Filomat 39:25 (2025), 8915–8928
https://doi.org/10.2298/FIL2525915N

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

A generalized second order iterative algorithm for computing the Moore–Penrose inverse

Hemant Kumar Nashine[a,b], Munish Kansal[c,*], Manpreet Kaur[d], V. Kanwar[e], Reena Jain[a]

[a]*Mathematics Division, School of Advanced Sciences and Languages, VIT Bhopal University, Kothrikalan, Sehore, Madhya Pradesh - 466114, India*
[b]*Department of Mathematics and Applied Mathematics, University of Johannesburg, Kingsway Campus, Auckland Park 2006, South Africa*
[c]*Department of Mathematics, Thapar Institute of Engineering and Technology, Patiala-147004, India*
[d]*Department of Mathematics, Lovely Professional University, Phagwara-144411, India*
[e]*University Institute of Engineering and Technology, Panjab University, Chandigarh-160014, India*

Abstract. In this paper, we propose a generalized second order iterative algorithm for computing the Moore–Penrose inverse. The method arises from the second Penrose equation $XAX = X$ and is further generalized by two real parameters. A detailed theoretical analysis is conducted to show that, under certain conditions, the new approach possesses linear, quadratic, and cubic convergence. As a result, various linear and quadratic convergence schemes can be extracted. Efficiency analysis of the method is considered to state its relation with respect to the condition number and number of iterations. We provide adequate examples to validate the new iterative scheme including matrices produced from real-life problems. Moreover, the applicability of method is also examined on one-dimensional heat problems. The convergence and error analysis, as well as the average CPU time analysis, are also given.

1. Introduction and preliminaries

Let $\mathbb{C}^{m\times n}$ and $\mathbb{C}_r^{m\times n}$ denote the set of all complex $m \times n$ matrices and all complex $m \times n$ matrices with rank r, respectively. Let A^t, A^*, $R(A)$, $N(A)$ and $rank(A)$ represent the transpose, the conjugate transpose, the range space, the null space and the rank of the matrix $A \in \mathbb{C}^{m\times n}$, respectively.

For $A \in \mathbb{C}^{m\times n}$, the Moore–Penrose inverse is denoted by $A^\dagger$ and is defined as

$$AA^\dagger = P_{R(A)},\ A^\dagger A = P_{R(A^\dagger)},$$

where $P_{R(A)}$ denotes the orthogonal projection onto range space $R(A)$. The unique matrix $A^\dagger$ satisfies the following four equations

$$(i)\ AXA = A, \quad (ii)\ XAX = X, \quad (iii)\ (AX)^* = AX, \quad (iv)\ (XA)^* = XA. \tag{1}$$

2020 *Mathematics Subject Classification*. Primary 65F30; Secondary 15A09, 47J25.
Keywords. Generalized inverse, Moore–Penrose inverse, convergence analysis, efficiency index, order of convergence
Received: 17 January 2023; Revised: 16 September 2024; Accepted: 30 March 2025
Communicated by Predrag Stanimirović
* Corresponding author: Munish Kansal
Email addresses: hknashine@gmail.com, hemantkumar.nashine@vitbhopal.ac.in (Hemant Kumar Nashine), munish.kansal@thapar.edu (Munish Kansal), mkaur.math@gmail.com (Manpreet Kaur), vmithil@yahoo.co.in (V. Kanwar), reenakhatod@gmail.com, reena.jain@vitbhopal.ac.in (Reena Jain)
ORCID iDs: https://orcid.org/0000-0002-0250-9172 (Hemant Kumar Nashine), https://orcid.org/0000-0002-7502-6091 (Munish Kansal), https://orcid.org/0000-0002-5829-4043 (Manpreet Kaur), https://orcid.org/0000-0002-7923-1324 (V. Kanwar), https://orcid.org/0000-0001-5018-4475 (Reena Jain)

One of the most commonly used direct methods to evaluate $A^\dagger$ is the singular value decomposition (SVD) method. For $A \in \mathbb{C}^{m\times n}$, the SVD method is a factorization of the form $A = U\Sigma V^*$, where U is an $m\times m$ complex unitary matrix, Σ is an $m \times n$ rectangular diagonal matrix with non-negative real numbers on the diagonal and V is an $n\times n$ complex unitary matrix. Then, $A^\dagger$ can be written as $A^\dagger = V\Sigma^\dagger U^*$, where $\Sigma^\dagger$ is the generalized inverse of Σ obtained by replacing every non-zero diagonal entry in Σ with its reciprocal and then transposing the resulting matrix. The most frequently used iterative method for approximating A^{-1} is the famous Newton's method originated in Schulz [1] as

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

As usual, I denotes the identity matrix of an appropriate order. The eigenvalues of $A \in \mathbb{C}_r^{n\times n}$ are given by

$$\lambda_1(A) \geq \cdots \geq \lambda_r(A) > \lambda_{r+1}(A) = \cdots = \lambda_n(A) = 0. \tag{3}$$

It is further established that the eigenvalues of $I - AX_0$ must have a magnitude less than 1 to ensure its second order convergence. Moreover, it satisfies error inequality, $\|R_{k+1}\| \leq \|A\|\|R_k\|^2$, where residuals are defined as $R_k = I - AX_k$ [2]. Ben-Israel and Greville [3] used (2) and the initial approximation $X_0 = \delta A^*$, where δ satisfies

$$0 < \delta < \frac{2}{\lambda_1(AA^*)}, \tag{4}$$

to compute $A^\dagger$. The success of iterative algorithms in converging to $A^\dagger$ depends significantly on the choice of the initial estimate X_0. Several approaches for selecting this initial approximation have been suggested in literature [4–6]. Petković and Stanimirović [7] developed an iterative method for computing the Moore–Penrose inverse of an arbitrary matrix $A \in \mathbb{C}^{m\times n}$, given by

$$X_{k+1} = (1+\beta)X_k - \beta X_k A X_k, \quad k = 0, 1, \ldots, \tag{5}$$

with initial approximation $X_0 = \beta A^*$, where $\beta \in (0, 1]$ be an appropriate real number. They showed that the method has a linear convergence for $\beta < 1$, while the well-known Schulz method is retrieved for $\beta = 1$. The method (5) is based on the second and fourth Penrose equations, i.e., $XAX = X$ and $(XA)^* = XA$. Srivastava and Gupta [8] extended the work of Petković and Stanimirović [7] and proposed the following third order iterative algorithm

$$\begin{cases} X_0 = \beta A^*, \\ X_{k+1} = X_k(I + \alpha(2I - 3AX_k + (AX_k)^2), \quad k = 0, 1, \ldots, \end{cases} \tag{6}$$

to compute $A^\dagger$ by using appropriate values of $\alpha, \beta \in \mathbb{R}$.

On the other hand, hyperpower matrix iterations of order p represent a general series form of matrices:

$$X_{k+1} = X_k(1 + V_k + {V_k}^2 + \cdots + {V_k}^{p-1}), \quad V_k = 1 - AX_k, \quad k = 0, 1, \ldots, \tag{7}$$

and several studies [9–14] have focused on developing more efficient versions of this method. For instance, Pan et al. [10] presented an 18th-order hyperpower method that requires only seven matrix multiplication, as outlined below:

$$\begin{cases} R_k = I - AX_k, \\ M_k = (1 + c_1R_k^2 + R_k^4)(I + c_2R_k^2 + R_k^4), \\ T_k = M_k + c_3R_k^2, \quad S_k = M_k + d_1R_k^2 + d_2R_k^4, \\ X_{k+1} = X_k((I + R_k)((T_kS_k) + \mu R_k^2 + \psi R_k^4)), \end{cases} \tag{8}$$

where $c_1 = \frac{1}{4}\left(\sqrt{27 - 2\sqrt{93}} + 1\right)$, $c_2 = \frac{1}{4}\left(1 - \sqrt{27 - 2\sqrt{93}}\right)$, $c_3 = \frac{1}{496}(5\sqrt{93} - 93)$, $d_1 = \frac{1}{496}(-93 - 5\sqrt{93})$, $d_2 = -\frac{\sqrt{93}}{4}$, $\mu = \frac{3}{8}$, $\psi = \frac{321}{1984}$. Soleimani et al. [15] discussed the improvements to the method in (7) for

$p = 30$ and highlights the applicability of such schemes to real-world problems. Indeed, these higher-order algorithms produce more accurate solutions with fewer iterations. However, designing the algorithm with same order that yields improved results is a difficult challenge. Motivated by the development of (6), based on the second Penrose equation influence us to work in this direction. Thus, in this paper, we develop and analyze a second order iterative family for computing Moore–Penrose inverse. The main aim of the paper is to discover a better alternative of the quadratic convergent iterative scheme other than the Schulz method.

The manuscript is organized as follows. Section 2 deals with the derivation and explanation of the proposed iterative scheme for computing generalized inverses. Following that, the convergence behavior of the method toward the Moore–Penrose inverse is discussed in detail. Section 3 addresses the computational complexity of the method by using the concept of a computational efficiency index and the necessary iterations. Section 4 shows the efficacy of the proposed work by applying the method to different types of matrices obtained from several real-life models, such as statically determinate truss problems and one-dimensional heat problems. Finally, section 5 presents the concluding points.

2. A new second order iterative method

Assume that $A \in \mathbb{C}^{m\times n}$ and $X = A^{\dagger} \in \mathbb{C}^{n\times m}$. We begin with only Penrose equation (*ii*) and for arbitrary $\alpha, \beta \in \mathbb{R}$, we obtain

$$\begin{aligned} X &= X - \alpha(XAX - X) + \beta\left(2X - 3XAX + X(AX)^2\right) \\ &= (1 + \alpha + 2\beta)X - (\alpha + 3\beta)XAX + \beta X(AX)^2) \\ &= X\left((1 + \alpha + 2\beta)I - (\alpha + 3\beta)AX + \beta(AX)^2\right) \\ &= X(aI + bAX + c(AX)^2), \end{aligned}$$

where $a = 1 + \alpha + 2\beta$, $b = -(\alpha + 3\beta)$, $c = \beta$ such that $a + b + c = 1$.

Thus, a generalized second order iterative method can be presented as

$$X_{k+1} = X_k\left(aI + bAX_k + c(AX_k)^2\right), \tag{9}$$

where $a = 1 + \alpha + 2\beta$, $b = -(\alpha + 3\beta)$, $c = \beta$ such that $a + b + c = 1$.

Lemma 2.1. *For all $k \geq 0$, the sequence $\{X_k\}$ generated by (9) with $X_0 = \delta A^*$ satisfies*

(*i*) $X_kA = (X_kA)^*$,
(*ii*) $A^{\dagger}AX_k = X_k$,
(*iii*) $X_kAA^{\dagger} = X_k$.

Proof. We shall prove this lemma by mathematical induction. Clearly, statement (*i*) holds for $k = 0$, since we have $X_0A = \delta A^*A = (X_0A)^*$. Assume that it holds for some positive integer k, i.e., $X_kA = (X_kA)^*$. To show that it also holds for $k + 1$, consider

$$\begin{aligned} (X_{k+1}A)^* &= a(X_kA)^* + b((X_kA)^*)^2 + c((X_kA)^*)^3 \\ &= aX_kA + b(X_kA)^2 + c(X_kA)^3 \\ &= X_k\left(aI + bX_kA + c(X_kA)^2\right)A \\ &= X_{k+1}A. \end{aligned}$$

Thus, the first statement is true for $k + 1$. To prove the second statement for $k = 0$, we find

$$A^{\dagger}AX_0 = \delta A^{\dagger}AA^* = \delta(A^{\dagger}A)^*A^* = \delta A^* = X_0.$$

Assume it holds for some k, i.e., $A^{\dagger}AX_k = X_k$. Then, we have

$$A^{\dagger}AX_{k+1} = A^{\dagger}A\left(X_k\left(aI + bX_kA + c(X_kA)^2\right)\right) = X_{k+1}.$$

This shows that the second statement holds for $k+1$.

In a similar manner, the third statement can also be proved. □

Now, we are in a position to discuss the convergence analysis of the iterative scheme (9) with the starting value $X_0 = \delta A^*$ and show that it converges to the Moore–Penrose inverse $X = A^{\dagger}$.

Theorem 2.2. *Let $\mathbf{0} \neq A \in \mathbb{C}^{m\times n}$, $X = A^{\dagger}$, the initial approximation $X_0 = \delta A^*$, for arbitrary real number δ be such that the residual $R_0 = (X_0 - X)A$ satisfies $\|R_0\| < 1$. The sequence $\{X_k\}$ generated by (9) for $\alpha \geq 0$ starting with $X_0 = \delta A^*$ converges to the Moore–Penrose inverse $A^{\dagger}$. It exhibits linear convergence for $\alpha + \beta \neq 1$, quadratic convergence for $\alpha + \beta = 1$ and third order convergence for $\alpha = 0$ and $\beta = 1$. The first, second, and third order error terms are given by*

$$error_1 = (1-\alpha-\beta)E_k,\ error_2 = -\alpha E_kAE_k \text{ and } error_3 = \beta E_k(AE_k)^2,$$

where $E_k = X_k - A^{\dagger}$ denotes the error matrix.

Proof. To prove the first part of the theorem, it suffices to verify that $\|X_k - X\| \to 0$ when $k \to +\infty$. Using the properties of the Moore–Penrose inverse X and the results of Lemma 2.1, we obtain

$$\begin{aligned}\|X_{k+1} - X\| &= \|X_{k+1}AX - XAX\| \\ &\leq \|X_{k+1}A - XA\|\|X\|.\end{aligned}$$

Applying the scheme (9) and Lemma 2.1, we have

$$\begin{aligned}X_{k+1}A - XA &= aX_kA + bX_kAX_kA + cX_k(AX_k)^2A - XA \\ &= aX_kA - XA + b(X_kA)^2 + c(X_kA)^3 \\ &= (1+\alpha+2\beta)X_kA - XA - (\alpha+3\beta)(X_kA)^2 + \beta(X_kA)^3 \\ &= X_kA - XA + \alpha\left[X_kA - (X_kA)^2\right] + \beta\left[2X_kA - 3(X_kA)^2 + (X_kA)^3\right] \\ &= X_kA - XA - \alpha\left[(X_kA - XA)^2 + (X_kA - XA)\right] + \beta\left[(X_kA - XA)^3 - (X_kA - XA)\right].\end{aligned}$$

After rearranging terms, we have

$$X_{k+1}A - XA = (X_kA - XA) - \alpha\left[(X_kA - XA)^2 + (X_kA - XA)\right] + \beta\left[(X_kA - XA)^3 - (X_kA - XA)\right].$$

Thus, the sequence of residual matrices defined by $R_k = X_kA - XA$ satisfies the following recurrence relation:

$$\begin{aligned}R_{k+1} &= R_k - \alpha[R_k^2 + R_k] + \beta[R_k^3 - R_k] \\ &= (1-\alpha-\beta)R_k - \alpha R_k^2 + \beta R_k^3. \qquad (10)\end{aligned}$$

Let $s_k = \|R_k\|$. Now, for the convergence of the sequence $\{X_k\}$, we require that $s_k \to 0$ as $k \to \infty$. This can be shown by mathematical induction. Clearly, it holds for $k = 0$ as $s_0 = \|R_0\| = \|X_0A - XA\| < 1$. Assuming it holds for some k, i.e., $s_k < 1$, we show that it for $k+1$. Taking the norm on recurrence relation (10), we get

$$s_{k+1} \leq (1-\alpha-\beta)s_k + \alpha s_k^2 + \beta s_k^3 \leq s_k, \qquad (11)$$

whenever $\alpha \geq 0$, $\beta \geq 0$ and $0 < \alpha + \beta \leq 1$. This completes the proof by induction, since $s_{k+1} \leq s_k < 1$. Moreover, inequality (11) implies $s_{k+1} \leq s_k$ holds for $k = 0, 1, \ldots$, indicating that s_k is a decreasing sequence. Since $s_k \geq 0$ is bounded, we conclude that s_k is convergent and $s_k \to s$ when $k \to +\infty$. Moreover, $0 \leq s < 1$ holds. Using earlier inequality (11), we obtain

$$s \leq (1-\alpha-\beta)s + \alpha s^2 + \beta s^3.$$

The last inequality implies that either $s = 1$ or $s = 0$. Hence, we conclude s = 0. This proof that $s_k \to 0$ as $k \to +\infty$ and consequently, $\|X_k - X\| \leq s_k\|X\|$, which leads to $X_k \to X$ when $k \to +\infty$. This establishes the convergence of method described in (9), and this confirms the first part of the theorem.

Next, substituting $X_k = X + E_k$ into (9), one can obtain

$$\begin{aligned} E_{k+1} + X &= (E_k + X)\left[aI + bA(E_k + X) + c(A(E_k + X))^2\right] \\ &= a(E_k + X) + b(E_k + X)(AE_k + AX) + c(E_k + X)(AE_k + AX)^2. \end{aligned}$$

This simplifies to

$$\begin{aligned} E_{k+1} &= aE_k + (a-1)X + b[E_kAE_k + E_kAX + XAE_k + XAX] + c(E_k + X)[AXAX + (AE_k)^2 + AXAE_k + AE_kAX] \\ &= aE_k + (a-1)X + b[E_kAE_k + E_kAX + XAE_k + X] + c(E_k + X)[AX + (AE_k)^2 + AE_k + AE_kAX] \\ &= aE_k + (a-1)X + b[E_kAE_k + E_kAX + XAE_k + X] \\ &\quad + c(E_kAX + E_k(AE_k)^2 + E_kAE_k + (E_kA)^2X + X + X(AE_k)^2 + XAE_k + XAE_kAX). \end{aligned}$$

We can further break this down into error terms as

$$\begin{aligned} error_1 =& aE_k + (a-1)X + b(E_kAX + XAE_k + X) + c(E_kAX + X + XAE_k + XAE_kAX), \\ error_2 =& bE_kAE_k + c(E_kAE_k + (E_kA)^2X + X(AE_k)^2), \\ error_3 =& cE_k(AE_k)^2. \end{aligned}$$

Using $E_k = X_k - X$ and the results from Lemma 2.1, the $error_1$, $error_2$ and $error_3$ can be expressed as

$$\begin{aligned} error_1 =& (1 - \alpha - \beta)E_k, \\ error_2 =& -\alpha E_kAE_k, \\ error_3 =& \beta E_k(AE_k)^2. \end{aligned}$$

Obviously, $error_1$ vanishes if and only if $\alpha + \beta = 1$. While $error_2$ will be zero only in the case of $\alpha = 0$. Hence, using these observations with the condition $\alpha \geq 0$, we conclude that the proposed scheme has at least quadratic convergence for the set of values $\alpha, \beta \in [0, 1] \subset \mathbb{R}$ with the condition $\alpha + \beta = 1$. Moreover, the scheme has cubic convergence for $\alpha = 0$ and $\beta = 1$. □

Theorem 2.3. *Iterative scheme (9) with initial approximation $X_0 = \delta A^*$ results in the following relation*

$$\lim_{k\to\infty} \frac{s_{k+1}}{s_k} = 1 - \alpha - \beta,$$

where $s_k = \|R_k\|$.

Proof. Recalling the recurrence relation (10)

$$R_{k+1} = (1 - \alpha - \beta)R_k - \alpha R_k^2 + \beta R_k^3. \tag{12}$$

Taking the norm on the above recurrence relation, we have

$$\|R_{k+1}\| = \|(1 - \alpha - \beta)R_k - \alpha R_k^2 + \beta R_k^3\|. \tag{13}$$

This leads to the following inequality

$$\begin{aligned} \|R_{k+1}\| &\leq |1 - \alpha - \beta|\|R_k\| + |\alpha|\|R_k\|^2 + |\beta|\|R_k\|^3, \\ \frac{\|R_{k+1}\|}{\|R_k\|} &\leq |1 - \alpha - \beta| + |\alpha|\|R_k\| + |\beta|\|R_k\|^2. \end{aligned}$$

On the other hand, from (13), we also have

$$\begin{aligned} ||R_{k+1}|| &\geq |1-\alpha-\beta|||R_k|| - |\alpha|||R_k||^2 - |\beta|||R_k||^3, \\ \frac{||R_{k+1}||}{||R_k||} &\geq |1-\alpha-\beta| - |\alpha|||R_k|| - |\beta|||R_k||^2. \end{aligned}$$

Consequently, these inequalities bounds in terms of $s_k = ||R_k||$ indicates

$$|1-\alpha-\beta| - |\alpha|s_k - |\beta|s_k^2 \leq \frac{s_{k+1}}{s_k} \leq |1-\alpha-\beta| + |\alpha|s_k + |\beta|s_k^2. \tag{14}$$

As s_k approaches zero, Theorem (2.2) permits us to deduce, by taking a limit of (14), $\frac{s_{k+1}}{s_k} \to 1-\alpha-\beta$ as k approaches infinity. This completes the proof. □

3. Theoretical complexity

This section emphasizes the study of measuring the theoretical efficiency of the iterative method. Several factors contribute to an algorithm's performance, including the local convergence order, the number of iterations, and the count of matrix-matrix multiplications involved in evaluating the matrix inverse. We analyze the computational complexity of our method and compared with its opponents using the concept of an efficiency index. The computational efficiency index [16] (*CEI*) of the p^{th}-order iterative method is measured as

$$CEI = p^{\frac{1}{d}}, \tag{15}$$

where d denotes the total computational cost per step. In this context, the computational cost d relies on the distinct count of matrix products at each iteration k and the number of iterations used for convergence s. In particular, for the p^{th}-order iterative methods, this cost is calculated as ks (for more detail, refer [17, 18]). Theoretically, the approximate value of s for Schulz method is given by Söderström and Stewart [19] as follows

$$s \approx 2\log_2 \kappa_2(A),$$

where $\kappa_2(A)$ is the condition number of A with 2-norm. This idea is further extended by Soleymani [17] for p^{th}-order Schulz-type method and provides the following approximation

$$s \approx 2\log_p \kappa_2(A). \tag{16}$$

To determine the efficiency index using equations (15) and (16) of the iterative scheme (9), we consider different parametric values and abbreviated them as follows. For $\beta = 1$ & $\alpha = 0$, scheme (9) is denoted by CM_1, which defines the well-known cubically convergent Chebyshev matrix method [20]. The second order proposed scheme is denoted by SM_2, SM_3, and SM_4 for $\beta = 0.8$ & $\alpha = 0.2$, $\beta = \alpha = 0.5$, and $\beta = 0$ & $\alpha = 1$, respectively. The method SM_4 corresponds to the well-known quadratically convergent Newton-Schulz method [1]. For $\beta = 1$ & $\alpha = 0.1$, the linear convergence method is tested, which is referred to as LM_5. We compare the results with the same-order matrix inversion finders proposed by Petković and Stanimirović [7] (denoted by *PS*) and Stanimirović et al. [5] (denoted by *SM*) as well as eighteenth order method (8) (denoted by EM_{18}). The estimated *CEI* values for each technique are given in Table 1. Overall, the methods SM_{18} and CM_1 demonstrate a better efficiency index. Among the quadratic convergence schemes, the SM_4 possesses a higher *CEI* than its counterparts; but, its performance in numerical testing (refer to section 4) does not meet expectations on its theoretical efficiency index when compared to other tested quadratic methods.

Table 1: Comparisons of *CEI* for different iterative methods

Methods	EM_{18}	CM_1	SM_2	SM_3	SM_4	SM	LM_5	PS
$s=1$	$18^{\frac{1}{7}} \approx 1.51$	$3^{\frac{1}{3}} \approx 1.44$	$2^{\frac{1}{3}} \approx 1.26$	$2^{\frac{1}{3}} \approx 1.26$	$2^{\frac{1}{2}} \approx 1.41$	$2^{\frac{1}{3}} \approx 1.26$	1	1
$s=2$	1.23	1.21	1.12	1.12	1.19	1.12	1	1
$s=3$	1.15	1.13	1.08	1.08	1.09	1.08	1	1

4. Numerical section

This section provides the numerical behavior of the proposed matrix iterative method (9). The performance of the scheme is studied by implementing it to real-life problems and compared with the results obtained from existing techniques. We displayed the number of iterations k, last three errors $e_k = \|X_{k+1} - X_k\|$, computational order of convergence ρ, and computational time T (in seconds). The approximated value of ρ is measured by using the following expression

$$\rho \approx \frac{\ln(\|X_{k+1} - X_k\|/\|X_k - X_{k-1}\|)}{\ln(\|X_k - X_{k-1}\|/\|X_{k-1} - X_{k-2}\|)}, \ k = 2, 3, \ldots. \tag{17}$$

In order to measure these comparison components, the initial approximations $X_0 = \frac{1}{\|A\|_1\|A\|_\infty} A^*$ and stopping criteria $\|X_{k+1} - X_k\|_\infty < \tau$, where τ represents the tolerance, are employed. Furthermore, the technical and symbolic computation software `Mathematica` [21] version 11 is used to conduct the numerical findings up to 250 significant digits.

Example 4.1. *Consider the following rank-deficient matrix*

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 1 \\ 1 & 3 & 4 & 6 & 2 \\ 2 & 3 & 4 & 5 & 3 \\ 3 & 4 & 5 & 6 & 4 \\ 4 & 5 & 6 & 7 & 6 \\ 6 & 6 & 7 & 7 & 8 \end{bmatrix}. \tag{18}$$

The numerical results are displayed in Table 2 with $\tau = 10^{-30}$. A notable difference in the iteration count between the existing and proposed linear and quadratic order iterative schemes is observed. As inspected from the finding, SM_2 performs much better than other quadratic convergent schemes.

Table 2: Comparisons of iterative methods using example 4.1

Method	k	e_{k-2}	e_{k-1}	e_k	ρ	T
CM_1	15	5.47478(−10)	1.29839(−30)	1.7319(−92)	3.0000	0.218
SM_2	16	2.79245(−7)	1.38725(−15)	3.42365(−32)	2.0000	0.391
SM_3	18	9.66005(−9)	4.15032(−18)	7.66103(−37)	2.0000	0.344
SM_4	23	3.06156(−13)	8.33755(−27)	6.18344(−54)	2.0000	0.376
SM	34	3.01258(−12)	1.21094(−24)	1.95654(−49)	2.0000	0.671
LM_5	40	2.47729(−29)	2.47729(−30)	2.47729(−31)	1.0000	0.421
PS	1038	1.13112(−30)	1.04629(−30)	9.67816(−31)	1.0000	0.769

Example 4.2. *Consider the statically determinate truss problem [6, 22] shown in Figure 1. To determine the resultant forces on the roller and stationary support with the members and the reaction forces, we model the problem into the linear system $Ax = b$, where*

$$A = \begin{bmatrix} a_1 & 0 & a_3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ a_2 & 0 & a_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -a_2 & -a_6 & 0 & 0 & 0 & 0 & 0 & a_6 & a_2 & 0 & 0 & 0 \\ -a_1 & a_5 & 0 & 0 & 1 & 0 & 0 & a_5 & -a_1 & 0 & 0 & 0 \\ 0 & a_6 & -a_4 & a_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -a_5 & -a_3 & a_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -a_6 & 0 & a_6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -a_5 & -1 & a_5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -a_6 & a_4 & -a_6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & a_5 & -a_3 & -a_5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -a_4 & 0 & a_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_3 & 0 & a_1 & 0 & 1 & 0 \end{bmatrix},$$

$$x = \begin{bmatrix} F_1 & F_2 & F_3 & F_4 & F_5 & F_6 & F_7 & F_8 & F_9 & F_V & F_R & F_H \end{bmatrix}^t,$$
$$b = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 500 & 0 & 1000 & 0 & 500 & 0 & 0 \end{bmatrix}^t,$$

and $a_1 = \sin(\theta_1)$, $a_2 = \cos(\theta_1)$, $a_3 = \sin(\theta_2)$, $a_4 = \cos(\theta_2)$, $a_5 = \sin(\theta_3)$, $a_6 = \cos(\theta_3)$, $\theta_1 = \arctan(\frac{1}{2})$, $\theta_2 = \arctan(\frac{4}{3})$ *and* $\theta_3 = \arctan(\frac{1}{3})$ *and then solve it using proposed iterative family. The computational components are evaluated with a tolerance of* $\tau = 10^{-30}$ *and compared with other schemes. These results, shown in Table 3, demonstrate that the proposed quadratic convergent methods provide more accurate solutions in less time and iterations than the well-known Schulz and SM methods. Moreover, by observing linear convergent schemes, the presented method* LM_5 *converges quite rapidly compared to the PS technique.*

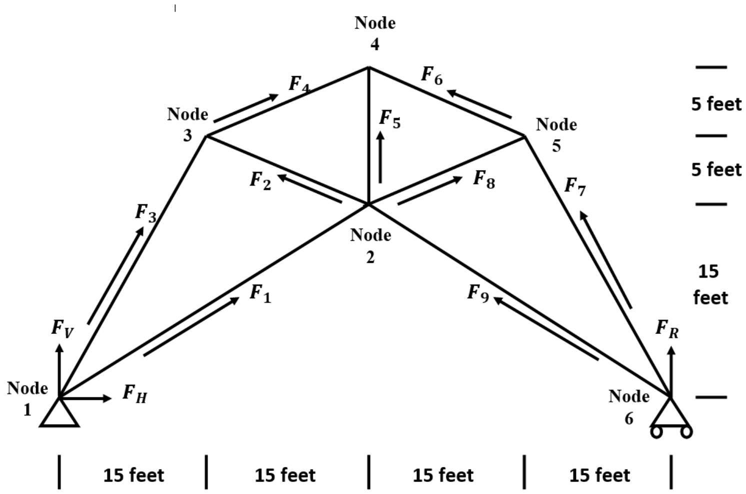


Figure 1: Forces on a statically determinate truss

Table 3: Comparisons of iterative methods using example 4.2

Method	k	e_{k-2}	e_{k-1}	e_k	ρ	T
CM_1	10	3.55009(−8)	6.77114(−25)	4.69817(−75)	3.0000	0.265
SM_2	11	3.27723(−8)	2.64251(−17)	1.71805(−35)	2.0001	0.296
SM_3	12	1.76698(−7)	1.92047(−15)	2.26860(−31)	2.0000	0.312
SM_4	15	2.96536(−10)	1.08175(−20)	1.43955(−41)	2.0000	0.311
SM	21	1.29021(−12)	3.07176(−25)	1.74116(−50)	2.0000	1.516
LM_5	36	4.24312(−29)	4.24312(−30)	4.24312(−31)	1.0000	0.469
PS	961	1.11829(−30)	1.03442(−30)	9.56835(−31)	1.0000	5.675

Example 4.3. *Consider the partial differential equation (pde):*

$$\frac{\partial U}{\partial t} = \frac{\partial^2 U}{\partial x^2} \quad (0 < x < 1,\ 0 < t \le 0.1), \tag{19}$$

satisfying the initial condition

$$U = \sin \pi x \ \ \textit{when } t = 0 \textit{ for } 0 \le x \le 1,$$

and the boundary condition

$$U = 0 \ \ \textit{at } x = 0 \textit{ and } 1 \textit{ for } t > 0.$$

Our aim is to find the approximate U using finite-difference methods. In particular, we applied the Crank-Nicolson implicit method on equation (19) *to evaluate U at n points, which further yields the following approximated equation*

$$\frac{U_{i,j+1} - U_{i,j}}{k} = \frac{1}{2}\left\{\frac{U_{i+1,j+1} - 2U_{i,j+1} + U_{i-1,j+1}}{h^2} + \frac{U_{i+1,j} - 2U_{i,j} + U_{i-1,j}}{h^2}\right\}$$

implies

$$-rU_{i-1,j+1} + (2+2r)U_{i,j+1} - rU_{i+1,j+1} = rU_{i-1,j} + (2-2r)U_{i,j} + rU_{i+1,j}$$

where $r = k/h^2$. The following linear system of equations is obtained by taking step sizes $h = 0.1$ and $k = 0.01$,

$$AU = b, \tag{20}$$

where $A = \begin{bmatrix} B_1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ B_2 & B_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & B_2 & B_1 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \ddots & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & B_2 & B_1 \end{bmatrix}$, $B_1 = \begin{bmatrix} 4 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 4 \end{bmatrix}$, $\mathbf{0}$ *is a zero matrix*

of order 9×9, $B_2 = \begin{bmatrix} 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{bmatrix}$,

$U = [U_{11}, U_{21}, U_{31}, U_{41}, \ldots, U_{91}, U_{12}, U_{22}, U_{23}, \ldots, U_{29}, \ldots, U_{1,10}, U_{2,10}, \ldots, U_{9,10}]^t$, *and* $b = [\sin 0.2\pi, \sin 0.1\pi + \sin 0.3\pi, \sin 0.2\pi + \sin 0.4\pi, \sin 0.3\pi + \sin 0.5\pi, \sin 0.4\pi + \sin 0.6\pi, \sin 0.5\pi + \sin 0.7\pi, \sin 0.6\pi + \sin 0.8\pi, \sin 0.7\pi + \sin 0.9\pi, \sin 0.8\pi, 0, 0, \ldots, 0]^t$.

In order to check the applicability of the proposed iterative methods (9) *on solving PDE* (19)*, the obtained linear system* (20) *is considered. The numerical outcomes for the coefficient matrices are calculated using* $\tau = 10^{-30}$*, are displayed in Table 4. The experimental data concludes that the proposed scheme, for each considered parametric value, provides superior results as compared to the same order existing methods. In addition to this, the final approximate values of* U*, accurate to four decimal places, are equal to* $\big[0.2802, 0.5329, 0.7335, 0.8623, 0.9067, 0.8623, 0.7335, 0.5329, 0.2802, 0.2540, 0.4832, 0.6651, 0.7818, 0.8221, 0.7818, 0.6651, 0.4832, 0.2540, 0.2303, 0.4381, 0.6030, 0.7089, 0.7453, 0.7089, 0.6030, 0.4381, 0.2303, 0.2088, 0.3972, 0.5467, 0.6427, 0.6758, 0.6427, 0.5467, 0.3972, 0.2088, 0.1893, 0.3602, 0.4957, 0.5827, 0.6127, 0.5827, 0.4957, 0.3602, 0.1893, 0.1717, 0.3265, 0.4494, 0.5284, 0.5556, 0.5284, 0.4494, 0.3265, 0.1717, 0.1557, 0.2961, 0.4075, 0.4791, 0.5037, 0.4791, 0.4075, 0.2961, 0.1557, 0.1411, 0.2684, 0.3695, 0.4344, 0.4567, 0.4345, 0.3695, 0.2684, 0.1411, 0.1280, 0.2434, 0.3350, 0.3938, 0.4141, 0.3938, 0.3350, 0.2434, 0.1280, 0.1160, 0.2207, 0.3037, 0.3571, 0.3754, 0.3571, 0.3037, 0.2207, 0.1160\big]$*, are obtained using the method* CM_1*. Overall, we can conclude that the developed scheme serves as a better alternative compared to existing linear and quadratic order iterative methods.*

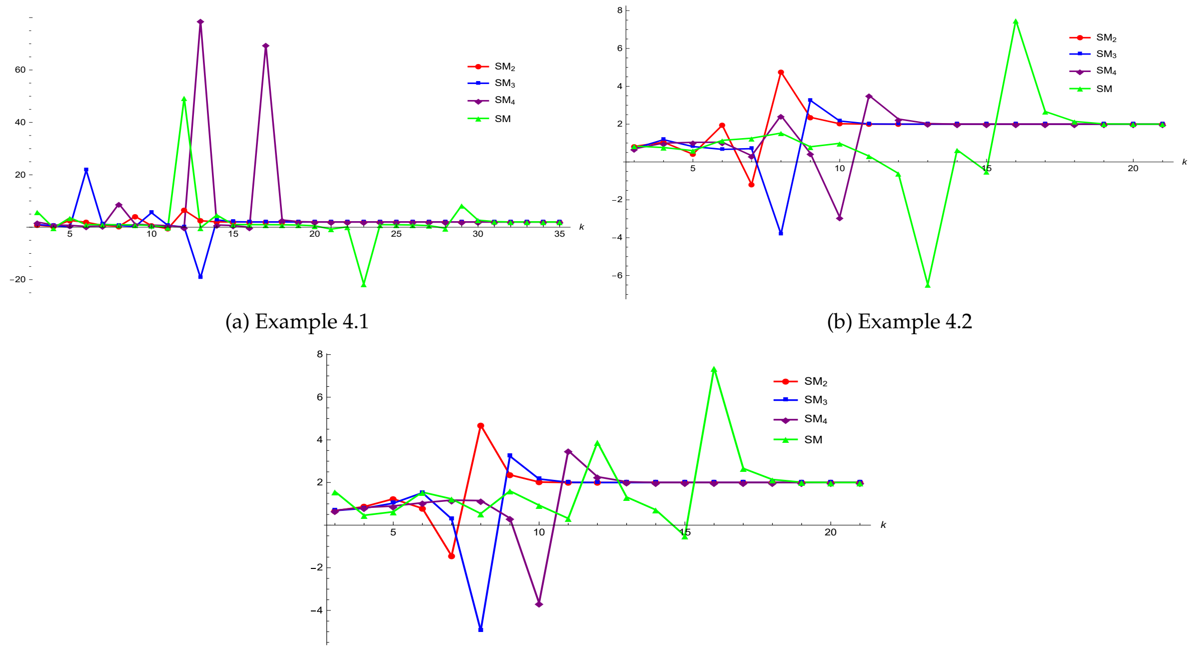
The motivation behind this problem is to assess the applicability of Schulz-type solvers for determining the solution of partial differential equations (PDEs). The resulting linear system is sparse, making such solvers well-suited for efficient computation. However, solving the linear system arising from discretization can still be challenging due to factors such as large system size and poor conditioning. Thus, these methods serve as a robust alternative when direct solvers or traditional iterative methods are inefficient due to the characteristics of the system matrix. Moreover, we found that investigating the numerical implementation of the proposed technique over a particular case of PDE (19) *is quite fruitful. Thus, one can also determine the solutions for elliptic, parabolic, and hyperbolic PDEs using the proposed method, even when the coefficient matrix of a modeled problem in a linear system becomes singular or rectangular.*

Table 4: Comparisons of iterative methods using example 4.3

Method	k	e_{k-2}	e_{k-1}	e_k	ρ	T
CM_1	10	1.42806(−8)	2.44389(−25)	1.22485(−75)	3.0000	18.063
SM_2	11	1.33766(−8)	1.03667(−17)	6.22630(−36)	2.0001	22.453
SM_3	12	7.21643(−8)	7.54283(−16)	8.24059(−32)	2.0000	26.375
SM_4	15	1.17688(−10)	4.01223(−21)	4.66328(−42)	2.0008	28.048
SM	21	4.76395(−13)	9.86153(−26)	4.2257(−51)	2.0002	406.594
LM_5	36	1.78668(−29)	1.78668(−30)	1.78668(−31)	1.0000	61.641
PS	950	1.11661(−30)	1.03287(−30)	9.55401(−31)	1.0000	1421.78

Apart from testing the scheme on the basis of accuracy, we also compared the computational convergence behavior of new second order schemes with existing approaches. The comparison is illustrated in Figure 2 using Examples 4.1, 4.2, and 4.3. These figures illustrate the performance of the iterative methods in terms of computational order of convergence with respect to iterations. According to Figure 2(a), the SM_2, SM_3, SM_4, and SM methods reach to convergence phase after 14, 15, 20 and 32 iterations, respectively. Figures 2(b) and 2(b)(c) demonstrate that each of the methods SM_2, SM_3, SM_4, and SM, require about 9, 10, 13 and 17 iterations, respectively, to meet the theoretical convergence order. Eventually, the proposed methods SM_2 and SM_3, reach the convergence phase earlier than the Schulz method SM_4 and SM methods. As expected from the data displayed in Tables 2, 3, and 4, the performance of SM_2 and SM_3 in each example is comparatively better in terms of both convergence phase and accurate solution than SM_4 and SM.

Example 4.4. *In this test problem, we measure the performance of the developed scheme* (9) *on a variety of application matrices. These test data are taken from Matrix Market Library [23], which provides the matrix generation tools and*



(a) Example 4.1

(b) Example 4.2

(c) Example 4.3

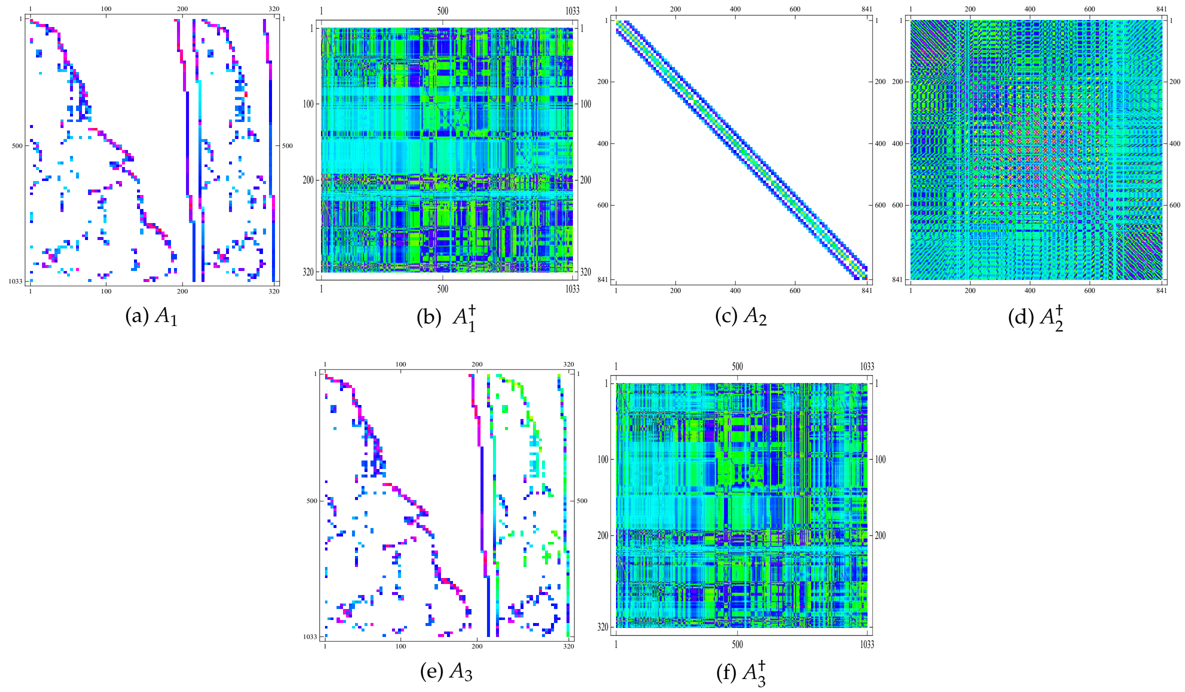
Figure 2: Iterations versus computational convergence order

services. The descriptions of the tested matrices are given in Table 5. By reason of the large order of matrices, their visual representation along with the Moore–Penrose inverse, are shown in Figure 3.

The numerical outcomes using $\tau = 10^{-8}$ *are demonstrated in Tables 6-8. It can be seen that the proposed quadratic schemes are competing with the existing same order methods. Moreover,* LM_5 *exhibits good results as compared to the linear order method PS. Overall,* SM_2 *is the more acceptable method in view of quadratic convergence.*

Table 5: Details of matrices

A_i	Matrix	Size	Entries	Type
A_1	ILLC1033	1033×320	4732	real unsymmetric
A_2	YOUNG1C	841×841	4089	complex symmetric indefinite
A_3	WELL1033	1033×320	4732	real unsymmetric



(a) A_1 (b) $A_1^\dagger$ (c) A_2 (d) $A_2^\dagger$

(e) A_3 (f) $A_3^\dagger$

Figure 3: Visual representation of sparse matrices along with its approximate inverse obtained by SM_2

Table 6: Comparisons of iterative methods using example 4.4 for A_1

Method	k	e_{k-2}	e_{k-1}	e_k	ρ	T
CM_1	23	747.551	1.76989(−1)	1.04919(−9)	3.07736	14.376
SM_2	25	67.6965	1.89715(−2)	1.57868(−9)	1.99293	15.500
SM_3	28	44.7166	2.06033(−2)	4.43591(−9)	1.99817	17.672
SM_4	36	51.1454	5.39120(−2)	5.98222(−8)	2.00019	21.781
SM	58	21.4820	1.42526(−2)	2.36813(−8)	2.01222	606.188
LM_5	32	2.30217(−7)	2.32912(−8)	2.54935(−9)	0.99426	20.642
PS	641	1.13256(−8)	1.04783(−8)	9.69217(−9)	0.994763	648.093

Table 7: Comparisons of iterative methods using example 4.4 for A_2

Method	k	e_{k-2}	e_{k-1}	e_k	ρ	T
CM_1	14	7.71696(−2)	1.80231(−4)	2.28837(−12)	3.1608	28.657
SM_2	16	3.56565(−4)	1.59122(−8)	2.96861(−15)	2.02597	29.109
SM_3	17	1.71856(−2)	9.42404(−5)	2.77657(−9)	2.09762	30.813
SM_4	22	4.03845(−3)	1.02478(−5)	6.56577(−11)	2.03159	31.001
SM	34	4.21653(−3)	1.67911(−5)	2.68443(−10)	2.03835	141.641
LM_5	19	1.20178(−7)	1.20177(−8)	1.20177(−9)	1.0000	37.641
PS	377	1.13983(−8)	1.05435(−8)	9.7527(−9)	1.0000	2558.69

Table 8: Comparisons of iterative methods using example 4.4 for A_3

Method	k	e_{k-2}	e_{k-1}	e_k	ρ	T
CM_1	15	3.25384	1.35865(−4)	6.43416(−13)	3.03500	11.891
SM_2	16	2.9489	3.56957(−3)	4.99004(−9)	2.00700	15.032
SM_3	18	1.26217	1.59852(−3)	2.49175(−9)	2.00428	12.251
SM_4	23	6.64651(−1)	8.76376(−4)	1.48562(−9)	2.00381	13.796
SM	36	1.04859(−1)	3.18485(−5)	5.97821(−9)	2.01270	281.735
LM_5	22	3.02028(−7)	3.02028(−8)	3.02031(−9)	0.999994	16.719
PS	459	1.11774(−8)	1.03391(−8)	9.56365(−9)	0.99997	505.313

5. Conclusions

In this paper, we have established a generic iterative method for evaluating the generalized inverse of a matrix. The construction of the method is based on the second Penrose equation. A theoretical investigation has been conducted to determine its convergence phase. It was further found that the proposed family generates several first and second order iterative methods. The most notable case occur when the parameters are set to $\beta = 1$ and $\alpha = 0$, yielding a cubic convergent iterative method with a higher efficiency index. Furthermore, the well-known Schulz and Chebyshev matrix iterative methods can be derived from the presented algorithm. In order to justify the theoretical results, numerical testing were performed on different types of matrices using the Matrix Market Library services. Furthermore, the applicability of the method was explored through practical problems, such as statically determinate truss systems and one-dimensional parabolic problems, by modeling them as linear systems. The obtained results were compared with those of existing methods and it was observed that the new schemes SM_2, SM_3, and LM_5, perform effectively and are competing with their counterparts of the same order.

Acknowledgments

The authors would like to sincerely thank the referees for their valuable comments and suggestions.

Conflicts of Interest

The authors declare that they have no conflict of interest

References

[1] G. Schulz. Iterative berechung der reziproken matrix. *ZAMM Z. Angew. Math. Mech.*, 13(1):57–59, 1933.
[2] Adi Ben Israel and Dan Cohen. On iterative computation of generalized inverses and associated projections. *SIAM J. Numer. Anal.*, 3(3):410–419, 1966.
[3] A. Ben-Israel and T. N. E. Greville. *Generalized Inverses: Theory and Applications*. Springer, New York, Second edition, 2003.
[4] V. Pan and R. Schreiber. An improved Newton iteration for the generalized inverse of a matrix, with applications. *SIAM J. Sci. Stat. Comp.*, 12(5):1109–1130, 1991.
[5] Predrag S. Stanimirović, Vasilios N. Katsikis, Shwetabh Srivastava, and Dimitrios Pappas. A class of quadratically convergent iterative methods. *RACSAM*, 113(4):3125–3146, 2019.
[6] Manpreet Kaur, Munish Kansal, and Sanjeev Kumar. An efficient matrix iterative method for computing Moore–Penrose inverse. *Mediterr. J. Math.*, 18(2), 2021. Article No. 42.
[7] Marko D. Petković and Predrag S. Stanimirović. Iterative method for computing the Moore–Penrose inverse based on Penrose equations. *J. Comput. Appl. Math.*, 235(6):1604–1613, 2011.
[8] Shwetabh Srivastava and Dharmendra Kumar Gupta. A third order iterative method for $A^{\dagger}$. *Int. J. Comput. Sci. Math.*, 4(2):140–151, 2013.
[9] Predrag S Stanimirović, Ashim Kumar, and Vasilios N Katsikis. Further efficient hyperpower iterative methods for the computation of generalized inverses $A^{(2)}_{T,S}$. *RACSAM*, 113(4):3323–3339, 2019.

[10] Victor Y Pan, Fazlollah Soleymani, and Liang Zhao. An efficient computation of generalized inverse of a matrix. *Appl. Math. Comput*, 316:89–101, 2018.

[11] Ashim Kumar, Predrag S Stanimirović, Fazlollah Soleymani, Mihailo Krstić, and Kostadin Rajković. Factorizations of hyperpower family of iterative methods via least squares approach. *Comput. Appl. Math.*, 37:3226–3240, 2018.

[12] Marko D Petković and Miodrag S Petković. Hyper-power methods for the computation of outer inverses. *J. Comput. Appl. Math.*, 278:110–118, 2015.

[13] F Soleymani, Predrag S Stanimirović, and F Khaksar Haghani. On hyperpower family of iterations for computing outer inverses possessing high efficiencies. *Linear Algebra Appl.*, 484:477–495, 2015.

[14] Munish Kansal, Manpreet Kaur, Litika Rani, and Lorentz Jäntschi. A cubic class of iterative procedures for finding the generalized inverses. *Mathematics*, 11(13):3031, 2023.

[15] Farahnaz Soleimani, Predrag S Stanimirović, and Fazlollah Soleymani. Some matrix iterations for computing generalized inverses and balancing chemical equations. *Algorithms*, 8(4):982–998, 2015.

[16] J. F. Traub. *Iterative Methods for the Solution of Equations*. Prentice-Hall: Englewood Cliffs, New Jersey, 1964.

[17] Fazlollah Soleymani. On finding robust approximate inverses for large sparse matrices. *Linear Multilinear Algebra*, 62(10):1314–1334, 2014.

[18] Munish Kansal, Sanjeev Kumar, and Manpreet Kaur. An efficient matrix iteration family for finding the generalized outer inverse. *Appl. Math. Comput.*, 430, 2022. Article Id: 127292.

[19] Torsten Söderström and G. W. Stewart. On the numerical properties of an iterative method for computing the Moore–Penrose generalized inverse. *SIAM J. Numer. Anal.*, 11(1):61–74, 1974.

[20] Hou-Biao Li, Ting-Zhu Huang, Yong Zhang, Xing-Ping Liu, and Tong-Xiang Gu. Chebyshev-type methods and preconditioning techniques. *Appl. Math. Comput.*, 218(2):260–270, 2011.

[21] M. Trott. *The Mathematica Guidebook for Programming*. Springer, New York, 2013.

[22] Manpreet Kaur and Munish Kansal. An efficient class of iterative methods for computing generalized outer inverse $M^{(2)}_{T,S}$. *J. Appl. Math. Comput.*, 64(1):709–736, 2020.

[23] Matrix Market, https://math.nist.gov/matrixmarket.