International Journal of Applied Mathematics

Volume 35 No. 3 2022, 365-380
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)
doi: http://dx.doi.org/10.12732/ijam.v35i3.1

A NEW FAMILY OF SECOND-ORDER ITERATIVE METHODS FOR COMPUTING THE MOORE PENROSE INVERSE BASED ON PENROSE EQUATIONS

Zainab Abu Iram[1], Ali Zein[2 §]

[1,2] Department of Applied Mathematics
Palestine Polytechnic University
Hebron, PALESTINE

Abstract: A new family of second-order iterative algorithms for computing the Moore-Penrose inverse is developed. The construction of this algorithm is based on the usage of Penrose Equations (1) and (2). Convergence properties are considered. Numerical results are also presented and a comparison with Newton's method is made. It is observed that the new methods require less number of iterations than that of Newton's method. In addition, numerical experiments show that these methods are more effective than Newton's method when the number of columns increases than the number of rows.

AMS Subject Classification: 15A09, 65F30
Key Words: generalized inverse; Moore-Penrose inverse; iterative method; Penrose equations

1. Introduction

The Moore-Penrose inverse finds many applications in engineering and applied problems such as signal and image processing, control of robot manipulators

Received: December 20, 2021

[§]Correspondence author

and statistical regression analysis, see [9, 8, 6].

The Moore-Penrose inverse of an $m \times n$ complex matrix A, denoted by $A^\dagger$, is a unique $n \times m$ matrix X satisfying the following four Penrose equations:

$$(1)\ AXA = A, \qquad (2)\ XAX = X,$$

$$(3)\ (AX)^* = AX, \qquad (4)\ (XA)^* = XA.$$

The direct methods such as singular value decomposition (SVD) is the most commonly used to compute $A^\dagger$. For $A \in \mathbb{C}^{m\times n}$, the SVD method is a factorization of the form

$$A = UDV^*,$$

where, U is an $m \times m$ complex unitary matrix, D 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. The diagonal entries of D are known as the singular values of A. Then, the matrix $A^\dagger$ can be written as

$$A^\dagger = VD^\dagger U^*,$$

where, $D^\dagger$ obtained by replacing every nonzero diagonal entry in D by its reciprocal and then transposing the resulting matrix, for more details, see e.g. [3].

This method is very accurate but time intensive since it requires a large amount of computational resources, especially in large matrices. Hence, various numerical methods have been developed to compute Moore-Penrose inverse.

Iterative methods have attracted more attention in recent years, see [2, 4, 12, 13, 7] and the references cited therein. According to the order of convergence, Petkovic and Stanimirovic in [7] developed a first-order iterative algorithm to find Moore-Penrose inverse based on Penrose equations (2) and (4).

The most frequently used iterative method for approximating the inverse of a matrix is the famous Newton's method [10]

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

This method is a second-order iterative methods. Shultz in [10] found that the eigenvalues of $I - AX_0$ must have magnitude less than 1 to ensure the convergence.

Li et al. in [4] investigated the following third-order method, known as the Chebyshev method,

$$X_{k+1} = X_k(3I - AX_k(3I - AX_k)), \quad k = 0, 1, 2, \ldots.$$

Toutounian and Soleymani [14] introduced the following fourth-order method that involves 5 matrix multiplications

$$X_{k+1} = \frac{1}{2}X_k[9I - AX_k(16I - AX_k(14I - AX_k(6I - AX_k)))],$$

$k = 0, 1, 2, \ldots$.

Esmaeili et al. in [2] proposed new fourth-order method to compute the Moore-Penrose inverse as follows

$$X_{k+1} = X_k[9I - 26AX_k + 34(AX_k)^2 - 21(AX_k)^3 + 5(AX_k)^4],$$

$k = 0, 1, 2, \ldots$.

Srivastava and Gupta in [12] introduced a class of higher-order iterative methods using only the Penrose equation (2) by extending the iterative method described in [7]. It is clear that the class of higher order iterative methods [12] coincide with Newton's method if the order is reduced to 2.

In addition, various iterative methods have been developed based on the matrix equation $f(X) = X^{-1} - A = 0$, see e.g. [4, 11].

In this paper we establish a new family of second order iterative methods to compute the Moore-Penrose inverse by using Penrose Equations (1) and (2). These methods are written in terms of *p-th* root of a square matrix AX_k. Then approximations for the *p-th* root of a square matrix is used in computation. A wide set of numerical tests show that these methods require less number of iterations than Newton's method. In addition, numerical experiments show that these methods are more efficient than Newton's method when the number of columns increases than the number of rows. In this case the CPU time of our methods is also less than Newton's method.

This paper is organized as follows. Section 1 is the introduction. In Section 2, our new family of second-order iterative methods for computing the Moore-Penrose inverse is introduced. Some lemmas and convergence analysis of the methods are also established. In Section 3, several numerical examples are given.

2. The iterative method

Let $A \in \mathbb{C}^{m\times n}$ and $X = A^{\dagger} \in \mathbb{C}^{n\times m}$. We use Equations (1) and (2) to obtain

$$X = XAX = X(AXAX)^{\frac{1}{2}} = X(AX)^{\frac{1}{2}}.$$

Hence, we have

$$X = X - 2(X(AX)^{\frac{1}{2}} - X).$$

From the last equation we get the following iterative method

$$X_{k+1} = X_k - 2X_k((AX_k)^{\frac{1}{2}} - I). \tag{1}$$

Assume the starting value for the iterative method (1) is

$$X_0 = \alpha A^*, \tag{2}$$

for an appropriate real number α.

Continuing in a similar manner, this can further be extended to a family of second-order iterative methods, given by

$$X_{k+1} = X_k - pX_k((AX_k)^{\frac{1}{p}} - I), \quad p \in \{2, 3, 4, \dots\}. \tag{3}$$

Lemma 1. *The iterative schemes* (1) *and* (3) *satisfy the following relations:*

$$XAX_k = X_k, \tag{4a}$$
$$X_kAX = X_k, \tag{4b}$$

where $k \geq 0$.

Proof. We use mathematical induction. For $k = 0$ we have $X_0 = \alpha A^*$ and all statements in (4) hold. Assume the statements are true for some integer k. Now we prove the statements for $k + 1$. For (4a), we have

$$\begin{aligned} XAX_{k+1} &= XA(X_k - 2X_k((AX_k)^{\frac{1}{2}} - I)) \\ &= XAX_k - 2XAX_k((AX_k)^{\frac{1}{2}} - I) \\ &= X_k - 2X_k((AX_k)^{\frac{1}{2}} - I) \\ &= X_{k+1}. \end{aligned}$$

We prove (4b) in a similar way

$$\begin{aligned} X_{k+1}AX &= (X_k - 2X_k((AX_k)^{\frac{1}{2}} - I))AX \\ &= X_kAX - 2(X_k(AX_k)^{\frac{1}{2}}AX - X_kAX) \\ &= X_k - 2(X_k(AX_k)^{\frac{1}{2}}(AX)^{\frac{1}{2}} - X_k) \end{aligned}$$

$$\begin{aligned} &= X_k - 2(X_k(AX_kAX)^{\frac{1}{2}} - X_k) \\ &= X_k - 2(X_k(AX_k)^{\frac{1}{2}} - X_k) \\ &= X_{k+1}. \end{aligned}$$

Proceeding in a similar manner, (4) can easily be proved for (3). This completes the proof of the lemma. □

Next, we follow the idea of [2] to prove that the matrix sequence X_k defined by the iterative method (1) and the starting value (2), converges to the Moore-Penrose inverse $X = A^{\dagger}$. The following well-known results are used.

Lemma 2. ([3]) *Let $M \in \mathbb{C}^{n\times n}$ and $\epsilon > 0$ be given. There is at least one matrix norm $\|.\|$ such that*

$$\rho(M) \leq \|M\| \leq \rho(M) + \epsilon,$$

where $\rho(M) = \max\{|\lambda_1(M)|, ..., |\lambda_n(M)|\}$ denotes the spectral radius of M.

Lemma 3. ([13]) *If $P \in \mathbb{C}^{n\times n}$ and $Q \in \mathbb{C}^{n\times n}$ are such that $P = P^2$ and $PQ = QP$, then*

$$\rho(PQ) \leq \rho(Q).$$

Let us consider the following singular value decomposition of the matrix A of $rank(A) = r \leq \min\{m, n\}$:

$$A = U \begin{bmatrix} S & 0 \\ 0 & 0 \end{bmatrix} V^*, \quad S = diag(\sigma_1, ..., \sigma_r), \quad \sigma_1 \geq ... \geq \sigma_r > 0,$$

where σ_i are the singular values of A.

It is well known that

$$A^{\dagger} = V \begin{bmatrix} S^{-1} & 0 \\ 0 & 0 \end{bmatrix} U^*,$$

where U and V are unitary matrices. Using $X_0 = \alpha A^*$, in which α is a constant, we can deduce that each iterate of the method (1) has a singular value decomposition of the form

$$X_k = VS_kU^*, \quad S_k = diag(s_1^{(k)}, ..., s_r^{(k)}),$$

where $S_0 = \alpha S$, we have

$$VS_{k+1}U^* = VS_kU^* - 2VS_kU^*((USS_kU^*)^{\frac{1}{2}} - I)$$

$$= 3VS_kU^* - 2VS_kU^*(USS_kU^*)^{\frac{1}{2}}$$
$$= 3VS_kU^* - 2VS_kU^*U(SS_k)^{\frac{1}{2}}U^*.$$

Hence,

$$S_{k+1} = 3S_k - 2S_k(SS_k)^{\frac{1}{2}}.$$

Therefore, the diagonal matrices $R_k = SS_k = diag(r_1^{(k)}, ..., r_r^{(k)})$ satisfy

$$R_{k+1} = g(R_k) = 3R_k - 2R_k(R_k)^{\frac{1}{2}},$$

that means

$$r_i^{(k+1)} = g(r_i^{(k)}) = 3r_i^{(k)} - 2r_i^{(k)\frac{3}{2}}. \tag{5}$$

In general, for (3) we have

$$S_{k+1} = (p+1)S_k - pS_k(SS_k)^{\frac{1}{p}}.$$

Therefore, the diagonal matrices $R_k = SS_k = diag(r_1^{(k)}, ..., r_r^{(k)})$ satisfy

$$R_{k+1} = g(R_k) = (p+1)R_k - pR_k(R_k)^{\frac{1}{p}},$$

hence

$$r_i^{(k+1)} = g(r_i^{(k)}) = (p+1)r_i^{(k)} - pr_i^{(k)\frac{p+1}{p}}. \tag{6}$$

Theorem 4. *For any initial point* $r^{(0)} \in \left(0, \frac{16}{9}\right)$, *the sequence* $r^{(k+1)} = g(r^{(k)})$ *is second order convergent to* $r = 1$, *in which the function* $g(r)$ *is defined by* (5).

Proof. The fixed points and the critical points of $g(r)$ are:

$$g(r) = r \quad \Rightarrow \quad r = 0, 1,$$

$$g'(r) = 0 \quad \Rightarrow \quad r = 1.$$

We can find that 1 is local maximizer of $g(r)$. It is easy to see that the interval $\left(\frac{4}{9}, \frac{16}{9}\right)$ is mapped into itself.

Moreover, $g(r)$ is a continuous function on the interval $\left(\frac{4}{9}, \frac{16}{9}\right)$, and $|g'(r)| < 1$ on this interval.

We conclude that the sequence $r^{(k+1)} = g(r^{(k)})$ is convergent to $r = 1$, see [1, Page 62]. For the interval $\left(0, \frac{4}{9}\right]$ the sequence $r^{(k+1)} = g(r^{(k)}) > r^{(k)}$, increasing and bounded above, see Figure 1. Hence we obtain convergent for any $r^{(0)} \in \left(0, \frac{16}{9}\right)$. On the other hand,

$$g(1) = 1 \quad g'(1) = 0,$$

implies that the convergence is second order, see [1, Page 81]1. □

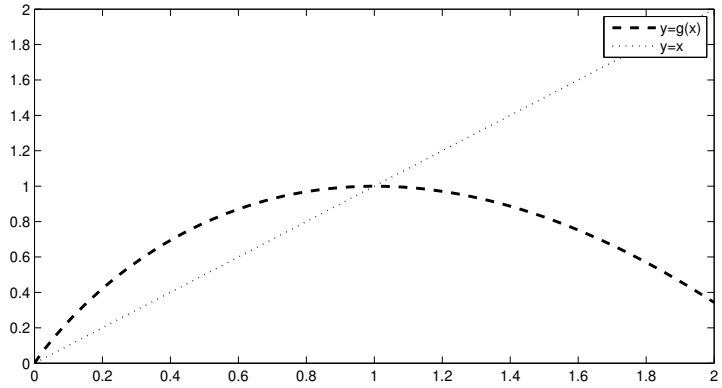

Figure 1: Graph of the function $y = g(x)$ and the line $y = x$.

Considering Theorem 4, we conclude that if $\alpha\sigma_1^2 = r_1^{(0)} \in \left(0, \frac{16}{9}\right)$, where σ_1^2 denotes the largest singular value of A, then $\alpha\sigma_i^2 = r_i^{(0)} \in \left(0, \frac{16}{9}\right)$, for all i, and

$$\lim_{k\to\infty} R_k = I.$$

Hence,

$$\lim_{k\to\infty} S_k = S^{-1},$$

so

$$\lim_{k\to\infty} X_k = V \begin{bmatrix} S^{-1} & 0 \\ 0 & 0 \end{bmatrix} U^* = A^{\dagger}.$$

Hence, the following theorem is proved.

Theorem 5. *Let A be an $m \times n$ nonzero complex matrix. If the initial approximation X_0 is defined by:*

$$X_0 = \alpha A^*, \quad with \;\; 0 < \alpha < \frac{\frac{16}{9}}{\sigma_1^2}, \tag{7}$$

then the iterative method (1) *converges to $A^\dagger$ with second order.*

In addition, we have the following theorem.

Theorem 6. *Let A be an $m \times n$ nonzero complex matrix. If the initial approximation X_0 is defined by:*

$$X_0 = \alpha A^*, \quad with \;\; 0 < \alpha < \frac{\frac{16}{9}}{\sigma_1^2}, \tag{8}$$

then

$$\|A(X - X_0)\| < 1.$$

Proof. Take $P = AX$ and $Q = I - AX_0$. Since $P^2 = P$ and

$$\begin{aligned} PQ = AX - AXAX_0 &= AX - AX_0 \\ &= AX - AX_0AX \\ &= (I - AX_0)AX \\ &= QP, \end{aligned}$$

from Lemma 3 we can conclude that

$$\begin{aligned} \rho(A(X - X_0)) \leq \rho(I - AX_0) &= \rho(I - \alpha AA^*) \\ &= \max_{1 \leq i \leq r} |1 - \alpha \lambda_i(AA^*)| \\ &= \max_{1 \leq i \leq r} |1 - \alpha \sigma_i^2|. \end{aligned}$$

By using (7), we conclude that

$$\rho(A(X - \alpha A^*)) \leq \max_{1 \leq i \leq r} |1 - \alpha \sigma_i^2| < 1.$$

Then from Lemma 2, we have

$$\|A(X - X_0)\| < 1.$$

□

The results of Theorem 4 are extended in the next theorem.

Theorem 7. *For any initial point* $r^{(0)} \in \left(\left(0, \left(\frac{p+2}{p+1}\right)^p\right)\right.$, *the sequence* $r^{(k+1)} = g(r^{(k)})$ *is second order convergent to* $r = 1$, *in which the function* $g(r)$ *is defined by* (6).

Proof. The proof is similar to that of Theorem 4. The general behaviour of $g(r)$ defined in (6) is similar to the case when $p = 2$. See Figure 2 which is the graph of (6) when $p = 3$, $p = 6$ and $p = 12$. □

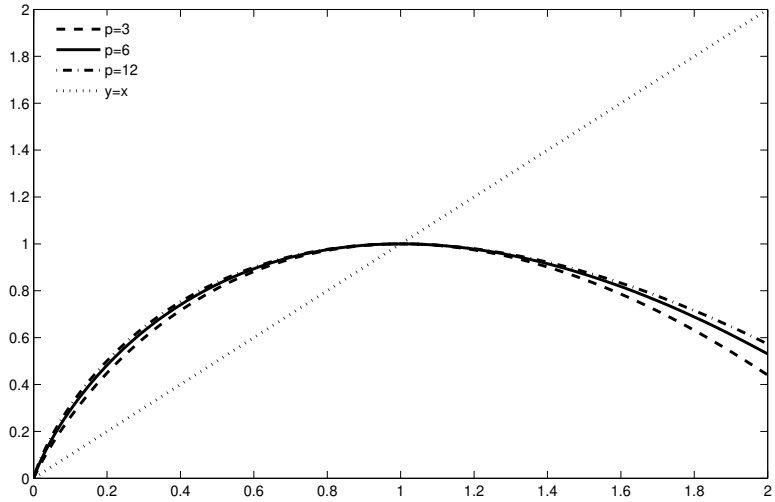

Figure 2: Graphs of the function $y = g(x)$ with different values of p and the line $y = x$.

Our method uses the *p-th* root of a square matrix AX_k. One can find several algorithms to compute this. In this work, we replace the *p-th* root by finite terms from power series expansion for a matrix of form $(I + B)^{\frac{1}{p}}$ which is given in the next remark.

Remark 8. Let $p \geq 2$ be an integer and $\|B\| < 1$, the power series expansion can be applied to define the matrix *p-th* root of the matrix $(I + B)$

as

$$(I+B)^{\frac{1}{p}} = \sum_{n=0}^{\infty} \frac{\frac{1}{p}\left(\frac{1}{p}-1\right)\dots\left(\frac{1}{p}-n+1\right)}{n!} B^n, \tag{9}$$

see e.g. [5].

By approximating $(AX_k)^{\frac{1}{2}}$ or $(I+(AX_k-I))^{\frac{1}{2}}$ with n terms of (9) we obtain the following iterative method

$$X_{k+1} = X_k - 2X_k\left(\frac{1}{2}(AX_k-I) - \frac{1}{8}(AX_k-I)^2 + \dots + \frac{1}{n!}\frac{1}{2}\left(\frac{1}{2}-1\right)\dots\left(\frac{1}{2}-n+1\right)(AX_k-I)^n\right). \tag{10}$$

With the starting value $X_0 = \alpha A^*$, where $0 < \alpha < \frac{\frac{16}{9}}{\sigma_1^2}$, then $\rho(I-AX_0) < 1$.

In the following section, several examples are given to show the efficient of our method. We use (10) up to $n = 2$. We find that if we use more terms in (10) the number of iteration decreases. But, after $n = 4$ the number of iteration is fixed.

3. Numerical results

In this section, numerical examples are worked out to demonstrate the efficacy of our second-order method by using MATLAB program, we use Matlab (R2013b).

We used the maximal residual norm criterion as in [12], for a tolerance $\epsilon = 10^{-8}$,

$$res(X) = \max\{\|AX_kA - A\|_F, \|X_kAX_k - X_k\|_F, \|(AX_k)^* - AX_k\|_F, \|(X_kA)^* - X_kA\|_F\} \le \epsilon,$$

where $\|.\|_F$ the Frobenius norm of a matrix.

Example 1. Consider the ill-conditional Hilbert matrix A of order (5×5) given by

$$A = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{bmatrix}.$$

The choice $\alpha = 0.8$ satisfies the convergence criteria given by

$$\max_{1 \le i \le 3} |1 - \alpha \lambda_i (A^* A)| = 0.9999 < 1,$$

since the eigenvalues of the matrix A^*A are

$$(\lambda_1, \lambda_2, \lambda_3) = (2.4556, 0.0435, 0.0001).$$

The iterative method (1) generates a sequence of iterates $\{X_k\}$ after 39 steps converging to the Moore-Penrose inverse $A^\dagger$ given by

$$A^\dagger = \begin{bmatrix} 25 & -300 & 1050 & -1400 & 630 \\ -300 & 4800 & -18900 & 26880 & -12600 \\ 1050 & -18900 & 79380 & -117600 & 56700 \\ -1400 & 26880 & -117600 & 179200 & -88200 \\ 630 & -12600 & 56700 & -88200 & 44100 \end{bmatrix}.$$

While the Newton method needs 42 iterations to have the same result.

Example 2. For the ill-conditional Hilbert matrix A of order (5×5) the iterative methods (3) are used for different values of p. The comparison of number of iterations are plotted in Figure 6. We note that for $p \ge 10$ the number of required iteration still fixed.

Example 3. We compute the inverse random square matrix A, where A are randomly generated as follows

$$A = 20r \text{ and } (600 + n, 600 + n) - 10r \text{ and } (600 + n, 600 + n),$$

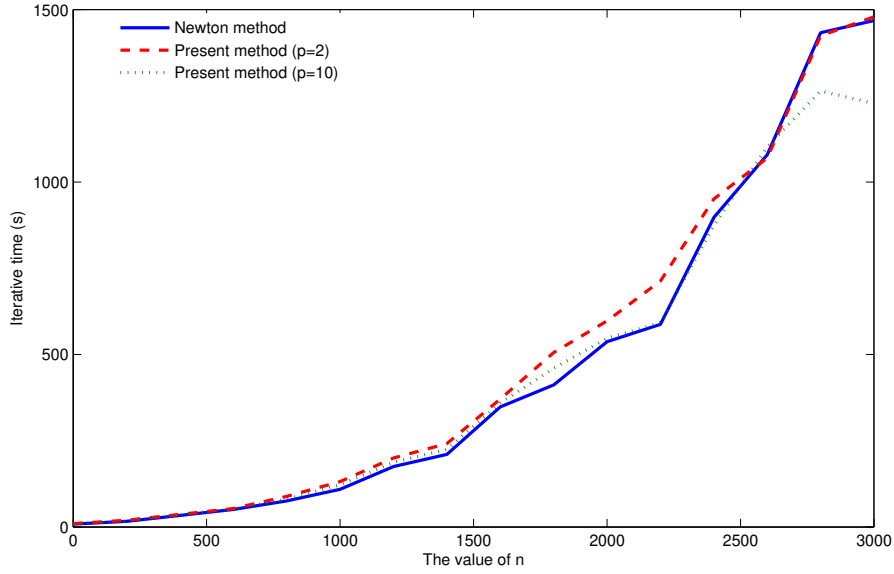

Figure 3: The results of comparisons in terms of computational time, Example 3.

and the value of n is $n = 0, 100, 200, 300, 400, \ldots, 3000$.

The number of iterations and the CPU time required for convergence are compared in Figures 7 and 3, respectively.

We see that the required number of iterations for the current method is less than that of Newton's method. But the computational time is almost the same when $p = 10$.

We noted that for the matrices $A_{m\times n}$ with $m < n$ the current methods also require less time. Next example illustrates this idea.

Example 4. We compute the inverse random square matrix A, where A are randomly generated as follows

$$A = 20r \text{ and } (500, 1000 + n) - 10r \text{ and } (500, 1000 + n),$$

and the value of n is $n = 0, 100, 200, 300, 400, \ldots, 3000$.

The number of iterations and the CPU time required for convergence are compared in Figures 4 and 5, respectively.

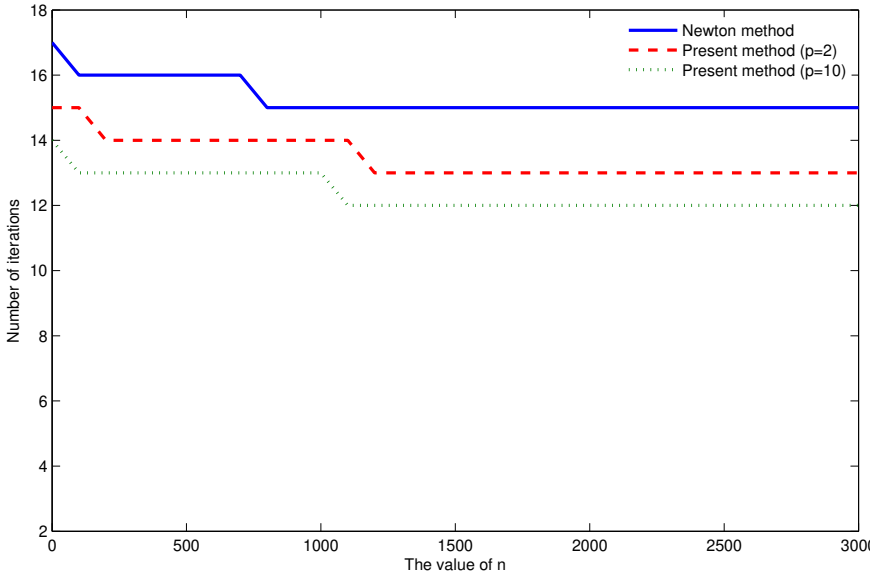

Figure 4: Comparison number of iteration, Example 4.

Figure 5 shows that as the number of columns become larger than the number of rows, the required computational times for current methods become smaller than that of the Newton method.

4. Conclusion

A family of second-order iterative methods were developed based on Penrose equations (1) and (2) and written interms of $p-th$ root of matrix AX_k. Convergence properties were considered and numerical tests were made. Numerical results show that the number of iterations of current methods always less than that of Newton's method. Also, it is observed that the CPU time compared with Newton's method decreases when the number of columns is larger than the number of rows, this makes the current methods more efficient for such cases.

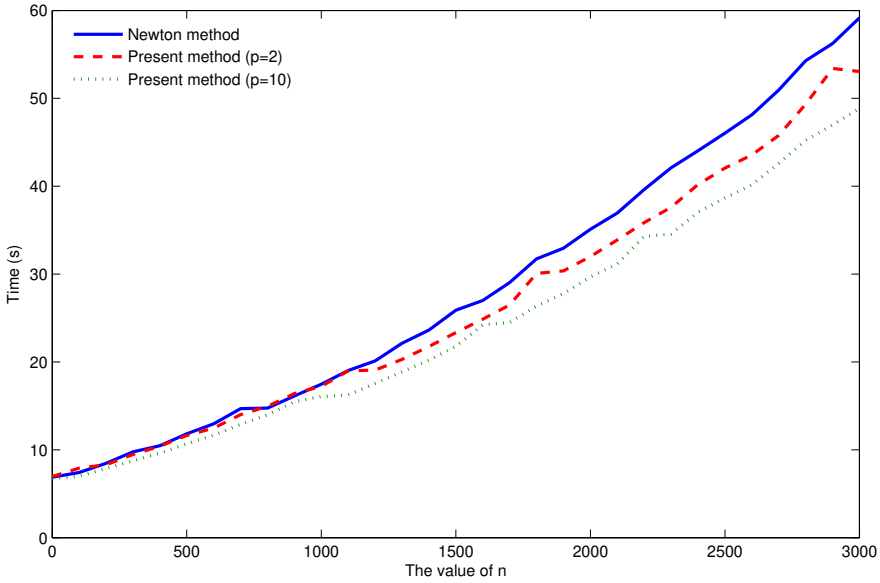

Figure 5: The results of comparisons in terms of computational time, Example 4.

References

[1] R. Burden, J. Faires, *Numerical Analysis*, 9th Ed. Brooks/Cole, Cengage Learning, Boston (2011).

[2] H. Esmaeili, R. Erfanifar and M. Rashidi, A Fourth-order iterative method for computing the Moore-Penrose inverse, *Journal of Hyperstructures*, **6**, No 1 (2017), 52-67.

[3] R. Kress, *Numerical Analysis*, Springer-Verlag, New York (1998).

[4] H. Li, T. Huang, Y. Zhang, X. Liu, and T. Gu, Chebyshev-type methods and preconditioning techniques, *Applied Mathematics and Computation*, **218** (2011), 260-270.

[5] J. A. Marrero, R. Ben Taher, Y. El Khatabi, and M. Rachidi, On explicit formulas of the principal matrix pth root by polynomial decompositions, *Applied Mathematics and Computation*, **242** (2014), 435-443.

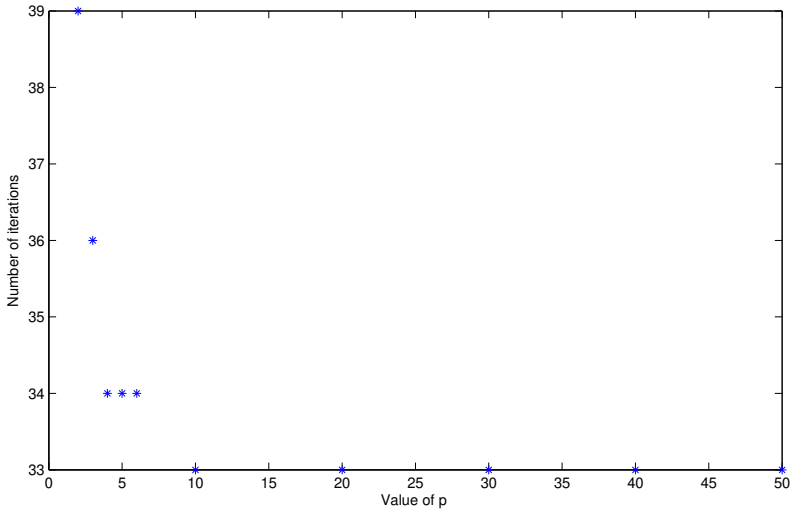

Figure 6: Number of iterations versus the value of p.

[6] S. Miljkovic, M. Miladinovic, P. Stanimirovic, I. Stojanovic, Application of the pseudo-inverse computation in reconstruction of blurred images, *Filomat*, **26** (2012), 453-465.

[7] M. Petkovic, and P. Stanimirovic, Iterative method for computing theMoore-Penrose inverse based on Penrose equations, *Journal of Computational and Applied Mathematics*, **235** (2011), 1604-1613.

[8] R. Potthast and P. b. Graben, Inverse problems in neural field theory, *SIAM Journal on Applied Dynamical Systems*, **8** (2009), 1405-1433.

[9] L. Sciavicco, B. Siciliano, *Modelling and Control of Robot Manipulators*, Springer-Verlag, London (2000).

[10] G. Shultz, iterative Berechmmg der reziproken matrix, *Z. Angew. Math. Mech.*, **13** (1933), 57-59.

[11] F. Soleymani, M. Sharifi, and S. Shateyi, Approximating the inverse of a square matrix with application in computation of the Moore-Penrose Inverse, *Journal of Applied Mathematics*, **2014** (2014), 2-8.

[12] S. Srivastava, D.K. Gupta, Higher order iterations for Moore-Penrose in-

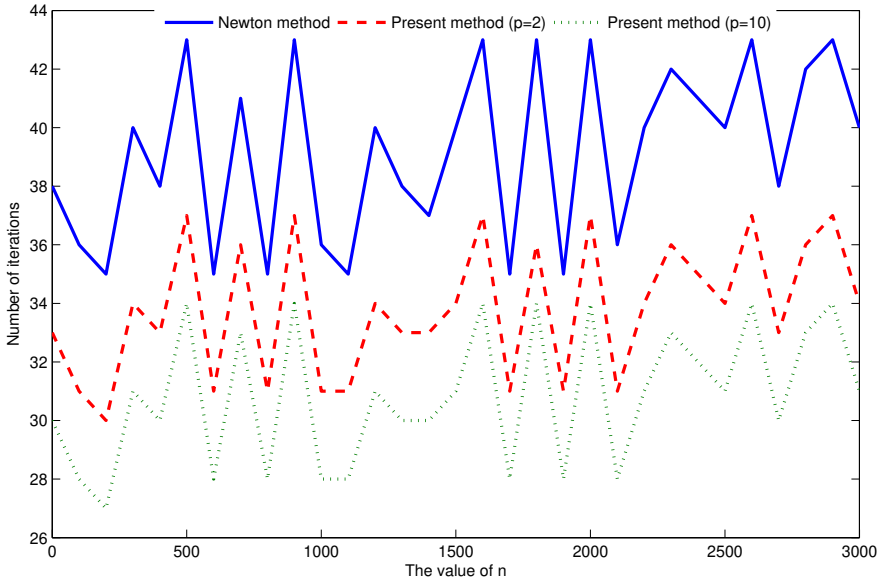

Figure 7: Comparison number of iteration, Example 3.

verses, *Journal of Applied Mathematics and Informatics*, **32** (2014), 171-184.

[13] P. Stanimirovic, D. Cvetkovic-Ilic, Successive matrix squaring algorithm for computing outer inverses, *Applied Mathematics and Computation*, **203** (2008), 19-29.

[14] F. Toutounian, F. Soleymani, An iterative method for computing the approximate inverse of a square matrix and the Moore-Penrose inverse of a non-square matrix, *Applied Mathematics and Computation*, **224** (2013), 671-680.