Filomat 32:9 (2018), 3181–3198
https://doi.org/10.2298/FIL1809181A

Published by Faculty of Sciences and Mathematics,
University of Niš, Serbia
Available at: http://www.pmf.ni.ac.rs/filomat

Delayed Over-Relaxation in Iterative Schemes to Solve Rank Deficient Linear System of (Matrix) Equations

Arezo Ameri[a], Fatemeh Panjeh Ali Beik[b]

[a]*Department of Mathematics, Islamic Azad University of Kerman, Kerman, Iran*
[b]*Department of Mathematics, Vali-e-Asr University of Rafsanjan, PO Box 518, Rafsanjan, Iran*

Abstract. Recently in [Journal of Computational Physics, 321 (2016), 829–907], an approach has been developed for solving linear system of equations with nonsingular coefficient matrix. The method is derived by using a delayed over-relaxation step (DORS) in a generic (convergent) basic stationary iterative method. In this paper, we first prove semi-convergence of iterative methods with DORS to solve singular linear system of equations. In particular, we propose applying the DORS in the Modified HSS (MHSS) to solve singular complex symmetric systems and in the Richardson method to solve normal equations. Moreover, based on the obtained results, an algorithm is developed for solving coupled matrix equations. It is seen that the proposed method outperforms the relaxed gradient-based (RGB) method [Comput. Math. Appl. 74 (2017), no. 3, 597–604] for solving coupled matrix equations. Numerical results are examined to illustrate the validity of the established results and applicability of the presented algorithms.

1. Introduction

Consider the following consistent linear system of equations,

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

where $A \in \mathbb{R}^{n\times n}$ (possibly singular) and $b \in \mathbb{R}^n$ are known and $x \in \mathbb{R}^n$ is an unknown vector.

For a given $n \times n$ matrix A, the decomposition $A = M - N$ is called splitting, if $M \in \mathbb{R}^{n\times n}$ is nonsingular. Corresponding to a given splitting $A = M - N$, a basic stationary iterative method to solve (1) has the succeeding general form:

$$x_{k+1} = \mathcal{G}x_k + f, \qquad k = 0, 1, 2, \ldots, \tag{2}$$

where $\mathcal{G} = M^{-1}N$ is called iteration matrix, $f = M^{-1}b$ and the initial guess x_0 is given.

In [1], a delayed iterative scheme to solve $Ax = b$ is given by:

$$\begin{aligned} \bar{x}_{k+1} &= \mathcal{G}x_k + f, \\ x_{k+1} &= \omega\bar{x}_{k+1} + (1-\omega)x_{k-1}, \qquad k = 0, 1, 2, \ldots, \end{aligned} \tag{3}$$

2010 *Mathematics Subject Classification*. 65F10
Keywords. Iterative method, delayed over-relaxation step, semi-convergence, coupled matrix equations
Received: 18 August 2017; Accepted: 22 February 2018
Communicated by Dijana Mosić
Email address: f.beik@vru.ac.ir, beik.fatemeh@gmail.com (Fatemeh Panjeh Ali Beik)

where ω is a given positive parameter. Notice that for $\omega = 1$, the iterative scheme (3) reduces to (2). In the case that the spectral radius of $\mathcal{G}$ is strictly less than one ($\rho(\mathcal{G}) < 1$) and A is nonsingular, the convergence properties of (3) have been discussed in [1].

In view of the discussions in [1], the efficiency of using a delayed over-relaxation step (DORS) on convergence speed of stationary iterative methods have been mainly studied under the assumption that the coefficient matrix A is a (square) positive definite matrix. Here we are interested in seeing whether this technique is applicable for solving consistent linear system of equations with possibly non-square coefficient matrix. More precisely, the derived results for applying DORS in Richardson method encouraged us to propose an algorithm for solving $Ax = b$ where A is a rectangular and possibly rank deficient matrix. As a natural way, the results are finally exploited to obtain an iterative scheme to solve coupled matrix equations. In particular, it is illustrated that the proposed algorithm surpasses both the GB method and the recently proposed method in [21] to solve coupled matrix equations.

Before ending this section, we give a brief overview on the concept of "semi-convergence". For a given square matrix $\mathcal{G}$, suppose that $\lim_{k\to\infty} \mathcal{G}^k = L$, if $L < \infty$ then $\mathcal{G}$ is said to be *semi-convergent*. In the case that $L = 0$ then $\mathcal{G}$ is called a *convergent* matrix. Here we recall that the matrix $\mathcal{G}$ is convergent if and only if $\rho(\mathcal{G}) < 1$. A given matrix $\mathcal{G}$ is semi-convergent iff

- $\rho(\mathcal{G}) = 1$,
- $\text{index}(I - \mathcal{G}) = 1$ which means that $\text{rank}(I - \mathcal{G}) = \text{rank}(I - \mathcal{G})^2$,
- If $\mu \in \sigma(\mathcal{G})$ with $|\mu| = 1$ then $\mu = 1$, (i.e., $\upsilon(\mathcal{G}) = \{|\mu| : \mu \in \sigma(\mathcal{G}), \mu \neq 1\} < 1$)

here $\sigma(\mathcal{G})$ denotes the spectrum of $\mathcal{G}$, for future details, one may refer to [7].

In order to discuss on the semi-convergence of an iterative method, we need to also recall the following theorem from [7].

Theorem 1.1. *Consider the splitting $A = M - N \in \mathbb{R}^{n\times n}$. The iterative method (2) converges to some solution x^* to $Ax = b$ for each x_0 if and only if $\mathcal{G}$ is semi-convergent.*

The remainder of this paper is organized as follows. In the next section, first, we prove that if $\mathcal{G}$ is semi-convergent, then (3) converges to a solution of $Ax = b$ in the case that A is singular. Then, as an example, we propose an algorithm by applying DORS in the MHSS method to solve singular complex symmetric systems. In the last part of the second section, we briefly describe the application of DORS on Richardson method to solve normal equations. Section 3 is devoted to presenting an algorithm with the DORS for solving coupled matrix equations. Numerical results are reported in Section 4 which demonstrate applicability of using a DORS for speeding up the convergence of stationary iterative methods. Finally, we briefly state our conclusions in Section 5.

2. DORS in Iterative Methods for Singular Linear Systems

This section is divided into three parts. In the first part, we mainly establish semi-convergence of the iterative methods with DORS to solve singular linear system of equations. Then as an example, we propose using DORS in the MHSS method to solve singular complex symmetric linear systems. Finally we present a convergent and parameter free algorithm with DORS to solve linear system $Ax = b$ where A is an arbitrary rectangular matrix.

2.1. Semi-convergence of iterative schemes with DORS

In this part, we prove that under a certain condition, the iterative method (3) converges to a solution of (1) where A is assumed to be singular. To this end, we first rewrite (3) as follows:

$$\begin{pmatrix} x_{k+1} \\ x_k \end{pmatrix} = \begin{pmatrix} \omega\mathcal{G} & (1-\omega)I \\ I & 0 \end{pmatrix} \begin{pmatrix} x_k \\ x_{k-1} \end{pmatrix} + \begin{pmatrix} \omega f \\ 0 \end{pmatrix}, \tag{4}$$

($k = 0, 1, 2, \ldots$). As seen the iteration matrix of iterative method (4) is given by

$$\tilde{\mathcal{G}} = \begin{pmatrix} \omega\mathcal{G} & (1-\omega)I \\ I & 0 \end{pmatrix}. \tag{5}$$

Now, from Theorem 1.1, it reveals that for proving the convergence of (3) in the case that A is a singular matrix, we need to show that $\tilde{\mathcal{G}}$ is semi-convergent. To this end, we first recall the following useful lemma.

Lemma 2.1. [23] *Consider the quadratic equation $x^2 - bx + c = 0$, where b and c are real numbers. Both roots of the equation are less than one in modulus if and only if $|c| < 1$ and $|b| < 1 + c$.*

The following theorem provides a sufficient condition for the semi-convergence of $\tilde{\mathcal{G}}$.

Theorem 2.2. *Let $\mathcal{G}$ be a semi-convergent matrix and $\omega \in (0, 2)$ is a fixed parameter. Then the matrix $\tilde{\mathcal{G}}$ given by* (5) *is semi-convergent.*

Proof. Here, without loss of generality, we assume that $\omega \neq 1$. We give the proof into two steps for more clarification.

Step I. Let λ be an arbitrary eigenvalue of $\tilde{\mathcal{G}}$. Therefore, it can be seen that in general the following relation holds between the eigenvalues of $\tilde{\mathcal{G}}$ and $\mathcal{G}$, (see [1])

$$\lambda^2 - \omega\ell\lambda + (\omega - 1) = 0, \tag{6}$$

where ℓ is an eigenvalue of $\mathcal{G}$. The above relation gives two solutions in the form:

$$\lambda^{\pm} = \frac{\omega\ell}{2} \pm \frac{1}{2}\sqrt{\omega^2\ell^2 - 4(\omega - 1)}. \tag{7}$$

Notice that our used strategy in the rest of this steps yields a different and short proof, from that given in [1, Proposition 1], for the convergence of the delayed iterative schemes when $\mathcal{G}$ is convergent, see Case 1.

By the assumption, $\mathcal{G}$ is semi-convergent which implies that $\rho(\mathcal{G}) = 1$ and $|\ell| < 1$ for all $\ell \neq 1$. We consider the following two cases.

- Case 1. If $|\ell| < 1$ then considering the quadratic equation (6) from Lemma 2.1, we get $|\lambda^{\pm}| < 1$ for $\omega \in (0, 2)$.
- Case 2. If $\ell = 1$ then from (7):
  - For $1 < \omega < 2$, we have
    $$\lambda^{\pm} = \frac{\omega}{2} \pm \frac{\omega}{2} \mp 1,$$
    which implies that $\lambda^{+} = \omega - 1$ and $\lambda^{-} = 1$.
  - For $0 < \omega < 1$ we have
    $$\lambda^{\pm} = \frac{\omega}{2} \mp \frac{\omega}{2} \pm 1,$$
    which implies that $\lambda^{+} = 1$ and $\lambda^{-} = \omega - 1$.

From the discussions of this step, we may deduce that $\rho(\tilde{\mathcal{G}}) = 1$ and $|\lambda| < 1$ for all $\lambda \neq 1$.

Step II. In this part, we show that $\operatorname{rank}((I-\tilde{\mathcal{G}})^2) = \operatorname{rank}(I-\tilde{\mathcal{G}})$. It can be observed that

$$\begin{aligned} I-\tilde{\mathcal{G}} &= \begin{pmatrix} I-\omega\mathcal{G} & -(1-\omega)I \\ -I & I \end{pmatrix} \\ &= \begin{pmatrix} \omega I & (1-\omega)I \\ 0 & -I \end{pmatrix}\begin{pmatrix} I-\mathcal{G} & 0 \\ I & -I \end{pmatrix}. \end{aligned}$$

Consequently, we see that $\operatorname{rank}(I-\tilde{\mathcal{G}})$ does not deponed on $\omega \neq 0$, i.e.,

$$\operatorname{rank}(I-\tilde{\mathcal{G}}) = \operatorname{rank}(\mathcal{W}),$$

where

$$\mathcal{W} = \begin{pmatrix} I-\mathcal{G} & 0 \\ I & -I \end{pmatrix}.$$

Now, it can be deduced that

$$\operatorname{rank}(I-\tilde{\mathcal{G}}) = \operatorname{rank}(I-\mathcal{G}) + n.$$

On the other hand,

$$\operatorname{rank}((I-\tilde{\mathcal{G}})^2) = \operatorname{rank}(\mathcal{I}\mathcal{W}\mathcal{I}\mathcal{W}) = \operatorname{rank}(\mathcal{W}\mathcal{I}\mathcal{W}),$$

in which $\mathcal{I}$ is a nonsingular matrix defined as follows:

$$\mathcal{I} = \begin{pmatrix} \omega I & (1-\omega)I \\ 0 & -I \end{pmatrix}.$$

By the following computations and invoking the assumption that $\operatorname{rank}((I-\mathcal{G})^2) = \operatorname{rank}(I-\mathcal{G})$, we get

$$\begin{aligned} \operatorname{rank}((I-\tilde{\mathcal{G}})^2) &= \operatorname{rank}(\mathcal{W}\mathcal{I}\mathcal{W}) \\ &= \operatorname{rank}\left(\begin{pmatrix} (I-\mathcal{G})(I-\omega\mathcal{G}) & -(1-\omega)(I-\mathcal{G}) \\ 2I-\omega\mathcal{G} & (\omega-2)I \end{pmatrix}\right) \\ &= \operatorname{rank}\left(\begin{pmatrix} (I-\mathcal{G})(I-\omega\mathcal{G}) & -(1-\omega)(I-\mathcal{G}) \\ 2I-\omega\mathcal{G} & (\omega-2)I \end{pmatrix}\begin{pmatrix} I & 0 \\ I & I \end{pmatrix}\right) \\ &= \operatorname{rank}\left(\begin{pmatrix} \omega(I-\mathcal{G})^2 & -(1-\omega)(I-\mathcal{G}) \\ \omega(I-\mathcal{G}) & (\omega-2)I \end{pmatrix}\right) \\ &= \operatorname{rank}\left(\begin{pmatrix} I & -(I-\mathcal{G}) \\ 0 & I \end{pmatrix}\begin{pmatrix} \omega(I-\mathcal{G})^2 & -(1-\omega)(I-\mathcal{G}) \\ \omega(I-\mathcal{G}) & (\omega-2)I \end{pmatrix}\right) \\ &= \operatorname{rank}\left(\begin{pmatrix} 0 & (I-\mathcal{G}) \\ \omega(I-\mathcal{G}) & (\omega-2)I \end{pmatrix}\right) \\ &= \operatorname{rank}\left(\begin{pmatrix} I & -\frac{1}{\omega-2}(I-\mathcal{G}) \\ 0 & I \end{pmatrix}\begin{pmatrix} 0 & (I-\mathcal{G}) \\ \omega(I-\mathcal{G}) & (\omega-2)I \end{pmatrix}\right) \\ &= \operatorname{rank}\left(\begin{pmatrix} -\frac{\omega}{\omega-2}(I-\mathcal{G})^2 & 0 \\ \omega(I-\mathcal{G}) & (\omega-2)I \end{pmatrix}\right) \\ &= \operatorname{rank}((I-\mathcal{G})^2) + n \\ &= \operatorname{rank}(I-\mathcal{G}) + n. \end{aligned}$$

From Steps I and II, we may conclude the results immediately. □

Before ending this subsection, we recall the following two basic theorems which are useful for determining the unique minimum norm least–squares solution in an algorithm. In the following we use the notation "Range(B)" to denote the span of the column vectors of a given matrix B.

Theorem 2.3. [17] *Let $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$, and*

$$\mathcal{X} = \{x \in \mathbb{R}^n : x = \underset{y\in\mathbb{R}^n}{\operatorname{argmin}} \|Ay - b\|_2\}.$$

Then $x \in \mathcal{X}$ if and only if $A^TAx = A^Tb$. Moreover, $x^* = A^+b$ is the unique solution of the problem

$$\min_{x\in\mathcal{X}} \|x\|_2,$$

where A^+ is the pseudoinverse of A.

Note the vector $x^* = A^+b$ in Theorem 2.3 is said to be the least Euclidean norm solution.

Theorem 2.4. [18] *Let $A \in \mathbb{R}^{m\times n}$ and $b \in Range(A)$. Suppose that the system of linear equations $Ax = b$ has a solution $x^* \in Range(A^T)$. Then x^* is a unique least Euclidean norm solution of the system of linear equations.*

2.2. MHSS iterative method with DORS

From discussions in the previous part, it revealed that if the stationary iterative method (2) is semi-convergent, then the corresponding iterative scheme (3) converges to a solution of (1). In the literature the semi-convergence of several iterative schemes have been studied so far, therefore they can be effectively used with DORS to solve singular linear system of equations. Here we only consider the modified MHHS method with DORS between several possible approaches. Observing the performance of DORS in other semi-convergence stationary iterative methods left to conscientious readers.

Let us consider the singular linear system $Ax = b$ where $A \in \mathbb{C}^{n\times n}$ is a complex symmetric matrix of the form $A = W + iT$. In [8], the MHSS method for solving this kind of linear system is given as follows:

The MHSS iteration method: Let $x_0 \in \mathbb{C}^n$ be an arbitrary initial guess. For $k = 0, 1, 2, \ldots$ until the sequence of iterates x_k converges, compute the $(k+1)$-th iterate x_{k+1} using the following iterative methods:

$$\begin{cases} (\alpha I + W)\overline{x} = (\alpha I - iT)x_k + b, \\ (\alpha I + T)x_{k+1} = (\alpha I + iW)\overline{x} - ib, \end{cases} \tag{8}$$

where α is a given positive parameter. Suppose that $\tau_{\min}$ and $\tau_{\max}$ are the minimum and the maximum nonzero eigenvalues of the matrices W and T, respectively. In [8, Corollary 2.2], it has been established that the quasi-optimum value for the α is given by $\alpha_* = \sqrt{\tau_{\min}\tau_{\max}}$ which minimizes the upper bound of of the semi-convergence factor for the MHSS iterative scheme; see [8] for further details.

In the case that the matrices W and T are both symmetric positive semidefinite, the semi-convergence of the MHSS method is proved in [8]. From the reported numerical experiments in [8], it is observed that the MHSS method surpasses the HSS method [2] to solve the mentioned class of singular linear system of equations.

For two complex vectors x and y of the same size, we consider the inner product $\langle x, y\rangle = \mathrm{Re}(y^Hx)$ where y^H denotes the conjugate transpose of y. This is a well-defined inner product on $\mathbb{C}^n$. The induced norm is the well-known Euclidean vector norm.

Assume that x_k has been computed, hence we set $x_{k+1} = x_{k-1} + \omega(\bar{x} - x_{k-1})$. Consequently, we have

$$\begin{aligned} r_{k+1} &= b - Ax_{k+1} \\ &= b - \omega A\bar{x} - (1-\omega)Ax_{k-1} \\ &= \omega(b - A\bar{x}) + (1-\omega)(b - Ax_{k-1}) \\ &= \omega\bar{r} + (1-\omega)r_{k-1} \\ &= r_{k-1} + \omega(\bar{r} - r_{k-1}) \\ &= r_{k-1} - \omega(r_{k-1} - \bar{r}). \end{aligned} \tag{9}$$

The value of ω can be determined by using the idea of one dimensional oblique projection technique (Minimal Residual method), [19, Chapter 5] as follows. We comment that, the idea is essentially MR-DOR (Minimal Residual with DOR step) which was originally described in [1].

In fact the $(k+1)$-th iterate is computed so that

$$\|b - Ax_{k+1}\|_2 = \min_{y \in \mathcal{S}} \|b - Ay\|_2,$$

where $\mathcal{S} = \{y \mid y = \bar{\omega}\bar{x} + (1-\bar{\omega})x_{k-1} \quad \text{for some real } \bar{\omega}\}$. It is not difficult to observe that the preceding optimality condition is equivalent to find ω by imposing the orthogonality condition

$$\langle r_{k+1}, r_{k-1} - \bar{r} \rangle = 0,$$

the above orthogonality condition implies that

$$\begin{aligned} 0 &= \langle r_{k+1}, r_{k-1} - \bar{r} \rangle \\ &= \langle r_{k-1}, r_{k-1} - \bar{r} \rangle - \omega \langle r_{k-1} - \bar{r}, r_{k-1} - \bar{r} \rangle, \end{aligned}$$

which results that

$$\omega = \frac{\langle r_{k-1}, r_{k-1} - \bar{r} \rangle}{\langle r_{k-1} - \bar{r}, r_{k-1} - \bar{r} \rangle}.$$

In conclusion, as a possible iterative scheme with DORS for complex symmetric linear system of equations $Ax = b$ with $A = W + iT$, we propose Algorithm 1. The resulting method is called the DMHHS method. The performance of the algorithm is numerically compared to the MHSS method in Section 4. We comment that Lines 5 and 6 are equivalent to one iterate of the MHSS method.

Algorithm 1: DMHHS method to solve $Ax = b$ where $A = W + iT$.

1 **Input:** The coefficient matrix A, W and T, the right-hand side b, $\alpha, \omega > 0$ and x_0.
2 Set $x_0 = 0$ and compute x_1 by applying one iterate of (8);
3 **begin**
4 **for** k=1,2,. . . , *until convergence* **do**
5 Solve $(\alpha I + W)\tilde{x} = (\alpha I - iT)x_k + b$ to find $\tilde{x}$;
6 Solve $(\alpha I + T)\bar{x} = (\alpha I + iW)\tilde{x} - ib$ to find $\bar{x}$;
7 $\bar{r} = b - A\bar{x}$;
8 $\omega = \frac{\mathrm{Re}(r_{k-1}^H(r_{k-1}-\bar{r}))}{\mathrm{Re}((r_{k-1}-\bar{r})^H(r_{k-1}-\bar{r}))}$;
9 $x_{k+1} = \omega\bar{x} + (1-\omega)x_{k-1}$;
10 $r_{k+1} = \omega\bar{r} + (1-\omega)r_{k-1}$;
11 **end**
12 **end**

2.3. Richardson method with DORS for normal equations

In this section we propose an iterative method to solve linear system of equations (1) where A is possibly rectangular and rank deficient matrix.

Let us consider the well-known Richardson method [19] to solve $A^TAx = A^Tb$ which is given as follows:

$$x_{k+1} = x_k + \mu A^T(b - Ax_k) = (I - \mu A^TA)x_k + \mu A^Tb, \qquad k = 0, 1, 2, \ldots, \tag{10}$$

where μ is a given positive real parameter and $A \in \mathbb{R}^{n\times m}$.

In [20, Theorem 2.5], Salkuyeh and Beik established a sufficient condition for the convergence of the iterative method (10). More precisely, it has been shown that the iteration matrix corresponding to (10), i.e., $\mathcal{G} = (I - \mu A^T A)$, is semi-convergent. In addition, the optimal value for the parameter μ is obtained for which (10) reaches to its best convergence rate.

Here, we first consider application of the Richardson method with DORS for solving $A^T Ax = A^T b$. The corresponding iterative scheme is of the form (3) with $\mathcal{G} = (I - \mu A^T A)$ and $f = \mu A^T b$ where μ is a given positive parameter. Therefore, the Richardson method with DORS is given as follows:

$$\begin{aligned} \bar{x}_{k+1} &= x_k + \mu A^T r_k, \\ x_{k+1} &= \omega \bar{x}_{k+1} + (1-\omega) x_{k-1}, \quad k = 1, 2, \ldots, \end{aligned} \tag{11}$$

where $r_k = b - Ax_k$ is the k-th residual vector.

In general, finding the optimal parameter in the iterative schemes (10) and (11) needs information about the extreme eigenvalues of $A^T A$ which would be too expensive in practice; see [19]. Notice that when the coefficient matrix is singular, the minimum nonzero and maximum eigenvalues must be found to determine the optimum parameter μ in (10), see [20]. To overcome this drawback, one may choose the parameters μ and ω in progressive manner instead of using fixed parameters. In fact, we mainly use the idea of one-dimensional projection techniques in the following manner [19, Chapter 5].

Let x_k be the k-th approximate solution, we find μ^* in (11) so that $\bar{r}^*_{k+1} = b - A\bar{x}^*_{k+1}$ satisfies

$$\|\bar{r}^*_{k+1}\|_2 = \min_{x \in \tilde{\mathcal{S}}} \|b - Ax\|_2, \tag{12}$$

where $\bar{x}^*_{k+1} = x_k + \mu^* A^T r_k$ and $\tilde{\mathcal{S}} = \{x \mid x = x_k + \mu A^T r_k\}$. It can be seen that (12) holds iff

$$\left\langle \bar{r}^*_{k+1}, Ap_k \right\rangle_2 = 0,$$

where $p_k = A^T r_k$. Evidently, we have $\bar{r}^*_{k+1} = r_k - \mu^* Ap_k$. Therefore, under the assumption $\langle Ap_k, Ap_k \rangle_2 \neq 0$, the preceding orthogonality condition implies that

$$\mu^* = \frac{\langle r_k, Ap_k \rangle_2}{\langle Ap_k, Ap_k \rangle_2}.$$

Notice that the proposed way for choosing μ can also be used while using (10). Therefore the results of the following proposition hold when μ is chosen in both iterative schemes (10) and (11) by the mentioned projection technique at each step. We comment that μ^* is derived using the idea of Minimal Residual method [19] which is also used in [1]. The proof is straightforward, hence we omit it.

Proposition 2.5. *Assume that x^* is a solution of $Ax = b$. Then,*

$$\langle p_k, x^* - x_k \rangle_2 = \langle r_k, r_k \rangle_2 ,$$

and

$$\langle Ap_k, r_k \rangle_2 = \langle p_k, p_k \rangle_2 ,$$

where $p_k = A^T r_k$, $r_k = b - Ax_k$ and x_k is the k-th approximate solution computed by either (10) *or* (11)*.*

Remark 2.6. *From the first relation of previous proposition, we can immediately conclude that $p_k = 0$ implies that $r_k = 0$ which shows that x_k is the exact solution of $Ax = b$. The second relation in Proposition 2.5 reveals that $Ap_k = 0$ implies $p_k = 0$ which, as pointed earlier, ensures that x_k is the exact solution of $Ax = b$. Therefore the earlier assumption $\langle Ap_k, Ap_k \rangle_2 \neq 0$ is not a kind of restriction. Because $Ap_k = 0$ results x_k is the exact solution of $Ax = b$.*

In order to determine parameter ω in iterative scheme (11), again, one can use the idea of one-dimensional techniques. Notice that in the delayed step of (11), we have

$$x_{k+1} = \omega\bar{x}_{k+1} + (1-\omega)x_{k-1}.$$

Or equivalently,

$$x_{k+1} = x_{k-1} + \omega d_k,$$

where $d_k = \bar{x}_{k+1} - x_{k-1}$. Here we find ω^* so that

$$\|r^*_{k+1}\|_2 = \min_{x\in\hat{\mathcal{S}}} \|b - Ax\|_2, \tag{13}$$

where $r^*_{k+1} = b - Ax^*_{k+1}$ and $x^*_{k+1} = x_{k-1} + \omega^* d_k$ and $\hat{\mathcal{S}} = \{x \mid x = x_{k-1} + \omega d_k\}$. It can be verified that (13) satisfies iff

$$\left\langle r^*_{k+1}, Ad_k \right\rangle_2 = 0.$$

If $\langle Ad_k, Ad_k\rangle_2 \neq 0$, it is not difficult to see that above orthogonality condition implies that

$$\omega^* = \frac{\langle r_k, Ad_k\rangle_2}{\langle Ad_k, Ad_k\rangle_2}.$$

Note that $Ad_k = r_{k-1} - \bar{r}_{k+1}$. Hence $\langle Ad_k, Ad_k\rangle_2 = 0$ implies $\bar{r}_{k+1} = r_{k-1}$. From the way that μ is chosen, we have $\|\bar{r}_{k+1}\|_2 \leq \|r_k\|_2$ and the way was exploited for choosing ω in the previous step results $\|r_k\|_2 \leq \|r_{k-1}\|_2$. This shows as soon as $Ad_k = 0$, the following equality holds

$$\|\bar{r}_{k+1}\|_2 = \|r_k\|_2.$$

It can be seen that the above relation concludes $\langle r_k, Ap_k\rangle_2 = 0$. Now from Proposition 2.5 and Remark 2.6, we deduce that $\|\bar{r}_{k+1}\|_2 = \|r_k\|_2$ implies x_k is the exact solution of $Ax = b$.

Now using the above discussions, we present Algorithm 2. Considering the way of choosing μ and ω in the above progressive manner, it is not difficult to establish the following proposition. The proof follows from straightforward computations, so it is omitted.

Proposition 2.7. *Assume that k steps of Algorithm 2 is performed. Suppose that $\hat{x} = x_k + \hat{\omega}p_k$ where $\hat{\omega}$ is an arbitrary constant. Then*

$$\|r_{k+1}\|_2 \leq \|\hat{r}\|_2, \tag{14}$$

where r_{k+1} and $\hat{r}$ are respectively residual corresponding to x_{k+1} and $\hat{x}$.

Remark 2.8. *From Proposition 2.7, we deduce that Algorithm 2 outperforms the Richardson method for solving singular normal equations. In fact, this follows from the fact that the k-th approximate solution computed by the Richardson method belongs to $x_k + span\{p_k\}$. Notice that if we choose μ in the above proposed way and set $\omega = 1$ then the resulting method still converges faster than Richardson method.*

Proposition 2.9. *The sequence of approximate solutions $\{x_k\}_{k=1}^{\infty}$, computed by Algorithm 2, converges to a solution of $Ax = b$.*

Proof. Assume that k steps of the algorithm have been performed. From the earlier discussions, it has been seen that $\|\bar{r}_{k+1}\|_2 \leq \|r_k\|_2$. Therefore, in view of Proposition 2.7, we deduce $\|r_{k+1}\|_2 \leq \|\bar{r}_{k+1}\|_2 \leq \|r_k\|_2$. As seen in the algorithm, we have $t_k = Ad_k$. From the previous discussions, it is known that $\langle \bar{r}_{k+1}, t_k\rangle_2 = 0$ which shows that

$$\begin{aligned}\langle \bar{r}_{k+1}, \bar{r}_{k+1}\rangle_2 &= \langle r_k - \mu t_k, \bar{r}_{k+1}\rangle_2 \\ &= \langle r_k, \bar{r}_{k+1}\rangle_2 .\end{aligned}$$

Algorithm 2: Richardson method with DORS for $A^TAx = A^Tb$ (A^TA is possibly singular).

Input: The coefficient matrix A, the right-hand side b and x_0.
Set $p_0 = A^Tr_0$ and $t_0 = Ap_0$ where $r_0 = b - Ax_0$;
Compute $x_1 = x_0 + \mu p_0$ where $\mu = \frac{\langle r_k, t_0\rangle_2}{\langle t_0, t_0\rangle_2}$;
begin
  for k=1,2,. . . , *until convergence* **do**
    $r_k = b - Ax_k$;
    $p_k = A^Tr_k$;
    $t_k = Ap_k$;
    $\mu = \frac{\langle r_k, t_k\rangle_2}{\langle t_k, t_k\rangle_2}$;
    $\bar{x} = x_k + \mu p_k$;
    $\bar{r} = r_k - \mu t_k$;
    $\phi_k = r_{k-1} - \bar{r}$;
    $\omega = \frac{\langle r_{k-1}, \phi_k\rangle_2}{\langle \phi_k, \phi_k\rangle_2}$;
    $x_{k+1} = x_{k-1} + \omega(\bar{x} - x_{k-1})$;
    $r_{k+1} = r_{k-1} - \omega\phi_k$;
  end
end

Now, it can be observed that

$$\begin{aligned}\langle r_{k+1}, r_{k+1}\rangle_2 &\leq \langle \bar{r}_{k+1}, \bar{r}_{k+1}\rangle_2 \\ &= \langle r_k, \bar{r}_{k+1}\rangle_2 \\ &= \langle r_k, r_k\rangle_2 - \mu \langle r_k, Ap_k\rangle_2 \\ &= \langle r_k, r_k\rangle_2 \left(1 - \frac{\langle r_k, Ap_k\rangle_2^2}{\langle Ap_k, Ap_k\rangle_2 \langle r_k, r_k\rangle_2}\right).\end{aligned}$$

Here we comment that the well-known Cauchy-Schwarz inequality implies

$$1 - \frac{\langle r_k, Ap_k\rangle_2^2}{\langle Ap_k, Ap_k\rangle_2 \langle r_k, r_k\rangle_2} \leq 1.$$

Notice that the above inequality holds strictly when $\langle r_k, Ap_k\rangle_2 \neq 0$. Otherwise $\langle r_k, Ap_k\rangle_2 = 0$ concludes $p_k = 0$ which implies that x_k is the exact solution of $Ax = b$ by Proposition 2.5 and Remark 2.6. Therefore, $\|r_k\|_2$ decreases monotonically toward zero at each step of the algorithm and this fact completes the proof. □

We end this part with the following brief remark which shows that if we choose a special kind of initial guess in Algorithm 2, then it converges to the unique least-norm solution of $Ax = b$.

Remark 2.10. *We comment here that if we choose the initial guess $x_0 \in Range(A^T)$ (for simplicity x_0 can be chosen as a zero vector) then the sequence of approximate solutions x_k for $k = 1, 2, \ldots$, belongs to $Range(A^T)$. Hence if the iterative scheme* (11) *is convergent, the produced sequence of approximate solutions converges to the unique least-norm solution by Theorems 2.3 and 2.4.*

3. An Algorithm with DORS for Matrix Equations

In this section, we briefly show how the results in Subsection 2.3 can be used to improve the speed of convergence of the well-known gradient-based (GB) iterative method to solve matrix equations. Our

examined numerical tests illustrate that applying the DORS can significantly decrease the number of iterations that the GB method requires to be convergent. Meanwhile, there is no need to assume that the mentioned problem has a unique solution. Comparing to GB-type methods, the following proposed algorithm is free of any parameters. For further dissuasions on improving the convergence speed of the GB iterative method and its cyclic variant in progressive way, we refer the reader to [5, 6].

Several iterative methods have been proposed for solving linear matrix equations in the literature; for instance see [3, 4, 9–15] and the references therein. The earlier cited works have been mainly proposed the GB approaches to solve different kinds of matrix equations under the restrictions that the problem has a unique solution.

More recently, Sheng and Sun [21] proposed an algorithm to solve the following coupled matrix equations

$$(A_1XB_1, A_2XB_2) = (F_1, F_2), \tag{15}$$

where the matrices A_i, B_i and F_i for $i = 1, 2$ are given with suitable dimensions. The following iterative scheme (namely Algorithm 3) has been proposed for solving (15). The convergence of Algorithm 3 has been proved under the assumptions that the coupled matrix equations (15) have a unique solution,

$$0 < \mu_1 < \frac{1}{\bar{\omega}\|A_1\|_2^2\|B_1\|_2^2} \quad \text{and} \quad 0 < \mu_1 < \frac{1}{(1-\bar{\omega})\|A_2\|_2^2\|B_2\|_2^2},$$

where $0 < \bar{\omega} < 1$ is given. As seen there is no suggestion for choosing optimum values of μ_1, μ_2 and $\bar{\omega}$. Obviously finding suitable values of these parameters is not easy in general. On the other hand the theoretical results have been only proved under the restriction of existence of a unique solution. In [21], the reported numerical results show that the RGB method surpasses the GB method by choosing suitable parameters.

Algorithm 3: The relaxed gradient-based (RGB) iterative algorithm [21].

Input: The matrices A_1, A_2, B_1, B_2 F_1 and F_2.
Choose the initial guess $X(0)$, the parameters μ_1, μ_2 and appropriative positive number ω such that $0 < \bar{\omega} < 1$;
begin
  for *k=1,2,. . . , until convergence* **do**
    $X_1(k) = X(k-1) - \mu_1 A_1^T(A_1X(k-1)B_1 - F_1)B_1^T$;
    $X_2(k) = X(k-1) - \mu_2 A_2^T(A_2X(k-1)B_2 - F_2)B_2^T$;
    $X(k) = \bar{\omega}X_1(k) + (1-\bar{\omega})X_2(k)$;
  end
end

Consider the following general coupled Sylvester matrix equations

$$\sum_{j=1}^{q} A_{ij}X_jB_{ij} = C_i, \qquad i = 1, \ldots, p, \tag{16}$$

where $A_{ij} \in \mathbb{R}^{r_i \times n_j}$, $B_{ij} \in \mathbb{R}^{m_j \times k_i}$ and $C_i \in \mathbb{R}^{r_i \times k_i}$ are given matrices and $X_j \in \mathbb{R}^{n_j \times m_j}$ are the unknown matrices for $j = 1, 2, \ldots, q$.

For a given matrix $X \in \mathbb{R}^{n \times p}$, in the sequel, the notation "vec(X)" stands for a vector of dimension np obtained by stacking the columns of the matrix X. Using the "vec(.)" operator and properties of Kronecker product, it is seen that solving (16) is equivalent to solving the following linear system of equations,

$$\sum_{j=1}^{q} (B_{ij}^T \otimes A_{ij})\text{vec}(X_j) = \text{vec}(C_i), \qquad i = 1, \ldots, p, \tag{17}$$

where $\otimes$ denotes the well-known Kronecker product. For simplicity, we rewrite (17) as follows:

$$\mathcal{M}\mathcal{X} = \mathcal{B}, \tag{18}$$

in which $\mathcal{M}$ is a block matrix and its (i, j)-th block is given by $M_{ij} = B_{ij}^T \otimes A_{ij}$ ($i = 1, \ldots, p$ and $j = 1, 2, \ldots, q$).

In [20], it has been described that the GB algorithm for solving (16) is in fact the matrix form of the Richardson method to solve normal equations $\mathcal{M}^T\mathcal{M}\mathcal{X} = \mathcal{M}^T\mathcal{B}$. Also, it has been shown that the restriction of the existence of a unique solution can be ignored when the convergence of the GB algorithm is studied for solving (16). An interval for the fixed parameter μ in the GB algorithm has been established for which the algorithm converges to a solution of (16). In addition, the following optimum value for the fixed parameter μ has been determined,

$$\mu_{opt} = \frac{2}{\sigma_1^2 + \sigma_r^2}, \tag{19}$$

where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r$ are the nonzero singular values of $\mathcal{M}$. However, as seen, it may become too expensive to compute the optimum value of μ in general situations.

Although we do not form the linear system (17) in practice, it helps us to figure out how Algorithm 2 can be extended to the matrix form for solving (16). Now we summarize the extension of Algorithm 2 for solving (16) in Algorithm 4. The derivation of the matrix form Algorithm 2 follows from the same techniques used in [20], therefore we omit the details.

Remark 3.1. *In the numerical experiments by DGB-version 1, we refer to Algorithm 4. In order to test how a DORS works by itself in the algorithm, in Lines 5 and 11, one may compute μ by* (19) *which is the optimum value of the parameter in the GB method. In this case we call Algorithm 4 by DGB-version 2. Numerical results show that both DGB-version 1 and DGB-version 2 need less number of iterations than the GB method to be convergent.*

4. Numerical Experiments

In this section we examine some numerical test problems to illustrate the validity of the theoretical results and the applicability of proposed algorithms. We comment that all of the reported experiments were performed on a 64-bit 2.45 GHz core i7 processor and 8.00GB RAM using MATLAB version 8.3.0532.

As see in the first subsection of Section 2, it is proved that if we have a semi-convergent iterative method after applying the DORS, it remains semi-convergent. We give the following example to numerically test this fact. As seen in both MHSS and Algorithm 1, one may need to solve two linear systems of the form appeared in Lines 5 and 6 of Algorithm 1. Here we used a sparse Cholesky factorization with the symmetric approximate minimum degree (SYMAMD) reordering to solve these systems.

In the sequel, under "Iter" and "CPU", we report the required number of iterations and CPU-times for reaching mentioned stooping criterions, respectively.

Example 4.1. [2, 8] *Let $n = m^2$ for a given m and consider the singular linear system $Ax = b$ where $A = W + iT \in \mathbb{C}^{n\times n}$ such that*

$$W = I_m \otimes V_c + V_c \otimes I_m \in \mathbb{R}^{n\times n} \quad \text{and} \quad T = \frac{\gamma}{2m}\left(I_m \otimes U_c + U_c \otimes I_m\right) \in \mathbb{R}^{n\times n},$$

with

$$\begin{aligned} V_c &= V - \left(e_1 e_m^T + e_m e_1^T\right) \in \mathbb{R}^{m\times m}, \\ U_c &= U - \left(e_1 e_{m-1}^T + e_{m-1} e_1^T + e_a e_m^T + e_m e_a^T\right) \in \mathbb{R}^{m\times m}, \end{aligned}$$

where

$$V = tridiag(-1, 2, -1) \in \mathbb{R}^{m\times m} \quad \text{and} \quad U = pentadiag(-1, -1, 4, -1, -1) \in \mathbb{R}^{m\times m},$$

Algorithm 4: DGB for solving (16).

Input: The coefficient matrix A_{ij}, B_{ij}, the right-hand sides C_i for $i = 1, \dots, p$ and $j = 1, 2, \dots, q$.
Set $X_j(0) = 0$ and $R_i(0) = C_i$ for $i = 1, \dots, p$ and $j = 1, 2, \dots, q$;
Set $P_j(0) = \sum_{i=1}^{p} A_{ij}^T R_i(0) B_{ij}^T$ for $j = 1, 2, \dots, q$;
Set $T_i(0) = \sum_{j=1}^{q} A_{ij} P_j(0) B_{ij}$ for $i = 1, \dots, p$;
Compute $X_j(1) = X_j(0) + \mu P_j(0)$ for $j = 1, 2, \dots, q$ where $\mu = \frac{\sum_{i=1}^{p} \langle R_i(0), T_i(0) \rangle_F}{\sum_{i=1}^{p} \langle T_i(0), T_i(0) \rangle_F}$;
begin
  for *k=1,2,. . . , until convergence* **do**
    $R_i(k) = C_i - \sum_{j=1}^{q} A_{ij} X_j(k) B_{ij}$ for $i = 1, 2, \dots, p$;
    $P_j(k) = \sum_{i=1}^{p} A_{ij}^T R_i(k) B_{ij}^T$ for $j = 1, 2, \dots, q$;
    $T_i(k) = \sum_{j=1}^{q} A_{ij} P_j(k) B_{ij}$ for $i = 1, \dots, p$;
    $\mu = \frac{\sum_{i=1}^{p} \langle R_i(k), T_i(k) \rangle_F}{\sum_{i=1}^{p} \langle T_i(k), T_i(k) \rangle_F}$;
    $\bar{X}_j = X_j(k) + \mu P_j(k)$ for $j = 1, 2, \dots, q$;
    $\bar{R}_i = R_i(k) - \mu T_i(k)$ for $i = 1, 2, \dots, p$;
    $\Phi_i = R_i(k-1) - \bar{R}_i$ for $i = 1, 2, \dots, p$;
    $\omega = \frac{\sum_{i=1}^{p} \langle R_i(k-1), \Phi_i \rangle_F}{\sum_{i=1}^{p} \langle \Phi_i, \Phi_i \rangle_F}$;
    $X_j(k+1) = X_j(k-1) + \omega(\bar{X} - X_j(k-1))$ for $j = 1, 2, \dots, q$;
    $R_i(k+1) = R_i(k-1) - \omega \Phi_i$ for $i = 1, 2, \dots, p$;
  end
end

here I_m stands for the identity matrix of dimension m, e_i denotes the i-th column of I_m ($i = 1, m-1, m$) and $e_a = e_1 + e_2$. The right-hand side b is constructed so that $\hat{x} = (1, 2, \dots, n)^T$ is a solution of $Ax = b$, i.e., $b = A\hat{x}$.

We use the MHSS method [8] and Algorithm 1 to solve singular linear system of equations $Ax = b$ in this example. The iterations are terminated as soon as the k-th approximate solution x_k satisfies

$$\frac{\|b - Ax_k\|_2}{\|b\|_2} < 10^{-6},$$

and x_0 is taken to be zero. The corresponding numerical results are reported in Table 1. Our obtained results illustrate that the DMHSS method converges for solving the singular linear system mentioned in this example.

As seen, the disclosed results in Table 1 confirms that the semi-convergence of MHSS method implies the semi-convergence of the DMHSS method. Nevertheless, in this case, improvements induced by the use of DORS in the MHSS method is quite moderate.

In the sequel, test examples we compare the performance of the GB, RGB, DGB-version 1 and DGB-version 2. The explanation about the last two terms is given in Remark 3.1. As seen both DGB-version 1 and DGB-version 2 outperform the GB method and the recently proposed RGB method [21].

Example 4.2. [21, Example 4.1] *Consider the succeeding coupled matrix equations*

$$A_i X B_i = F_i, \qquad i = 1, 2, \tag{20}$$

		MHSS method			DMHSS method		
(m,n)	(γ,α)	Iter	CPU	RES	Iter	CPU	RES
(64,4096)	$(10^1,0.09)$	73	0.617	8.9619E-07	61	0.513	9.2362E-07
	$(10^2,0.33)$	83	0.683	9.7154E-07	67	0.568	8.3753E-07
	$(10^3,1.33)$	49	0.413	8.1816E-07	39	0.409	8.1322E-07
	$(10^4,1.08)$	111	0.922	9.6670E-07	64	0.547	9.9524E-07
(80,6400)	$(10^1,0.07)$	86	1.329	9.0562E-07	66	1.065	8.7853E-07
	$(10^2,0.24)$	102	1.608	9.9764E-07	76	1.284	9.9173E-07
	$(10^3,0.98)$	60	0.936	9.6464E-07	58	0.925	8.8036E-07
	$(10^4,0.65)$	98	1.539	9.6905E-07	91	1.415	9.1675E-07
(96,9216)	$(10^1,0.05)$	91	2.064	9.1716E-07	71	1.613	8.8935E-07
	$(10^2,0.18)$	122	2.798	9.6067E-07	87	2.006	8.6586E-07
	$(10^3,0.70)$	72	1.633	9.9390E-07	67	1.522	8.8512E-07
	$(10^4,0.75)$	93	2.112	9.3202E-07	81	1.851	9.5765E-07

Table 1: Numerical results for solving Example 4.1.

where

$$A_1 = \begin{bmatrix} 1.00 & 0.00 \\ 3.00 & 2.00 \\ -2.00 & 5.00 \\ 4.00 & -1.00 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1.00 & 0.50 \\ -2.00 & 1.00 \\ 1.00 & 1.10 \end{bmatrix},$$

$$B_1 = \begin{bmatrix} 3.00 & 0.30 & 6.50 & 1.40 \\ -1.00 & 1.00 & -2.00 & 1.20 \\ 1.00 & -2.00 & 2.00 & 0.50 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 1.00 & 1.10 & 0.80 \\ 1.50 & 1.10 & 0.40 \\ 0.10 & -1.50 & -3.00 \end{bmatrix},$$

and

$$F_1 = \begin{bmatrix} 4.00 & -3.70 & 8.50 & 5.30 \\ 17.00 & -0.90 & 37.50 & 26.50 \\ 4.50 & 32.90 & 13.00 & 15.90 \\ 13.50 & -19.90 & 28.00 & 15.90 \end{bmatrix}, \quad F_2 = \begin{bmatrix} 7.125 & 2.025 & -4.600 \\ -2.950 & 8.850 & 20.400 \\ 10.515 & 5.895 & -1.240 \end{bmatrix}.$$

It can be verified that the coupled matrix equations (20) *has a unique solution given by*

$$X^* = \begin{bmatrix} 1.00 & 2.00 & 3.00 \\ 2.00 & 2.50 & -1.00 \end{bmatrix}.$$

We use GB, RGB [21]*, DGB-version 1 and DGB-version 2 to solve* (20)*. The iteration steps were terminated as soon as*

$$\|X(k) - X^*\|_F < 10^{-8},$$

where $X(0)$ *is taken to be zero matrix. Here we comment that in the GB method the optimal parameter is chosen and for RGB method, we used the best parameters based on the reported results in* [21]*. The GB, RGB, DGB-version 1*

and DGB-version 2 take 425, 195, 6 and 40 iterations, respectively, for convergence with respect to the mentioned stopping criterion. For more details the convergence histories of the algorithms are plotted in Figure 1 in which

$$\delta_k = \sqrt{\frac{\|F_1 - A_1X(k)B_1\|_F^2 + \|F_2 - A_2X(k)B_2\|_F^2}{\|F_1\|_F^2 + \|F_1\|_F^2}}, \tag{21}$$

where $X(k)$ is the k-th computed approximate solution.

Example 4.3. [20, Example 4.2] *Consider the following coupled matrix equations*

$$\begin{cases} A_{11}X_1B_{11} + A_{12}X_2B_{12} = C_1 \\ A_{21}X_1B_{21} + A_{22}X_2B_{22} = C_2 \end{cases} \tag{22}$$

where

$$A_{11} = \begin{bmatrix} 1 & 2 \\ -3 & -6 \end{bmatrix}, \quad B_{11} = \begin{bmatrix} -1 & -1 \\ 2 & 1 \\ -5 & 1 \end{bmatrix},$$

$$A_{12} = \begin{bmatrix} 2 & 1 & 3 \\ 1 & -1 & 0 \end{bmatrix}, \quad B_{12} = \begin{bmatrix} 2 & 9 \\ 0 & -3 \end{bmatrix},$$

$$A_{21} = \begin{bmatrix} 1 & 2 \\ -3 & -6 \\ 1 & 2 \end{bmatrix}, \quad B_{21} = \begin{bmatrix} -1 & -1 & -2 \\ 3 & 1 & -1 \\ 2 & -1 & 1 \end{bmatrix},$$

and

$$C_1 = \begin{bmatrix} 2 & 83 \\ 54 & 57 \end{bmatrix}, \quad C_2 = \begin{bmatrix} 9 & -6 & 15 \\ 65 & 44 & 37 \\ -19 & -28 & 1 \end{bmatrix}.$$

Using, the "vec(.)" operator, $\mathcal{M}\mathcal{X} = \mathcal{B}$ corresponds to (22) *such that*

$$\mathcal{M} = \begin{bmatrix} B_{11}^T \otimes A_{11} & B_{12}^T \otimes A_{12} \\ B_{21}^T \otimes A_{21} & B_{22}^T \otimes A_{22} \end{bmatrix},$$

where $\mathcal{X} = (vec(X_1); vec(X_2))$ and $\mathcal{B} = (vec(C_1); vec(C_2))$. It can be verified that $\mathcal{M}$ is rank deficient and GB method is semi-convergent and its the optimum value is $\mu_{opt} = 0.00109$ which is computed by (19)

Here we compare the performance of GB, DGB-version 1 and DGB-version 2 to solve (22)*. We used the following stopping criterion*

$$\max\left\{\frac{\|R_1(k)\|_F}{\|C_1\|_F}, \frac{\|R_2(k)\|_F}{\|C_2\|_F}\right\} < 10^{-6},$$

where

$$R_1(k) = C_1 - (A_{11}X_1(k)B_{11} + A_{12}X_2(k)B_{12}) \quad \text{and} \quad R_2(k) = C_2 - (A_{21}X_1(k)B_{21} + A_{22}X_2(k)B_{22}),$$

in which $X_i(k)$ $(k = 1, 2)$ is the k-th computed approximate solution. The initial guess matrices $X_1(0)$ and $X_2(0)$ are both taken to be zero. The GB, DGB-version 1 and DGB-version 2 require 207, 9 and 29 iterations, respectively, to satisfy the preceding stopping criterion. For more clarification, we displayed the convergence behaviour of the algorithms in Figure 1 in which

$$\eta_k = \sqrt{\frac{\|R_1(k)\|_F^2 + \|R_2(k)\|_F^2}{\|C_1\|_F^2 + \|C_2\|_F^2}}.$$

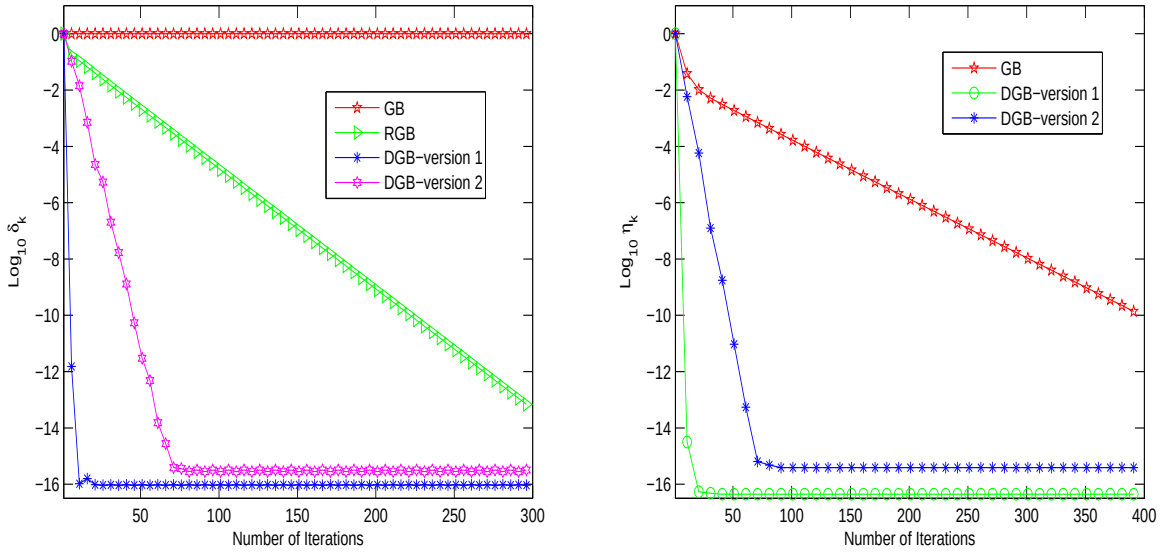

Figure 1: Convergence history; left: Example 4.2; right: Example 4.3

In the above two examples, it is observed that DORS can improve the rate of convergence of GB-type algorithms significantly. The size of the previous examples are too small, so required CPU-times of the algorithms are too close. Now, in order to show the effect of the reduction of the number of iterations on the CPU time, we examine the following test problem with larger dimensions. Due to the larger size of the problem in comparison to the previous two examples, it is expensive to estimate the optimum of value of the GB method. Here we set $\omega = 1$ in Algorithm 4 to observe how the DORS works in the algorithm. In practice, it is known that Algorithm 4 with $\omega = 1$ is as fast as the GB method with its optimum convergence factor.

Example 4.4. [22, Example 4.3] *Here we mention the Sylvester matrix equation,*

$$AX + XB = C, \tag{23}$$

where A,B and C are $n \times n$ matrices and generated by MATLAB *function such that $A = triu(rand(n,n),1) + diag(\alpha + diag(rand(n)))'$, $B = A'$, $C = AX^* + X^*B$ where $X^* = rand(n,n)$. We set $\alpha = 6$, for which the resulting systems is very ill-conditioned. Our experiments, after several runs of the code, illustrate that DGB-version 1 (Algorithm 4) converges faster than the accelerated Jacobi-gradient based iterative (AJGI) algorithm; see [22, Table 3] in which $n = 60$. We comment that AJGI method relies on some parameters which finding their optimum values is an open problem, whereas DGB-version 1 is free of parameter. In the implementation of the algorithms, we used the following stopping criterion,*

$$\bar{\eta}_k := \frac{\|C - AX(k) - X(k)B\|_F}{\|C\|_F} < 10^{-13},$$

where $X(k)$ is the kth approximate solution and $X(0)$ is taken to be zero.

Finally we examine the performance of Algorithm 4 for an image deblurring problem.

Table 2: Numerical comparison results for Example 4.4.

n	Method	ω	$\|X(k) - X^*\|_F / \|X^*\|_F$	$\bar{\eta}_k$	CPU-time(s)	Iteration
60	Algorithm 4	-	3.6608E-14	4.0038E-14	0.0216	51
	Algorithm 4	1	1.1065E-13	9.3335E-14	0.0832	364
100	Algorithm 4	-	9.4566E-14	8.5870E-14	0.0596	79
	Algorithm 4	1	1.4965E-13	9.9664E-14	0.5947	1144
200	Algorithm 4	-	1.2837E-13	8.8398E-14	0.3377	167
	Algorithm 4	1	2.0020E-13	9.9890E-14	11.4891	5337

Method	ω	CPU	Iter
Algorithm 4	-	1.2067	219
Algorithm 4	1	48.3243	10000

Table 3: Numerical results for solving Example 4.5.

Example 4.5. *In this example, we mainly test the applicability of the DORS in Algorithm 4 for the image deblurring problem*

$$A_c X A_r^T = B, \tag{24}$$

where $A_c \in \mathbb{R}^{m\times m}$, $A_r \in \mathbb{R}^{n\times n}$ *and the recorded blurred image* $B \in \mathbb{R}^{m\times n}$ *are given and the unknown* $X \in \mathbb{R}^{m\times n}$ *is the desired sharp image, see* [16] *for further details. We work on Challenge 2 from* [16]. *For this problem* $m = 260$, $n = 300$, $cond(A_c) = 6.2679 \times 10^5$ *and* $cond(A_r) = 1.6739 \times 10^5$. *The blurred image is displayed in Figure 2.*

Here we mainly aim to illustrate the effectiveness of a DORS in an iterative method. Notice that Algorithm 4 without DORS (with $\omega = 1$*) is the matrix form of the steepest decent method. We stopped the iterations as soon as the number of iterates reaches to* 10000 *or*

$$\theta_k = \frac{\|B - A_c X(k) A_r^T\|_F}{\|B\|_F} < 0.0015,$$

where $X(k)$ *is the k-th approximation and* $X(0)$ *is taken to be zero. The restored images plotted in Figure 3. From Table 3 and Figure 3, it is obvious that the DORS can improve the convergence speed of an iterative method. Here we comment that Algorithm 4 with* $\omega = 1$ *converges too slowly so that the computed approximate solutions can not satisfy* $\theta_k < 0.0015$ *in less than* 1000 *iterations.*

5. Conclusion

We have established a sufficient condition for the semi-convergence of an iterative method with DORS to solve the consistent linear system of equations $Ax = b$ where A is singular. In the case that A is a rectangular matrix and possibly rank deficient, we proposed a parameter free approach using the Richardson method with DORS for normal equations. Moreover the discussed results have exploited to construct an algorithm for solving coupled matrix equations. In addition we numerically examined the proposed algorithms to illustrate their effectiveness and to compare their performance with some of the existing approaches in the literature. The performance of the algorithm has been numerically tested for an image deblurring problem.

Figure 2: Blurred image for Example 4.5

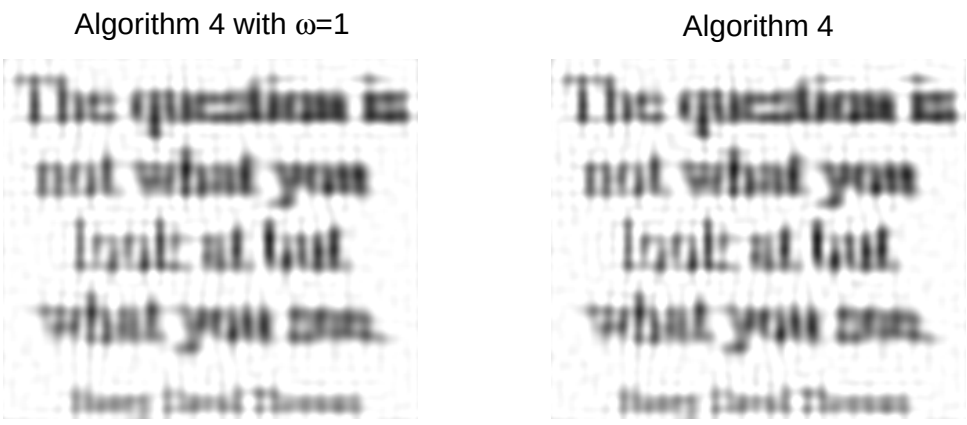

Figure 3: Restored images by Algorithm 4 for Example 4.5

Acknowledgments

The authors would like to express their sincere gratitude to anonymous referees for their helpful comments which have improved the quality of the paper. The authors are also grateful to Professor Dijana Mosić for managing the review process.

References

[1] M. Antuono and G. Colicchio, Delayed over-relaxation for iterative methods, Journal of Computational Physics 321 (2016) 829–907.
[2] Z. Z. Bai, On semi-convergence of Hermitian and skew-Hermitian splitting methods for singular linear systems, Computing 89 (2010) 171–197.
[3] F. P. A. Beik and D. K. Salkuyeh, On the global Krylov subspace methods for solving general coupled matrix equation, Computers & Mathematics with Applications 62 (2011), no. 11, 4605–4613.
[4] F. P. A. Beik, D. K. Salkuyeh and M. M. Moghadam, Gradient-based iterative algorithm for solving the generalized coupled Sylvester-transpose and conjugate matrix equations over reflexive (anti-reflexive) matrices, Transactions of the Institute of Measurement and Control 36 (2014) 99–110.
[5] F. P. A. Beik, A modified iterative algorithm for the (Hermitian) reflexive solution of the generalized Sylvester matrix equation, Transactions of the Institute of Measurement and Control 36 (2014), no. 6, 815–827.
[6] F. P. A. Beik and D. K. Salkuyeh, A cyclic iterative approach and its modified version to solve coupled Sylvester-transpose matrix equations, Linear Multilinear Algebra 65 (2016) 2406–2423.
[7] A. Berman and R. J. Plemmons, Nonnegative Matrices in the Mathematical Sciences, Academic Press, New York, third edition, 1979. Reprinted by SIAM, Philadelphia, 1994.
[8] F. Chen and Q. Q. Liu, On semi-convergence of modified HSS iteration methods, Numerical Algorithms 64 (2013) 507–518.
[9] M. Dehghan and M. Hajarian, Two iterative algorithms for solving coupled matrix equations over reflexive and anti-reflexive matrices, Computational & Applied Mathematics 31 (2012) 353–371.
[10] F. Ding and T. Chen, Iterative least-squares solutions of coupled Sylvester matrix equations, Systems & Control Letters 54 (2005) 95–107.
[11] F. Ding and T. Chen, On iterative solutions of general coupled matrix equations, SIAM Journal on Control and Optimization 44 (2006) 2269-2284.
[12] F. Ding, Coupled-least-squares identification for multivariable systems, IET Control Theory and Applications 7 (2013) 68–79.
[13] F. Ding, P. X. Liu and J. Ding, Iterative solutions of the generalized Sylvester matrix equations by using the hierarchical identification principle, Applied Mathematics and Computation 197 (2008) 41–50.
[14] J. Ding, Y. Liu and F. Ding, Iterative solutions to matrix equations of the form $A_iXB_i = F_i$, Computers & Mathematics with Applications 59 (2010) 3500–3507.
[15] M. Hajarian, Efficient iterative solutions to general coupled matrix equations, International Journal of Automation and Computing 10 (2013) 481–486.
[16] P. C. Hansen, J. G. Nagy and D. P. O'Leary, Deblurring Images: Matrices, Spectra, and Filtering. SIAM, Philadelphia, 2006.
[17] A. J. Laub, Matrix Analysis for Scientists and Engineers, SIAM, Philadelphia, 2005.
[18] Z. Peng and Y. Peng, An efficient iterative method for solving the matrix equation $AXB + CYD = E$, Numerical Linear Algebra with Applications 13 (2006) 473–485.
[19] Y. Saad, Iterative Methods for Sparse Linear Systems, PWS press, New York, 1995.
[20] D.K. Salkuyeh and F. P. A. Beik, On the gradient based algorithm for solving the general coupled matrix equations, Transactions of the Institute of Measurement and Control 36 (2014) 375–381.
[21] X. Sheng and W. Sun, The relaxed gradient based iterative algorithm for solving matrix equations, Computers & Mathematics with Applications 74 (2017) 597–604.
[22] Z. Tian, M. Tian, C. Gu and X. Hao, An accelerated Jacobi-gradient based iterative algorithm for solving Sylvester matrix equations. Filomat 31 (2017) 2381–2390.
[23] D. M. Young, Iterative Solution for Large Linear Systems, Academic Press, New York, 1971.