AIMS Mathematics

http://www.aimspress.com/journal/Math

AIMS Mathematics, 8(5): 11781–11798.
DOI: 10.3934/math.2023596
Received: 27 December 2022
Revised: 22 February 2023
Accepted: 15 March 2023
Published: 17 March 2023

Research article

Gradient-descent iterative algorithm for solving exact and weighted least-squares solutions of rectangular linear systems

Kanjanaporn Tansri and Pattrawut Chansangiam*

Department of Mathematics, School of Science, King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, Thailand

* **Correspondence:** Email: pattrawut.ch@kmitl.ac.th; Tel: +66935266600;
Fax: +6602329840011 ext. 284.

Abstract: Consider a linear system $Ax = b$ where the coefficient matrix A is rectangular and of full-column rank. We propose an iterative algorithm for solving this linear system, based on gradient-descent optimization technique, aiming to produce a sequence of well-approximate least-squares solutions. Here, we consider least-squares solutions in a full generality, that is, we measure any related error through an arbitrary vector norm induced from weighted positive definite matrices W. It turns out that when the system has a unique solution, the proposed algorithm produces approximated solutions converging to the unique solution. When the system is inconsistent, the sequence of residual norms converges to the weighted least-squares error. Our work includes the usual least-squares solution when $W = I$. Numerical experiments are performed to validate the capability of the algorithm. Moreover, the performance of this algorithm is better than that of recent gradient-based iterative algorithms in both iteration numbers and computational time.

Keywords: gradient-descent; iterative method; least-squares solution; weighted norm; convergence analysis
Mathematics Subject Classification: 65F10, 15A12, 15A60, 26B25

1. Introduction and literature reviews

Throughout, let us denote by $\mathbb{R}^{m\times n}$ the set of all m-by-n real matrices, and $\mathbb{R}^n$ the set of all n-dimensional real vectors. For any real symmetric matrix A, its largest/smallest eigenvalues are denoted by $\lambda_{\max}(A)$ and $\lambda_{\min}(A)$, respectively. We consider a linear algebraic system

$$Ax = b, \tag{1.1}$$

where A is a given coefficient matrix, b is a known constant vector, and x is an unknown vector. Theory of linear systems is a fundamental part of linear algebra. Computations of solutions of the linear system are an attractive research topic in numerical linear algebra, and play an essential role in data science, statistics, theoretical physics, signal processing, control engineering, economics, etc; see e.g. [1]. In an ideal case, we can solve for an exact solution of the linear system using traditional methods, such as, Gaussian elimination and matrix factorization. When the system has no solution, we can seek for a least-squares solution which minimizes the squared norm-error $\|Ax - b\|^2$. Direct computational methods for this case are using Moore-Penrose inverses and solving the associated normal equation. However, such direct methods are suitable only when the coefficient matrix A is of small dimension. For large/moderate systems, well-approximate solutions are enough in practical applications.

1.1. Stationary iterative methods

There are many ideas to create an iterative scheme for producing approximate solutions of Eq (1.1). A group of stationary iterative methods are formulated by a recursive equation

$$x^{(i+1)} = Tx^{(i)} + c,$$

where T is an iteration matrix derived from A, and c is a vector derived from A and b. Then the sequence $x^{(i)}$ of approximate solutions converges to the exact solution for any initial vector $x^{(0)}$ if and only if T has spectral radius less than 1; see e.g. [2, Ch. 9]. Classical stationary iterative methods are the Jacobi method, Gauss-Siedel method, and SOR method. All three of them start with decomposing $A = D + L + U$, where D, L, U are the diagonal/lower/upper-triangular parts of A, respectively. A group of methods developed from these methods include: JOR method [3], ESOR method [4], AOR method [5], weighted Jacobi method [6], and scheduled relaxation Jacobi method [7,8]. Unfortunately, the above iterative methods are guaranteed to be applicable when coefficient matrices are of specific forms. For example, AOR method is applicable when A is an irreducible matrix with weak diagonal dominance.

1.2. Gradient-based iterative methods

A general form of stationary iterative methods was discussed in the work of Ding and Chen [9]:

$$x^{(i+1)} = x^{(i)} + \mu G(b - Ax^{(i)}), \tag{1.2}$$

where G is an associated matrix and $\mu > 0$ is a convergent factor. Equation (1.2) includes the Jacobi method when $G = D^{-1}$ and $\mu = 1$. When $G = L + U$ and $\mu = 1$, Eq (1.2) reduces to Gauss-Siedel method. When $G = A^T$, this method is called the gradient-based iterative (GI) method.

Algorithm 1: GI algorithm for solving Eq (1.1).

```
Choose $x^{(0)} \in \mathbb{R}^n$ and compute $x^{(i+1)} = x^{(i)} + \mu A^T(b - Ax^{(i)})$,
update $i$;
```

According to [9], with $0 < \mu < 2/\lambda_{\max}(AA^T)$, the generated sequence $x^{(i)}$ converges to the exact solution for any given initial $x^{(0)}$. When Eq (1.1) is inconsistent and $\operatorname{rank} A = n$, we shall seek for an LS solution as follows.

```
Algorithm 2: LSI algorithm for solving Eq (1.1).
    Choose $x^{(0)} \in \mathbb{R}^n$ and compute $x^{(i+1)} = x^{(i)} + \mu(A^TA)^{-1}A^T(b - Ax^{(i)})$,
    update $i$;
```

The work [10] shows that, for $0 < \mu < 2$, the iterative solution $x^{(i)}$ produced by Algorithm 2 converges to the LS solution for any initial $x^{(0)}$.

In the last decade, many authors adapted the idea of GI method to derive iterative methods for linear matrix equations, such as the Sylvester matrix equation $AX + XB = C$. Note that the Sylvester equation includes the linear system (1.1) as a special case. In the following, we rewrite certain matrix algorithms to be fit with the linear system.

```
Algorithm 3: RGI algorithm for solving Eq (1.1).
    Choose $x_1^{(0)}$ and $x_2^{(0)} \in \mathbb{R}^n$. Compute
    $x^{(i)} = \hat{\omega}x_1^{(i)} + (1 - \hat{\omega})x_2^{(i)}$,
    $x_1^{(i+1)} = x^{(i)} + (1 - \hat{\omega})\mu A^T(b - Ax^{(i)})$,
    $x_2^{(i+1)} = x^{(i)}$,
    update $i$;
```

The RGI algorithm was proposed in [11] by introducing a relaxed factor $\hat{\omega} \in (0, 1)$. This algorithm has been proved to be applicable when $0 < \mu < \{\hat{\omega}(1 - \hat{\omega})\lambda_{\max}(AA^T)\}^{-1}$.

```
Algorithm 4: MGI algorithm for solving Eq (1.1).
    Choose $x_1^{(0)}, x_2^{(0)} \in \mathbb{R}^n$ and compute $x^{(0)} = (x_1^{(0)} + x_2^{(0)})/2$. In the step of computing
    $x_1^{(i+1)} = x^{(i)} + \mu A^T(Ax^{(i)} - b)$,
    $x^{(i)} = (x_1^{(i+1)} + x_2^{(i)})/2$,
    $x_2^{(i+1)} = x^{(i)}$,
    $x^{(i+1)} = (x_1^{(i+1)} + x_2^{(i+1)})/2$,
    update $i$;
```

In [12], MGI algorithm was proposed and proved to be applicable when $0 < \mu < 2/\|A\|_F^2$.

```
Algorithm 5: AGBI algorithm for solving Eq (1.1).
    Choose $x_1^{(0)}, x_2^{(0)} \in \mathbb{R}^n$ and compute
    $x^{(i)} = (1 - \bar{\omega})x_1^{(i)} + \bar{\omega}x_2^{(i)}$,
    $x_1^{(i+1)} = x^{(i)} + \bar{\omega}\mu(b - Ax^{(i)})$,
    $x^{(i)} = (1 - \bar{\omega})x_1^{(i+1)} + \bar{\omega}x_2^{(i)}$,
    $x_2^{(i+1)} = x^{(i)}$,
    update $i$;
```

In [13], AGBI algorithm was proposed and proved to be applicable when $0 < \mu < 2/(\bar{\omega}\|A\|_2^2)$. Another interesting GI method which can be adapted for the linear system starts with decomposing the (square) coefficient matrices to be the sum of its diagonal part and others, and then applies GI technique; see e.g. [14–16]. See more GI algorithms to solve for exact or LS solutions in [17, 18] and references therein. Recently, GI technique is adapted with optimization, so that we can compute the best step size for each iteration; see e.g. [19–21].

1.3. *Our work*

This paper is a continuation of [19]. Instead of finding an exact solution, we look for a least-squares solution of inconsistent linear systems. We can extend the notion of LS solution by introducing a weighted norm $\|v\|^2 = v^T W v$, where W is a given positive definite matrix. In fact, every vector norm on $\mathbb{R}^n$ induced from inner products must be in this form. In most practical applications, the weighted matrix W is a diagonal matrix whose entries are weights of individual coordinates. Its off-diagonal entries can be cross-correlation weights between data values. We apply gradients and the steepest-descent optimization technique to derive an iterative algorithm, called gradient-descent iterative (GDI) algorithm. When the system (1.1) is inconsistent, the algorithm will construct a sequence of approximate LS solutions with respect to the weighted norm induced by W. It turns out that the sequence of residual errors $\|Ax^{(i)} - b\|_W^2$ converges to the weighted LS error. This means that $x^{(i)}$ is a well-approximate weighted LS solution, where i is a large-enough integer. When the system (1.1) is consistent, the algorithm will construct a sequence of approximate LS solutions converging to an exact solution.

The rest of this paper is organized as follows. In Section 2, we provide preliminaries that will be used in the analysis of matrix algorithms. In Section 3, we derive the GDI algorithm. We make a convergence analysis, including error estimations and convergence rates, to verify theoretical capability of the GDI algorithm; see Section 4. Numerical experiments for consistent/inconsistent cases are illustrated to show the performance of the algorithm, compared to the direct method and GI-type methods; see Section 5. Finally, we conclude the whole work in Section 6.

2. Auxiliary results from matrix analysis

Recall that the spectral norm $\|\cdot\|_2$ and the Frobenius norm $\|\cdot\|_F$ of matrix $A \in \mathbb{R}^{m\times n}$ are defined by

$$\|A\|_2 = \sqrt{\lambda_{\max}(A^T A)}, \qquad \|A\|_F = \sqrt{\operatorname{tr}(A^T A)}.$$

Lemma 2.1 (e.g. [22]). *For any conformable matrices A and B, we have*

1) $\|A^T\|_2 = \|A\|_2$,
2) $\|A^T A\|_2 = \|A\|_2^2$,
3) $\|A^T B\|_F \leq \|A\|_2 \|B\|_F$.

Recall that every inner product on $\mathbb{R}^m$ must be in the form

$$\langle x, y\rangle_W = y^T W x, \quad x, y \in \mathbb{R}^m,$$

where $W \in \mathbb{R}^{m\times m}$ is a positive definite matrix. The weighted inner product induces the weighted norm $\|x\|_W = \sqrt{x^T W x}$. If $W = I$, then the weighted ones are just the usual inner product $\langle x, y\rangle = y^T x$ and the usual norm $\|x\| = \sqrt{x^T x}$. The weighted matrix norm induced by W is defined by $\|A\|_W = \sqrt{\operatorname{tr}(A^T W A)}$ for any $A \in \mathbb{R}^{m\times n}$. In paticular, when $W = I$ we have $\|A\|_I = \|A\|_F$ for any $A \in \mathbb{R}^{m\times n}$.

Lemma 2.2. *Let $W \in \mathbb{R}^{m\times m}$ be positive definite. For any $x, y \in \mathbb{R}^m$ and $A, B \in \mathbb{R}^{m\times m}$, we get*

1) $\langle x, y\rangle = \left\langle W^{-\frac{1}{2}}x, W^{-\frac{1}{2}}y \right\rangle_W$,
2) $\|x\| = \|W^{-\frac{1}{2}}x\|_W$,

3) $\|A\|_W = \|W^{\frac{1}{2}}A\|_F$,
4) $\|AB\|_W \le \|W^{\frac{1}{2}}\|_2\|A\|_2\|W^{-\frac{1}{2}}B\|_W$.

Proof. Let us write $u = W^{-\frac{1}{2}}x$ and $v = W^{-\frac{1}{2}}y$. We have

$$\langle x, y\rangle = y^T x = (W^{\frac{1}{2}}v)^T(W^{\frac{1}{2}}u) = v^T W u = \langle u, v\rangle_W = \left\langle W^{-\frac{1}{2}}x, W^{-\frac{1}{2}}y\right\rangle_W.$$

In particular, we get $\|x\| = \|W^{-\frac{1}{2}}x\|_W$. Since $W \in \mathbb{R}^{m\times m}$ is positive definite matrix, we can write $W = (W^{\frac{1}{2}})^T W^{\frac{1}{2}}$ thus

$$\|A\|_W = \sqrt{\operatorname{tr}(A^T(W^{\frac{1}{2}})^T W^{\frac{1}{2}}A)} = \|W^{\frac{1}{2}}A\|_F.$$

From 3) of Lemma 2.1 and 1), 3) of Lemma 2.2, we obtain

$$\|AB\|_W = \|W^{\frac{1}{2}}AB\|_F \le \|W^{\frac{1}{2}}\|_2\|A\|_2\|B\|_F \le \|W^{\frac{1}{2}}\|_2\|A\|_2\|W^{-\frac{1}{2}}B\|_W.$$

□

Lemma 2.3 (e.g. [2]). *Suppose $W \in \mathbb{R}^{m\times m}$ is any positive definite matrix and $A \in \mathbb{R}^{m\times n}$ is of full-column rank (*$\operatorname{rank} A = n$*). Then the LS solution to the linear system Eq* (1.1) *that minimizes the weighted squared error $\|Ax - b\|_W^2$ is the unique solution x^* to the weighted normal equations*

$$A^T W A x^* = A^T W b \qquad \text{so that} \quad x^* = (A^T W A)^{-1} A^T W b.$$

In this case, the weighted LS error is given by

$$\|Ax^* - b\|_W^2 = \|b^T\|_W^2 - b^T x^* = \|b^T\|_W^2 - b^T W A (A^T W A)^{-1} A^T W b. \tag{2.1}$$

Let $f : \mathbb{R}^m \to \mathbb{R}$ be a strongly convex function, which means that there are two positive constants γ and Γ such that

$$\gamma I \le \nabla^2 f(x) \le \Gamma I, \quad \text{for all } x \in \mathbb{R}^m.$$

Here, the matrix inequality $A \le B$ for real symmetric matrices A and B means that the scalar inequality $x^T A x \le x^T B x$ holds for all $x \in \mathbb{R}^m$.

Lemma 2.4. *Let $W \in \mathbb{R}^{m\times m}$ be positive definite. Then, from the above notations about strongly convex functions, the following bounds hold for all $x, y \in \mathbb{R}^m$:*

$$f(y) \ge f(x) + \left\langle W^{-\frac{1}{2}}\nabla f(x), W^{-\frac{1}{2}}(y - x)\right\rangle_W + \frac{\gamma}{2}\|W^{-\frac{1}{2}}(y - x)\|_W^2, \tag{2.2}$$

$$f(y) \le f(x) + \left\langle W^{-\frac{1}{2}}\nabla f(x), W^{-\frac{1}{2}}(y - x)\right\rangle_W + \frac{\Gamma}{2}\|W^{-\frac{1}{2}}(y - x)\|_W^2. \tag{2.3}$$

Proof. From [23, Ch. 9], we have

$$f(y) \ge f(x) + \langle \nabla f(x), y - x\rangle + \frac{\gamma}{2}\|y - x\|^2,$$
$$f(y) \le f(x) + \langle \nabla f(x), y - x\rangle + \frac{\Gamma}{2}\|y - x\|^2.$$

We can rewrite the above usual inner products/norms in terms of weighted ones by using 1) and 2) of Lemma 2.2. □

3. GDI algorithm for exact or weighted LS solutions

Consider the linear system (1.1) when $A \in \mathbb{R}^{m\times n}$ is a given constant matrix, $b \in \mathbb{R}^m$ is a given constant vector, and $x \in \mathbb{R}^n$ is an unknown vector. Assume that $\operatorname{rank} A = n$, i.e., A is of full-column rank. In this section, we derive an iterative algorithm to solve (1.1) for an exact solution or a least-squares (LS) solution, based on gradients and steepest-descent technique. Here, we seek for a vector $x^* \in \mathbb{R}^n$ that minimizes the weighted LS error $\|Ax - b\|_W^2$, where $W \in \mathbb{R}^{m\times m}$ is a given positive definite matrix. When $\|Ax - b\|_W = 0$, the vector x^* becomes an exact solution of the system.

It is natural to measure any error related to iterative procedure using the weighted norm $\|\cdot\|_W$ induced by W. To measure the residual error occurred at each iteration, we consider the squared residual-error function $f : \mathbb{R}^m \to \mathbb{R}$, defined by

$$f(x) \;:=\; \|Ax - b\|_W^2. \tag{3.1}$$

Our recursive iteration will be in the form

$$x^{(i+1)} \;=\; x^{(i)} - \alpha_{(i+1)}\nabla f(x),$$

i.e., the next approximate solution $x^{(i+1)}$ is equal to the current one $x^{(i)}$ along with a suitable step size $\alpha_{(i+1)}$ in the direction of $-\nabla f(x)$. Now, we compute the gradient of $f(x)$ as follows:

$$\begin{aligned}\nabla f(x) &= \frac{d}{dx}((Ax-b)^T W(Ax-b))\\ &= \frac{d}{dx}(x^TA^TWAx - x^TA^TWb - b^TWAx + b^TWb)\\ &= 2A^TW(Ax-b).\end{aligned} \tag{3.2}$$

Thus, we obtain the following recursive formula:

$$x^{(i+1)} \;=\; x^{(i)} - 2\alpha_{(i+1)}A^TW(Ax^{(i)} - b). \tag{3.3}$$

To optimize the step size at each iteration, we define $\Phi_{(i+1)} : [0,\infty) \to \mathbb{R}$ by

$$\begin{aligned}\Phi_{(i+1)}(\alpha) \;:=\; f(x^{(i+1)}) &= \|Ax^{(i+1)} - b\|_W^2\\ &= \|A(x^{(i)} - 2\alpha_{(i+1)}A^TW(Ax^{(i)} - b)) - b\|_W^2.\end{aligned}$$

By substituting $c = b - Ax^{(i)}$ and $e = 2AA^TWc$, we have

$$\Phi_{(i+1)}(\alpha) \;=\; \|\alpha e - c\|_W^2, \quad \alpha > 0.$$

Its critical point can be obtained as follows:

$$\begin{aligned}0 &= \frac{d}{d\alpha}\Phi_{(i+1)}(\alpha) = \frac{d}{d\alpha}((\alpha e - c)^TW(\alpha e - c))\\ &= \frac{d}{d\alpha}(\alpha^2e^TWe - \alpha e^TWc - \alpha c^TWe + c^TWc) = 2\alpha e^TWe - 2e^TWc.\end{aligned} \tag{3.4}$$

Indeed, we get $\alpha = (e^T Wc)/(e^T We)$, so that the minimizer of $\Phi_{(i+1)}$ is given by

$$\alpha_{(i+1)} = \frac{(W^{\frac{1}{2}}AA^T W(b - Ax^{(i)}))^T(W^{\frac{1}{2}}(b - Ax^{(i)}))}{2(W^{\frac{1}{2}}AA^T W(b - Ax^{(i)}))^T(W^{\frac{1}{2}}AA^T W(b - Ax^{(i)}))}. \tag{3.5}$$

Algorithm 6: GDI algorithm for solving Eq (1.1).

```
$A \in \mathbb{R}^{m\times n}$, and $b \in \mathbb{R}^m$ ;
Set $i = 0$. Choose $x^{(0)} \in \mathbb{R}^n$. Compute $P = A^T W$, $M = W^{\frac{1}{2}}AP$.
for $i = 0, 1, 2, 3, \dots$ do
    $r^{(i)} = b - Ax^{(i)}$;
    if $\|r^{(i)}\|_W \leqslant \epsilon$ then
        $x^{(i)}$ is a solution; break;
    else
        $m_{(i)} = Mr^{(i)}$;
        $\alpha_{(i+1)} = m_{(i)}^T W^{\frac{1}{2}} r^{(i)}/(2m_{(i)}^T m_{(i)})$ ;
        $x^{(i+1)} = x^{(i)} - \alpha_{(i+1)}P(Ax^{(i)} - b)$ ;
    end
    update $i$;
end
```

Remark 3.1. In Algorithm 6, the matrices P, M and the vectors $r^{(i)}$, $m_{(i)}$ are introduced in order to avoid duplicated computation. Recall that the condition number of a matrix is the ratio between its largest and smallest singular values. When A is ill-conditioned (i.e., its condition number is large), then a small residual $\|r^{(i)}\|_W$ does not guarantee a small error of the solution. In this case, we suggest a user to impose an additional stopping rule, e.g. $\|x^{(i)} - x^{(i-1)}\|_W < \epsilon'$ where ϵ' is a toterance number.

4. Capability and performance of GDI algorithm

In this section, we make a convergence analysis for the proposed algorithm.

Theorem 1. *Suppose that Eq* (1.1) *is inconsistent and A is of full-column rank. Let $W \in \mathbb{R}^{m\times m}$ be positive definite. Denote the condition number of $W^{\frac{1}{2}}A$ by κ. Then, for any initial vector $x^{(0)}$, the sequence $x^{(i)}$ generated by Algorithm 6 satisfies*

(i) $\|Ax^{(i)} - b\|_W^2 \to \delta$ *as* $i \to \infty$, *where* δ *is the weighted LS error* (2.1)*;*

(ii) we have the following bounds:

$$\|r^{(i)}\|_W^2 \leq (1 - \kappa^{-2})\|r^{(i-1)}\|_W^2 - \delta\kappa, \tag{4.1}$$

$$\|r^{(i)}\|_W^2 \leq (1 - \kappa^{-2})^i\|r^{(0)}\|_W^2 - \delta + \delta(1 - \kappa^{-2})^i. \tag{4.2}$$

Proof. If there is an integer i such that $\nabla f(x^{(i)}) = 0$, then $x^{(i)}$ is the desire LS solution. Now, assume that $\nabla f(x^{(i)}) \neq 0$ for any i. We can compute the second-order derivative of f as follows:

$$\nabla^2 f(x) = 2A^T WA = 2(W^{\frac{1}{2}}A)^T(W^{\frac{1}{2}}A).$$

Since rank $A = n$ and W is invertible, we have rank $W^{\frac{1}{2}}A = n$. Thus, $\nabla^2 f(x)$ is positive definite, so that $0 < \lambda_{\min}(A^T WA) \leq \lambda_{\max}(A^T WA)$. From the spectral theory of matrices, we have

$$2\lambda_{\min}(A^T WA)I \quad \leq \quad \nabla^2 f(x) \quad \leq \quad 2\lambda_{\max}(A^T WA)I.$$

Hence f is strongly convex. From the convexity inequality (2.2) with $\gamma = 2\lambda_{\min}(A^T WA)$, we have that for all $y \in \mathbb{R}^m$,

$$\begin{aligned} f(y) \quad &\geq \quad f(x^{(i)}) + \left\langle W^{-\frac{1}{2}}\nabla f(x^{(i)}), W^{-\frac{1}{2}}(y - x^{(i)})\right\rangle_W + \frac{2\lambda_{\min}(A^T WA)}{2}\|W^{-\frac{1}{2}}(y - x^{(i)})\|_W^2 \\ &= \quad f(x^{(i)}) - \alpha_{(i+1)}\|W^{-\frac{1}{2}}\nabla f(x^{(i)})\|_W^2 + \alpha_{(i+1)}^2\lambda_{\min}(A^T WA)\|W^{-\frac{1}{2}}\nabla f(x^{(i)})\|_W^2. \end{aligned} \tag{4.3}$$

Minimizing the real-valued function on the right-hand side of (4.3) with respect to the variable $\alpha_{(i+1)}$ yields the minimizer $\alpha_{(i+1)} = 1/(2\lambda_{\min}(A^T WA))$. Thus, for all $y \in \mathbb{R}^m$, we obtain

$$\begin{aligned} f(y) \geq \quad & f(x^{(i)}) - \frac{1}{2\lambda_{\min}(A^T WA)}\|W^{-\frac{1}{2}}\nabla f(x^{(i)})\|_W^2 + \frac{1}{4\lambda_{\min}^2(A^T WA)}\lambda_{\min}(A^T WA)\|W^{-\frac{1}{2}}\nabla f(x^{(i)})\|_W^2 \\ = \quad & f(x^{(i)}) - \frac{1}{4\lambda_{\min}(A^T WA)}\|W^{-\frac{1}{2}}\nabla f(x^{(i)})\|_W^2. \end{aligned} \tag{4.4}$$

With the weighted LS solution $y = x^*$, we have

$$\delta \quad \geq \quad f(x^{(i)}) - \frac{1}{4\lambda_{\min}(A^T WA)}\|W^{-\frac{1}{2}}\nabla f(x^{(i)})\|_W^2, \tag{4.5}$$

where δ is the weighted LS error $\|Ax^* - b\|_W^2$. Similarly, from Eq (2.3), we have

$$f(y) \;\leq\; f(x^{(i)}) - \alpha_{(i+1)}\|W^{-\frac{1}{2}}\nabla f(x^{(i)})\|_W^2 + \alpha_{(i+1)}^2\lambda_{\max}(A^T WA)\|W^{-\frac{1}{2}}\nabla f(x^{(i)})\|_W^2 \tag{4.6}$$

for all $y \in \mathbb{R}^m$. Minimizing the real-valued function on the right-hand side of (4.6) yields the minimizer $\alpha_{(i+1)} = 1/(2\lambda_{\max}(A^T WA))$. With $y = x^{(i+1)}$, we get

$$f(x^{(i+1)}) \quad \leq \quad f(x^{(i)}) - \frac{1}{4\lambda_{\max}(A^T WA)}\|W^{-\frac{1}{2}}\nabla f(x^{(i)})\|_W^2. \tag{4.7}$$

From (4.5) and (4.7), we get

$$f(x^{(i+1)}) - \delta \quad \leq \quad \left(1 - \frac{\lambda_{\min}(A^T WA)}{\lambda_{\max}(A^T WA)}\right)(f(x^{(i)}) - \delta). \tag{4.8}$$

Let us denote

$$\beta \;=\; 1 - (\lambda_{\min}(A^T WA)/\lambda_{\max}(A^T WA)) = 1 - \kappa^{-2}. \tag{4.9}$$

Then, we can rewrite Eq (4.8) as

$$f(x^{(i+1)}) - \delta \quad \leq \quad \beta(f(x^{(i)}) - \delta). \tag{4.10}$$

By induction, we obtain

$$f(x^{(i)}) - \delta \quad \leq \quad \beta^i(f(x^{(0)}) - \delta), \qquad \text{for any} \;\; i \in \mathbb{N}. \tag{4.11}$$

Since $0 \leq \beta < 1$, we have $f(x^{(i)}) - \delta \to 0$ or $f(x^{(i)}) \to \delta$ as $i \to \infty$.

From (4.9), the inequalities (4.10) and (4.11) become (4.1) and (4.2), respectively. $\square$

Corollary 4.1. *Suppose that Eq* (1.1) *is inconsistent and A is of full-column rank. Let κ be the condition number of A. Then, for any initial vector $x^{(0)}$, the sequence $x^{(i)}$ generated by Algorithm 6 satisfies*

(i) $\|Ax^{(i)} - b\|^2 \to \delta$ *as* $i \to \infty$, *where* $\delta = \|b\|^2 - b^T A(A^T A)^{-1} A^T b$ *is the LS error;*
(ii) we have the following bounds:

$$\|r^{(i)}\|^2 \le (1-\kappa^{-2})\|r^{(i-1)}\|^2 - \delta\kappa, \tag{4.12}$$
$$\|r^{(i)}\|^2 \le (1-\kappa^{-2})^i\|r^{(0)}\|^2 - \delta + \delta(1-\kappa^{-2})^i. \tag{4.13}$$

Proof. This is a particular case of Theorem 1 when $W = I$. □

Theorem 2. *Suppose that Eq* (1.1) *is consistent and A is of full-column rank. Let x^* be the exact solution of* (1.1). *Let $W \in \mathbb{R}^{m\times m}$ be positive definite. Denote the condition number of $W^{\frac{1}{2}}A$ by κ. Then, for any initial vector $x^{(0)}$, the sequence $x^{(i)}$ generated by Algorithm 6 satisfies*

(i) $x^{(i)} \to x^*$ *as* $i \to \infty$;
(ii) estimations for residual vectors are as follows:

$$\|r^{(i)}\|_W \le (1-\kappa^{-2})^{\frac{1}{2}}\|r^{(i-1)}\|_W, \tag{4.14}$$
$$\|r^{(i)}\|_W \le (1-\kappa^{-2})^{\frac{i}{2}}\|r^{(0)}\|_W; \tag{4.15}$$

(iii) we have the following error estimations:

$$\|x^{(i)} - x^*\|_W \le \sqrt{\lambda_{\max}(W)}\kappa^2(1-\kappa^{-2})^{\frac{1}{2}}\|x^{(i-1)} - x^*\|, \tag{4.16}$$
$$\|x^{(i)} - x^*\|_W \le \sqrt{\lambda_{\max}(W)}\kappa^2(1-\kappa^{-2})^{\frac{i}{2}}\|x^{(0)} - x^*\|. \tag{4.17}$$

In particular, the convergence rate of Algorithm 6 is controlled by the term $\sqrt{1-\kappa^{-2}}$;
(iv) let $\epsilon > 0$. *Suppose* $x^{(0)} \ne x^*$ *and* $\kappa > 1$. *We have* $\|x^{(i)} - x^*\|_W < \epsilon$ *after* i^* *iterations, where*

$$i^* > \frac{2\log\epsilon - \log\lambda_{\max}(W) - 4\log\kappa - 2\log\|x^{(0)} - x^*\|}{\log(1-\kappa^{-2})}. \tag{4.18}$$

Proof. (i) The proof of is similar to that of Theorem 1. In this case, Eq (1.1) has a solution, i.e. $\delta = 0$. From Eqs (4.10) and (4.11), we get

$$f(x^{(i+1)}) \le \beta f(x^{(i)}), \tag{4.19}$$
$$f(x^{(i)}) \le \beta^i f(x^{(0)}), \qquad \text{for any } i \in \mathbb{N}. \tag{4.20}$$

Since $0 \le \beta < 1$, we have $f(x^{(i)}) = \|Ax^{(i)} - b\|_W^2 \to 0$. It follows that $Ax^{(i)} - b \to 0$. Since $b = Ax^*$, we have $Ax^{(i)} \to Ax^*$. Since A is of full-column rank, the matrix A^TA is invertible. Hence, $x^{(i)} \to x^*$ as $i \to \infty$.

(ii) From (4.9), the inequalities (4.19) and (4.20) become (4.14) and (4.15), respectively.

(iii) By using Lemmas 2.1, 2.2 and Eq (4.14), we have

$$\begin{aligned}\|x^{(i)} - x^*\|_W &= \|(A^TWA)^{-1}(A^TWA)x^{(i)} - (A^TWA)^{-1}(A^TWA)x^*\|_W \\ &= \|(A^TWA)^{-1}(A^TW^{\frac{1}{2}})(W^{\frac{1}{2}}Ax^{(i)} - W^{\frac{1}{2}}Ax^*)\|_W\end{aligned}$$

$$
\begin{aligned}
&\leq \|W^{\frac{1}{2}}\|_2\|(A^TWA)^{-1}\|_2\|A^TW^{\frac{1}{2}}\|_2\|Ax^{(i)} - Ax^*\|_W \\
&= \sqrt{\lambda_{\max}(W)}\|(A^TWA)^{-1}\|_2\|W^{\frac{1}{2}}A\|_2\|Ax^{(i)} - b\|_W \\
&\leq \sqrt{\lambda_{\max}(W)}\|(A^TWA)^{-1}\|_2\|W^{\frac{1}{2}}A\|_2\left(\sqrt{\beta}\|Ax^{(i-1)} - b\|_W\right) \\
&\leq (1-\kappa^{-2})^{\frac{1}{2}}\sqrt{\lambda_{\max}(W)}\|(A^TWA)^{-1}\|_2\|W^{\frac{1}{2}}A\|_2\|W^{\frac{1}{2}}A\|_2\|W^{-\frac{1}{2}}(x^{(i-1)} - x^*)\|_W \\
&= \sqrt{\lambda_{\max}(W)}\kappa^2(1-\kappa^{-2})^{\frac{1}{2}}\|x^{(i-1)} - x^*\|.
\end{aligned}
$$

By induction, the inequality (4.17) holds.

(iv) From (4.17), we get an error estimation

$$\|x^{(i)} - x^*\|_W \leq \sqrt{\lambda_{\max}(W)}\,\kappa^2(1-\kappa^{-2})^{\frac{i}{2}}\|x^{(0)} - x^*\| \to 0, \quad \text{as } i \to \infty.$$

It implies that for each $\epsilon > 0$

$$\sqrt{\lambda_{\max}(W)}\kappa^2(1-\kappa^{-2})^{\frac{i}{2}}\|x^{(0)} - x^*\| < \epsilon, \quad \text{for all } i \geq i^*, i^* \in \mathbb{N}.$$

Taking the 10-base logarithm on both sides yields the desired iteration number (4.18).

□

Theorem 2 includes the work [19] as a special case.

5. Numerical experiments

In this section, we verify theoretical results through numerical experiments implemented by MATLAB R2021a, on the same Mac operating system (M1 chip 8C CPU/8C GPU/8GB/512GB). We consider both consistent and inconsistent linear systems with square/non-square coefficient matrices of moderate/large dimension. The coefficient matrix A and the weighted positive definite matrix W we considered are the identity matrix (denoted by I),

- a tridiagonal matrix (denoted by tridiag),
- an upper-triangular portion included the main diagonal (denoted by triu),
- a lower-triangular portion included the main diagonal (denoted by tril).

We compare the performance of Algorithm 6 (GDI) with the direct method and well-known iterative methods in the literature. The algorithm performance is evaluated through the number of iterations, the norm of residual vectors, and the CPU time measured in seconds by *tic toc* functions on MATLAB.

5.1. Experiments for consistent linear systems

Example 1. *Consider a moderate-scaled linear system* (1.1) *with*

$$A = \text{tridiag}(1, -2, 0) \in \mathbb{R}^{50\times 50}, \quad b = (1, 0, \dots, 0)^T \in \mathbb{R}^{50}, \quad W = \text{tridiag}(1, 4, 1) \in \mathbb{R}^{50\times 50}.$$

This linear system has a unique solution. We take an initial point $x^{(0)} = (-0.1, -0.1, \dots, -0.1)^T \in \mathbb{R}^{50}$ *and set a tolerance error* $\epsilon = 10^{-3}$. *It turns out that Algorithm 6 produces a satisfactory solution (*3*-digit accuracy) within* 13 *iterations, consuming* 0.026620 *seconds. So, the iteration number satisfies* *the inequality* (4.18)*. Numerical results illustrated in Figure 1 and Table 1 both show the applicability of the algorithm. Theorem 2 says that the convergence rate of Algorithm 6 depends on the condition number of $W^{\frac{1}{2}}A$, namely, $\kappa = 1.8699$. So, with a well-conditioned coefficient matrix, the algorithm produced a satisfactory solution within a small iteration number.*

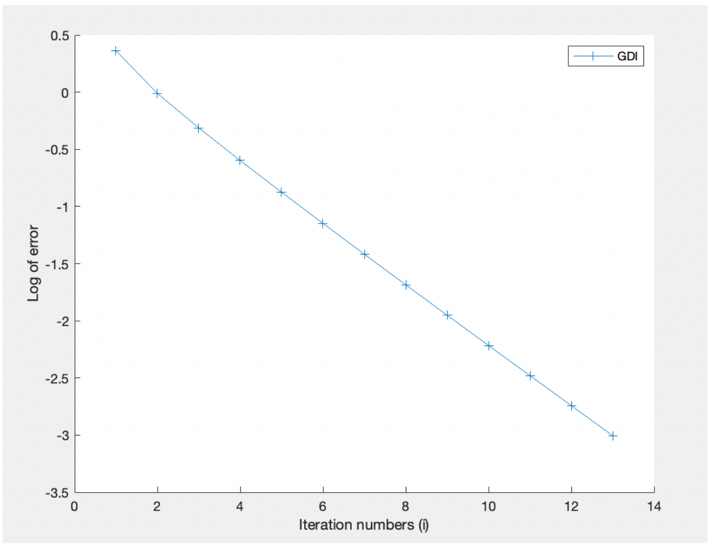

Figure 1. The logarithm of the relative error for Example 1.

Table 1. Numerical solutions for Example 1.

Method	Direct	GDI
Iterations	-	13
x_1	-0.5000	-0.5000
x_2	-0.2500	-0.2500
x_3	-0.1250	-0.1250
x_4	-0.0625	-0.0625
x_5	-0.0312	-0.0313
x_6	-0.0156	-0.0157
x_7	-0.0078	-0.0078
x_8	-0.0039	-0.0039
x_9	-0.0020	-0.0020
x_{10}	-0.0010	-0.0010
x_{11}	-0.0005	-0.0005
x_{12}	-0.0002	-0.0003
x_{13}	-0.0001	-0.0002
x_{14}	-0.0001	-0.0001
$x_{15}, \ldots, x_{74}$	0	-0.0001
$x_{75}, \ldots, x_{80}$	0	0
$\|\|r^{(i)}\|\|_W$	0	0.0009898876

Example 2. *Consider a large-scaled linear system* (1.1) *with*

$$A = \mathrm{tridiag}(1, -13, -7) \in \mathbb{R}^{80\times 80}, \quad b = (1, 1, \ldots, 1)^T \in \mathbb{R}^{80}, \quad W = I \in \mathbb{R}^{80\times 80}.$$

In fact, the linear system has a unique solution. We compare the performance of Algorithm 6 with well-known and recent iterative methods, namely GI (Algorithm 1), RGI (Algorithm 3), MGI (Algorithm 4), AGBI (Algorithm 5), and MJGI [14]. All algorithms are implemented using the same initial point $x^{(0)} = (-5, -5, \ldots, -5)^T \in \mathbb{R}^{80}$. *The numerical results are shown in Figure 2 and Table 2. After running* 29 *iterations, GDI method produces a well-approximated solution having the residual* $\|r^{(i)}\|_W = 0.00087 < \epsilon := 10^{-3}$, *consuming* 0.017624 *seconds. We see that the performance of GDI algorithm is better than that of other algorithms in both errors and computational time. Indeed, the performance of GDI algorithm (and, in fact, other ones) is governed by the condition number* $\kappa = 2.7097$.

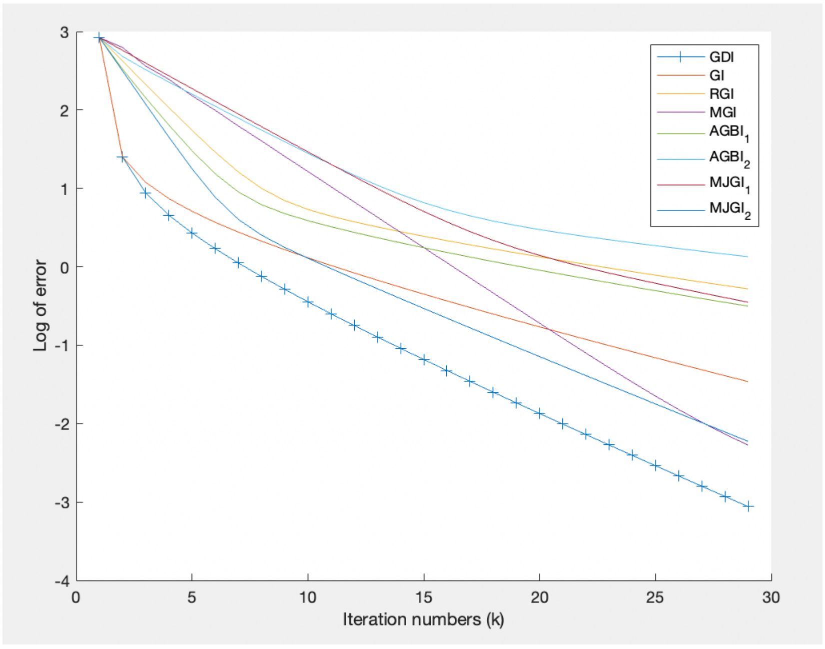

Figure 2. The logarithm of the relative error for Example 2.

Table 2. Numerical solutions for Example 2.

Method	Parameters		Iterations	Relative error	CPU time
	Convergent factor	weighted factor			
GDI	-		29	0.00087	0.021170
GI	$\mu = 0.0055$	-	29	0.034310	0.034310
RGI	$\mu = 0.0292$	$\omega = 0.6$	29	0.522498	0.054389
MGI	$\mu = 0.0055$	-	29	0.005316	0.027525
AGBI_1	$\mu = 0.0062$	$\omega = 0.3$	29	0.314652	0.027661
AGBI_2	$\mu = 0.0031$	$\omega = 0.6$	29	1.343812	0.026997
MJGI_1	$\mu = 0.0013$	-	29	0.352640	0.028018
MJGI_2	$\mu = 0.0025$	-	29	0.005942	0.024014

5.2. *Experiments for inconsistent linear systems*

Example 3. *Consider a moderate-scaled linear system* (1.1) *with*

$$A = \operatorname{triu}(35, -4, 3, 0, \dots, 0) \in \mathbb{R}^{50\times 45}, \; b = (1, 1, \dots, 1, -1)^T \in \mathbb{R}^{50},$$

and the weighted positive definite matrix $W = \operatorname{diag}(2, 1, 2, 1, \dots, 1) \in \mathbb{R}^{50\times 50}$. *We can check that* $\operatorname{rank} A = 45 \neq 46 = \operatorname{rank}[A \; b]$, *so that the linear system is inconsistent but has a unique LS solution. According to Eq* (2.1)*, the square root of weighted LS error is* 2.7017. *We perform GDI algorithm with initial value* $x^{(0)} = (7, 7, \dots, 7)^T \in \mathbb{R}^{45}$. *The numerical results in Figure 3 and Table 3 show that GDI algorithm produces a well-approximated solution after 16 iterations with error* 2.7017, *which is very close to the theoretical weighted LS error. Moreover, each coordinate value of* $x^{(16)}$ *is very close to that of the LS solution obtained from the direct method.*

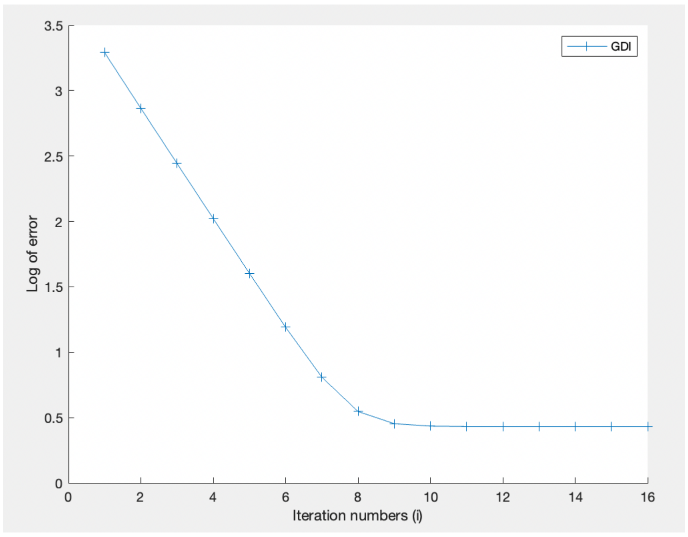

Figure 3. The logarithm of the relative error for Example 3.

Table 3. Numerical solutions for Example 3.

Method	Direct	GDI
Iterations	-	16
x_1	0.0286	0.0286
x_2	0.0318	0.0319
x_3	0.0298	0.0298
x_4	0.0292	0.0292
$x_5, \dots, x_{41}$	0.0294	0.0294
x_{42}	0.0296	0.0296
x_{43}	0.0294	0.0294
x_{44}	0.0268	0.0298
x_{45}	0.0280	0.0280
LS error	2.7017	2.7017

Example 4. *Consider a moderate-scaled linear system* (1.1) *with*

$$A = \text{triu}(15, -2, 1, 0, \ldots, 0) \in \mathbb{R}^{30\times 25}, \quad b = (1, 1, \ldots, 1, -1)^T \in \mathbb{R}^{30}, \quad W = I \in \mathbb{R}^{30\times 30}.$$

In this case, the linear system is inconsistent and has a unique LS solution. According to Eq (2.1)*, the square root of weighted LS error is given by* $\epsilon = \|Ax^* - b\|_W = 2.2371$*. We compare the performance of GDI algorithm with LSI method (Algorithm 2) with different parameters* $\mu_1 = 0.1$*,* $\mu_2 = 0.25$*, and* $\mu_3 = 0.5$*, using the same initial point* $x^{(0)} = 0 \in \mathbb{R}^{25}$*. The numerical results are presented in Figure 4, Tables 4 and 5. We can see that out of a total of* 4 *iterations, the GDI method takes the least computational time* (0.012849 *seconds) compared to the LSI method with different parameters. In addition, the value of each solution coordinate obtained from GDI method is close to the exact LS solution* x^* *than those obtained from LSI method.*

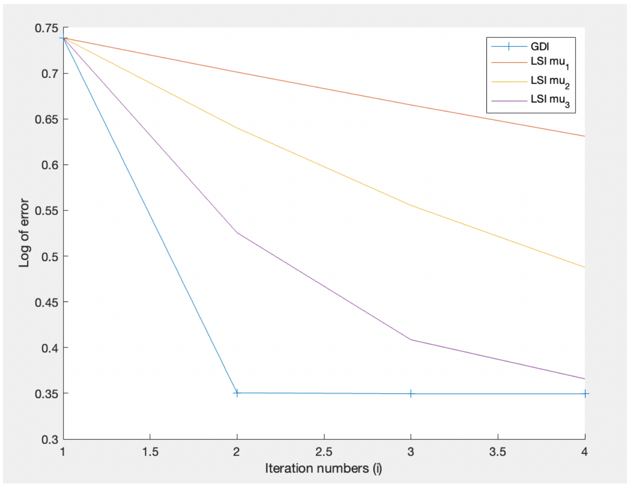

Figure 4. The logarithm of the relative error for Example 4.

Table 4. LS solutions for Example 4.

Method	Direct	GDI	LSI		
Parameter	-	-	$\mu_1 = 0.1$	$\mu_2 = 0.25$	$\mu_3 = 0.5$
Iterations	-	4	4	4	4
x_1	0.0714	0.0715	0.0194	0.0413	0.0625
x_2	0.0714	0.0713	0.0194	0.0413	0.0625
x_3	0.0714	0.0714	0.0194	0.0413	0.0625
x_4	0.0714	0.0713	0.0194	0.0413	0.0625
x_5	0.0714	0.0715	0.0194	0.0413	0.0625
$x_6, \ldots, x_{20}$	0.0714	0.0714	0.0194	0.0413	0.0625
x_{21}	0.0713	0.0714	0.0193	0.0412	0.0624
x_{22}	0.0713	0.0712	0.0193	0.0412	0.0624
x_{23}	0.0723	0.0722	0.0196	0.0418	0.0633
x_{24}	0.0756	0.0754	0.0205	0.0437	0.0661
x_{25}	0.0667	0.0667	0.0181	0.0385	0.0583

Table 5. Numerical solutions for Example 4.

Method	GDI	LSI		
Parameter	-	$\mu_1 = 0.1$	$\mu_2 = 0.25$	$\mu_3 = 0.5$
Iterations	4	4	4	4
LS error	2.23607	4.276216	3.073998	2.321772
CPU time	0.006919	0.009616	0.007964	0.008092

Example 5. *Consider a large-scaled linear system* (1.1) *with*

$$A = \mathrm{tril}(-3, -13, 7, 0, \ldots, 0) \in \mathbb{R}^{100\times 80}, \quad b = (1, 1, \ldots, 1, -1)^T \in \mathbb{R}^{100}, \quad W = I \in \mathbb{R}^{100\times 100}.$$

In fact, the linear system is inconsistent and has a unique LS solution. We compare the performance of GDI algorithm with LSI algorithm (Algorithm 2) with the same initial vector $x^{(0)} = 0 \in \mathbb{R}^{80}$*. Figure 5 shows that GDI produces a solution having the LS error* 4.41218*, which is equal to the theoretical LS error* (2.1) *within 5 iterations. However, the accuracy of each coordinate (compared to the direct method) in the solution is not guaranteed. From Tables 6 and 7, we see that GDI method takes 28 iterations to obtain an LS solution with 4-digit accuracy in each coordinate, consuming* 0.027191 *seconds. However, within the same iteration number, LSI method with three parameters (*$\mu_1 = 0.02$*,* $\mu_2 = 0.05$*,* $\mu_3 = 0.1$*) does not provide a satisfactory LS solution. Thus, GDI method is applicable and has a better performance than LSI method.*

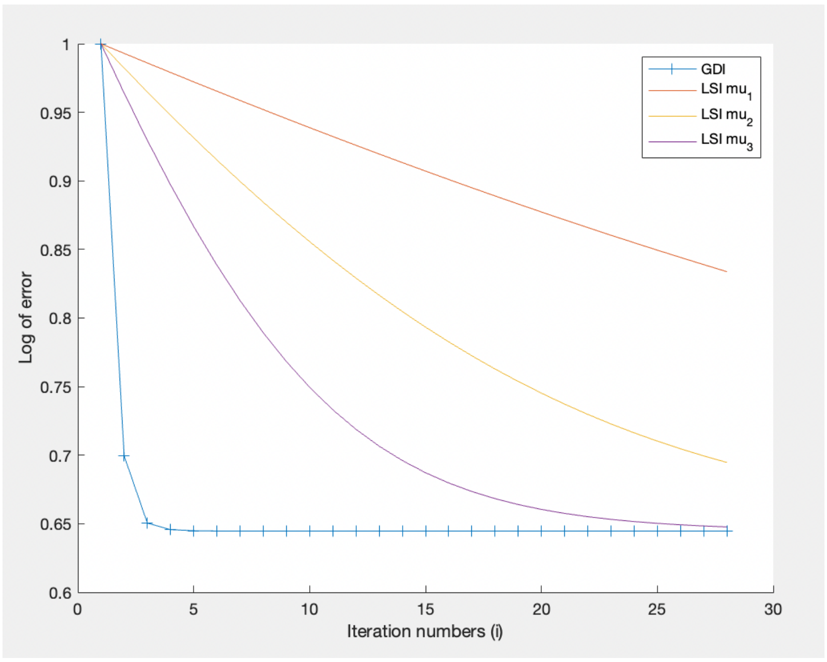

Figure 5. The logarithm of the relative error for Example 5.

Table 6. LS solutions for Example 5.

Method	Direct	GDI	LSI		
Parameter	-	-	$\mu_1 = 0.02$	$\mu_2 = 0.05$	$\mu_3 = 0.1$
Iterations	-	28	28	28	28
x_1	-0.0692	-0.0692	-0.0291	-0.0519	-0.0652
x_2	-0.0883	-0.0883	-0.0371	-0.0662	-0.0832
x_3	-0.1006	-0.1006	-0.0423	-0.0754	-0.0947
x_4	-0.1059	-0.1059	-0.0445	-0.0794	-0.0998
x_5	-0.1086	-0.1086	-0.0457	-0.0814	-0.1023
x_6	-0.1099	-0.1099	-0.0462	-0.0824	-0.1035
x_7	-0.1105	-0.1105	-0.0465	-0.0829	-0.1041
x_8	-0.1108	-0.1108	-0.0466	-0.0831	-0.1044
x_9	-0.1110	-0.1110	-0.0467	-0.0832	-0.1045
x_{10}	-0.1110	-0.1110	-0.0467	-0.0832	-0.1046
$x_{11}, \ldots, x_{20}$	-0.1111	-0.1111	-0.0467	-0.0833	-0.1046
$x_{21}, \ldots, x_{71}$	-0.1111	-0.1111	-0.0467	-0.0833	-0.1047
x_{72}	-0.1112	-0.1112	-0.0467	-0.0833	-0.1047
x_{73}	-0.1113	-0.1113	-0.0468	-0.0834	-0.1048
x_{74}	-0.1114	-0.1114	-0.0468	-0.0835	-0.1049
x_{75}	-0.1118	-0.1118	-0.0470	-0.0838	-0.1053
x_{76}	-0.1122	-0.1122	-0.0472	-0.0841	-0.1057
x_{77}	-0.1148	-0.1148	-0.0483	-0.0861	-0.1081
x_{78}	-0.1123	-0.1123	-0.0472	-0.0842	-0.1057
x_{79}	-0.1446	-0.1146	-0.0608	-0.1084	-0.1362
x_{80}	-0.0303	-0.0303	-0.0127	-0.0227	-0.0285

Table 7. Numerical solutions for Example 5.

Method	GDI	LSI		
Parameter	-	$\mu_1 = 0.02$	$\mu_2 = 0.05$	$\mu_3 = 0.1$
Iterations	28	28	28	28
LS error	4.41218	6.820422	4.951215	4.442936
CPU time	0.027191	0.028941	0.031406	0.030662

6. Conclusions

We consider the rectangular linear system (1.1) when the coefficient matrix is of full-column rank. So the system has a unique solution or a unique LS solution. Here, we consider LS solutions in a full generality, that is, we measure the error through any vector norm induced from weighted positive definite matrices W. We propose a gradient-descent iterative (GDI) algorithm to this linear system, aiming to produce a sequence of well-approximate exact or weighted LS solutions. In the recursive equation, the step size is optimized at each iteration so that the square residual error is minimized. It turns out that when Eq (1.1) has a unique solution, GDI algorithm produces approximate solutions converging to the unique solution. When Eq (1.1) is inconsistent, the sequence of residual norms converges to the weighted LS error. Moreover, with a well-conditioned coefficient matrix, the algorithm produced a satisfactory solution within a small iteration number. When $W = I$, the weighted LS error becomes the LS error with respect to the usual norm. Numerical experiments are performed to verify the capability of the proposed algorithm. Moreover, the performance of GDI algorithm is better than that of GI-type algorithms in both iteration numbers and computational time.

Acknowledgments

This research project is supported by National Research Council of Thailand (NRCT): (N41A640234). The authors would like to thank reviewers for useful suggestions and comments.

Conflict of interest

The authors declare there are no conflicts of interest.

References

1. W. D. James, *Applied numerical linear algebra*, Philadelphia: Society for Industrial and Applied Mathematics, 1997.
2. P. J. Olver, C. Shakiban, *Applied linear algebra*, New York: Springer, 2018.
3. D. M. Young, *Iterative solution of large linear systems*, New York: Academic Press, 1971.
4. P. Albrechtt, M. P. Klein, Extrapolated iterative methods for linear systems, *SIAM J. Numer. Anal.*, **21** (1984), 192–201. https://doi.org/10.1137/0721014
5. A. J. Hughes-Hallett, The convergence of accelerated overrelaxation iterations, *Math. Comput.*, **47** (1986), 219–223.
6. Y. Saad, *Iterative methods for sparse linear systems*, 2 Eds., Philadelphia: Society for Industrial and Applied Mathematics, 2003.
7. X. I. A. Yang, R. Mittal, Acceleration of the Jacobi iterative method by factors exceeding 100 using scheduled relaxation, *J. Comput. Phys.*, **274** (2014), 695–708. https://doi.org/10.1016/j.jcp.2014.06.010
8. J. E. Adsuara, I. Cordero-Carrión, P. Cerdá-Durán, M. A. Aloy, Scheduled Relaxation Jacobi method improvements and applications, *Comput. Phys.*, **321** (2016), 369–413. https://doi.org/10.1016/j.jcp.2016.05.053
9. F. Ding, T. Chen, Iterative least-squares solutions of coupled Sylvester matrix equations, *Syst. Control Lett.*, **54** (2005), 95–107. https://doi.org/10.1016/j.sysconle.2004.06.008
10. F. Ding, T. Chen, On iterative solutions of general coupled matrix equations, *SIAM J. Control Optim.*, **44** (2006), 2269–2284. https://doi.org/10.1137/S0363012904441350

11. Q. Niu, X. Wang, L. Z. Lu, A relaxed gradient based algorithm for solving Sylvester equation, *Asian J. Control*, **13** (2011), 461–464. https://doi.org/10.1002/asjc.328

12. X. Wang, L. Dai, D. Liao, A modified gradient based algorithm for solving Sylvester equation, *Appl. Math. Comput.*, **218** (2012), 5620–5628. https://doi.org/10.1016/j.amc.2011.11.055

13. Y. J. Xie, C. F. Ma, The accelerated gradient based iterative algorithm for solving a class of generalized Sylvester transpose matrix equation, *Appl. Math. Comput.*, **273** (2016), 1257–1269. https://doi.org/10.1016/j.amc.2015.07.022

14. N. Sasaki, P. Chansangiam, Modified Jacobi-gradient iterative method for generalized Sylvester matrix equation, *Symmetry*, **12** (2020), 1831. https://doi.org/10.3390/sym12111831

15. W. Fan, C. Gu, Z. Tian, Jacobi-gradient iterative algorithms for Sylvester matrix equations, *Proceedings of the 14th Conference of the International Linear Algebra Society*, Shanghai University, Shanghai, China, 2007.

16. Z. Tian, M. Tian, C. Gu, X. Hao, An accelerated Jacobi-gradient based iterative algorithm for solving Sylvester matrix equations, *Filomat*, **31** (2017), 2381–2390.

17. N. Boonruangkan, P. Chansangiam, Convergence analysis of a gradient iterative algorithm with optimal convergence factor for a generalized sylvester-transpose matrix equation, *AIMS Math.*, **6** (2021), 8477–8496. https://doi.org/10.3934/math.2021492

18. Z. Y. Li, Y. Wang, B. Zhou, G. R. Duan, Least squares solution with the minimum-norm to general matrix equations via iteration, *Appl. Math. Comput.*, **215** (2010), 3547–3562. https://doi.org/10.1016/j.amc.2009.10.052

19. A. Kittisopaporn, P. Chansangiam, The steepest descent of gradient-based iterative method for solving rectangular linear systems with an application to Poisson's equation, *Adv. Differ. Equ.*, **2020** (2020), 259. https://doi.org/10.1186/s13662-020-02715-9

20. A. Kittisopaporn, P. Chansangiam, Gradient-descent iterative algorithm for solving a class of linear matrix equations with applications to heat and Poisson equations, *Adv. Differ. Equ.*, **2020** (2020), 324. https://doi.org/10.1186/s13662-020-02785-9

21. A. Kittisopaporn, P. Chansangiam, Approximate solutions of the 2D space-time fractional diffusion equation via a gradient-descent iterative algorithm with Grünwald-Letnikov approximation, *AIMS Math.*, **7** (2022), 8471–8490. https://doi.org/10.3934/math.2022472

22. R. A. Horn, C. R. Johnson, *Topics in matrix analysis*, New York: Cambridge University Press, 1991. https://doi.org/10.1017/CBO9780511840371

23. S. P. Boyd, L. Vandenberghe, *Convex optimization*, Cambridge: Cambridge University Press, 2004.