Indian J Pure Appl Math (2026) 57:1900–1913
https://doi.org/10.1007/s13226-025-00753-1

ORIGINAL RESEARCH

A new efficient iterative method to compute the weighted Moore–Penrose inverse

Salman Sheikhi · Hamid Esmaeili

Received: 22 December 2023 / Accepted: 9 January 2025 / Published online: 27 January 2025
© The Indian National Science Academy 2025

Abstract In this paper, we introduce a new efficient iterative method to compute the weighted Moore–Penrose inverse of a singular or rectangular real (or complex) matrix, which requires five matrix multiplications per iteration. The proposed method has been shown to be fourth-order convergent to the weighted Moore–Penrose inverse. The average number of iterations, the average total number of matrix multiplications, and the average elapsed execution time of compared methods are computed using ten matrices in each dimension. The numerical results are based on random matrices of sizes $m \times m$ and $m \times (m+10)$, where $m = 200, 300, 400, 500, 600$. Based on this comparison, our new method can be considered as an efficient and fast method with low computational cost to compute the weighted Moore–Penrose inverse.

Keywords Weighted Moore–Penrose inverse · Iterative method · Fourth-order convergent · Matrix multiplication

Mathematics Subject Classification 65F10 · 65F20 · 65N99

1 Introduction

The computation of the generalized inverses of matrices or operators has been considered important in numerical and computational mathematics. Indeed, in case a matrix is singular or rectangular, diverse types of generalized inverses are preferred in numerous computational and theoretical problems [1]. The weighted Moore–Penrose inverse (or, WMP inverse, for short) of a complex matrix is one of the significant generalized inverses. When the dimension is finite, minimizing a Hermitian positive definite quadratic form $x^t x$ under linear restrictions, which t denotes the transpose, is in range of the accessible applications for the Moore–Penrose inverse. The WMP inverse has a chief effect in the indefinite linear least-square problems [2,3]. Especially, it is practical for minimizing the weighted linear least-square problem

$$\|Ax - b\|_W^2 = (Ax - b)^* W (Ax - b), \tag{1}$$

where W is a given positive definite matrix. When A is not a full column rank, this problem does not have a unique solution for x. In the set of all solutions, the one may be chosen which

$$\|x\|_Q^2 = x^* Q x \tag{2}$$

Communicated by Priyanka Grover.

Salman Sheikhi and Hamid Esmaeili contributed equally to this work.

S. Sheikhi · H. Esmaeili (✉)
Department of Mathematics, Faculty of Science, Bu-Ali Sina University, Hamedan, Iran
E-mail: esmaeili@basu.ac.ir
S. Sheikhi
E-mail: s.sheikhi@sci.basu.ac.ir

is the smallest, where Q is a second positive definite matrix and A^* denotes the conjugate transpose of A [1]. The foundation and significance of the WMP inverse for an arbitrary matrix has been considered in [1,4,5].

Assume $A \in \mathbb{C}^{m\times n}$ and positive definite matrices $M \in \mathbb{C}^{m\times m}$ and $N \in \mathbb{C}^{n\times n}$ be complex. There is a unique matrix $X \in \mathbb{C}^{n\times m}$ called the WMP inverse of A with respect to M and N, and denoted by $A^{\dagger}_{MN}$, if the following four conditions are confirmed:

$$AXA = A, \quad XAX = X, \quad (MAX)^* = MAX, \quad (NXA)^* = NXA. \tag{3}$$

It is attested that once M and N are Hermitian positive definite matrices, the unique WMP inverse $A^{\dagger}_{MN}$ exists [1]. In this paper, the matrices M and N are regarded as the Hermitian positive definite and consequently the WMP inverse is defined in a unique form. Especially, as $M = I_m$ and $N = I_n$, in which I_k denotes the identity matrix of the order k, the matrix X is entitled the Moore–Penrose inverse or the generalized pseudo-inverse of A and is denoted by $A^{\dagger}$, whereas (1) it is shortened to the conventional Penrose equations primitively ascribed to [6] as follows:

$$AXA = A, \quad XAX = X, \quad (AX)^* = AX, \quad (XA)^* = XA. \tag{4}$$

The WMP inverse, regarded as a generalized Moore–Penrose inverse, is widely applied to control system analysis, weighted least-squares problems, statistics, iterative methods, singular differential and difference equations, Markov chains, neural network problems, perturbation theory, and other subjects established in [7–11].

There is some current research that focuses on schemes to compute the (weighted) Moore–Penrose inverse of a matrix (see, e.g., [12–20]).

The weighted conjugate transpose matrix of A is denoted as follows:

$$A^{\#} = N^{-1}A^*M. \tag{5}$$

According to the weighted singular value decomposition, a primary method is founded on computing the WMP inverse [21]. Assume that $rank(A) = r$, the matrices $U \in \mathbb{C}^{m\times m}$ and $V \in \mathbb{C}^{n\times n}$ are exist and satisfied in $U^*MU = I_m$ and $V^*N^{-1}V = I_n$. Moreover,

$$A = U\begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix}V^*, \tag{6}$$

in which

$$D = diag(\sigma_1, \ldots, \sigma_r), \quad \sigma_1 \geq \cdots \geq \sigma_r > 0,$$

and $\sigma_1^2, \ldots, \sigma_r^2$ are the nonzero eigenvalues of $A^{\#}A$. Accordingly, we can define the WMP inverse $A^{\dagger}_{MN}$ by

$$A^{\dagger}_{MN} = N^{-1}V\begin{bmatrix} D^{-1} & 0 \\ 0 & 0 \end{bmatrix}U^*M. \tag{7}$$

Further,

$$\|A\|_{MN} = \sigma_1, \quad \|A^{\dagger}_{MN}\|_{NM} = \frac{1}{\sigma_r}, \tag{8}$$

where $\|A\|_{MN} = \|M^{\frac{1}{2}}AN^{-\frac{1}{2}}\|$.

To obviate restrictions of applying weighted singular value decomposition in order to compute the WMP inverse, some attempts have been made to apply iterative methods. Huang and Zhang [12], using the Schulz iterative method [22]

$$X_{k+1} = X_k(2I - AX_k), \quad k = 0, 1, 2, \ldots, \tag{9}$$

tried to compute the WMP inverse with the initial approximation

$$X_0 = \alpha A^{\#}, \quad 0 < \alpha < \frac{2}{\sigma_1^2}. \tag{10}$$

It is obvious that the Schulz iteration consists of only two matrix multiplications and the convergence rate is quadratic, Which means that for the residual matrix $E_k = I - AX_k$ we obtain $E_{k+1} = E_k^2$. This method



is numerically stable but convergence can be slow when the approximations X_k are not enough close to the solution. Based on this insufficiency higher-order iterative methods were created. Soleimani et al. [23] asserted the eighth-order convergence based on Householder's method [24] as follows:

$$\begin{cases} S_k = AX_k, \\ Y_k = S_k(-2I + S_k), \\ V_k = 2I + Y_k, \\ X_{k+1} = -X_k(-2I + S_k)V_k(2I + Y_kV_k), \quad n = 0, 1, \ldots, \end{cases} \tag{11}$$

in which X_0 is the same in (10) and has only six matrix multiplications. Soheili et al. [25] proposed a convergent twelfth-order method with eight matrix multiplications as follows:

$$\begin{cases} S_k = AX_k, \\ Y_k = -7I + S_k(9I + S_k(-5I + S_k)), \\ V_k = S_kY_k, \\ X_{k+1} = -\frac{1}{16}X_kY_k(4I + V_k)(8I + V_k(4I + V_k)), \quad n = 0, 1, \ldots. \end{cases} \tag{12}$$

Soleymani et al. [26] introduced a sixth-order method consisting of five matrix multiplications in the following form:

$$\begin{cases} S_k = AX_k, \\ Y_k = S_k(-I + S_k), \\ X_{k+1} = X_k(2I - S_k)(3I - 2S_k + Y_k)(I + Y_k), \quad n = 0, 1, \ldots. \end{cases} \tag{13}$$

Kaur et al. [20] proposed a new fifth-order hyperpower iterative method for approximating the WMP inverse of a given matrix, using four matrix multiplications per iteration, as follows:

$$\begin{cases} S_k = AX_k, \\ Y_k = S_k^2, \\ V_k = 5(I - S_k), \\ X_{k+1} = X_k(V_k - 5S_k + Y_k(5I + V_k + Y_k)), \quad k = 0, 1, \ldots. \end{cases} \tag{14}$$

In this paper, we propose a new iterative method to compute the WMP inverse which involves five matrix multiplications per iteration. It is shown that this method is consistently fourth-order convergent. Numerical experiments express that our method outperforms some existing methods (9), (11), (12), (13), and (14).

In the sequel, it is arranged as follows. In Sect. 2, we propose our new method to compute the WMP inverse and prove that it is convergent with the fourth order. In Sect. 3, some numerical examples are given to exhibit the performance of the presented method compared with other methods both in computing the Moore–Penrose inverse and the weighted Moore–Penrose inverse. For each of sizes $m \times (m + 10)$, $m = 200, 300, 400, 500, 600$, ten dense matrices were chosen randomly to make these comparisons, and also the methods are compared by solving a PDE problem. We represent the average number of iterations, the average total number of matrix multiplications, and the average CPU time elapsed of methods required to converge. Finally, some conclusions are outlined in Sect. 4.

2 A new iterative method

In this section, we first introduce a new iterative method to compute the WMP inverse and then prove its convergence. To this aim, we note that the class of Schulz-type methods, which used just matrix multiplications, are based on the following structure

$$X_{k+1} = X_kh(AX_k), \quad g(x) = xh(x), \quad s.t \quad g^{(i)}(1) = 0, \quad i = 1, 2, \ldots, p - 1,$$

in which $h(x)$ is a polynomial and $x = 1$ is the fixed point of $g(x)$. Thus, the sequence $\{X_k\}$ is pth-order convergent to the Moore–Penrose inverse of the matrix A. Therefore, consider the following polynomial

$$g(x) = \frac{1}{4}(44x - 185x^2 + 380x^3 - 410x^4 + 224x^5 - 49x^6). \tag{15}$$

In the following theorem, it is proved that the fixed point sequence $x_{k+1} = g(x_k)$ is fourth-order convergent to $x = 1$ for any $x_0 \in (0, 1]$.

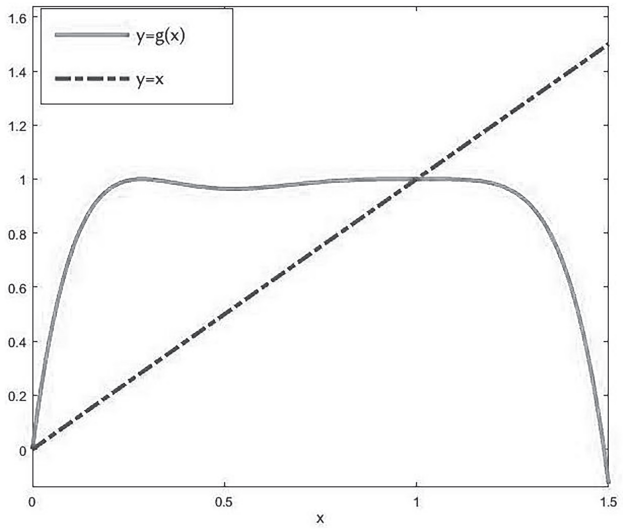


Fig. 1 Graphs of the line $y = x$ and the function $y = g(x)$

Theorem 1 *For any initial point $x_0 \in (0, 1]$, the sequence $x_{k+1} = g(x_k)$ is fourth-order convergent to $x = 1$, in which the function $g(x)$ is defined by* (15)*.*

Proof We can discern that the real fixed points and the critical points of $g(x)$ are as follows:

$$\begin{aligned} g(x) = x \quad &\Rightarrow \quad x = 0, 1, \\ g'(x) = 0 \quad &\Rightarrow \quad x = 1, \frac{2}{7}, \frac{11}{21}. \end{aligned}$$

Notice that also $g''(1) = g'''(1) = 0$. Since $g^{(4)}(1) < 0$ and $g''(\frac{2}{7}) < 0$, the points $x = 1$ and $x = \frac{2}{7}$ are the maximizers of the function $g(x)$. In addition, we note that $0 < g''(\frac{11}{21})$, so the point $x = \frac{11}{21}$ is the minimizer of $g(x)$. Based on the facts that $g(\frac{2}{7}) = g(1) = 1$ and $g(0) = 0 < g(\frac{11}{21}) \approx \frac{964}{1000}$, we conclude that the points $x = 1$ and $x = \frac{2}{7}$ are the global maximizers and the point $x = 0$ is the global minimizer of the function $g(x)$ on the interval $[0, 1]$ (see Fig. 1). Therefore, for any $x \in [0, 1]$ we have $0 \leq g(x) \leq 1$. It means that the function $g(x)$ maps the interval $[0, 1]$ into itself.

Now, we show that the sequence $x_{k+1} = g(x_k)$ is increasing for any $x_0 \in [0, 1]$. To this end, note that

$$g(x) - x = \frac{x(1-x)}{4}(49x^4 - 175x^3 + 235x^2 - 145x + 40).$$

Take

$$p(x) = 49x^4 - 175x^3 + 235x^2 - 145x + 40.$$

For the polynomial function $p(x)$, we have

$$\begin{aligned} p'(x) &= 196x^3 - 525x^2 + 470x - 145, \\ p''(x) &= 588x^2 - 1050x + 470, \end{aligned}$$

in which the value of the discriminant of $p''(x)$ is $\Delta = -2940 < 0$. It implies that $p''(x)$ is strictly positive and $p''(x) \geq \frac{2940}{4\times 588} = \frac{5}{4}$. Thus, $p'(x)$ is a strictly increasing function and has at most one real root. Besides, since $p'(1)p'(2) < 0$, $p'(x)$ has a real root γ in the interval $(1, 2)$. Hence, for any $x < \gamma$ we have $p'(x) < 0$, this means the function $p(x)$ is strictly decreasing for any $x < \gamma$. Since $p(0) = 40$ and $p(1) = 4$, $p(x)$ is strictly positive in the interval $[0, 1]$ and we have $p(x) \geq 4$ for any $x \in [0, 1]$. Now, since $g(x) - x = \frac{x(1-x)}{4}p(x)$, it could result in

$$\forall x \in [0, 1]: \quad g(x) - x \geq 0,$$

and therefore

$$\forall x_k \in [0,1]: \quad g(x_k) - x_k \geq 0 \quad \rightarrow \quad x_{k+1} = g(x_k) \geq x_k \quad \rightarrow \quad x_{k+1} \geq x_k.$$

So, the sequence $\{x_k\}$ is increasing and bounded from above by 1. Since the function $g(x)$ has only two fixed points $x = 0$ and $x = 1$, the sequence $\{x_k\}$ is convergent to $x = 1$ for any $x_0 \in (0, 1]$. Take $e_k = x_k - 1$ and note that,

$$e_{k+1} = x_{k+1} - 1 = g(x_k) - g(1) = -\frac{150}{4!}e_k^4 - \frac{2100}{5!}e_k^5 - \frac{8820}{6!}e_k^6 + \cdots.$$

It implies that the order of convergence of the sequence $x_{k+1} = g(x_k)$ is four. □

Corollary 1 *For any initial point $x_0 \in [1, 1.48757)$, the sequence $x_{k+1} = g(x_k)$ is fourth-order convergent to $x = 1$, in which the function $g(x)$ is defined by* (15).

Proof It is clear that $g'(x) < 0$ for any $1 < x$. In addition, the real roots of $g(x) = 0$ are as follows:

$$x = 0, \quad x \approx 1.4875726467,$$

(which computed by MATLAB). Since $g(1) = 1$, for any $x \in [1, 1.48757)$, we have $0 < g(x) \leq 1$. This means that with the initial point x_0 from this interval, the next iteration of the sequence $x_{k+1} = g(x_k)$ belongs to the interval $(0, 1]$. Now, according to Theorem 1, the sequence $\{x_k\}$ is convergent to $x = 1$ for any $x_0 \in [1, 1.48757)$. □

Using the function (15), we obtain the following iterative method to compute $A^{\dagger}_{MN}$:

$$X_{k+1} = \frac{1}{4}X_k(44I - 185AX_k + 380(AX_k)^2 - 410(AX_k)^3 + 224(AX_k)^4 - 49(AX_k)^5),$$

which can be rewritten as follows:

$$\begin{cases} S_k = AX_k, \\ Y_k = S_k(7S_k - 16I) + 11I, \\ X_{k+1} = \frac{1}{4}X_k(4I - Y_kS_k)Y_k, \quad n = 0, 1, \ldots. \end{cases} \tag{16}$$

The iterative method (16) falls in the class of Schulz-type methods, which uses matrix multiplications to generate approximate inverses for matrices. It requires an initial matrix to commence the process and can rapidly converge, which is an advantage in comparison with other methods. A crucial challenge of applying iterative methods to compute the WMP is related to the initial matrix. The initial matrix plays a critical role to provide convergence. Correspondingly, we must apply the following initial matrix

$$X_0 = \beta A^{\#}, \tag{17}$$

where β is a suitable constant. In the sequel, through a mathematical analysis we first show under which conditions the method (16) converges.

Lemma 1 *The sequence $\{X_k\}$ generated by* (16) *with the initial matrix* (17) *satisfies*

$$\begin{aligned} &(MAX_k)^* = MAX_k, \quad (NX_kA)^* = NX_kA, \\ &X_kAA^{\dagger}_{MN} = X_k, \qquad A^{\dagger}_{MN}AX_k = X_k. \end{aligned} \tag{18}$$

Proof We will prove the conclusion by induction on k. For $k = 0$, using (17), the first two equations can be verified easily. So, we only give a verification to the last two equations. Using the facts that $(AA^{\dagger}_{MN})^{\#} = AA^{\dagger}_{MN}$ and $(A^{\dagger}_{MN}A)^{\#} = A^{\dagger}_{MN}A$, we have

$$\begin{aligned} X_0AA^{\dagger}_{MN} &= \beta A^{\#}AA^{\dagger}_{MN} = \beta A^{\#}(AA^{\dagger}_{MN})^{\#} = \beta(AA^{\dagger}_{MN}A)^{\#} = \beta A^{\#} = X_0, \\ A^{\dagger}_{MN}AX_0 &= \beta A^{\dagger}_{MN}AA^{\#} = \beta(A^{\dagger}_{MN}A)^{\#}A^{\#} = \beta(AA^{\dagger}_{MN}A)^{\#} = \beta A^{\#} = X_0. \end{aligned}$$

Assume now that the conclusion holds for some $k > 0$. We show that it continues to hold for $k+1$. Using the iterative method (16), one has

$$\begin{aligned}(MAX_{k+1})^* &= \frac{1}{4}(MAX_k(44I - 185AX_k + 380(AX_k)^2 - 410(AX_k)^3 \\ &\quad + 224(AX_k)^4 - 49(AX_k)^5))^* \\ &= \frac{1}{4}(MS_k(44I - 185S_k + 380S_k^2 - 410S_k^3 + 224S_k^4 - 49S_k^5))^* \\ &= \frac{1}{4}(44(MS_k)^* - 185(MS_k^2)^* + 380(MS_k^3)^* - 410(MS_k^4)^* \\ &\quad + 224(MS_k^5)^* - 49(MS_k^6)^*).\end{aligned}$$

On the other hand, since $(MS_k)^* = MS_k$ and M is Hermitian positive definite ($M^* = M$), we obtain

$$\begin{aligned}(MS_k^2)^* = S_k^*(MS_k)^* = S_k^*(MS_k) = S_k^*M^*S_k \\ = (MS_k)^*S_k = (MS_k)S_k = MS_k^2.\end{aligned}$$

So,

$$\begin{aligned}(MAX_{k+1})^* &= \frac{1}{4}(44(MS_k) - 185(MS_k^2) + 380(MS_k^3) - 410(MS_k^4) \\ &\quad + 224(MS_k^5) - 49(MS_k^6)) \\ &= MS_{k+1} = MAX_{k+1}.\end{aligned}$$

Thus, the first equality in (18) holds for $k+1$, and the second equality can be proved in a similar way. For the third equality in (18), using the assumption that $X_kAA^\dagger_{MN} = X_k$ or equivalently $S_kAA^\dagger_{MN} = S_k$ and the iterative method (16), we could write down

$$\begin{aligned}X_{k+1}AA^\dagger_{MN} &= \frac{1}{4}X_k(44 - 185S_k + 380S_k^2 - 410S_k^3 + 224S_k^4 - 49S_k^5)AA^\dagger_{MN} \\ &= \frac{1}{4}(44X_kAA^\dagger_{MN} - 185X_kS_kAA^\dagger_{MN} + 380X_kS_k^2AA^\dagger_{MN} \\ &\quad - 410X_kS_k^3AA^\dagger_{MN} + 224X_kS_k^4AA^\dagger_{MN} - 49X_kS_k^5AA^\dagger_{MN}) \\ &= \frac{1}{4}(44(X_kAA^\dagger_{MN}) - 185X_k(S_kAA^\dagger_{MN}) + 380X_kS_k(S_kAA^\dagger_{MN}) \\ &\quad - 410X_kS_k^2(S_kAA^\dagger_{MN}) + 224X_kS_k^3(S_kAA^\dagger_{MN}) \\ &\quad - 49X_kS_k^4(S_kAA^\dagger_{MN})) \\ &= \frac{1}{4}(44X_k - 185X_kS_k + 380X_kS_k^2 - 410X_kS_k^3 + 224X_kS_k^4 \\ &\quad - 49X_kS_k^5) \\ &= \frac{1}{4}X_k(44 - 185S_k + 380S_k^2 - 410S_k^3 + 224S_k^4 - 49S_k^5) \\ &= X_{k+1}.\end{aligned}$$

Consequently, the third equality in (18) holds for $k+1$. The fourth equality can similarly be proved, and the desired result follows. □

Lemma 2 *Let A be the matrix* (6) *and take X_0 as in* (17)*. Considering the conditions of Lemma 1, for each approximate inverse produced by* (16)*, it holds that*

$$(N^{-1}V)^{-1}X_k(U^*M)^{-1} = \begin{bmatrix} D_k & 0 \\ 0 & 0 \end{bmatrix},$$

in which D_k is a diagonal matrix.

Proof Using (6) and the definition of $A^{\#}$, we have

$$X_0 = \beta A^{\#} = (N^{-1}V)\begin{bmatrix} D_0 & 0 \\ 0 & 0 \end{bmatrix}(U^*M),$$

in which

$$D_0 = \beta D$$

is a diagonal matrix. Therefore,

$$(N^{-1}V)^{-1}X_0(U^*M)^{-1} = \begin{bmatrix} D_0 & 0 \\ 0 & 0 \end{bmatrix}.$$

Now, we continue by using the principle of mathematical induction and Lemma 1. Assume that the conclusion holds for some $k > 0$, meaning that

$$(N^{-1}V)^{-1}X_k(U^*M)^{-1} = \begin{bmatrix} D_k & 0 \\ 0 & 0 \end{bmatrix}.$$

On the other hand, since

$$\begin{aligned} AX_k &= U\begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix}V^*N^{-1}V\begin{bmatrix} D_k & 0 \\ 0 & 0 \end{bmatrix}U^*M \\ &= U\begin{bmatrix} DD_k & 0 \\ 0 & 0 \end{bmatrix}U^*M, \end{aligned}$$

it results

$$\begin{aligned} (AX_k)^2 &= U\begin{bmatrix} DD_k & 0 \\ 0 & 0 \end{bmatrix}U^*MU\begin{bmatrix} DD_k & 0 \\ 0 & 0 \end{bmatrix}U^*M \\ &= U\begin{bmatrix} DD_k & 0 \\ 0 & 0 \end{bmatrix}^2U^*M, \end{aligned}$$

and other powers of AX_k could be acquired in a similar way. Thus, we have

$$\begin{aligned} &(N^{-1}V)^{-1}X_kAX_k(U^*M)^{-1} \\ &\quad= (N^{-1}V)^{-1}X_kU\begin{bmatrix} DD_k & 0 \\ 0 & 0 \end{bmatrix}U^*M(U^*M)^{-1} \\ &\quad= (N^{-1}V)^{-1}(N^{-1}V)\begin{bmatrix} D_k & 0 \\ 0 & 0 \end{bmatrix}U^*MU\begin{bmatrix} DD_k & 0 \\ 0 & 0 \end{bmatrix}U^*M(U^*M)^{-1} \\ &\quad= \begin{bmatrix} D_k & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} DD_k & 0 \\ 0 & 0 \end{bmatrix} \\ &\quad= \begin{bmatrix} D_k(DD_k) & 0 \\ 0 & 0 \end{bmatrix}, \end{aligned}$$

and similarly

$$(N^{-1}V)^{-1}X_k(AX_k)^2(U^*M)^{-1} = \begin{bmatrix} D_k(DD_k)^2 & 0 \\ 0 & 0 \end{bmatrix}.$$

Therefore,

$$\begin{aligned}
&(N^{-1}V)^{-1}X_{k+1}(U^*M)^{-1}\\
&= \frac{1}{4}(N^{-1}V)^{-1}X_k(44I - 185AX_k + 380(AX_k)^2 - 410(AX_k)^3\\
&\quad + 224(AX_k)^4 - 49(AX_k)^5)(U^*M)^{-1}\\
&= \frac{1}{4}(44(N^{-1}V)^{-1}X_k(U^*M)^{-1} - 185(N^{-1}V)^{-1}X_kAX_k(U^*M)^{-1}\\
&\quad + 380(N^{-1}V)^{-1}X_k(AX_k)^2(U^*M)^{-1}\\
&\quad - 410(N^{-1}V)^{-1}X_k(AX_k)^3(U^*M)^{-1}\\
&\quad + 224(N^{-1}V)^{-1}X_k(AX_k)^4(U^*M)^{-1}\\
&\quad - 49(N^{-1}V)^{-1}X_k(AX_k)^5(U^*M)^{-1})\\
&= \begin{bmatrix} D_{k+1} & 0\\ 0 & 0\end{bmatrix},
\end{aligned}$$

in which D_{k+1} is the following diagonal matrix:

$$\begin{aligned}
D_{k+1} = \varphi(D_k) = \frac{1}{4}D_k(44I - 185DD_k + 380(DD_k)^2 - 410(DD_k)^3\\
+ 224(DD_k)^4 - 49(DD_k)^5).
\end{aligned} \tag{19}$$

This completes the proof. □

In the next theorem, we prove that the method (16) is a fourth-order convergent.

Theorem 2 *Assume that A is an $m \times n$ matrix whose weighted singular value decomposition is given by* (7)*. Let furthermore the initial matrix be available by* (17) *in which*

$$0 < \beta \leq \frac{37}{25\sigma_1^2}. \tag{20}$$

Then, the sequence of iterates generated by (16) *converges to the WMP inverse $A^{\dagger}_{MN}$ with fourth-order.*

Proof By considering (7) and in order to establish this result, we must show that

$$\lim_{k\to\infty}(N^{-1}V)^{-1}X_k(U^*M)^{-1} = \begin{bmatrix} D^{-1} & 0\\ 0 & 0\end{bmatrix}. \tag{21}$$

Take

$$D_k = diag(d_1^{(k)}, \ldots, d_r^{(k)}),$$

in which

$$d_i^{(0)} = \beta\sigma_i, \quad i = 1, \ldots, r.$$

It results from Lemma 2 and relation (19) that

$$\begin{aligned}
d_i^{(k+1)} = \varphi(d_i^{(k)}) = \frac{1}{4}d_i^{(k)}(44I - 185\sigma_i d_i^{(k)} + 380(\sigma_i d_i^{(k)})^2 - 410(\sigma_i d_i^{(k)})^3\\
+ 224(\sigma_i d_i^{(k)})^4 - 49(\sigma_i d_i^{(k)})^5), \quad i = 1, \ldots, r.
\end{aligned} \tag{22}$$

Take $t_i^{(k)} = \sigma_i d_i^{(k)}$. Then, $t_i^{(k+1)} = g(t_i^{(k)})$ and, according to (20), we have $t_i^{(0)} = \beta\sigma_i^2 \in (0, \frac{37}{25}]$. Based on Theorem 1 and Corollary 1, we conclude that the convergence $\lim_{k\to\infty} t_i^{(k)} = 1$ holds with fourth-order. Hence, $d_i^{(k)}$ is also convergent to σ_i^{-1} with fourth-order. This shows the fourth-order of convergence for the presented method (16). □

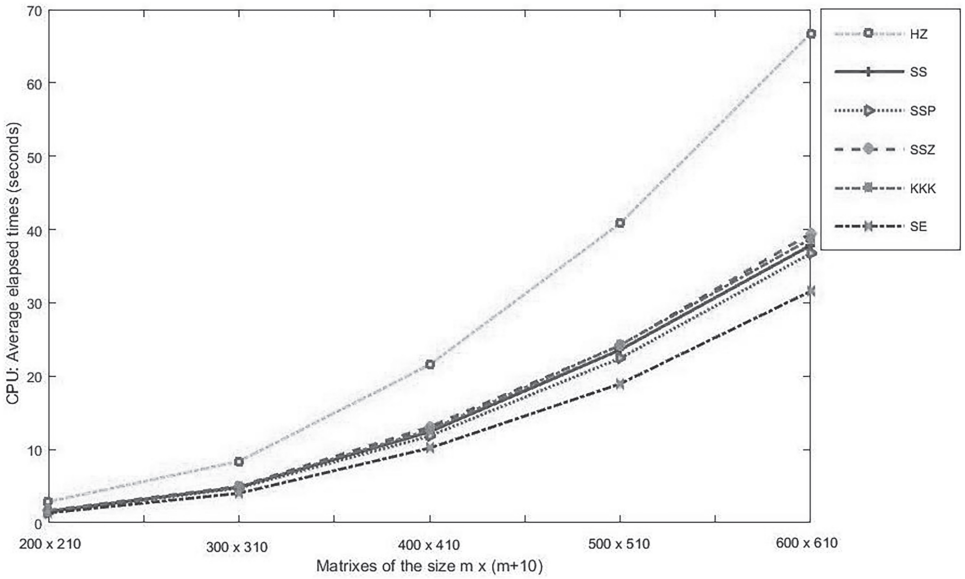


Fig. 2 The average elapsed time to compute the weighted Moore–Penrose inverse by different methods for Test 1

3 Numerical experiments

In this section, we will make some numerical comparisons of our method (16) (SE) with the methods (9) (HZ), (11) (SS), (12) (SSP), (13) (SSZ), and (14) (KKK). The calculations of this part have been done by MATLAB R2015b program on PC with specifications AMD, E2-1800, APU, 1.70 GHz.

We first consider two individual type of tests and then the methods are applied to solve PDEs. Test 1 is dedicated to comparing the algorithms to compute the weighted Moore–Penrose inverse, while Test 2 offers some comparison to compute the Moore–Penrose inverse. In both tests, for each $m = 200, 300, 400, 500, 600$, we have performed 10 tests and compared the average number of iterations (Iter), the average total number of the matrix multiplications (TMM), and the average elapsed execution time (CPU) in seconds. We have used the abbreviation (DIM) for dimension of each matrix in tables. Moreover, the initial matrix X_0 is used according to (17), with β from (20). The largest eigenvalue of $A^{\#}A$ is σ_1^2, so we obtain $\sigma_1^2 \leq \|A^{\#}A\|_\infty \leq \|A^{\#}\|_\infty \|A\|_\infty$. Thus, the choice

$$\beta = \frac{1}{\|A^{\#}\|_\infty \|A\|_\infty},$$

satisfies both (10) and (17). The stop criterion is

$$\frac{\|X_{k+1} - X_k\|_\infty}{1 + \|X_k\|_\infty} < 10^{-7},$$

and we set 100 as a limit for the maximum number of allowed iterations in our written codes.

Test 1 *In this test, the weighted Moore–Penrose inverse of randomly generated dense matrix A of the size* $m \times n, n = m + 10$ *is computed as follows:*

$$A = 10rand(m, n) - 10rand(m, n)$$

Randomly constructed different Hermitian positive definite matrices M and N are as follows:

$$M = qr(10rand(m, m) - 10rand(m, m)), \quad M = M^*M,$$
$$N = qr(10rand(n, n) - 10rand(n, n)), \quad N = N^*N.$$

The results of this test are demonstrated in Table 1 *and Figs.* 2, 3, *and* 4.

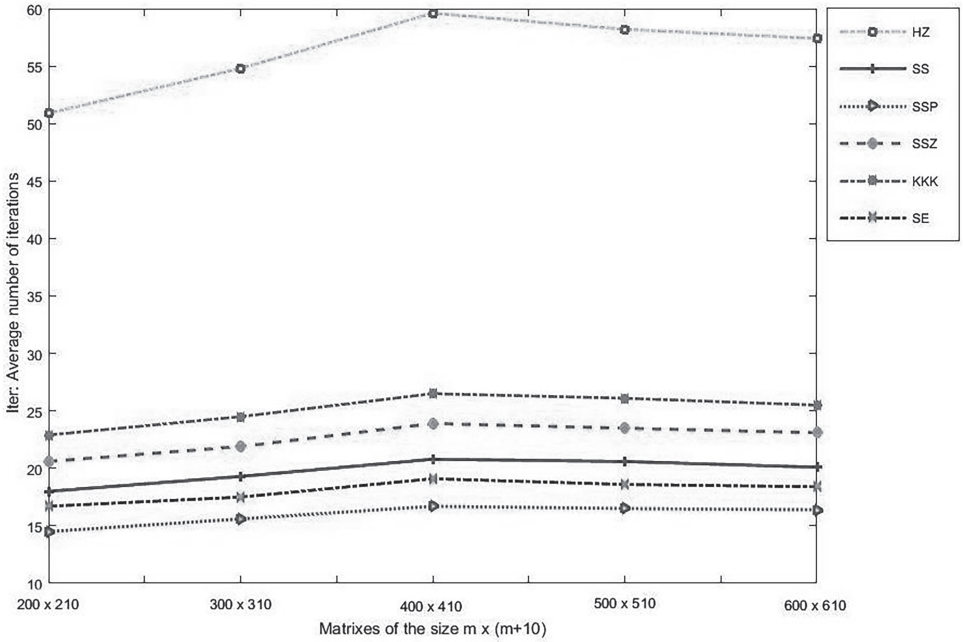


Fig. 3 The average number of iterations to compute the weighted Moore–Penrose inverse by different methods for Test 1

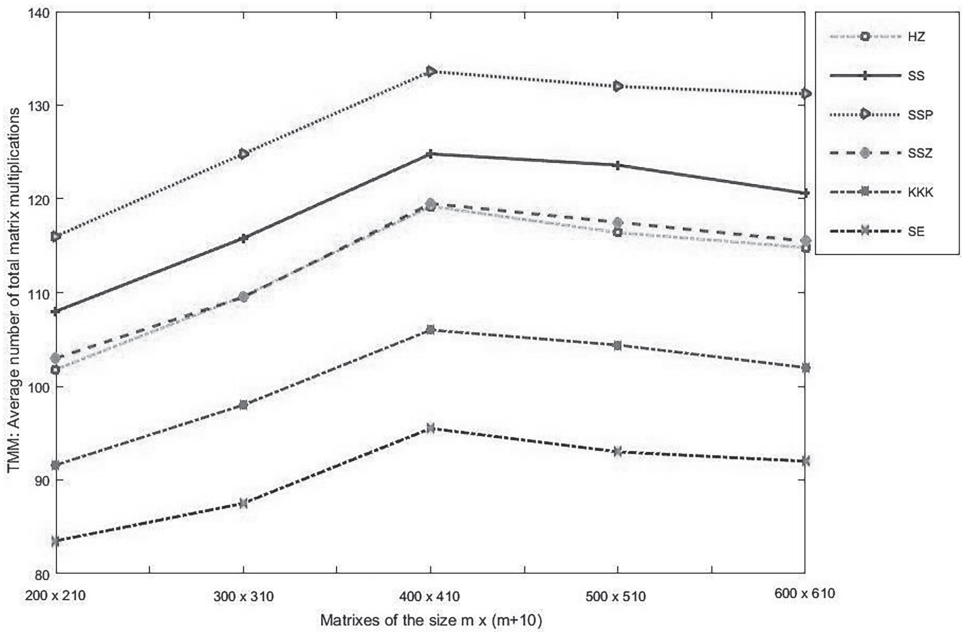


Fig. 4 The average total number of matrix multiplications to compute the weighted Moore–Penrose inverse by different methods for Test 1

Test 2 *In this test, the Moore–Penrose inverse of randomly generated dense matrix A of the size* $m \times n$, $n = m+10$ *is computed as follows:*

$$A = 10rand(m, n) - 10rand(m, n)$$

The results of this test are demonstrated in Table 2 *and Figs.* 5, 6, *and* 7.

Table 1 The results of comparisons for Test 1

Methods:	HZ	SS	SSP	SSZ	kkk	SE
DIM: 200 × 210						
CPU	$2.8722e+00$	$1.5478e+00$	$1.4499e+00$	$1.6749e+00$	$1.6246e+00$	$1.3448e+00$
Iter	$5.0900e+01$	$1.8000e+01$	$1.4500e+01$	$2.0600e+01$	$2.2900e+01$	$1.6700e+01$
TMM	$1.0180e+02$	$1.0800e+02$	$1.1600e+02$	$1.0300e+02$	$9.1600e+01$	$8.3500e+01$
DIM: 300 × 310						
CPU	$8.3724e+00$	$4.8464e+00$	$4.7437e+00$	$4.9537e+00$	$4.8873e+00$	$4.0220e+00$
Iter	$5.4800e+01$	$1.9300e+01$	$1.5600e+01$	$2.1900e+01$	$2.4500e+01$	$1.7500e+01$
TMM	$1.0960e+02$	$1.1580e+02$	$1.2480e+02$	$1.0950e+02$	$9.8000e+01$	$8.7500e+01$
DIM: 400 × 410						
CPU	$2.1539e+01$	$1.2405e+01$	$1.1821e+01$	$1.3070e+01$	$1.2634e+01$	$1.0182e+01$
Iter	$5.9600e+01$	$2.0800e+01$	$1.6700e+01$	$2.3900e+01$	$2.6500e+01$	$1.9100e+01$
TMM	$1.1920e+02$	$1.2480e+02$	$1.3360e+02$	$1.1950e+02$	$1.0600e+02$	$9.5500e+01$
DIM: 500 × 510						
CPU	$4.0836e+01$	$2.3580e+01$	$2.2435e+01$	$2.4178e+01$	$2.4229e+01$	$1.8943e+01$
Iter	$5.8200e+01$	$2.0600e+01$	$1.6500e+01$	$2.3500e+01$	$2.6100e+01$	$1.8600e+01$
TMM	$1.1640e+02$	$1.2360e+02$	$1.3200e+02$	$1.1750e+02$	$1.0440e+02$	$9.3000e+01$
DIM: 600 × 610						
CPU	$6.6702e+01$	$3.7843e+01$	$3.6720e+01$	$3.9455e+01$	$3.8694e+01$	$3.1583e+01$
Iter	$5.7400e+01$	$2.0100e+01$	$1.6400e+01$	$2.3100e+01$	$2.5500e+01$	$1.8400e+01$
TMM	$1.1480e+02$	$1.2060e+02$	$1.3120e+02$	$1.1550e+02$	$1.0200e+02$	$9.2000e+01$

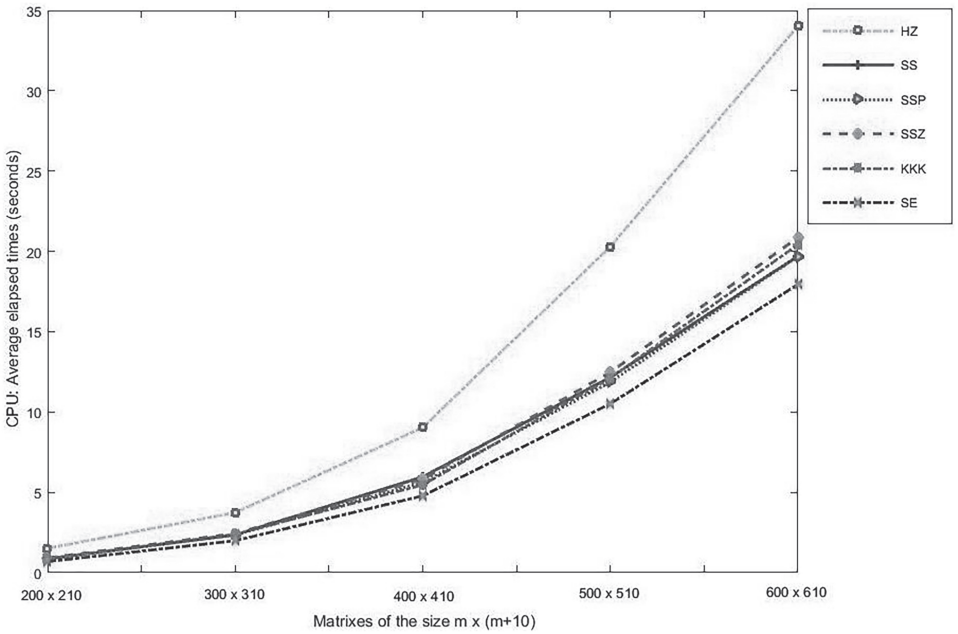


Fig. 5 The average elapsed time to compute the Moore–Penrose inverse by different methods for Test 2.

Test 3 [20] *Consider the following elliptic partial differential equation:*

$$\frac{\partial^2\phi}{\partial x^2}+\frac{\partial^2\phi}{\partial y^2}=32\phi(x,y)$$

where ϕ is a function of x and y. It is satisfied at every point inside the square formed by $x=\pm1$, $y=\pm1$ and subject to the following the boundary conditions:

$$\begin{aligned}&\phi(x,y)=0,\quad on\quad y=1,\quad -1\le x\le 1,\\&\phi(x,y)=1,\quad on\quad y=-1,\quad -1\le x\le 1,\\&\frac{\partial\phi}{\partial x}=-\frac{1}{2}\phi(x,y),\quad on\quad x=1,\quad -1\le y\le 1,\end{aligned}$$

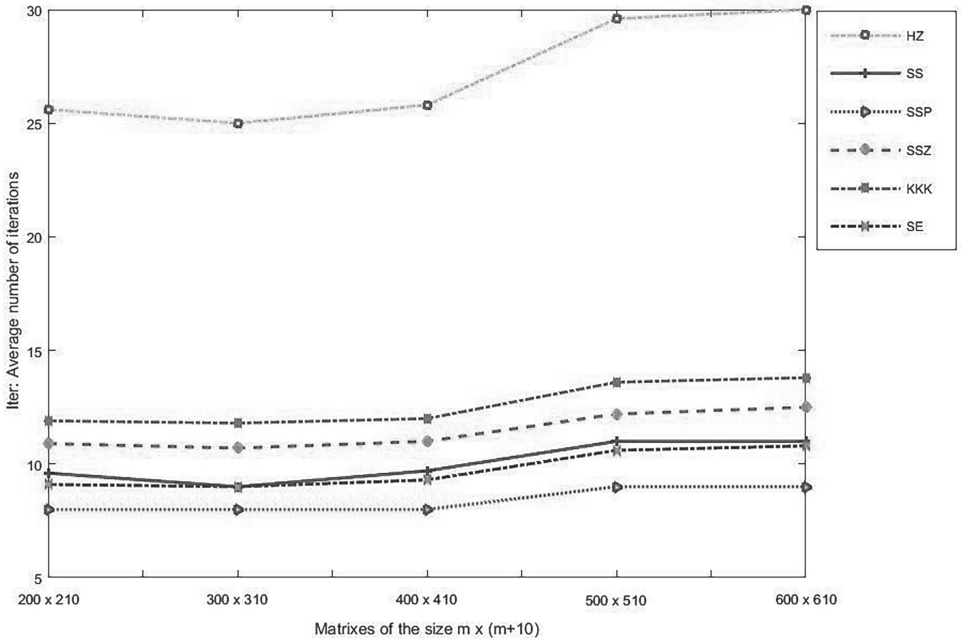


Fig. 6 The average number of iterations to compute the Moore–Penrose inverse by different methods for Test 2.

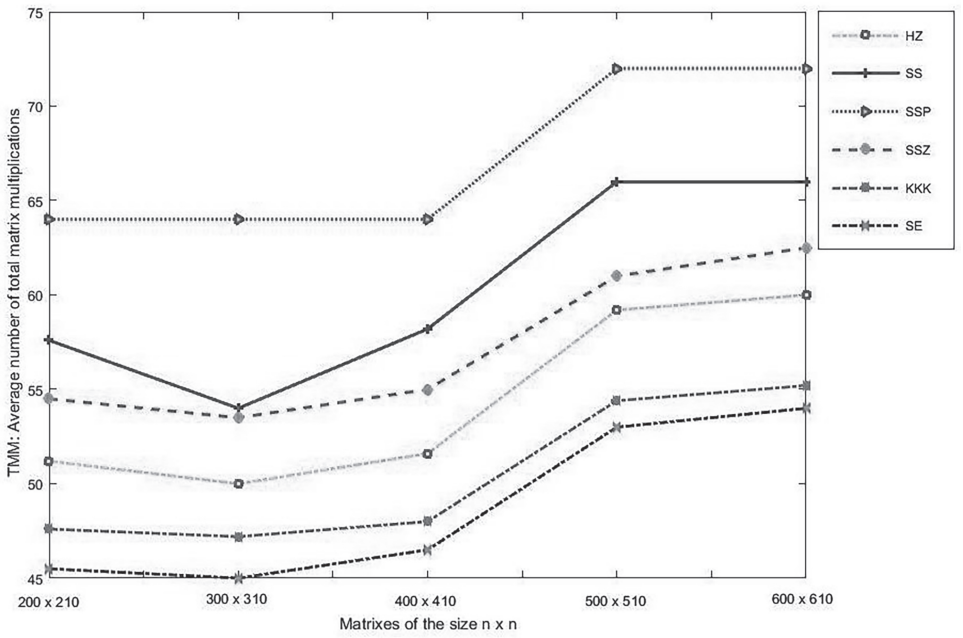


Fig. 7 The average total number of matrix multiplications to compute the Moore–Penrose inverse by different methods for Test 2.

$$\frac{\partial \phi}{\partial x} = \frac{1}{2}\phi(x, y), \quad on \quad x = -1, \quad -1 \leq y \leq 1.$$

Using the central difference formula on the above differential equation, one can obtain

$$-4\phi_{i,j} + \phi_{i+1,j} + \phi_{i-1,j} + \phi_{i,j+1} + \phi_{i,j-1} = 32h^2\phi_{i,j},$$

in which $\phi_{i,j} = \phi(x_i, y_j)$. *Considering square mesh sizes* $h = \frac{1}{16}$ *and* $h = \frac{1}{20}$, *we solve the system* $Az = b$ *to compute the approximate solution* $\phi(x_i, y_j)$. *To this end, we start the algorithm with the initial matrix* $X_0 =$

Table 2 The results of comparisons for Test 2

Methods:	HZ	SS	SSP	SSZ	kkk	SE
DIM: 200×210						
CPU	$1.5132e+00$	$8.5343e-01$	$8.7922e-01$	$9.1753e-01$	$8.7067e-01$	$6.9167e-01$
Iter	$2.5600e+01$	$9.6000e+00$	$8.0000e+00$	$1.0900e+01$	$1.1900e+01$	$9.1000e+00$
TMM	$5.1200e+01$	$5.7600e+01$	$6.4000e+01$	$5.4500e+01$	$4.7600e+01$	$4.5500e+01$
DIM: 300×310						
CPU	$3.7386e+00$	$2.3535e+00$	$2.3375e+00$	$2.4319e+00$	$2.3881e+00$	$1.9863e+00$
Iter	$2.5000e+01$	$9.0000e+00$	$8.0000e+00$	$1.0700e+01$	$1.1800e+01$	$9.0000e+00$
TMM	$5.0000e+01$	$5.4000e+01$	$6.4000e+01$	$5.3500e+01$	$4.7200e+01$	$4.5000e+01$
DIM: 400×410						
CPU	$9.0323e+00$	$5.9616e+00$	$5.6179e+00$	$5.8030e+00$	$5.4639e+00$	$4.7806e+00$
Iter	$2.5800e+01$	$9.7000e+00$	$8.0000e+00$	$1.1000e+01$	$1.2000e+01$	$9.3000e+00$
TMM	$5.1600e+01$	$5.8200e+01$	$6.4000e+01$	$5.5000e+01$	$4.8000e+01$	$4.6500e+01$
DIM: 500×510						
CPU	$2.0296e+01$	$1.2162e+01$	$1.1865e+01$	$1.2475e+01$	$1.2108e+01$	$1.0513e+01$
Iter	$2.9600e+01$	$1.1000e+01$	$9.0000e+00$	$1.2200e+01$	$1.3600e+01$	$1.0600e+01$
TMM	$5.9200e+01$	$6.6000e+01$	$7.2000e+01$	$6.1000e+01$	$5.4400e+01$	$5.3000e+01$
DIM: 600×610						
CPU	$3.4027e+01$	$1.9693e+01$	$1.9693e+01$	$2.0871e+01$	$2.0424e+01$	$1.7947e+01$
Iter	$3.0000e+01$	$1.1000e+01$	$9.0000e+00$	$1.2500e+01$	$1.3800e+01$	$1.0800e+01$
TMM	$6.0000e+01$	$6.6000e+01$	$7.2000e+01$	$6.2500e+01$	$5.5200e+01$	$5.4000e+01$

Table 3 The results of comparisons for Test 3

Methods:	HZ	SS	SSP	SSZ	kkk	SE
DIM: 527×527						
CPU	$3.0529e+00$	$2.9939e+00$	$3.0499e+00$	$2.6737e+00$	$2.8645e+00$	$2.6173e+00$
Iter	10	4	3	4	5	4
TMM	20	24	24	20	20	20
DIM: 819×819						
CPU	$9.3846e+00$	$1.1097e+01$	$1.1851e+01$	$1.0077e+01$	$1.0226e+01$	$9.2814e+00$
Iter	10	4	3	4	5	4
TMM	20	24	24	20	20	20

$\frac{1}{\|A\|_1\|A\|_\infty}A^*$ *and then* $z_k = X_k b$ *is computed in each iteration. The stop criterion*

$$\frac{\|r_k\|_2}{\|r_0\|_2} < 10^{-12}$$

is used, where $r_k = b - Az_k$ *is the kth residual. The results of this test are demonstrated in Table* 3.

According to the results of Tables 1, 2, and 3, we notice that method (16) (SE) performed, in terms of the average total number of matrix multiplications and the average elapsed execution time, better than other methods.

4 Conclusion

In this paper, we introduce a new iterative method to compute the weighted Moore–Penrose inverse. It is proved that this method converges with fourth-order. Although our method is not a high-order scheme, a wide set of random numerical experiments showed that the total number of matrix multiplications and elapsed execution time are considerably less than those of higher-order methods. So, our method could be considered as an efficient method.

Author contributions Both authors contributed equally.

Funding Not applicable.

Availability of data and materials Not applicable.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

1. A., Ben-Israel, and T.N.E., Greville, (2003), Generalized Inverses, Theory and Applications, second ed. Springer, NY.
2. S., Chandrasekaran, M., Gu, A.H., Sayed, (1998), A stable and efficient algorithm for the indefinite linear least-squares problem, SIAM J. Matrix Anal. Appl., **20**, 354–362.
3. S.F., Wang, B., Zheng, Z.P., Xiong, et al., (2009), The condition numbers for weighted Moore-Penrose inverse and weighted linear least squares problem, Appl. Math. Comput., **215**, 197–205.
4. M.E., Gulliksson, P.A., Wedin, and Y., Wei, (2000), Perturbation identities for regularized Tikhonov inverse and weighted pseudo inverse, BIT Num.math. **40**, 513–523.
5. W., Sun, and Y., Wei, (1998), Inverse order rule for weighted generalized inverse, SIAM J. Matrix Anal. Appl. **19**, 772–775.
6. R., Penrose, (1955), A generalized inverses for matrices, Proc. Cambridge Philo. Soc. **51**, 406–413.
7. A., Bojanczyk, N.J., Higham, and H., Patel, (2003), Solving the indefinite least squares problem by hyperbolic QR factorization, SIAM J. Matrix Anal. Appl. **24**, 914–931.
8. M., Gulliksson, X.O., Jin, and Y., Wei, (2002), Perturbation bounds for constrained and weighted least squares problems, Linear Algebra Appl. **349**, 221–232.
9. A., Kilicman, Z., Al Zhour, (2005), The general common exact solutions of coupled linear matrix and matrix differential equations, J. Anal. Comput. **1**, 15–30.
10. A., Kilicman, and Z., Al Zhour, (2007), Iterative solutions of coupled matrix convolution equations, Soochow J. Math **33**, 167–180.
11. Y., Wei, (1999), Perturbation bound of singular linear system, Appl. Math. Comput. **105**, 211–220.
12. F., Huang, and X., Zhang, (2006), An improved Newton iteration for the weighted Moore-Penrose inverse, Appl. Math. Comput. **174**, 1460–1486.
13. M.D., Petkovic, P.S., Stanimirovic, and M.B., Tasic, (2008), Effective partitioning method for computing weighted Moore–Penrose inverse, Comput. Math. Appl. **55**, 1720–1734.
14. S.K., Sen, H., Agarwal, and S., Sen, (2006), Chemical equation balancing: an integer programming approach, Math. Comput. Model. **44**, 678–691.
15. F., Soleymani, (2013), On a fast iterative method for approximate inverse of matrices, Commun. Korean Math. Soc. **28**, 407–418.
16. M.Z., Ullah, F., Soleymani, and A.S., Al-Fhaid, (2014), An efficient matrix iteration for computing weighted Moore–Penrose inverse, Appl. Math. Comput. **226**, 441–454.
17. H., Esmaeili, R., Erfanifar, and M., Rashidi, (2016), A new fourth-order method to compute the weighted Moore–Penrose inverse, Bull. Inter. Math. Virtual Inst., **6**, 269-280.
18. G.R., Wang, (1985), A new proof of Greville's method for computing the weighted MP inverse, J. Shanghai Normal University (Nat. Sci.), **3**, 32–38.
19. L., Weiguo, L., Juan, and Q., Tiantian, (2013), A family of iterative methods for computing Moore–Penrose inverse of a matrix, Linear Algebra Appl. **438**, 47–56.
20. M., Kaur, M., Kansal, and S., Kumar, (2020), An efficient hyperpower iterative method for computing weighted Moore–Penrose inverse, AIMS Math., **5**, 1680–1692.
21. C.F., Van Loan, (1976), Generalizing the singular value decomposition, SIAM J. Numer. Anal. **13**, 76–83.
22. G., Schulz, (1933), Iterative Berechnung der reziproken matrix, Z. Angew. Math. Mech. **13**, 57–59.
23. F., Soleimani, F., Soleymani, A., Cordero, and J.R., Torregrosa, (2014), On the extension of Householder's method for weighted Moore–Penrose inverse, Appl. Math. Comput. **231**, 407–413.
24. A.S., Householder, (1964), The Theory of Matrices in Numerical Analysis, Dover Publications, New York.
25. A.R., Soheili, F., Soleymani, and M.D., Petkovic, (2013), On the computation of weighted Moore–Penrose inverse using a high-order matrix method, Comput. Math. Appl., **66**, 2344–2351.
26. F., Soleymani, P.S., Stanimirovic, and M., Zaka Ullah, (2013), An accelerated iterative method for computing weighted Moore–Penrose inverse, Appl. Math. Comput., **222**, 365–371.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.