A Novel Iterative Method for Computing Generalized Inverse

Youshen Xia, Tianping Chen, and Jinjun Shan

Abstract

In this paper, we propose a novel iterative method for computing generalized inverse, based on a novel KKT formulation. The proposed iterative algorithm requires to make four matrix and vector multiplications at each iteration and thus has a low computational complexity. The proposed iterative method is proved to be globally convergent without any condition. Furthermore, for fast computing generalized inverse we present an acceleration scheme based on the proposed iterative method. The global convergence of the proposed acceleration algorithm is also proved. Finally, effectiveness of the proposed iterative algorithm is evaluated numerically.

Keywords: Generalized inverse, iterative method, global Convergence, KKT equations

I. INTRODUCTION

The generalized inverse of a matrix is the generalization of the inverse of a nonsingular matrix to the inverse of a singular matrix or rank-deficient rectangular matrix [1]. The applications of pseudo-inverse are diverse, such as linear equations, statistical regression analysis, filtering, signal and image processing, and control of robot manipulators [2,3,4].

Many methods for generalized inverse were developed. They are divided into types: the continuous-time recurrent neural networks and learning algorithms [4-8] and the numerical algorithms [9-19]. The continuous-time algorithm has emerged as parallel distributed computational models, but it has relatively slow speed due to its continuous-time feature. The numerical algorithms mainly include direct methods and iterative methods. The direct methods such as the singular value decomposition (SVD) algorithm and the factorization-based algorithm are more accurate and is thus the most

Youshen Xia is with College of Mathematics and Computer Science, Fuzhou University, China (ysxia@fzu.edu.cn), Tianping Chen is with Department of Mathematics, Fudan University, China (tchen@fudan.edu.cn), Jinjun Shan is with Department of Earth and Space Science and Engineering, York University, Canada (jjshan@yorku.ca)
commonly used method but it requires a large amount of computational resources. The iterative methods such as the orthogonal projection algorithms, the Newton iterative algorithm, and the higher-order convergent iterative method are more suitable for implementation in a high-level language or low-level language. On the other hand, the existing iteration algorithms require an initial condition to show its convergence. That is, the convergence depends on the initial approximation.

In this paper, we first convert generalized inverse problem into a matrix norm optimization problem and introduce a novel KKT formulation. Then we propose a new iterative method for calculating generalized inverse. The proposed iterative method requires to make four matrix and vector multiplications at each time iteration and thus has a low computational complexity. The proposed iterative method is proved to be globally convergent without any condition. Furthermore, for fast computing generalized inverse we present an acceleration scheme based on the proposed iterative method. The global convergence of the proposed acceleration algorithm is also proved. Finally, effectiveness of the proposed iterative algorithm is evaluated numerically.

The paper is organized as follows. In Section II, some equivalent formulations of the generalized inverse are discussed. In Section III, a new iterative method for calculating generalized inverse is presented and the global convergence of the proposed iterative algorithm is proved. Section IV gives an acceleration scheme. Section V gives three illustrative examples. The conclusion of this is given in Section VI.

II. FORMULATIONS

For any \( A \in R^{m \times n} \), there exists only one matrix \( X \in R^{n \times m} \) satisfying the Moore-Penrose condition below

\[
\begin{align*}
A &= AXA \\
X &= XAX \\
(AX)^T &= AX \\
(XA)^T &= XA,
\end{align*}
\]

(1)

\( X \) is called the generalized inverse matrix, denoted by \( A^+ \). If rank (\( A \)) =\( \min\{n, m\} \), then the generalized inverse can be expressed as

\[
A^+ = \begin{cases} 
(A^T A)^{-1} A^T, & \text{if } m > n, \\
A^{-1}, & \text{if } m = n, \\
A^T (AA^T)^{-1}, & \text{if } m < n.
\end{cases}
\]

(2)
since $A^TA$ or $AA^T$ is nonsingular. If rank$(A) < \min\{n, m\}$, the generalized inverse still exists but cannot be obtained by (2) due to the singularity of $A^TA$ and $AA^T$. According to the minimality of the Frobenius norm of the pseudo-inverse [20], the generalized inverse problem can be equivalent to the matrix-valued quadratic convex programming problem with equality constraints. Thus, if $m \geq n$ and $X^*$ is a solution of the following optimization problem

$$\begin{align*}
\text{minimize} & \quad \frac{1}{2}\|X\|_F^2 \\
\text{subject to} & \quad A^TAX = A^T,
\end{align*}$$

(3)

where $X \in \mathbb{R}^{n \times m}$ and $\|X\|_F$ is the Frobenius norm of $X$, then $A^+ = X^*$. When $m < n$ and $X^*$ is a solution of the following optimization problem

$$\begin{align*}
\text{minimize} & \quad \frac{1}{2}\|X\|_F^2 \\
\text{subject to} & \quad AA^TX = A,
\end{align*}$$

(4)

where $X \in \mathbb{R}^{m \times n}$, then $A^+ = (X^*)^T$. Furthermore, solving the matrix-valued quadratic convex programming problem can be equivalently decomposed into solving either $m$ or $n$ independent vector-valued quadratic programs. If $m > n$, we solve for $i = 1, \ldots, m$

$$\begin{align*}
\text{minimize} & \quad \frac{1}{2}\|x_i\|^2 \\
\text{subject to} & \quad A^TAX_i = a_i^T,
\end{align*}$$

(5)

where $x_i$ is the $i$th column of $X$ and $a_i$ is the $i$th row of $A$, and $\|x_i\| = (x_i^T x_i)^{\frac{1}{2}}$. If $m \leq n$, we solve for $i = 1, \ldots, n$

$$\begin{align*}
\text{minimize} & \quad \frac{1}{2}\|x_j\|^2 \\
\text{subject to} & \quad AA^Tx_j = a_j,
\end{align*}$$

(6)

where $x_j$ is the $j$th row of $X$ and $a_j$ is the $j$th column of $A$.

By KKT optimal condition [21] we see that $x_i^*$ is an optimal solution to (5) if and only if there exists $y_i^* \in \mathbb{R}^n$ such that $(x_i^*, y_i^*)$ satisfies

$$\begin{align*}
x_i^* - A^T Ay_i^* &= 0 \\
A^T Ax_i^* &= a_i^T,
\end{align*}$$

(7)

that is,

$$\begin{align*}
x_i^* &= A^T Ay_i^* \\
a_i^T &= (A^T A)^2 y_i^*.
\end{align*}$$

(8)
So, \( X^* \) is an optimal solution of (3) if and only if there exists \( Y^* \in \mathbb{R}^{n \times m} \) such that \((X^*, Y^*)\) satisfies

\[
\begin{align*}
X^* &= A^TAY^* \\
A^T &= (A^TA)^2Y^*.
\end{align*}
\]

Similarly, \( X^* \) is an optimal solution to (4) if and only if there exists \( Z^* \in \mathbb{R}^{m \times n} \) such that \((X^*, Z^*)\) satisfies

\[
\begin{align*}
X^* &= AATZ^* \\
A &= (AA^T)^2Z^*.
\end{align*}
\]

Therefore, solving the generalized inverse problem is equivalent to solving (9) and (10), respectively.

III. PROPOSED ITERATIVE METHOD

From the discussion in Section II we know that solving the generalized inverse problem can be converted into solving algebraic equation (9) and (10), respectively. We thus propose the following iterative algorithms:

**Iterative algorithm (I) \((m \geq n)\).** Given an initial matrix \( Y^{(0)} \in \mathbb{R}^{n \times m} \) and \( \epsilon > 0 \). For \( k = 0, 1, \ldots, L \) (the maximum iteration number)

(i) Compute for \( i = 1, \ldots, m \)

\[
h^k_i = \frac{\|a^T_i - B_1y_i^{(k)}\|^2}{\|a^T_i - B_1y_i^{(k)}\|^2} \|(I_n + B_1)(a^T_i - B_1y_i^{(k)})\|^2,
\]

where \( B_1 = (A^TA)^2 \), \( I_n \in \mathbb{R}^{n \times n} \) is an unit matrix (the identity matrix), \( A^T = [a_1, \ldots, a_m] \), and \( Y^{(k)} = [y_1^{(k)}, \ldots, y_m^{(k)}] \).

(ii) Compute

\[
Y^{(k+1)} = Y^{(k)} + W_n(A^T - B_1Y^{(k)})H^{(k)},
\]

where \( H^{(k)} = \text{diag}(h_1^{(k)}, \ldots, h_m^{(k)}) \) and \( W_n = (I_n + B_1) \).

(iii) If \( \|A^T - B_1Y^{(k+1)}\|_F^2 \leq \epsilon \) and \( k \geq L \), then return \( Y^{(k+1)} \). Otherwise, let \( k := k + 1 \), and go to (i).

**Iterative algorithm (II) \((m < n)\).** Given an initial matrix \( Z^{(0)} \in \mathbb{R}^{m \times n} \) and \( \epsilon > 0 \). For \( k = 0, 1, \ldots, L \)

(i) Compute for \( i = 1, \ldots, n \)

\[
h^k_i = \frac{\|a_i - B_2z_i^{(k)}\|^2}{\|a_i - B_2z_i^{(k)}\|^2} \|(I_m + B_2)(a^T_i - B_2z_i^{(k)})\|^2,
\]

\[
W_m = (I_m + B_2).
\]

(ii) Compute

\[
Y^{(k+1)} = Y^{(k)} + W_m(A^T - B_2Y^{(k)})H^{(k)}.
\]

(iii) If \( \|A^T - B_2Y^{(k+1)}\|_F^2 \leq \epsilon \) and \( k \geq L \), then return \( Y^{(k+1)} \). Otherwise, let \( k := k + 1 \), and go to (i).
where \( B_2 = (AA^T)^2 \), \( I_m \in \mathbb{R}^{m \times m} \) is an unit matrix, \( A = [a_1, ..., a_n] \) and \( Z^{(k)} = [z_1^{(k)}, ..., z_n^{(k)}] \).

(ii) Compute
\[
Z^{(k+1)} = Z^{(k)} + W_m(A - B_2Z^{(k)})H^{(k)},
\]
where \( H^{(k)} = \text{diag}(h_1^{(k)}, ..., h_n^{(k)}) \) and \( W_m = (I_m + B_2). \)

(iii) If \( \|A^T - B_2Z^{(k+1)}\|_F^2 \leq \epsilon \) and \( k \geq L \), then return \( Z^{(k+1)} \). Otherwise, let \( k := k + 1 \), and go to (i).

It is easy to see that the proposed iterative algorithms (I) and (II) require to make four matrix and vector multiplications at each time iteration.

Remark 1. The proposed iterative algorithm (I) may be viewed as a discrete-time form of the following continuous-time recurrent neural network
\[
dY(t) = W_n\{A(t)^T - (A(t)^T A(t))^2 Y(t)\}
\]
where \( W_n \) is an weighted matrix is defined in (11). The generalized inverse solution is then obtained by repeatedly iterative learning. Also, the proposed iterative algorithm (II) may be viewed as a discrete-time form of the following continuous-time recurrent neural network:
\[
dZ(t) = W_m\{A(t) - (A(t)A(t)^T)^2 Z(t)\}
\]
where \( W_n \) is an weighted matrix is defined in (12).

We now compare the proposed iterative method with two conventional iterative algorithms. First, two Newton iterative algorithms are given in [11]:

\[
\begin{align*}
X_0 &= \alpha_0 A^T \\
X_{k+1} &= 2X_k - X_kAX_k \quad (k = 0, 2, ...)
\end{align*}
\]

and

\[
\begin{align*}
X_0 &= \alpha_0 A^T \\
X_{k+1} &= h_0(2I - AX_k)X_k \quad (k = 0, 2, ...) \\
h_0 &= 2/(1 + (2 - \rho)\rho)
\end{align*}
\]

where \( \alpha_0 \) and \( \rho \) are design parameters and \( I \) is an unit matrix (the identity matrix). Both of them require two matrix operations per iteration and they have computation complexity \( O(N^3) \) per iteration. The two Newton iterative algorithms have a fast convergence rate. However, they require the initial condition for their convergence. In addition, the design parameter \( \alpha_0 \) must be set to be proper small with \( \alpha_0 < 2/\sigma_1^2 \) and \( \rho = 2\sigma_0/(\sigma_0 + \sigma_1) \) since \( \sigma_0 \) and \( \sigma_1 \) are the smallest and the
largest singular value of matrix $A$, respectively. Next, a family of higher-order convergent iterative methods for computing the Moore-Penrose inverse is given by

$$X_{k+1} = X_k[C_1^t - C_2^t(AX_k) + \ldots + (-1)^{t-1}C_t^t(AX_k)^{t-1}], \quad t = 2, 3, ..$$

It is seen that the higher-order iterative algorithm has a lower algorithm complexity $O((t-1)N^3)$ per iteration. Moreover, it also requires an initial approximation condition for convergence. In contrast, the proposed iterative algorithm requires an algorithm complexity $O(2N^3)$ per iteration and we can prove that it is globally convergent without any condition.

**Theorem 1.** (i) The sequence $\{Y^{(k)}\}$ generated by the iterative algorithm (I) with any initial point $Y^{(0)}$ is globally convergent to $Y^*$ satisfying (9). Moreover, $A^+ = (A^TA)^T$. (ii) The sequence $\{Z^{(k)}\}$ generated by the iterative algorithm (II) with any initial point $Z^{(0)}$ is globally convergent to $Z^*$ satisfying (10). Moreover, $A^+ = (AA^T)Z^T$.

**Proof.** Note that for $i = 1, \ldots, m$,

$$y_i^{(k+1)} = y_i^{(k)} + h_i^{(k)}(I_n + B_1)(a_i^T - (A^TA)^2y_i^{(k)}), \quad (15)$$

it follows that

$$\begin{align*}
\|y_i^{(k+1)} - y_i^*\|^2 &= \|y_i^{(k)} - y_i^*\|^2 + (h_i^{(k)})^2\|\langle I_n + B_1 \rangle (a_i^T - (A^TA)^2y_i^{(k)})\|^2 \\
&\quad + 2h_i^{(k)}(y_i^{(k)} - y_i^*)^T(a_i^T - (A^TA)^2y_i^{(k)}) \\
&\quad + 2h_i^{(k)}(y_i^{(k)} - y_i^*)^TB_1(a_i^T - (A^TA)^2y_i^{(k)})
\end{align*}$$

where $Y^* = [y_1^*, \ldots, y_m^*]^T$. Note that $a_i^T = (A^TA)^2y_i^*$ and $B_1 = (A^TA)^2$. Then

$$(y_i^{(k)} - y_i^*)^TB_1(a_i^T - (A^TA)^2y_i^{(k)}) = -\|a_i^T - (A^TA)^2y_i^{(k)}\|^2$$

and

$$(y_i^{(k)} - y_i^*)^TB(a_i^T - (A^TA)^2y_i^{(k)}) = -\|B(y_i^{(k)} - y_i^*)\|^2.$$

where $B = A^TA$. By the definition of $h_i^{(k)}$ we have

$$\begin{align*}
\|y_i^{(k+1)} - y_i^*\|^2 &= \|y_i^{(k)} - y_i^*\|^2 \\
&+ h_i^{(k)}\|\langle I_n + B_1 \rangle (a_i^T - (A^TA)^2y_i^{(k)})\|^2 \\
&- 2h_i^{(k)}\|a_i^T - (A^TA)^2y_i^{(k)}\|^2 - 2h_i^{(k)}\|B(y_i^{(k)} - y_i^*)\|^2 \\
&\leq \|y_i^{(k)} - y_i^*\|^2 - 2h_i^{(k)}\|a_i^T - (A^TA)^2y_i^{(k)}\|^2 \\
&+ h_i^{(k)}\|\langle I_n + B_1 \rangle (a_i^T - (A^TA)^2y_i^{(k)})\|^2 \\
&= \|y_i^{(k)} - y_i^*\|^2 - h_i^{(k)}\|a_i^T - (A^TA)^2y_i^{(k)}\|^2.
\end{align*}$$
Then

\[ \|Y^{(k+1)} - Y^*\|^2_F \leq \sum_{i=1}^m (\|y_i^{(k)} - \hat{y}_i\|^2 - h_i^{(k)} \|a_i^T - (A^TA)^2 y_i^{(k)}\|^2) \]

\[ = \|Y^{(k)} - Y^*\|^2_F - \frac{\|A^T - B_1 Y^{(k)}\|^4_F}{\|(I_n + B_1)(A^T - B_1 Y^{(k)})\|^2_F}. \]

Thus \( \{\|Y^{(k)} - Y^*\|^2\} \) is decreasing and \( \{Y^{(k)}\} \) is bounded. Let \( \gamma = 1/\|I_n + B_1\|^2_F \). Then for any positive integer \( N \) we have

\[ \gamma \sum_{k=0}^N \|A^T - B_1 Y^{(k)}\|^2_F \leq \sum_{k=0}^N (\|Y^{(k)} - Y^*\|^2_F - \|Y^{(k+1)} - Y^*\|^2_F) \]

\[ = \|Y^{(0)} - Y^*\|^2_F - \|Y^{(N+1)} - Y^*\|^2_F \]

\[ \leq \|Y^{(0)} - Y^*\|^2_F < +\infty. \]

Thus

\[ \lim_{k \to \infty} \|A^T - B_1 Y^{(k)}\|^2_F = 0, \]

and hence

\[ \lim_{k \to \infty} (A^TA)^2 Y^{(k)} = A^T. \]

Since \( \{Y^{(k)}\} \) is bounded, there exists subsequence \( \{k_j\} \) such that

\[ \lim_{j \to \infty} Y^{(k_j)} = \hat{Y}. \]

As \( \{\|Y^{(k)} - Y^*\|^2\} \) is decreasing, we have

\[ \lim_{k \to \infty} Y^{(k)} = \hat{Y} \]

where \( \hat{Y} \) satisfies \( A^+ = A^T A\hat{Y} \). It follows that

\[ \lim_{k \to \infty} \|X^{(k)} - A^+\|^2_F = \lim_{k \to \infty} \|A^T Y^{(k)} - A^T A\hat{Y}\|^2_F = 0. \]

Similarly, we can obtain the convergence of the iterative algorithm (II). The proof is complete.
IV. ACCELERATION SCHEME

There is a step size $h^k_i$ in the sample iterative algorithms of Section II, which very depends on the size of the matrix $A$. In the case of large-size matrices, $h^k_i$ may be very small and thus it will result in slow convergence. In order to speed up the proposed iterative algorithm, in this section we present the following improved algorithm:

**Improved iterative algorithm (I)** ($m \geq n$). Given an initial matrix $X(0) \in \mathbb{R}^{n \times m}$ and $\epsilon > 0$. For $k = 0, 1, ..., L$

(i) Compute

$$Y^{(k+1)} = Y^{(k)} + W_n (A^T - B_1 Y^{(k)}),$$

where $W_n = (I_n + (A^T A)^2)^{-1}$ and $B_1 = (A^T A)^2$.

(ii) If $\|A^T - B_1 Y^{(k+1)}\|_F^2 \leq \epsilon$ and $k \geq L$, then output $Y^{(k+1)}$. Otherwise, let $k := k + 1$, and go to (i).

**Improved iterative algorithm (II)** ($m < n$). Given an initial matrix $Z(0) \in \mathbb{R}^{m \times n}$ and $\epsilon > 0$. For $k = 0, 1, ..., L$

(i) Compute

$$Z^{(k+1)} = Z^{(k)} + W_m (A^T - B_2 Z^{(k)}),$$

where $W_m = (I_m + (AA^T)^2)^{-1}$ and $B_2 = (AA^T)^2$.

(ii) If $\|A^T - B_2 Z^{(k+1)}\|_F^2 \leq \epsilon$ and $k \geq L$, then output $Z^{(k+1)}$. Otherwise, let $k := k + 1$, and go to (i).

It should point out that when performing the proposed accelerated algorithm, we first compute the inverse of a matrix such as $(I_n + B_1)$ before iteration. The matrix is symmetric positive-definite with eigenvalue being always greater than 1 and thus is usually good condition. The Matrix Inversion Computation of Symmetric Positive-Definite Matrices have been well developed in the matrix computation Toolbox [22]. In addition to computing the inverse of a symmetric positive-definite matrix before iteration, the proposed improved iterative algorithm requires a computation complexity $O(N^3)$ per iteration. Because the improved algorithms (I) and (II) don’t include the step length and thus can speed up the proposed iterative algorithms (I) and (II) in Section III.

As for their global convergence, we have the following results.

**Theorem 2.** (i) The sequence $\{Y^{(k)}\}$ generated by the improved iterative algorithm (I) with any initial point $Y^{(0)}$ is globally convergent to a point $Y^*$ satisfying (9). Moreover, $A^+ = A^T A Y^*$. (ii)
The sequence \( \{Z^{(k)}\} \) generated by the improved iterative algorithm (II) with any initial point \( Z^{(0)} \) is globally convergent to \( Z^* \) satisfying (10). Moreover, \( A^+ = (AA^T Z^*)^T \).

Proof. Let \( Y^* = [y_1^*, ..., y_m^*] \) and let \( P = (I_n + B_1) \). From the improved iterative algorithm (I) we have for \( i = 1, ..., m \)

\[
\|y_i^{(k+1)} - y_i^*\|_P^2 = \|P(y_i^{(k+1)} - y_i^*)\|^2 = \|P(y_i^{(k)} - y_i^*)\|^2 \\
+ 2(y_i^{(k)} - y_i^*)^T (I_n + B_1)^2 (I_n + B_1)^{-1} (a_i^T - B_1 y_i^{(k)}) \\
+ \| (a_i^T - B_1 y_i^{(k)}) \|^2 \\
= \|y_i^{(k)} - y_i^*\|_P^2 - 2(y_i^{(k)} - y_i^*)^T (B_1 y_i^{(k)} - a_i^T) \\
- 2(y_i^{(k)} - y_i^*)^T B_1 (B_1 y_i^{(k)} - a_i^T) + \| (a_i^T - B_1 y_i^{(k)}) \|^2.
\]

Because \( a_i^T = B_1 y_i^* \),

\[
(y_i^{(k)} - y_i^*)^T (B_1 y_i^{(k)} - a_i^T) \geq 0.
\]

It follows that

\[
\|y_i^{(k+1)} - y_i^*\|_P^2 \\
\quad \leq \|y_i^{(k)} - y_i^*\|_P^2 - \|a_i^T - B_1 y_i^{(k)}\|^2.
\]

The rest of this proof is similar to Theorem 1. Similarly, we can obtain the convergence of the improved iterative algorithm (II). The proof is complete.

V. ILLUSTRATIVE EXAMPLES

In this section, we give illustrative examples to demonstrate the effectiveness of the proposed sample iterative algorithm. The simulation is conducted in MATLAB. To test the accuracy of the proposed algorithm we use the error formula: \( \| AXA - A \|^2 \), \( \| XAX - X \|^2 \), \( \| AX - (AX)^T \|^2 \), and \( \| XA - (XA)^T \|^2 \) where \( \| \cdot \| \) denotes the \( l_2 \) norm of the matrix.

**Example 1.** Consider a \( 6 \times 4 \) rank-deficient matrix below

\[
A = \begin{pmatrix}
-1 & 0 & 1 & 2 \\
-1 & 1 & 0 & -1 \\
0 & -1 & 1 & 3 \\
0 & 1 & -1 & -3 \\
1 & -1 & 0 & 1 \\
1 & 0 & -1 & -2
\end{pmatrix}.
\]
The rank of matrix $A$ is 2 and its generalized inverse is given by

$$A^+ = \frac{1}{102} \begin{pmatrix} -15 & -18 & 3 & -3 & 18 & 15 \\ 8 & 13 & -5 & 5 & -13 & -8 \\ 7 & 5 & 2 & -2 & -5 & -7 \\ 6 & -3 & 9 & -9 & 3 & -6 \end{pmatrix}.$$  

We first perform the sample iterative method (11). Let $\epsilon = 10^{-9}$. After 2500 iterations, we get $\hat{X}$ with the error $\|A^+ - \hat{X}\|_F \leq 10^{-8}$. Next, we perform the proposed accelerated algorithm (16). After 5 iterations, we get $\hat{X}$ with the same error above. Furthermore, for comparative purpose we perform the Newton iterative algorithm and the full rank QR-factorization algorithm. Computed results are listed in Table 1. From the computed results, we can see that like the full rank QR-factorization algorithm, the proposed iterative algorithms (16) can give the better solution. The Newton iterative algorithm has a faster convergence speed, but it requires an initial condition.

**Example 2.** Consider the following $110 \times 10$ matrix

$$P = \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \\ 1 & 1 & 0 & \ldots & 0 \\ 1 & 1 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \ldots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \ldots & 1 \end{bmatrix}$$

which is a full-rank matrix. For comparative purpose we perform the proposed iterative algorithm, the Newton iterative algorithm, and the full rank QR-factorization algorithm. Computed results are listed in Table 2. From the computed results, we can see that like the full rank QR-factorization algorithm, the proposed iterative algorithm (16) can give the better solution. The Newton iterative algorithm has a faster convergence speed, but it requires an initial condition.
Example 3. Consider random test matrices with large condition number. Let the test matrix be a $100 \times 100$ matrix given by the function matrix $(3, 100)$ from paper [16], where the condition number is $8.11 \times 10^{17}$. For comparative purpose we perform the proposed iterative algorithm (16), the Newton iterative algorithm, and the full rank QR-factorization algorithm. The obtained numerical results are reported in Table 3. From the computed results, we see that the proposed iterative algorithm (16) can get a better solution. The Newton iterative algorithm has a faster convergence speed, but it requires an initial condition.

Example 4. Consider a set of singular test matrices that includes 4 singular matrices (called chow, prolate,hilb and vand matrices, respectively), of size $200 \times 200$, obtained from the function matrix in the Matrix Computation Toolbox (mctoolbox) [22]. The condition number of these matrices range from order $1015$ to $10135$. In paper [16], the QR-factorization algorithm has been compared with several conventional algorithms by using these singular test matrices. For comparative purpose we here compare the proposed iterative algorithm with the high-order iterative algorithm. The obtained numerical results are reported in Table 3 where the order of the higher-order method is taken as $t = 6$ and $t = 8$. From the computed results, we see that the proposed iterative method gives better accurate results for all matrices and has a faster speed than the high-order iterative algorithm.

### Table II
**Computed results of the full-rank matrix in Example 2**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Initial point</th>
<th>$|AX - A|_2$</th>
<th>$|AX - X|_2$</th>
<th>$|AX - (AX)^T|_2$</th>
<th>$|XAX - (XAX)^T|_2$</th>
<th>CPU time(s)</th>
<th>Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Newton algorithm</td>
<td>$X_0 = 10^{-1}A^T$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>0.015</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>$X_0 = 10^{-1}A^T$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>0.015</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>$X_0 = O$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>0.015</td>
<td>27</td>
</tr>
<tr>
<td>New algorithm (16)</td>
<td>$X_0 = 10^{-3}A^T$</td>
<td>5.29 $\times$ $10^{-14}$</td>
<td>3.11 $\times$ $10^{-15}$</td>
<td>1.76 $\times$ $10^{-15}$</td>
<td>9.9 $\times$ $10^{-15}$</td>
<td>0.016</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td>$X_0 = 10^{-3}A^T$</td>
<td>3.24 $\times$ $10^{-14}$</td>
<td>1.888 $\times$ $10^{-15}$</td>
<td>1.1 $\times$ $10^{-15}$</td>
<td>5.99 $\times$ $10^{-13}$</td>
<td>0.015</td>
<td>130</td>
</tr>
<tr>
<td></td>
<td>$X_0 = O$</td>
<td>2.21 $\times$ $10^{-14}$</td>
<td>1.26 $\times$ $10^{-15}$</td>
<td>9.83 $\times$ $10^{-16}$</td>
<td>3.964 $\times$ $10^{-14}$</td>
<td>0.016</td>
<td>150</td>
</tr>
<tr>
<td>QR algorithm</td>
<td>2.75 $\times$ $10^{-13}$</td>
<td>13.02 $\times$ $10^{-15}$</td>
<td>8.77 $\times$ $10^{-14}$</td>
<td>8.08 $\times$ $10^{-15}$</td>
<td>0.015</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table III
**Computed results of the bad behaved condition matrix in Example 3**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Initial point</th>
<th>$|AX - A|_2$</th>
<th>$|AX - X|_2$</th>
<th>$|AX - (AX)^T|_2$</th>
<th>$|XAX - (XAX)^T|_2$</th>
<th>CPU time(s)</th>
<th>Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Newton algorithm</td>
<td>$X_0 = 10^{-3}A^T$</td>
<td>12.84</td>
<td>0.004</td>
<td>1.26 $\times$ $10^{-17}$</td>
<td>1.32 $\times$ $10^{-17}$</td>
<td>0.015</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>$X_0 = 10^{-3}A^T$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>0.015</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>$X_0 = O$</td>
<td>0.1234</td>
<td>0.7633</td>
<td>4.5 $\times$ $10^{-15}$</td>
<td>2.37 $\times$ $10^{-13}$</td>
<td>0.016</td>
<td>8</td>
</tr>
<tr>
<td>New algorithm (16)</td>
<td>$X_0 = 10^{-3}A^T$</td>
<td>0.1218</td>
<td>0.7607</td>
<td>7.88 $\times$ $10^{-15}$</td>
<td>6.78 $\times$ $10^{-12}$</td>
<td>0.015</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>$X_0 = 10^{-3}A^T$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>0.015</td>
<td>8</td>
</tr>
<tr>
<td>QR-based algorithm</td>
<td>0.0103</td>
<td>1.67 $\times$ $10^8$</td>
<td>0.015</td>
<td>0.0077</td>
<td>0.015</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

11
TABLE IV
COMPUTED RESULTS OF FOUR SINGULAR TEST MATRICES IN EXAMPLE 4

<table>
<thead>
<tr>
<th>Chow matrix</th>
<th>initial point</th>
<th>( |XAX - A|^2 )</th>
<th>( |XAX - X|^2 )</th>
<th>( |XAX - (AX)^T|^2 )</th>
<th>( |XAX - (AX)^T|^2 )</th>
<th>CPU time(sec)</th>
<th>Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>High-order algorithm ( t = 8 )</td>
<td>( X_0 = A^{T}/\sigma )</td>
<td>( 1.4 \times 10^{-16} )</td>
<td>( 7.1 \times 10^{-17} )</td>
<td>( 1.1 \times 10^{-16} )</td>
<td>( 1.9 \times 10^{-17} )</td>
<td>1.77</td>
<td>21</td>
</tr>
<tr>
<td>High-order algorithm ( t = 6 )</td>
<td>( X_0 = A^{T}/\sigma )</td>
<td>( 2.2 \times 10^{-14} )</td>
<td>( 7.5 \times 10^{-16} )</td>
<td>( 9.1 \times 10^{-16} )</td>
<td>( 1.5 \times 10^{-14} )</td>
<td>1.75</td>
<td>26</td>
</tr>
<tr>
<td>New algorithm (16)</td>
<td>( X_0 = A^{T}/\sigma )</td>
<td>( 1.2 \times 10^{-15} )</td>
<td>( 4.7 \times 10^{-15} )</td>
<td>( 3.8 \times 10^{-14} )</td>
<td>( 5.5 \times 10^{-13} )</td>
<td>1.89</td>
<td>128</td>
</tr>
<tr>
<td>Prolate matrix</td>
<td>initial point</td>
<td>( |XAX - A|^2 )</td>
<td>( |XAX - X|^2 )</td>
<td>( |XAX - (AX)^T|^2 )</td>
<td>( |XAX - (AX)^T|^2 )</td>
<td>CPU time(sec)</td>
<td>Iteration</td>
</tr>
<tr>
<td>High-order algorithm ( t = 8 )</td>
<td>( X_0 = A^{T}/\sigma )</td>
<td>( 8.1 \times 10^{-4} )</td>
<td>( 1.5 \times 10^{-2} )</td>
<td>( 3.5 \times 10^{-4} )</td>
<td>( 4.6 \times 10^{-4} )</td>
<td>0.422</td>
<td>6</td>
</tr>
<tr>
<td>High-order algorithm ( t = 6 )</td>
<td>( X_0 = A^{T}/\sigma )</td>
<td>( 2.6 \times 10^{-4} )</td>
<td>( 4.1 \times 10^{-4} )</td>
<td>( 2.7 \times 10^{-4} )</td>
<td>( 4.5 \times 10^{-4} )</td>
<td>0.438</td>
<td>8</td>
</tr>
<tr>
<td>New algorithm (16)</td>
<td>( X_0 = A^{T}/\sigma )</td>
<td>( 1.9 \times 10^{-2} )</td>
<td>( 7.1 \times 10^{-1} )</td>
<td>( 1.7 \times 10^{-10} )</td>
<td>( 5.0 \times 10^{-10} )</td>
<td>0.055</td>
<td>19</td>
</tr>
<tr>
<td>Hilbert matrix</td>
<td>initial point</td>
<td>( |XAX - A|^2 )</td>
<td>( |XAX - X|^2 )</td>
<td>( |XAX - (AX)^T|^2 )</td>
<td>( |XAX - (AX)^T|^2 )</td>
<td>CPU time(sec)</td>
<td>Iteration</td>
</tr>
<tr>
<td>High-order algorithm ( t = 8 )</td>
<td>( X_0 = A^{T}/\sigma )</td>
<td>( 3.7 \times 10^{-3} )</td>
<td>( 2.3 \times 10^{-3} )</td>
<td>( 6.3 \times 10^{-3} )</td>
<td>( 4.7 \times 10^{-3} )</td>
<td>0.975</td>
<td>7</td>
</tr>
<tr>
<td>High-order algorithm ( t = 6 )</td>
<td>( X_0 = A^{T}/\sigma )</td>
<td>( 1.5 \times 10^{-3} )</td>
<td>( 5.0 \times 10^{-3} )</td>
<td>( 9.2 \times 10^{-3} )</td>
<td>( 8.3 \times 10^{-3} )</td>
<td>0.975</td>
<td>7</td>
</tr>
<tr>
<td>New algorithm (16)</td>
<td>( X_0 = A^{T}/\sigma )</td>
<td>( 0.25 )</td>
<td>( 0.47 )</td>
<td>( 7.8 \times 10^{-17} )</td>
<td>( 4.6 \times 10^{-10} )</td>
<td>0.046</td>
<td>17</td>
</tr>
<tr>
<td>Vand matrix</td>
<td>initial point</td>
<td>( |XAX - A|^2 )</td>
<td>( |XAX - X|^2 )</td>
<td>( |XAX - (AX)^T|^2 )</td>
<td>( |XAX - (AX)^T|^2 )</td>
<td>CPU time(sec)</td>
<td>Iteration</td>
</tr>
<tr>
<td>High-order algorithm ( t = 8 )</td>
<td>( X_0 = A^{T}/\sigma )</td>
<td>( 2.2 \times 10^{-2} )</td>
<td>( 6.4 \times 10^{-3} )</td>
<td>( 5.2 \times 10^{-14} )</td>
<td>( 1.1 \times 10^{-12} )</td>
<td>0.579</td>
<td>8</td>
</tr>
<tr>
<td>High-order algorithm ( t = 6 )</td>
<td>( X_0 = A^{T}/\sigma )</td>
<td>( 1.0 \times 10^{-3} )</td>
<td>( 1.1 \times 10^{-3} )</td>
<td>( 8 \times 10^{-14} )</td>
<td>( 8.6 \times 10^{-13} )</td>
<td>0.547</td>
<td>10</td>
</tr>
<tr>
<td>New algorithm (16)</td>
<td>( X_0 = A^{T}/\sigma )</td>
<td>( 0.17 )</td>
<td>( 0.75 )</td>
<td>( 8.2 \times 10^{-10} )</td>
<td>( 9.1 \times 10^{-14} )</td>
<td>0.039</td>
<td>15</td>
</tr>
</tbody>
</table>

VI. Conclusion

In this paper, we proposed a new iterative method for computing generalized inverse. Different from other iterative methods in literatures, the proposed iterative method is based on a new KKT condition. The global convergence of the proposed iterative algorithm is proved without any condition. Furthermore, for fast computing generalized inverse we presented an improved iterative algorithm for acceleration convergence. The global convergence of the proposed improved algorithm is also analyzed. Finally, effectiveness of the proposed iterative algorithm is evaluated numerically.

References


19) W.G. Li, Z. Li, A family of iterative methods for computing the approximate inverse of a square matrix and inner inverse of a non-square matrix, *Applied Mathematics and Computa-

