equation, we have

$$R^{-\mathrm{T}}A^{\mathrm{T}}AR^{-1}R\boldsymbol{x} = R^{-\mathrm{T}}A^{\mathrm{T}}\boldsymbol{b},$$

or

$$\tilde{A}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{b}}, \tag{2.3}$$

where $\tilde{A} = R^{-\mathrm{T}}A^{\mathrm{T}}AR^{-1}$, $\tilde{\boldsymbol{x}} = R\boldsymbol{x}$, $\tilde{\boldsymbol{b}} = R^{-\mathrm{T}}A^{\mathrm{T}}\boldsymbol{b}$.

Then, the conjugate gradient (CG) method is applied to (2.3).

2.4. CR-LS(k) method

Zhang and Oyanagi [10], [11], [12] proposed a different type of method in which they applied the Orthomin(k) method to (1.1) by introducing an $n \times m$ mapping matrix B, instead of solving the normal equation, so that the condition number becomes smaller and improved convergence of the iterative method is expected. This method is called the CR-LS(k) method. The algorithm of the CR-LS(k) method is as follows.

Algorithm 2.4. CR-LS(k) method

Choose $\boldsymbol{x}_0$.
$\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$, $\boldsymbol{p}_0 = B\boldsymbol{r}_0$
for $i = 0, 1, 2, \ldots$ **until convergence**
  $\alpha_i = \frac{(\boldsymbol{r}_i, A\boldsymbol{p}_i)}{(A\boldsymbol{p}_i, A\boldsymbol{p}_i)}$
  $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i\boldsymbol{p}_i$
  $\boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i A\boldsymbol{p}_i$
  for $j = 0$ **to** $\min(k-1, i)$
    $\beta_{i,i-j} = -\frac{(AB\boldsymbol{r}_{i+1}, A\boldsymbol{p}_{i-j})}{(A\boldsymbol{p}_{i-j}, A\boldsymbol{p}_{i-j})}$
  end
  $\boldsymbol{p}_{i+1} = B\boldsymbol{r}_i + \sum_{j=0}^{\min(k-1,i)} \beta_{i,i-j}\boldsymbol{p}_{i-j}$
end

---

3. The GMRES(k)-LS method

The GMRES(k) method [9] is an efficient and robust Krylov subspace method for solving systems of linear equations

$$A\boldsymbol{x} = \boldsymbol{b},$$

where A is square, nonsingular and nonsymmetric.

In this section, we propose algorithms which apply the GMRES(k) method to the linear least squares problem (1.1) by using an $n \times m$ mapping matrix B.

If one were to apply the GMRES(k) method directly to the linear least squares problem in which A is an $m \times n$ rectangular matrix, and the initial residual $\boldsymbol{r}_0$ is an m-dimensional vector, one cannot create a Krylov subspace, just by multiplying $\boldsymbol{r}_0$ by A. There are two possible ways of overcoming this problem.

3.1. GMRES(k)-LS method 1

The first method is to use an $n \times m$ mapping matrix B to create a Krylov subspace $\langle \boldsymbol{r}_0, AB\boldsymbol{r}_0, \ldots, (AB)^{i-1}\boldsymbol{r}_0 \rangle$ in the (larger) m-dimensional space, where AB is an $m \times m$ matrix, as in the CR-LS(k) method [10], [11], [12], and to apply the GMRES(k) method using this Krylov subspace.

First note the following lemma.

Lemma 3.1.

$$\min_{\boldsymbol{x} \in \mathbf{R}^n} \|\boldsymbol{b} - A\boldsymbol{x}\|_2 \leq \min_{\boldsymbol{z} \in \mathbf{R}^m} \|\boldsymbol{b} - AB\boldsymbol{z}\|_2, \tag{3.1}$$

holds, where the equality holds if $\operatorname{rank} B = n$.

Proof. If $\operatorname{rank} B = n$, $\{B\boldsymbol{z} \mid \boldsymbol{z} \in \mathbf{R}^m\} = \mathbf{R}^n$. Hence, the equality holds in (3.1). □

Therefore, we will assume that $\operatorname{rank} B = n$. Note that if $\operatorname{rank} B = n$, there exists $\boldsymbol{z}_0$ such that $B\boldsymbol{z}_0 = \boldsymbol{x}_0$. Thus, consider solving

$$\min_{\boldsymbol{z} \in \mathbf{R}^m} \|\boldsymbol{b} - AB\boldsymbol{z}\|_2 = \min_{\boldsymbol{x} \in \mathbf{R}^n} \|\boldsymbol{b} - A\boldsymbol{x}\|_2$$

using the GMRES(k) method with initial approximation $\boldsymbol{z} = \boldsymbol{z}_0$, where $B\boldsymbol{z}_0 = \boldsymbol{x}_0$. Then, we have the following algorithm.

Algorithm 3.1. GMRES(k)-LS method 1

Choose $\boldsymbol{x}_0$.
$*$ $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$
$\boldsymbol{v}_1 = \frac{\boldsymbol{r}_0}{\|\boldsymbol{r}_0\|_2}$
for $i = 1, 2, \ldots, k$ **until convergence**
  $h_{j,i} = (AB\boldsymbol{v}_i, \boldsymbol{v}_j)$ $(j = 1, 2, \ldots, i)$
  $\hat{\boldsymbol{v}}_{i+1} = AB\boldsymbol{v}_i - \sum_{j=1}^{i} h_{j,i}\boldsymbol{v}_j$
  $h_{i+1,i} = \|\hat{\boldsymbol{v}}_{i+1}\|_2$
  $\boldsymbol{v}_{i+1} = \frac{\hat{\boldsymbol{v}}_{i+1}}{h_{i+1,i}}$
  Find $\boldsymbol{y}_i \in \mathbf{R}^i$ which minimizes $\|\boldsymbol{r}_i\|_2 = \|\|\boldsymbol{r}_0\|_2 \boldsymbol{e}_i - \bar{H}_i \boldsymbol{y}\|_2$.
end
$\boldsymbol{x}_k = \boldsymbol{x}_0 + B[\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]\boldsymbol{y}_k$
$\boldsymbol{x}_0 = \boldsymbol{x}_k$
Go to $*$.

Here, $\bar{H}_i = (h_{pq}) \in \mathbf{R}^{(i+1)\times i}$, and $\boldsymbol{e}_i = (1, 0, \ldots, 0)^{\mathrm{T}} \in \mathbf{R}^{i+1}$.

The line $\boldsymbol{x}_k = \boldsymbol{x}_0 + B[\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]\boldsymbol{y}_k$ in Algorithm 3.1 corresponds to $\boldsymbol{z}_k = \boldsymbol{z}_0 + [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]\boldsymbol{y}_k$, where $\boldsymbol{x}_k = B\boldsymbol{z}_k$ and $\boldsymbol{x}_0 = B\boldsymbol{z}_0$.

Now note the following theorem due to Brown and Walker (Theorem 2.4 of [3]).

THEOREM 3.2. *Let* $\tilde{A} \in \mathbf{R}^{m\times m}$. *The GMRES method determines a least squares solution of*

$$\min_{\boldsymbol{z}\in\mathbf{R}^m} \|\boldsymbol{b} - \tilde{A}\boldsymbol{z}\|$$

for all $\boldsymbol{b} \in \mathbf{R}^m$ *and for all initial approximation* $\boldsymbol{z}_0 \in \mathbf{R}^m$, *if and only if*

$$\mathcal{N}(\tilde{A}) = \mathcal{N}(\tilde{A}^{\mathrm{T}}).$$

Here $\mathcal{N}(\,\cdot\,)$ denotes the null space.

Then, we have the following theorem.

THEOREM 3.3. *Let* $\operatorname{rank} A = \operatorname{rank} B = n$. *Then, the GMRES-LS method 1 determines the least squares solution of* (1.1) *for arbitary* $\boldsymbol{b}$ *and* $\boldsymbol{x}_0$, *if and only if* B *can be expressed as*

$$B = CA^{\mathrm{T}}, \tag{3.2}$$

where C *is a nonsingular matrix.*

Note that the GMRES-LS method 1 corresponds to the GMRES(k)-LS method 1 with $k = \infty$ (no restarts).

Proof. It is sufficient to prove that the GMRES method determines the least squares solution of

$$\min_{\boldsymbol{z}\in\mathbf{R}^m} \|\boldsymbol{b} - AB\boldsymbol{z}\|_2$$

for arbitary $\boldsymbol{b}$ and $\boldsymbol{z}_0$, where $B\boldsymbol{z}_0 = \boldsymbol{x}_0$, if and only if $B = CA^{\mathrm{T}}$ where C is nosingular.

Further, from Theorem 3.3, it is sufficient to prove that $\mathcal{N}(AB) = \mathcal{N}(B^{\mathrm{T}}A^{\mathrm{T}})$. Let $A = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n]$ and $B^{\mathrm{T}} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n]$. Then, note that $\boldsymbol{z} \in \mathcal{N}(AB) \iff AB\boldsymbol{z} = \boldsymbol{0} \iff B\boldsymbol{z} = \boldsymbol{0} \iff \boldsymbol{z} \perp \langle \boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\rangle$, where the second equivalence is due to $\operatorname{rank} A = n$. Note also that, $\boldsymbol{z} \in \mathcal{N}(B^{\mathrm{T}}A^{\mathrm{T}}) \iff B^{\mathrm{T}}A^{\mathrm{T}}\boldsymbol{z} = \boldsymbol{0} \iff A^{\mathrm{T}}\boldsymbol{z} = \boldsymbol{0} \iff \boldsymbol{z} \perp \langle \boldsymbol{a}_1, \ldots, \boldsymbol{a}_n\rangle$, where the second equivalence is due to $\operatorname{rank} B^{\mathrm{T}} = \operatorname{rank} B = n$.

Hence, $\mathcal{N}(AB) = \mathcal{N}(B^{\mathrm{T}}A^{\mathrm{T}}) \iff \langle \boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\rangle = \langle \boldsymbol{a}_1, \ldots, \boldsymbol{a}_n\rangle \iff B^{\mathrm{T}} = AC^{\mathrm{T}}$ where C^{T}: nonsingular $\iff B = CA^{\mathrm{T}}$ where C: nonsingular.

Here we note that there is a related method of applying the GMRES method to overdetermined systems due to Calvetti, Lewis and Reichel [4]. The method essentially appends $m - n$ zero column vectors to A to obtain an $m \times m$ square singular matrix $\tilde{A} = [A, 0]$, and then applies the GMRES method to the least squares problem

$$\min_{\boldsymbol{z}\in\mathbf{R}^m} \|\boldsymbol{b} - \tilde{A}\boldsymbol{z}\| \quad \Big(= \min_{\boldsymbol{x}\in\mathbf{R}^n} \|\boldsymbol{b} - A\boldsymbol{x}\|\Big).$$

This method can be considered as a special case of our GMRES(k)-LS method 1 with $B = [\mathrm{I}_n, 0]$. However, this special choice of B does not necessarily meet the condition (3.2) in Theorem 3.3, so that their method may break down before reaching the least squares solution of (1.1).

3.2. GMRES(k)-LS method 2

The other alternative is to use the $n \times m$ mapping matrix B to map $\boldsymbol{r}_0$ to $\tilde{\boldsymbol{r}}_0 = B\boldsymbol{r}_0$, and then to create a Krylov subspace $\langle \tilde{\boldsymbol{r}}_0, BA\tilde{\boldsymbol{r}}_0, \ldots, (BA)^{i-1}\tilde{\boldsymbol{r}}_0 \rangle$ in the (smaller) n-dimensional space, where BA is an $n \times n$ matrix, and to apply the GMRES(k) method using this Krylov subspace as follows.

Algorithm 3.2. GMRES(k)-LS method 2

Choose $\boldsymbol{x}_0$.
$*$ $\tilde{\boldsymbol{r}}_0 = B(\boldsymbol{b} - A\boldsymbol{x}_0)$
$\boldsymbol{v}_1 = \frac{\tilde{\boldsymbol{r}}_0}{\|\tilde{\boldsymbol{r}}_0\|_2}$
for $i = 1, 2, \ldots, k$ **until convergence**
$h_{j,i} = (BA\boldsymbol{v}_i, \boldsymbol{v}_j)$ $(j = 1, 2, \ldots, i)$
$\hat{\boldsymbol{v}}_{i+1} = BA\boldsymbol{v}_i - \sum_{j=1}^{i} h_{j,i}\boldsymbol{v}_j$
$h_{i+1,i} = \|\hat{\boldsymbol{v}}_{i+1}\|_2$
$\boldsymbol{v}_{i+1} = \frac{\hat{\boldsymbol{v}}_{i+1}}{h_{i+1,i}}$
Find $\boldsymbol{y}_i \in \mathbf{R}^i$ which minimizes $\|\tilde{\boldsymbol{r}}_i\|_2 = \|\|\tilde{\boldsymbol{r}}_0\|_2 \boldsymbol{e}_i - \bar{H}_i \boldsymbol{y}\|_2$.
end
$\boldsymbol{x}_k = \boldsymbol{x}_0 + [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]\boldsymbol{y}_k$
$\boldsymbol{x}_0 = \boldsymbol{x}_k$
Go to $*$.

Here, $\bar{H}_i$, and $\boldsymbol{e}_i$ are defined as in Algorithm 3.1.

As will be shown in the numerical experiments, it turns out that the GMRES(k)-LS method 2, which uses the Krylov space $\langle \tilde{\boldsymbol{r}}_0, BA\tilde{\boldsymbol{r}}_0, \ldots, (BA)^{i-1}\tilde{\boldsymbol{r}}_0 \rangle$ in the (smaller) n-dimensional space requires less computational work compared to the GMRES(k)-LS method 1.

Note that the GMRES(k)-LS method 2 is equivalent to applying the GMRES(k) method to

$$BA\boldsymbol{x} = B\boldsymbol{b}, \tag{3.3}$$

where BA is an $n \times n$ matrix.

The following theorem gives the necessary and sufficient condition for the mapping matrix B, in order that equation (3.3) is equivalent to the original least squares problem (1.1).

THEOREM 3.4. *Let* $\operatorname{rank} A = n$. *Then, the solution of* (3.3) *is a (least squares) solution of* (1.1) *if and only if* B *can be expressed as*

$$B = CA^{\mathrm{T}},$$

where C *is a nonsingular matrix.*

Proof. $\boldsymbol{x}$ is a least squares solution of (1.1) if and only if $\boldsymbol{x}$ satisfies the normal equation (1.2) which is equivalent to

$$A^{\mathrm{T}}\boldsymbol{r} = \boldsymbol{0}, \tag{3.4}$$

where $\boldsymbol{r} = \boldsymbol{b} - A\boldsymbol{x}$.

(3.4), in turn, is equivalent to

$$\langle \boldsymbol{a}_1, \ldots, \boldsymbol{a}_n \rangle \perp \boldsymbol{r}, \tag{3.5}$$

where $A = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n]$.

On the other hand, (3.3) is equivalent to

$$B\boldsymbol{r} = \boldsymbol{0},$$

which, in turn, is equivalent to

$$\langle \boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \rangle \perp \boldsymbol{r}, \tag{3.6}$$

where $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ are the column vectors of B^{T}.

Now, (3.5) and (3.6) are equivalent, if and only if $\langle \boldsymbol{a}_1, \ldots, \boldsymbol{a}_n \rangle$ and $\langle \boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \rangle$ are the same subspace, which is true, if and only if

$$A = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n] = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n]C' = B^{\mathrm{T}}C',$$

where C' is a nonsingular $n \times n$ matrix.

Hence, (3.3) and (1.2) are equivalent if and only if $A = B^{\mathrm{T}}C'$, where C' is a nonsingular $n \times n$ matrix, or, in other words, $AC'^{-1} = B^{\mathrm{T}}$ or $B = C'^{-\mathrm{T}}A^{\mathrm{T}}$.

Thus, the solution of (3.3) is a least squares solution of (1.1), if and only if (3.2) where $C = C'^{-\mathrm{T}}$ is nonsingular.

Hence, we have the following.

THEOREM 3.5. *Let* $\operatorname{rank} A = n$. *Then, the GMRES-LS method 2 determines the least squares solution of* (1.1) *for arbitary* $\boldsymbol{b}$ *and* $\boldsymbol{x}_0$ *if and only if* B *can be expressed as*

$$B = CA^{\mathrm{T}},$$

where C *is a nonsingular matrix.*

Here, the GMRES-LS method 2 is the GMRES(k)-LS method 2 with $k = \infty$ (no restarts).

Proof. The necessity comes from Theorem 3.4.

Next, we show the sufficiency. If $\operatorname{rank} A = n$, $A^{\mathrm{T}}A$ is nonsingular, and if C is nonsingular, $BA = CA^{\mathrm{T}}A$ is nonsingular. Hence, the method, which is equivalent to the GMRES method applied to (3.3), gives the least squares solution of (1.1). □

From Theorem 3.3 and Theorem 3.5, the necessary and sufficient condition for B is the same for the GMRES-LS method 1 and the GMRES-LS method 2, as follows.

Corollary 3.6. *Let* $\operatorname{rank} A = \operatorname{rank} B = n$. *Then, the GMRES-LS method 1 and the GMRES-LS method 2 determine the least squares solution of* (1.1) *for arbitary* $\boldsymbol{b}$ *and* $\boldsymbol{x}_0$ *if and only if* B *can be expressed as*

$$B = CA^{\mathrm{T}},$$

where C *is a nonsingular matrix.*

3.3. Incomplete QR decomposition

Besides satisfying the condition (3.2), it is natural to choose a mapping matrix B which satisfies $BA \approx \mathrm{I}_n$, so that we may expect fast convergence of the iterative method.

In [12], $B = (\operatorname{diag}(A^{\mathrm{T}}A))^{-1}A^{\mathrm{T}}$ was proposed for the CR-LS(k) method, where $\operatorname{diag}(A^{\mathrm{T}}A)$ is the diagonal matrix obtained by taking the diagonal part of $A^{\mathrm{T}}A$, and is nonsingular when A is full rank. It is obvious that this choice of B fulfills the condition (3.2). For this B, (3.3) is equivalent to carrying out the preconditioning of row diagonal scaling to the normal equation (1.2).

In this paper, we consider using the incomplete QR decomposition of A given by

$$A = QR + E,$$

where Q is an $m \times n$ matrix, R is an $n \times n$ upper triangular matrix, and E is the error matrix. The column vectors of Q may not be perfectly orthonormal. Previous methods for solving the linear least squares problem (1.1) using the incomplete QR decomposition used R as the preconditioning matrix for the normal equation $A^{\mathrm{T}}A\boldsymbol{x} = A^{\mathrm{T}}\boldsymbol{b}$ [6]. That is, to solve

$$(R^{-\mathrm{T}}A^{\mathrm{T}}AR^{-1})R\boldsymbol{x} = R^{-\mathrm{T}}A^{\mathrm{T}}\boldsymbol{b}, \tag{3.7}$$

or

$$\tilde{A}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{b}},$$

where $\tilde{A} = R^{-\mathrm{T}}A^{\mathrm{T}}AR^{-1}$, $\tilde{\boldsymbol{x}} = R\boldsymbol{x}$, $\tilde{\boldsymbol{b}} = R^{-\mathrm{T}}A^{\mathrm{T}}\boldsymbol{b}$, using the conjugate gradient method.

In our case, we set $B = R^{-1}Q^{\mathrm{T}}$. In the GMRES(k)-LS method 2, this is equivalent to solving

$$R^{-1}Q^{\mathrm{T}}A\boldsymbol{x} = R^{-1}Q^{\mathrm{T}}\boldsymbol{b} \tag{3.8}$$

using the GMRES(k) method. We expect that (3.8) is better conditioned compared to (3.7).

If we take $E = 0$, the QR decomposition is performed completely, i.e. $A = QR$, so that $\boldsymbol{x} = R^{-1}Q^{\mathrm{T}}\boldsymbol{b}$, and only one iteration is required, but the computational work and storage would become prohibitive for large problems. Hence, we set $E \neq 0$, and do an incomplete QR decomposition. The nearer the incomplete QR decomposition is to the complete QR decomposition, the required number of iterations will decrease, but more time and memory would be required to do the incomplete QR decomposition. Hence, we need to strike a balance.

Consider the QR decomposition using the modified Gram–Schmidt process. In this paper, we will realize the incomplete QR decomposition by making the current column vector of Q to be orthonormal to the previous l column vectors, where the incompleteness of the decomposition is adjusted by the parameter l. If $l = n - 1$, it is equivalent to the complete QR decomposition. We call this procedure the IMGS(l) method. The algorithm is as follows, where $\boldsymbol{a}_i^{(1)} = \boldsymbol{a}_i$ $(i = 1, \ldots, n)$.

Algorithm 3.3. IMGS(l) method

for $i = 1, 2, \ldots, n$
  $r_{ii} = \left\|\boldsymbol{a}_i^{(i)}\right\|_2$, $\boldsymbol{q}_i = \frac{\boldsymbol{a}_i^{(i)}}{r_{ii}}$
  for $j = i + 1, \ldots, \min(i + l, n)$
    $r_{ij} = \boldsymbol{q}_i^{\mathrm{T}}\boldsymbol{a}_j^{(i)}$
    $\boldsymbol{a}_j^{(i+1)} = \boldsymbol{a}_j^{(i)} - r_{ij}\boldsymbol{q}_i$
  end
end

At each step of the IMGS(l) method, the column vector of Q is computed as a linear combination of the column vectors of A such that

$$\langle \boldsymbol{q}_1, \ldots, \boldsymbol{q}_n \rangle = \langle \boldsymbol{a}_1, \ldots, \boldsymbol{a}_n \rangle.$$

Thus, we have $Q = A\tilde{C}$, where $\tilde{C}$ is a nonsingular matrix. Thus, we have

$$Q^{\mathrm{T}} = \tilde{C}^{\mathrm{T}}A^{\mathrm{T}}, \tag{3.9}$$

where $\tilde{C}^{\mathrm{T}}$ is nonsingular.

Now, consider

$$B = R^{-1}Q^{\mathrm{T}} \tag{3.10}$$

obtained by the IMGS(l) decomposition. Since $r_{ii} \neq 0$, $i = 1, \ldots, n$, $R = (r_{ij})$ is nonsingular. From (3.10) and (3.9), we have

$$B = R^{-1}Q^{\mathrm{T}} = R^{-1}\tilde{C}^{\mathrm{T}}A^{\mathrm{T}} = CA^{\mathrm{T}},$$

where $C = R^{-1}\tilde{C}^{\mathrm{T}}$ is nonsingular. Hence, B satisfies the condition (3.2).

Here, we note that IMGS(0) is equivalent to the diagonal scaling proposed in [12].

Lemma 3.7. *The* IMGS(0) *method is equivalent to the diagonal scaling given by* $B = (\mathrm{diag}(A^{\mathrm{T}}A))^{-1}A^{\mathrm{T}}$.

Proof. From Algorithm 3.3, the IMGS(0) method is given by

for $i = 1, 2, \ldots, n$
  $r_{ii} = \|\boldsymbol{a}_i\|_2$, $\boldsymbol{q}_i = \frac{\boldsymbol{a}_i}{r_{ii}}$
end

which gives $R = \mathrm{diag}(\|\boldsymbol{a}_1\|_2, \ldots, \|\boldsymbol{a}_n\|_2)$ and $Q = [\boldsymbol{a}_1/\|\boldsymbol{a}_1\|_2, \ldots, \boldsymbol{a}_n/\|\boldsymbol{a}_n\|_2]$, so that

$$B = R^{-1}Q^{\mathrm{T}} = \begin{bmatrix} \dfrac{\boldsymbol{a}_1^{\mathrm{T}}}{\|\boldsymbol{a}_1\|_2^2} \\ \vdots \\ \dfrac{\boldsymbol{a}_n^{\mathrm{T}}}{\|\boldsymbol{a}_n\|_2^2} \end{bmatrix}.$$

On the other hand, $\mathrm{diag}(A^{\mathrm{T}}A) = \mathrm{diag}(\|\boldsymbol{a}_1\|_2^2, \ldots, \|\boldsymbol{a}_n\|_2^2)$, so that

$$(\mathrm{diag}(A^{\mathrm{T}}A))^{-1}A^{\mathrm{T}} = \begin{bmatrix} \dfrac{\boldsymbol{a}_1^{\mathrm{T}}}{\|\boldsymbol{a}_1\|_2^2} \\ \vdots \\ \dfrac{\boldsymbol{a}_n^{\mathrm{T}}}{\|\boldsymbol{a}_n\|_2^2} \end{bmatrix}.$$

Hence, $B = R^{-1}Q^{\mathrm{T}} = (\mathrm{diag}(A^{\mathrm{T}}A))^{-1}A^{\mathrm{T}}$. □

4. Numerical experiments

In this section, we give numerical experiment results solving linear least squares problems

$$\min_{\boldsymbol{x}} \|A\boldsymbol{x} - \boldsymbol{b}\|_2, \quad A \in \mathbf{R}^{m\times n},\ m \geq n,$$

in order to demonstrate the effectiveness of the proposed method. The following sparse matrices were used for A.

The sets of matrices are newly made artificial matrices. These matrices were generated in order to test the methods for problems with varying condition numbers

and size. They were produced by the MATLAB command "`sprandn`" which makes an $m \times n$ nonsymmetric matrix of a certain density and condition number. The non-zero elements of the matrices are random numbers of normal distribution, whose mean is zero and variance is one, and the pattern of the non-zero elements is also random. The characteristics of these matrices are shown in Table 1 and 2, where "Condition" denotes the condition number, and "Density" is the percentage of non-zero elements.

Table 1. Characteristics of the smaller test matrices

No.	Name	m	n	Density	Condition
1	RANDS2	1000	320	4.9 %	2×10^2
2	RANDS4	1000	320	4.9 %	1×10^4
3	RANDS6	1000	320	4.9 %	1×10^6
4	RANDS8	1000	320	4.9 %	1×10^8

Table 2. Characteristics of the larger test matrices

No.	Name	m	n	Density	Condition
5	RANDL1	10000	1000	1.5 %	6×10^1
6	RANDL2	10000	1000	1.5 %	4×10^2
7	RANDL3	10000	1000	1.5 %	3×10^3
8	RANDL4	10000	1000	1.5 %	3×10^4
9	RANDL5	10000	1000	1.5 %	2×10^5
10	RANDL6	10000	1000	1.5 %	2×10^6
11	RANDL7	10000	1000	1.5 %	2×10^7

We set the initial approximate solution to $\boldsymbol{x}_0 = \boldsymbol{0}$, and the convergence criterion was $\|A^{\mathrm{T}}\boldsymbol{r}\|_2/\|A^{\mathrm{T}}\boldsymbol{b}\|_2 < 10^{-6}$, where $\boldsymbol{r} = \boldsymbol{b} - A\boldsymbol{x}$ is the residual. For the right hand vector $\boldsymbol{b}$, each of its components were generated by a random number generator following the normal distribution, such that $\boldsymbol{b} \in \mathbf{R}^m$ but $\boldsymbol{b} \notin \mathcal{R}(\mathcal{A})$, in general.

The following methods were compared.

(1) The CGLS method, which is equivalent to applying the conjugate gradient (CG) method to the normal equation

$$A^{\mathrm{T}}A\boldsymbol{x} = A^{\mathrm{T}}\boldsymbol{b}.$$

(2) The method of applying the CG method to the normal equation using the IMGS preconditioning (Jennings' version) [2], [6]. It is equivalent to applying the CG method to

$$R^{-\mathrm{T}}A^{\mathrm{T}}AR^{-1}R\boldsymbol{x} = R^{-\mathrm{T}}A^{\mathrm{T}}\boldsymbol{b},$$

or to

$$\tilde{A}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{b}},$$

where $\tilde{A} = R^{-\mathrm{T}}A^{\mathrm{T}}AR^{-1}$, $\tilde{\boldsymbol{x}} = R\boldsymbol{x}$, and $\tilde{\boldsymbol{b}} = R^{-\mathrm{T}}A^{\mathrm{T}}\boldsymbol{b}$.

(3) The method of using the IMGS(l) method of Section 3.3 for the preconditioning, in place of the IMGS method in (2).
(4) The proposed GMRES(k)-LS method 1 with the IMGS(l) as the preconditioner for $B = R^{-1}Q^{\mathrm{T}}$.
(5) The proposed GMRES(k)-LS method 2 with the IMGS(l) as the preconditioner for $B = R^{-1}Q^{\mathrm{T}}$. It is equivalent to solving

$$R^{-1}Q^{\mathrm{T}}A\boldsymbol{x} = R^{-1}Q^{\mathrm{T}}\boldsymbol{b}$$

by the GMRES(k) method.

Besides, for comparison, we showed the computation result of the direct method using the modified Gram–Schmidt orthogonalization considering sparsity.

We coded the algorithm in MATLAB 6 and ran the experiments on an IBM PC 1.70 GHz, 1.50 GB RAM.

First, we tested the IMGS(l) method in combination with the GMRES-LS method 2, changing the parameter values for the preconditioners. The results are shown in Table 3, Table 4, Table 5 and Table 6. The results show the computational time and the number of iterations required to satisfy $\|A^{\mathrm{T}}\boldsymbol{r}\|_2/\|A^{\mathrm{T}}\boldsymbol{b}\|_2 < 10^{-6}$. The fastest case is indicated by $*$.

Table 3. Computation time required for different parameter values for the IMGS(l) method (time in seconds)

Matrix	l	Preconditioning time	Iteration time	Number of iterations	Total time
RANDS2	0	0.01	2.07	131	*2.08
	1	0.51	5.06	130	5.57
	10	1.97	6.01	128	7.98
	50	7.86	5.53	119	13.39
	100	15.36	4.06	91	19.42
	200	31.66	2.03	48	33.69
	300	45.42	0.53	9	45.95
	320	45.57	0.26	1	45.83
RANDS4	0	0.01	10.83	316	*10.84
	1	0.53	19.17	316	19.70
	10	2.02	21.46	316	23.48
	50	7.82	20.86	310	28.68
	100	15.15	17.77	278	32.92
	200	31.27	7.56	148	38.83
	300	44.89	0.92	18	45.81
	320	45.00	0.28	1	45.28

Table 4. Computation time required for different parameter values for the IMGS(l) method (time in seconds)

Matrix	l	Preconditioning time	Iteration time	Number of iterations	Total time
RANDS6	0	0.01	10.94	320	*10.95
	1	0.54	19.55	320	20.09
	10	1.97	21.84	320	23.81
	50	7.88	21.85	320	29.73
	100	15.25	22.04	320	37.29
	200	32.32	10.65	192	42.97
	300	46.00	1.14	24	47.14
	320	46.44	0.28	1	46.72
RANDS8	0	0.01	11.11	320	*11.12
	1	0.48	19.05	320	19.53
	10	1.95	21.89	320	23.84
	50	7.85	21.72	320	29.57
	100	15.23	21.90	320	37.13
	200	31.83	12.10	211	43.93
	300	45.26	1.33	29	46.59
	320	45.70	0.26	1	45.96

Table 5. Computation time required for different parameter values for the IMGS(l) method (time in seconds)

Matrix	l	Preconditioning time	Iteration time	Number of iterations	Total time
RANDL1	0	0.23	4.37	57	*4.60
	1	28.15	34.79	57	62.94
	10	120.59	80.92	57	201.51
	50	490.01	81.06	57	571.07
	100	1106.30	73.36	51	1179.66
RANDL2	0	0.24	25.03	221	*25.27
	1	26.76	131.76	221	158.52
	10	116.74	309.09	220	425.83
	50	481.50	305.67	217	787.17
	100	1149.00	265.00	187	1414.00
RANDL3	0	0.24	73.63	438	*73.87
	1	23.91	273.65	437	297.56
	10	116.82	630.51	435	747.33
	50	482.22	636.00	431	1118.22
	100	1115.50	614.32	426	1729.82
RANDL4	0	0.22	317.29	980	*317.51
	1	26.27	761.67	980	787.94
	10	125.29	1540.30	984	1665.59
	50	498.30	1554.70	987	2053.00
	100	1174.90	1548.90	984	2723.80

Table 6. Computation time required for different parameter values for the IMGS(l) method (time in seconds)

Matrix	l	Preconditioning time	Iteration time	Number of iterations	Total time
RANDL5	0	0.21	311.17	971	*311.38
	1	24.01	708.68	971	732.69
	10	126.37	1557.10	977	1683.47
	50	500.78	1550.20	976	2050.98
	100	1180.20	1558.00	981	2738.20
RANDL6	0	0.24	325.47	994	*325.71
	1	20.71	697.42	995	718.13
	10	125.48	1584.00	995	1709.48
	50	494.23	1591.50	996	2085.73
	100	1161.20	1584.20	995	2745.40
RANDL7	0	0.26	320.25	983	*320.51
	1	22.50	688.95	984	711.45
	10	123.18	1560.50	983	1683.68
	50	475.58	1573.30	986	2048.88
	100	1139.80	1582.00	992	2721.80

In MATLAB, the matrices and the vectors were handled by a sparse data structure. In this structure, each non-zero element has three data (row, column, and the element), and the structure has the data only for all non-zero elements, so the data of the zero element is not recorded. As a result, the consumption of the memory can be suppressed, since the zero elements can be omitted. Also, since the multiplication by zero is not calculated, the computing time can be saved. Hence, in the IMGS(l) method of Algorithm 3.3, the data is updated so that only the non-zero elements are maintained and calculated.

The MATLAB implementation of IMGS(l)-GMRES-LS 2 method is as follows.

```
function iteriter = GMRES-LS2(A,b,restrt,p)
[m n] = size(A);
max_it = 1000;
tol = 0.000001;
[Q,R] = imgram(A,p);
iter = 0;
iteriter = 0;
x = sparse(n,1);
Ri = R^-1;
d = Ri*(Q'*b);
bnrm2 = norm(d);
tolb = tol*norm(A'*b);
r = d-(Ri*((Q')*(A*x)));
```

```
m = restrt;
V(1:n,1:m+1) = sparse(n,m+1);
H(1:m+1,1:m) = sparse(m+1,m);
cs(1:m) = sparse(m,1);
sn(1:m) = sparse(m,1);
e1 = sparse(n,1);
e1(1) = 1.0;
for iter = 1:max_it,
  r = d-(Ri*((Q')*(A*x)));
  V(:,1) = r / norm(r);
  s = norm(r)*e1;
  for i = 1:m,
  w = Ri*((Q')*(A*V(:,i)));
  for k = 1:i,
    H(k,i) = w'*V(:,k);
    w = w-H(k,i)*V(:,k);
  end
  H(i+1,i) = norm(w);
  V(:,i+1) = w / H(i+1,i);
  for k = 1:i-1,
    temp = cs(k)*H(k,i)+sn(k)*H(k+1,i);
    H(k+1,i) = -sn(k)*H(k,i)+cs(k)*H(k+1,i);
    H(k,i) = temp;
  end
  [cs(i),sn(i)] = rotmat(H(i,i),H(i+1,i));
  temp = cs(i)*s(i);
  s(i+1) = -sn(i)*s(i);
  s(i) = temp;
  H(i,i) = cs(i)*H(i,i)+sn(i)*H(i+1,i);
  H(i+1,i) = 0.0;
  iteriter = iteriter+1;
  y1 = H(1:i,1:i) \ s(1:i);
  x1 = x+V(:,1:i)*y1;
  if (norm(A'*(b-A*x1)) <= tolb),
    E = abs(s(i+1));
    y = y1;
    x = x1;
    break;
    end
  end
  if (norm(A'*(b-A*x)) <= tolb), break, end
  y = H(1:m,1:m) \ s(1:m);
  x = x+V(:,1:m)*y;
  r = d-Ri*((Q')*(A*x));
```

```
  s(i+1) = norm(r);
  E = s(i+1);
  if (norm(A'*(b-A*x)) <= tolb), break, end;
end
```

In the IMGS(l) method, as l is increased, the incompleteness of the incomplete QR decomposition becomes small, and the number of iterations tends to decrease, but the computation time for the decomposition increases. It turns out that $l = 0$, which is equivalent to the diagonal scaling $B = (\mathrm{diag}(A^{\mathrm{T}}A))^{-1}A^{\mathrm{T}}$ is optimal in terms of total time.

However, we did not consider the choice of the l column vectors in the computation of IMGS(l). We just chose the first l vectors, so the effect of preconditioing may be limited. If we were to optimize the choice of the column vectors, the preconditioning may become more effective. This is left for future research.

Jennings' version of the IMGS (Algorithm 2.3) and Saad's version of the IMGS (Algorithm 2.3) was also tested with various parameters, but it required more computational time compared to IMGS(0), so we do not give the results here.

For all the following experiments, l was fixed at $l = 0$ for the IMGS(l) method.

Next, we demonstrate the effect of changing the restarting cycle k of the GMRES(k)-LS method 2 in Table 7, Table 8 and Table 9. In each cell of the table, the middle number (iter) is the number of iterations, and the bottom number (time) is the computation time (in seconds). The † indicates that convergence was not obtained within the number of iterations.

Although restarting is effective for less ill-conditioned problems "RANDS2," "RANDL1," "RANDL2" and "RANDL3," the full GMRES is faster for more ill-conditioned problems. Hence, in the following, we used $k = \infty$ (the full GMRES).

The computation results for the smaller test matrices of Table 1 are shown in Table 10. In each cell of the table, the upper number is the number of iterations, and the lower number is the computation time (in seconds) required for convergence.

In relatively well-conditioned problem, the CG method with IMGS(0) preconditioning (3) was the fastest. For the very ill-conditioned problem "RANDS8," the CG method with IMGS(0) preconditioning (3) required many iterations and computation time. On the other hand, the GMRES 2 method with IMGS(0) (5) converged quickly and was the fastest. (2) required the least number of iterations, but was computationally expensive, due to the time for preconditioning and backward and forward substitutions.

Next, experiment results for larger test matrices of Table 2 are given in Table 11, respectively. Similar results are observed, where the GMRES 2 method with IMGS(0) preconditioning of (5) was the fastest for the severely ill-conditioned problems "RANDL6" and "RANDL7."

Table 7. Results for changing k of the GMRES(k)-LS method 2

Matrix	k	Iteration	Time
RANDS2	≥ 131	131	2.07
	100	180	2.10
	50	221	1.74
	20	335	1.92
	10	293	*1.37
	5	555	3.20
RANDS4	≥ 316	316	*10.81
	300	734	21.54
	250	1196	31.60
	200	1390	30.57
	150	2241	38.65
	100	3192	39.53
	50	4244	35.56
RANDS6	≥ 320	320	*11.17
	300	2697	88.85
	250	17989	491.16
	200	44398	982.93
	150	†100000	
RANDS8	≥ 320	320	*11.11
	300	124133	4091.02

Table 8. Results for changing k of the GMRES(k)-LS method 2

Matrix	k	Iteration	Time
RANDL1	≥ 57	57	4.03
	50	59	4.10
	40	59	3.94
	30	59	*3.87
	20	61	4.07
	10	70	4.83
RANDL2	≥ 221	221	25.10
	200	223	23.14
	150	240	21.40
	100	238	*19.11
	50	278	19.58
	10	402	28.46
RANDL3	≥ 438	438	75.84
	400	474	68.52
	300	511	63.58
	200	582	61.76
	100	714	*58.62
	50	983	70.12
	10	1759	124.75

Table 9. Results for changing k of the GMRES(k)-LS method 2

Matrix	k	Iteration	Time
RANDL4	≥ 980	980	*320.55
	900	2445	691.91
	800	3951	1050.61
RANDL5	≥ 971	971	*315.76
	900	2670	799.94
	800	3965	1060.35
RANDL6	≥ 994	994	*330.60
	900	16989	5060.63
RANDL7	≥ 983	983	*322.09
	900	8998	2690.57

Table 10. Results for smaller test matrices

Matrix	(1) CG	(2) CG IMGS	(3) CG IMGS(0)	(4) GMRES 1 IMGS(0)	(5) GMRES 2 IMGS(0)	Direct
RANDS2	434	30	147	128	131	
	0.69	13.23	*0.34	3.55	1.66	47.84
RANDS4	5456	162	896	315	316	
	8.74	27.73	*1.99	20.56	8.84	49.05
RANDS6	39523	612	3814	320	320	
	82.46	89.77	*8.14	21.12	9.14	51.94
RANDS8	†100000	1567	14399	324	320	
		248.73	36.40	20.97	*9.05	47.10

Table 11. Results for larger test matrices

Matrix	(1) CG	(2) CG IMGS	(3) CG IMGS(0)	(4) GMRES 1 IMGS(0)	(5) GMRES 2 IMGS(0)	Direct
RANDL1	131	19	58	59	57	
	3.34	413.07	*1.75	8.57	2.38	24904
RANDL2	791	43	243	232	221	
	21.92	533.92	*6.72	115.75	15.84	24478
RANDL3	5227	109	586	458	438	
	145.23	646.13	*16.54	400.53	49.93	24361
RANDL4	30607	535	3980	977	980	
	846.38	1095.94	*112.78	1904.73	245.52	24478
RANDL5	†1000000	884	7901	981	971	
		1488.77	*225.60	1977.32	234.86	25675
RANDL6	†1000000	2340	24399	†1000	994	
		3231.45	688.95		*240.29	25383
RANDL7	†1000000	4998	51560	†1000	983	
		6789.62	1452.47		*223.13	23903

Fig. 1 shows the relative error vs. iterations, and Fig. 2 shows the relative residual vs. iterations for the case "RANDL1" in Table 4 and 11. Fig. 3 shows the relative error vs. iterations, and Fig. 4 shows the relative residual vs. iterations for the case "RANDL6" in Table 4 and 11. Again, it is shown that the GMRES converges better than CG with the same IMGS(0) preconditioning for the severely ill-conditioned problem "RANDL6."

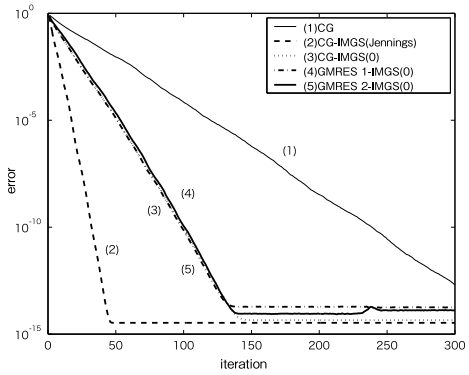

Fig. 1. Relative error versus the number of iterations for the problem RANDL1

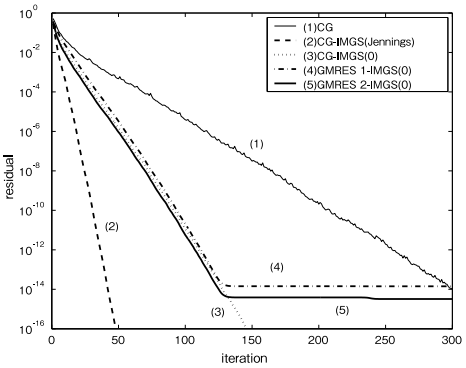

Fig. 2. Relative residual versus the number of iterations for the problem RANDL1

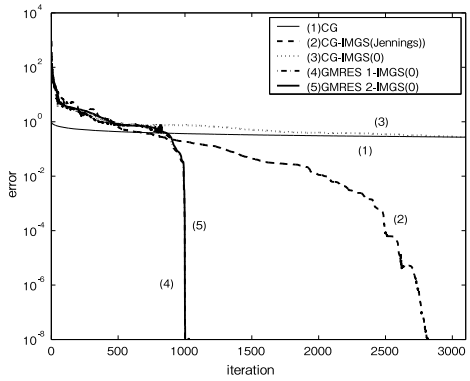

Fig. 3. Relative error versus the number of iterations for the problem RANDL6

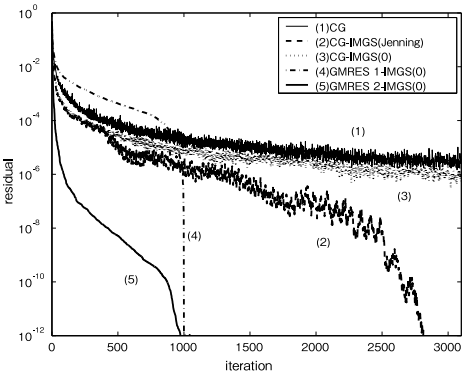

Fig. 4. Relative residual versus the number of iterations for the problem RANDL6

Comparing (3) and (5), it is observed that when using the same preconditioning of IMGS(0), our approach of applying the GMRES method requires less iterations and computation time compared to the usual approach of applying CG to the normal equation. This may be explained by the fact that the GMRES performs the Gram–Schmidt orthogonalization explicitly, where as the CG relies on the three term recurrence, so that GMRES is more robust against loss of orthogonality due to rounding error, especially in ill-conditioned problems. (cf. convergence graphs in

Fig. 3 and Fig. 4.) For less ill conditioned problems "RANDS4" and "RANDS6," the same phenomenon is observed in the sense that GMRES requires less iterations than CG, but CG (3) is faster than GMRES 2 (5) in computation time because the (modified) Gram-Schimidt process of the GMRES is time consuming.

GMRES 1 and GMRES 2 showed similar convergence behaviours, but GMRES 2 was faster. This is because GMRES 1 uses Krylov subspaces generated by BA in the smaller n-dimensional space, where as GMRES 1 uses Krylov subspaces generated by AB in the larger m-dimensional space.

In all cases, the direct method was too time consuming compared to the iterative method. For ill conditioned problems, the number of GMRES iterations approaches the column dimension n. Even then, the GMRES-LS 2 method was much faster than the direct method.

From these experiments, we conclude that the proposed GMRES-LS 2 method with IMGS(0) preconditioning, which is equivalent to applying the GMRES method to the diagonally scaled normal equation, becomes superior to previous methods particularly for severely ill-conditioned problems.

5. Conclusions

In this paper, we proposed two methods for applying the GMRES method to the least squares problem by using a mapping matrix B. The first method applies GMRES to $AB\boldsymbol{z} = \boldsymbol{b}$, and the second method applies GMRES to $BA\boldsymbol{x} = B\boldsymbol{b}$. Next, we gave the necessary and sufficient condition which should be satisfied by B, in order that the methods give a least squares solution. The necessary and sufficient conditions for B are common for the two methods.

As an example for B, we proposed using IMGS(l), which is a kind of incomplete QR decomposition. In fact, IMGS(0) and GMRES applied to $BA\boldsymbol{x} = B\boldsymbol{b}$, which is equivalent to applying the GMRES method to the diagonally scaled normal equation, turned up to be the fastest.

Numerical experiments show that among the proposed method the simplest case $l = 0$ gives best results, and converges faster than previous methods particularly when the coefficient matrix is severely ill-conditioned. On the other hand, CG-IMGS(0) was the best for well-conditioned problems. The preconditioner IMGS(0) is equivalent to the case $B = (\mathrm{diag}(A^{\mathrm{T}}A))^{-1}A^{\mathrm{T}}$, so $\mathrm{diag}(A^{\mathrm{T}}A)^{-1}A^{\mathrm{T}}$ was the best preconditioner among IMGS(l) and Jennnings' IMGS(τ).

Acknowledgements. We would like to thank Professor Kokichi Sugihara for an important remark which lead to Theorem 3.4, and Dr. Shao-Liang Zhang for discussions and useful information. Also, we would like to thank the referees for useful comments.

References

[1] M. Benzi and M. Tuma, A robust preconditioner with low memory requirements for large sparse least squares problems. SIAM J. Sci. Comput., **25** (2003), 499–512.

[2] A. Björck, Numerical Methods for Least Squares Problems. SIAM, Philadelphia, 1996.

[3] P.N. Brown and H.F. Walker, GMRES on (nearly) singular systems. SIAM J. Matrix Anal. Appl., **18** (1997), 37–51.

[4] D. Calvetti, B. Lewis, and L. Reichel, GMRES-type methods for inconsistent systems. Linear Algebra Appl., **316** (2000), 157–169.

[5] M.R. Hestenes and E. Stiefel, Methods of conjugate gradients for solving linear systems. J. Res. Natl. Bur. Stand., **49** (1952), 409–436.

[6] A. Jennings and M.A. Ajiz, Incomplete methods for solving $A^{\mathrm{T}}Ax = b$. SIAM J. Sci. Stat. Comput., **5** (1984), 978–987.

[7] National Institute of Standards, Matrix Market: test matrices database, available on line at `http://math.nist.gov/MatrixMarket/data/`, Gaithersburg, MD.

[8] Y. Saad, Preconditioning techniques for nonsymmetric and indefinite linear systems. J. Comput. Appl. Math., **24** (1988), 89–105.

[9] Y. Saad and M.H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Stat. Comput., **7** (1986), 856–869.

[10] S.-L. Zhang, Generalization of the Conjugate Residual Method. Ph.D. Thesis, University of Tsukuba, Japan, 1989 (in Japanese).

[11] S.-L. Zhang and Y. Oyanagi, A necessary and sufficient convergence condition of orthomin(k) methods for least squares problem with weight. Ann. Inst. Statist. Math., **42** (1990), 805–811.

[12] S.-L. Zhang and Y. Oyanagi, Orthomin(k) method for linear least squares problem. J. Inf. Proc., **14** (1991), 121–125.