Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2014, Article ID 731562, 8 pages
http://dx.doi.org/10.1155/2014/731562

Research Article

Approximating the Inverse of a Square Matrix with Application in Computation of the Moore-Penrose Inverse

F. Soleymani,[1] M. Sharifi,[1] and S. Shateyi[2]

[1] *Department of Mathematics, Zahedan Branch, Islamic Azad University, Zahedan, Iran*
[2] *Department of Mathematics and Applied Mathematics, School of Mathematical and Natural Sciences, University of Venda, Private Bag X5050, Thohoyandou 0950, South Africa*

Correspondence should be addressed to S. Shateyi; stanford.shateyi@univen.ac.za

Received 3 January 2014; Revised 13 February 2014; Accepted 27 February 2014; Published 31 March 2014

Academic Editor: Juan R. Torregrosa

This paper presents a computational iterative method to find approximate inverses for the inverse of matrices. Analysis of convergence reveals that the method reaches ninth-order convergence. The extension of the proposed iterative method for computing Moore-Penrose inverse is furnished. Numerical results including the comparisons with different existing methods of the same type in the literature will also be presented to manifest the superiority of the new algorithm in finding approximate inverses.

1. Introduction

The main purpose of this paper is to present an efficient method in terms of speed of convergence, while its convergence can be easily achieved and also be economic for large sparse matrices possessing sparse inverses. We also discuss the extension of the new scheme for finding the Moore-Penrose inverse of singular and rectangular matrices.

Finding a matrix inverse is important in some practical applications such as finding a rational approximation for the Fermi-Dirac functions in the density functional theory [1], because it conveys significant features of the problems dealing with.

We further mention that in some certain circumstances, the computation of a matrix inverse is necessary. For example, there are many ways to encrypt a message, whereas the use of coding has become particularly significant in recent years. One way to encrypt or code a message is to use matrices and their inverses. Indeed, consider a fixed invertible matrix A. Convert the message into a matrix B such that AB is possible to perform. Send the message generated by AB. At the other end, they will need to know A^{-1} in order to decrypt or decode the message sent.

The direct methods such as Gaussian elimination with partial pivoting or LU decomposition require a reasonable time to compute the inverse when the size of the matrices is high. To illustrate further, the Gaussian elimination with partial pivoting method cannot be highly parallelized and this restricts its applicability in some cases. In contrary, Schulz-type methods, which could be applied for large sparse matrices (possessing sparse inverses [2]) by preserving the sparsity feature and can be parallelized, are in focus in such cases.

Some known methods were proposed for the approximation of a matrix inverse, such as Schulz scheme. The oldest scheme, that is, the Schulz method (see [3]), is defined as

$$V_{n+1} = V_n\left(2I - AV_n\right), \quad n = 0, 1, 2, \ldots, \tag{1}$$

wherein I is the identity matrix. Note that [4] mentions that Newton-Schulz iterations can also be combined with wavelets or hierarchical matrices to compute the diagonal elements of A^{-1} independently.

The proposed iteration relies on matrix multiplications, which destroy sparsity, and therefore the Schulz-type methods are less efficient for sparse inputs possessing dense inverses. However, a numerical dropping is usually applied to V_{n+1} to keep the approximated inverse sparse. Such a strategy is useful for preconditioning.

To provide more iterations from this classification of methods, we remember that W. Li and Z. Li in [5] proposed

$$V_{n+1} = V_n\left(3I - 3AV_n + (AV_n)^2\right), \quad n = 0, 1, 2, \ldots. \tag{2}$$

In 2011, Li et al. in [6] represented (2) in the following form:

$$V_{n+1} = V_n\left(3I - AV_n\left(3I - AV_n\right)\right), \quad n = 0, 1, 2, \ldots \tag{3}$$

and also proposed another iterative method for finding A^{-1} as comes next:

$$V_{n+1} = \left[I + \frac{1}{4}\left(I - V_nA\right)\left(3I - V_nA\right)^2\right]V_n, \quad n = 0, 1, 2, \ldots. \tag{4}$$

Much of the application of these solvers (specially the Schulz method) for structured matrices was investigated in [7] while the authors in [8] showed that the matrix iteration of Schulz is numerically stable. Further discussion about such iterative schemes might be found at [9–12].

The Schulz-type matrix iterations are dependent on the initial matrix V_0 too much. In general, we construct the initial guess for square matrices as follows. For a general matrix $A^{m\times m}$ satisfying no structure, we choose as in [13]

$$V_0 = \frac{A^T}{m\|A\|_1\|A\|_\infty}, \tag{5}$$

or $V_0 = \alpha I$, where I is the identity matrix, and $\alpha \in \mathbb{R}$ should adaptively be determined such that $\|I - \alpha A\| < 1$. For diagonally dominant (DD) matrices, we choose as in [14] $V_0 = \mathrm{diag}(1/a_{11}, 1/a_{22}, \ldots, 1/a_{mm})$, wherein a_{ii} is the diagonal entry of A. And for a symmetric positive definite matrix and by using [15], we choose $V_0 = (1/\|A\|_F)I$, whereas $\|\cdot\|_F$ is the Frobenius norm.

For rectangular or singular matrices, one may choose $V_0 = A^*/(\|A\|_1\|A\|_\infty)$ or $V_0 = A^T/(\|A\|_1\|A\|_\infty)$, based on [16]. Also, we could choose $V_0 := \alpha A^T$, where $0 < \alpha < 2/\rho(A^TA)$ or

$$V_0 = \alpha A^*, \tag{6}$$

with $0 < \alpha < 2/\rho(A^*A)$ [17]. Note that these choices could also be considered for square matrices. However, the most efficient way for producing V_0 (for square nonsingular matrices) is the hybrid approach presented in Algorithm 1 of [18].

The rest of this paper is organized as follows. The main contributions of this article are given in Sections 2 and 3. In Section 2, we analyze a new scheme for matrix inversion while a discussion about the applicability of the scheme for Moore-Penrose inverse will be given in Section 3. Section 4 will discuss the performance and the efficiency of the new proposed method numerically in comparison with the other schemes. Finally, concluding remarks are presented in Section 5.

2. Main Result

The derivation of different Schulz-type methods for matrix inversion relies on iterative (one- or multipoint) methods for the solution of nonlinear equations [19, 20]. For instance, imposing Newton's iteration on the matrix equation $AV = I$ would result in (1), as fully discussed in [21]. Here, we apply the following *new* nonlinear solver on the matrix equation $f(V) = V^{-1} - A = 0$:

$$\begin{aligned} Y_n &= V_n - f'(V_n)^{-1}f(V_n) - 2^{-1} \\ &\quad\times\left(f'(V_n)^{-1}f''(V_n)\right)\left(f'(V_n)^{-1}f(V_n)\right)^2, \\ Z_n &= Y_n - 2^{-1}f'(Y_n)^{-1}f(Y_n), \\ V_{n+1} &= Y_n - f'(Z_n)^{-1}f(Y_n), \quad n = 0, 1, 2, \ldots. \end{aligned} \tag{7}$$

Note that (7) is novel and constructed in such a way to produce a new matrix iterative method for finding generalized inverses efficiently. Now, the following iteration method for matrix inversion could be obtained:

$$\begin{aligned} V_{n+1} &= \frac{1}{4}V_n\left(3I + AV_n\left(-3I + AV_n\right)\right) \\ &\quad\times\left(4I - \left(-I + AV_n\right)^3\right. \\ &\quad\left.\times\left(-3I + AV_n\left(3I + AV_n\left(-3I + AV_n\right)\right)\right)^2\right), \\ &\qquad n = 0, 1, 2, \ldots. \end{aligned} \tag{8}$$

The obtained scheme includes matrix power, which is certainly of too much cost. To remedy this, we rewrite the obtained iteration as efficiently as possible to also reduce the number of matrix-matrix multiplications in what follows:

$$\begin{aligned} \psi_n &= AV_n, \\ \zeta_n &= 3I + \psi_n\left(-3I + \psi_n\right), \\ \upsilon_n &= \psi_n\zeta_n, \\ V_{n+1} &= -\frac{1}{4}V_n\zeta_n\left(-13I + \upsilon_n\left(15I + \upsilon_n\left(-7I + \upsilon_n\right)\right)\right), \\ &\qquad n = 0, 1, 2, \ldots, \end{aligned} \tag{9}$$

whereas I is the identity matrix and the sequence of matrix iterates $\{V_n\}_{n=0}^{n=\infty}$ converges to A^{-1} for a good initial guess.

It should be remarked that the convergence of any order for nonsingular square matrices is generated in Section 6 of Chapter 2 of [22], whereas the general way for the rectangular matrices was discussed in Chapter 5 of [23] and the recent paper of [24]. In those constructions, always a convergence order ρ will be attained by ρ times of matrix-matrix products, such as (1) which reaches the order 2 using two matrix-matrix multiplications.

Two important matters must be mentioned at this moment to ease up the perception of why a higher order (efficient) method such as (9) with 7 matrix-matrix products

to reach at least the convergence order 9 is practical. First, by following the comparative index of informational efficiency for inverse finders [25], defined by $E = \rho/\theta$, wherein ρ and θ stand for the convergence order and the number of matrix-matrix products, then the informational index for (9), that is, $9/7 \approx 1.28$, beats its other competitors, $2/2 = 1$ of (1), $3/3 = 1$ of (2)-(3), and $3/4 = 0.75$ of (4). And second, the significance of the new scheme will be displayed in its implementation. To illustrate further, such iterations depend totally on the initial matrices. Though there are certain and efficient ways for finding V_0, in general such initial approximations take a *high* number of iterations (see e.g., Figure 3, the blue color) to arrive at the convergence phase. On the other hand, each cycle of the implementation of such Schulz-type methods includes one stopping criterion based on the use of a matrix norm, and this would impose further burden and load in general, for the low order methods in contrast to the high order methods, such as (9). Because the computation of a matrix norm (usually $\|\cdot\|_2$ for dense complex matrices and $\|\cdot\|_F$ for large sparse matrices) takes time and therefore higher number of steps/iterations (which is the result of lower order methods), it will be costlier than the lower number of steps/iterations of high order methods. Hence, the higher order (efficient) iterations are mostly better solvers in terms of computational time to achieve the desired accuracy.

After this complete discussion of the need and efficacy of the new scheme (9), we are about to prove the convergence behavior of (9) theoretically in what follows.

Theorem 1. *Let $A = [a_{ij}]^{m\times m}$ be a nonsingular complex or real matrix. If the initial approximation V_0 satisfies*

$$\|I - AV_0\| < 1, \tag{10}$$

then the iterative method (9) *converges with at least ninth order to A^{-1}.*

Proof. Let (10) hold, and for the sake of simplicity assume that $E_0 = I - AV_0$ and $E_n = I - AV_n$. It is straightforward to have

$$E_{n+1} = I - AV_{n+1} = \frac{1}{4}\left[3E_n^9 + E_n^{10}\right]. \tag{11}$$

Hence, by taking an arbitrary norm from both sides of (11), we obtain

$$\|E_{n+1}\| \le \frac{1}{4}\left(3\|E_n\|^9 + \|E_n\|^{10}\right). \tag{12}$$

The rest of the proof is similar to Theorem 2.1 of [26], and it is hence omitted. Finally, $I - AV_n \to 0$, when $n \to \infty$, and thus $V_n \to A^{-1}$, as $n \to \infty$. So the sequence $\{\|E_n\|_2\}$ is strictly monotonic decreasing. Moreover, if we denote $\epsilon_n = V_n - A^{-1}$, as the error matrix in the iterative procedure (9), then we could easily obtain the following error inequality:

$$\|\epsilon_{n+1}\| \le \left(\frac{1}{4}\left[3\|A\|^8 + \|A\|^9\|\epsilon_n\|\right]\right)\|\epsilon_n\|^9. \tag{13}$$

The error inequality (13) reveals that the iteration (9) converges with at least ninth order to A^{-1}. The proof is now complete. □

Some features of the new scheme (9) are as follows.

(1) The new method can be taken into consideration for finding approximate inverse (preconditioners) of complex matrices. Using a dropping strategy, we could keep the sparsity of a matrix to be used as a preconditioner. Such an action will be used throughout this paper for sparse matrices using the command Chop[exp, tolerance] in Mathematica, to preserve the sparsity of the approximate inverses V_i, for any i.

(2) Unlike the traditional direct solvers such as GEPP, the new scheme has the ability to be parallelized.

(3) In order to further reduce the computational burden of the matrix-by-matrix multiplications per computing step of the new algorithm when dealing with sparse matrices, in the new scheme by using Sparse Array[] command in Mathematica, we will preserve the sparsity features of the output approximate inverse in a reasonable computational time. Additionally, for some special matrices, such as Toeplitz or Vandermonde types of matrices, further computational savings are possible as discussed by Pan in [27]. See for more information [28].

(4) If the initial guess commutes with the original matrix, then all matrices in the sequence $\{V_n\}_{n=0}^{n=\infty}$ satisfy the same property of commutation. Notice that only some initial matrices guarantee that in all cases the sequence commutes with the matrix A.

In the next section, we present some analytical discussion about the fact that the new method (9) could also be used for computing the Moore-Penrose generalized inverse.

3. An Iterative Method for Moore-Penrose Inverse

It is well known in the literature that the Moore-Penrose inverse of a complex matrix $A \in \mathbb{C}^{m\times k}$ (also called pseudo-inverse), denoted by $A^\dagger \in \mathbb{C}^{k\times m}$, is a matrix $V \in \mathbb{C}^{k\times m}$ satisfying the following conditions:

$$\begin{aligned} AVA &= A, \qquad VAV = V, \\ (AV)^* &= AV, \qquad (VA)^* = VA, \end{aligned} \tag{14}$$

wherein A^* is the conjugate transpose of A. This inverse uniquely exists.

Various numerical solution methods have been developed for approximating Moore-Penrose inverse (see e.g., [29]). The most comprehensive review of such iterations could be found in [17], while the authors mentioned that iterative Schulz-type methods can be taken into account for finding pseudoinverse. For example, it is known that (1) converges to the pseudoinverse in the general case if $V_0 := \alpha A^*$, where $0 < \alpha < 2/\rho(A^*A)$ and $\rho(\cdot)$ denotes the spectral radius.

Accordingly, it is known that the new scheme (9) converges to the Moore-Penrose inverse. In order validate this

```
n = 30000;
A = SparseArray[{Band[{195, 10000}] -> −I, Band[{1, 1}] -> 19.,
  Band[{1000, 2500}] -> 2.1, Band[{−60, −1800}] -> 1.1,
  Band[{−600, 170}] -> 2. + I, Band[{−1350, 250}] -> −5.3},
   {n, n}, 0.]
```

Algorithm 1

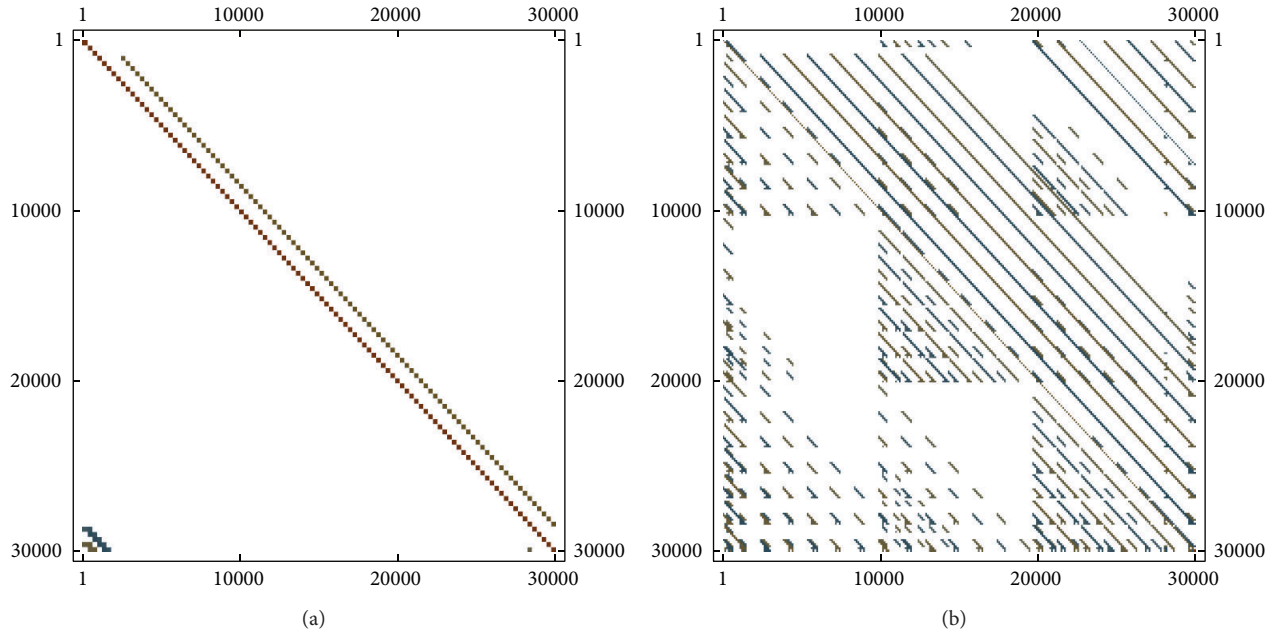

Figure 1: The matrix plot (a) and the inverse (b) of the large complex sparse 30000 × 30000 matrix A in Example 7.

fact analytically, we provide some important theoretics in what follows.

Lemma 2. *For the sequence* $\{V_n\}_{n=0}^{n=\infty}$ *generated by the iterative Schulz-type method* (9) *and the initial matrix* (6)*, for any* $n \geq 0$*, it holds that*

$$\begin{aligned} \left(AV_n\right)^* &= AV_n, \qquad \left(V_nA\right)^* = V_nA, \\ V_nAA^\dagger &= V_n, \qquad A^\dagger AV_n = V_n. \end{aligned} \tag{15}$$

Proof. The proof of this lemma is based on mathematical induction. Such a procedure is similar to Lemma 3.1 of [30], and it is hence omitted. □

Theorem 3. *For the rectangular complex matrix* $A \in \mathbb{C}^{m\times k}$ *and the sequence* $\{V_n\}_{n=0}^{n=\infty}$ *generated by* (9)*, for any* $n \geq 0$*, using the initial approximation* (6)*, the sequence converges to the pseudoinverse* $A^\dagger$ *with ninth order of convergence.*

Proof. We consider $\mathbb{E}_n = V_n - A^\dagger$, as the error matrix for finding the Moore-Penrose inverse. Then, the proof is similar to the proof of Theorem 4 in [18] and it is hence omitted. We finally have

$$\left\|\mathbb{E}_{n+1}\right\| \leq \left\|A^\dagger\right\| \|A\|^9 \left\|\mathbb{E}_n\right\|^9. \tag{16}$$

Thus, $\|V_n - A^\dagger\| \to 0$; that is, the obtained sequence of (9) converges to the Moore-Penrose inverse as $n \to +\infty$. This ends the proof. □

Theorem 4. *Considering the same assumptions as in Theorem 3, then the iterative method* (9) *has asymptotical stability for finding the Moore-Penrose generalized inverse.*

Proof. The steps of proving the asymptotical stability of (9) are similar to those which have recently been taken for a general family of methods in [31]. Hence, the proof is omitted. □

Remark 5. It should be remarked that the generalization of our proposed scheme for generalized outer inverses, that is, $A_{T,S}^{(2)}$, is straight forward according to the recent work [32].

Remark 6. The new iteration (9) is free from matrix power in its implementation and this allows one to apply it for finding generalized inverses easily.

4. Computational Experiments

Using the computer algebra system Mathematica 8 [33], we now compare our iterative algorithm with other Schulz-type methods.

Table 1: Results of comparisons for Example 7.

	Methods			
	(1)	(3)	(4)	(9)
Convergence rate	2	3	3	7
Iteration number	3	2	2	1
Residual norm	8.32717×10^{-7}	1.21303×10^{-7}	5.10014×10^{-8}	9.7105×10^{-8}
No. of nonzero ele.	591107	720849	800689	762847
Time (in seconds)	4.22	4.05	9.32	4.18

```
Id = SparseArray[{{i_, i_} -> 1.}, {n, n}];
V = DiagonalMatrix@SparseArray[1/Normal[Diagonal[A]]];
Do[V1 = SparseArray[V];
   V2 = Chop[A.V1]; V3 = 3 Id + V2.(-3 Id + V2);
   V4 = SparseArray[V2.V3];
   V = Chop[-(1/4) V1.V3.SparseArray[-13 Id
   + V4.(15 Id + V4.(-7 Id + V4))]];
   Print[V];L[i]= N[Norm[Id - V.A, 1]];
   Print["The residual norm is:"
     Column[{i}, Frame -> All, FrameStyle -> Directive[Blue]]
     Column[{L[i]}, Frame -> All, FrameStyle -> Directive[Blue]]];
   , {i, 1}]; //AbsoluteTiming
```

Algorithm 2

```
n = 10000;
A = SparseArray[{Band[{-700, -200}] -> 1., Band[{1, 1}] -> -1.5,
   Band[{1, -400}] -> .9, Band[{1, -400}] -> -1.,
   Band[{2000, 200}] -> 1.}, {n, n}, 0.]
```

Algorithm 3

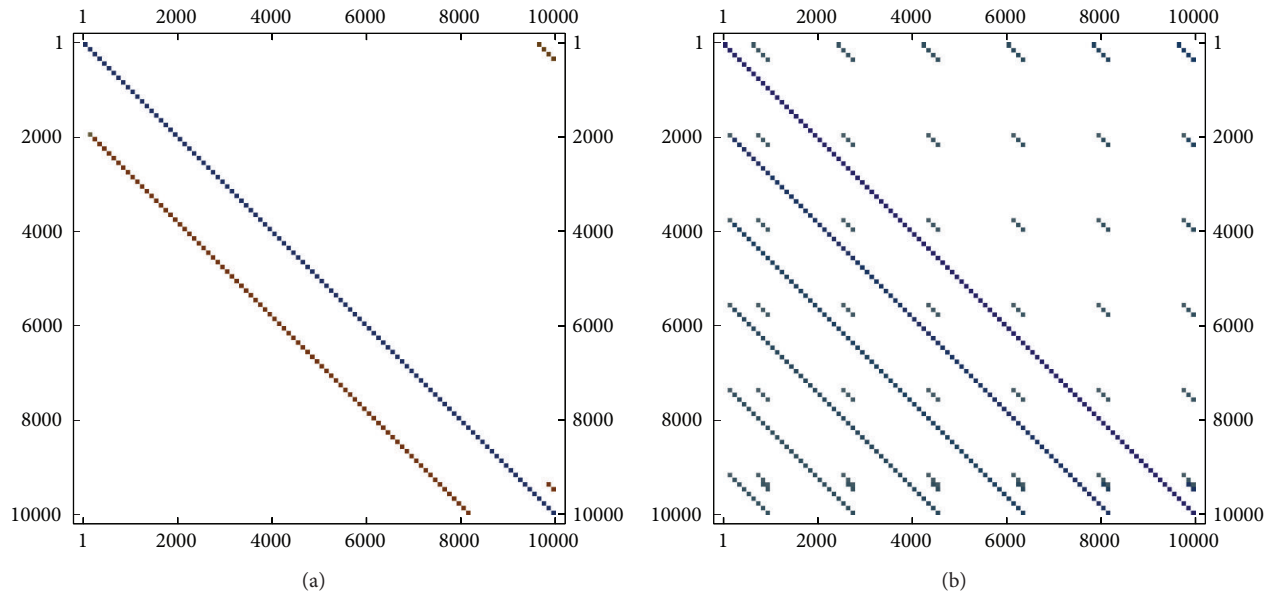

Figure 2: The matrix plot (a) and the inverse (b) of the large sparse 10000×10000 matrix A in Example 8.

Table 2: Results of comparisons for Example 8.

	Methods			
	(1)	(3)	(4)	(9)
Convergence rate	2	3	3	7
Iteration number	10	7	6	3
Residual norm	1.29011×10^{-10}	1.63299×10^{-9}	1.18619×10^{-11}	7.68192×10^{-10}
No. of nonzero ele.	41635	42340	41635	41635
Time (in seconds)	2.23	2.15	2.26	1.95

Table 3: Results of elapsed computational time for solving Example 9.

Methods	GMRES	PGMRES-(1)	PGMRES-(3)	PGMRES-(4)	PGMRES-(9)
The total time	0.29	0.14	0.14	0.09	0.07

For numerical comparisons in this section, we have used the methods (1), (3), (4), and (9). We implement the above algorithms on the following examples using the built-in precision in Mathematica 8.

Example 7. Let us consider the large complex sparse matrix of the size 30000 × 30000 with 79512 nonzero elements possessing a sparse inverse as in Algorithm 1 ($I = \sqrt{-1}$), where the matrix plot of A showing its structure has been drawn in Figure 1(a).

Methods like (9) are powerful in finding an approximate inverse or in finding robust approximate inverse preconditioners in low number of steps, in which the output form of the approximate inverse is also *sparse*.

In this test problem, the stopping criterion is $\|I - V_n A\|_1 \leq 10^{-7}$. Table 1 shows the number of iterations for different methods. As the programs were running, we found the running time using the command Absolute Timing[] to report the elapsed CPU time (in seconds) for this experiment. Note that the initial guess has been constructed using $V_0 = \text{diag}(1/a_{11}, 1/a_{22}, \ldots, 1/a_{mm})$ in Example 7. The plot of the approximate inverse obtained by applying (9) has been portrayed in Figure 1(b). The reason which made (4) the worse method is in the fact of computing matrix power which is time consuming for sparse matrices. In the tables of this paper "No. of nonzero ele." stands for the number of nonzero elements.

In Algorithm 2, we provide the written Mathematica 8 code of the new scheme (9) in this example to clearly reveals the simplicity of the new scheme in finding approximate inverses using a threshold `Chop`[exp, 10^{-10}].

Example 8. Let us consider a large sparse 10000 × 10000 matrix (with 18601 nonzero elements) as shown in Algorithm 3.

Table 2 shows the number of iterations for different methods in order to reveal the efficiency of the proposed iteration with a special attention to the elapsed time. The matrix plot of A in this case showing its structure alongside its approximate inverse has been drawn in Figure 2, respectively. Note that in Example 8, we have used an initial value constructed by $V_0 = A^*/(\|A\|_1 \|A\|_\infty)$. The stopping criterion is same as the previous example.

The new method (9) is totally better in terms of the number of iterations and the computational time. In this paper, the computer specifications are Microsoft Windows XP Intel(R), Pentium(R) 4, CPU 3.20 GHz, and 4 GB of RAM.

Matrices used up to now are large and sparse enough to clarify the applicability of the new scheme. Anyhow, for some classes of problems, there are collections of matrices that are used when proving new iterative linear system solvers or new preconditioners, such as Matrix Market, http://math.nist.gov/MatrixMarket/. In the following test we compare the computational time of solving a practical problem using the above source.

Example 9. Consider the linear system of $Ax = b$, in which A is defined by A = Example Data ["Matrix", "Bai/pde900"] and the right hand side vector is b = Table [1., {i, 1, 900}]. In Table 3, we compare the consuming time (in seconds) of solving this system by GMRES solver and its left preconditioned system $V_1 Ax = V_1 b$. In Table 3, for example, PGMRES-(9) shows that the linear system $V_1 Ax = V_1 b$, in which V_1 has been calculated by (9), is solved by GMRES solver. The tolerance is the residual norm to be less than 10^{-14} in this test. The results support fully the efficiency of the new scheme (9) in constructing robust preconditioners.

Example 10. This experiment evaluates the applicability of the new method for finding Moore-Penrose inverse of 20 random sparse matrices (possessing sparse pseudo-inverses) $m \times k = 1200 \times 1500$ as shown in Algorithm 4.

Note that the identity matrix should then be an $m \times m$ matrix and the approximate pseudoinverses would be of the size $k \times m$. In this example, the initial approximate Moore-Penrose inverse is constructed by $\mathtt{V_0}$ = `ConjugateTranspose [A[j]] * (1./((Singular ValueList[A[j], 1] [[1]])^2))` for each random test matrix. And the stopping criterion is $||\mathtt{V_{n+1}} - \mathtt{V_n}||_1 \leq 10^{-8}$. We only compared methods (1) denoted by "Schulz", (3) denoted by "KMS," and the new iterative scheme (9) denoted by "PM."

```
m = 1200; k = 1500; number = 20;
Table[A[l] = SparseArray[{Band[{400, 1}, {m, k}] -> Random[] − I,
Band[{1, 200}, {m, k}] -> {1.1, -Random[]},
Band[{−95, 100}] -> −0.02, Band[{−100, 500}] -> 0.1},
                                {m, k}, 0.];, {l, number}];
```

Algorithm 4

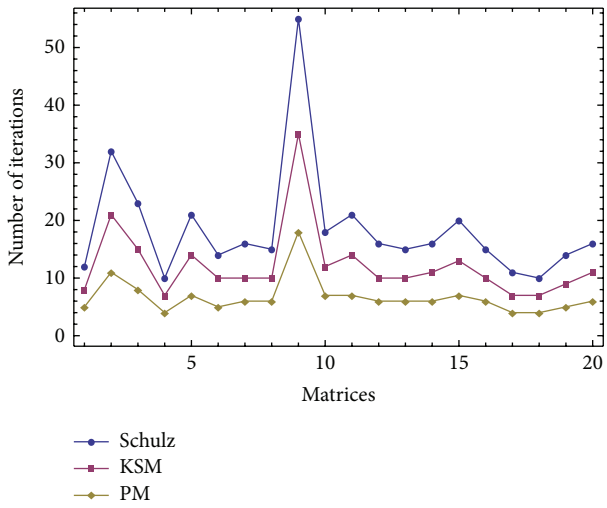

Figure 3: The results of comparisons for Example 10 in terms of the number of iterations.

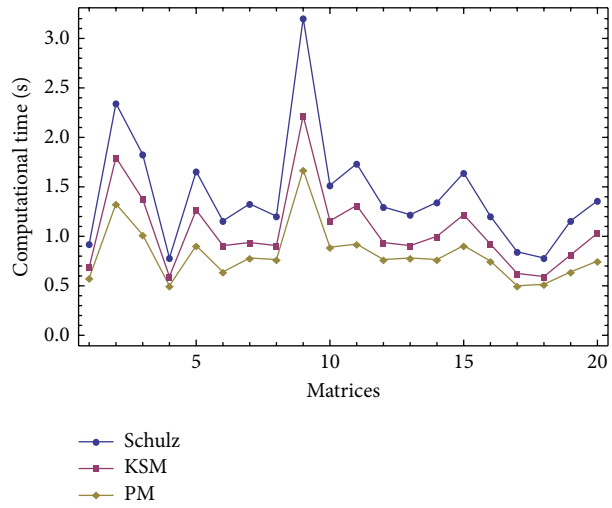

Figure 4: The results of comparisons for Example 10 in terms of the elapsed time.

The results for the number of iterations and the running time are compared in Figures 3-4. They show a clear advantage of the new scheme in finding the Moore-Penrose inverse.

5. Conclusions

In this paper, we have developed an iterative method in inverse finding of matrices, which has the capability to be applied on real or complex matrices with preserving the sparsity feature in matrix-by-matrix multiplications using a threshold. This allows us to interestingly reduce the computational time and usage memory in applying the new iterative method. We have shown that the suggested method (9) reaches ninth order of convergence. The extension of the scheme for pseudoinverse has also been discussed in the paper. The applicability of the new scheme was illustrated numerically in Section 4.

Finally, according to the numerical results obtained and the concrete application of such solvers, we can conclude that the new method is efficient.

The new scheme (9) has been tested for matrix inversion. However, we believe that such iterations could also be applied for finding the inverse of an integer in modulo p^n [34]. Following this idea will, for sure, open a new topic of research which is a combination of Numerical Analysis and Number Theory. We will focus on this trend of research for future studies.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The authors thank the referees for their valuable comments and for the suggestions to improve the readability of the paper.

References

[1] R. B. Sidje and Y. Saad, "Rational approximation to the Fermi-Dirac function with applications in density functional theory," *Numerical Algorithms*, vol. 56, no. 3, pp. 455–479, 2011.

[2] F. Soleymani and P. S. Stanimirović, "A higher order iterative method for computing the Drazin inverse," *The Scientific World Journal*, vol. 2013, Article ID 708647, 11 pages, 2013.

[3] G. Schulz, "Iterative Berechnung der Reziproken matrix," *Zeitschrift für angewandte Mathematik und Mechanik*, vol. 13, pp. 57–59, 1933.

[4] L. Lin, J. Lu, R. Car, and E. Weinan, "Multipole representation of the Fermi operator with application to the electronic structure analysis of metallic systems," *Physical Review B—Condensed Matter and Materials Physics*, vol. 79, no. 11, Article ID 115133, 2009.

[5] W. Li and Z. Li, "A family of iterative methods for computing the approximate inverse of a square matrix and inner inverse of a non-square matrix," *Applied Mathematics and Computation*, vol. 215, no. 9, pp. 3433–3442, 2010.

[6] H.-B. Li, T.-Z. Huang, Y. Zhang, X.-P. Liu, and T.-X. Gu, "Chebyshev-type methods and preconditioning techniques," *Applied Mathematics and Computation*, vol. 218, no. 2, pp. 260–270, 2011.

[7] W. Hackbusch, B. N. Khoromskij, and E. E. Tyrtyshnikov, "Approximate iterations for structured matrices," *Numerische Mathematik*, vol. 109, no. 3, pp. 365–383, 2008.

[8] T. Söderström and G. W. Stewart, "On the numerical properties of an iterative method for computing the Moore-Penrose generalized inverse," *SIAM Journal on Numerical Analysis*, vol. 11, pp. 61–74, 1974.

[9] X. Liu and Y. Qin, "Successive matrix squaring algorithm for computing the generalized inverse $A^{(2)}_{T,S}$," *Journal of Applied Mathematics*, vol. 2012, Article ID 262034, 12 pages, 2012.

[10] X. Liu and S. Huang, "Proper splitting for the generalized inverse $A^{(2)}_{T,S}$ and its application on Banach spaces," *Abstract and Applied Analysis*, vol. 2012, Article ID 736929, 9 pages, 2012.

[11] X. Liu and F. Huang, "Higher-order convergent iterative method for computing the generalized inverse over Banach spaces," *Abstract and Applied Analysis*, vol. 2013, Article ID 356105, 5 pages, 2013.

[12] X. Liu, H. Jin, and Y. Yu, "Higher-order convergent iterative method for computing the generalized inverse and its application to Toeplitz matrices," *Linear Algebra and Its Applications*, vol. 439, no. 6, pp. 1635–1650, 2013.

[13] J. Rajagopalan, *An iterative algortihm for inversion of matrices [M.S. thesis]*, Concordia University, Quebec, Canada, 1996.

[14] L. Grosz, "Preconditioning by incomplete block elimination," *Numerical Linear Algebra with Applications*, vol. 7, no. 7-8, pp. 527–541, 2000.

[15] G. Codevico, V. Y. Pan, and M. Van Barel, "Newton-like iteration based on a cubic polynomial for structured matrices," *Numerical Algorithms*, vol. 36, no. 4, pp. 365–380, 2004.

[16] V. Pan and R. Schreiber, "An improved Newton iteration for the generalized inverse of a matrix, with applications," *SIAM Journal on Scientific and Statistical Computing*, vol. 12, no. 5, pp. 1109–1130, 1991.

[17] A. Ben-Israel and T. N. E. Greville, *Generalized Inverses*, Springer, 2nd edition, 2003.

[18] F. Khaksar Haghani and F. Soleymani, "A new high-order stable numerical method for matrix inversion," *The Scientific World Journal*, vol. 2014, Article ID 830564, 10 pages, 2014.

[19] F. Soleymani, S. Karimi Vanani, H. I. Siyyam, and I. A. Al-Subaihi, "Numerical solution of nonlinear equations by an optimal eighth-order class of iterative methods," *Annali dell'Universitá di Ferrara. Sezione VII. Scienze Matematiche*, vol. 59, no. 1, pp. 159–171, 2013.

[20] X. Wang and T. Zhang, "High-order Newton-type iterative methods with memory for solving nonlinear equations," *Mathematical Communications*, vol. 19, pp. 91–109, 2014.

[21] H. Hotelling, "Analysis of a complex of statistical variables into principal components," *Journal of Educational Psychology*, vol. 24, no. 6, pp. 417–441, 1933.

[22] E. Isaacson and H. B. Keller, *Analysis of Numerical Methods*, John Wiley & Sons, New York, NY, USA, 1966.

[23] E. V. Krishnamurthy and S. K. Sen, *Numerical Algorithms—Computations in Science and Engineering*, Affiliated East-West Press, New Delhi, India, 2007.

[24] L. Weiguo, L. Juan, and Q. Tiantian, "A family of iterative methods for computing Moore-Penrose inverse of a matrix," *Linear Algebra and Its Applications*, vol. 438, no. 1, pp. 47–56, 2013.

[25] F. Soleymani, "A fast convergent iterative solver for approximate inverse of matrices," *Numerical Linear Algebra with Applications*, 2013.

[26] M. Zaka Ullah, F. Soleymani, and A. S. Al-Fhaid, "An efficient matrix iteration for computing weighted Moore-Penrose inverse," *Applied Mathematics and Computation*, vol. 226, pp. 441–454, 2014.

[27] V. Y. Pan, *Structured Matrices and Polynomials: Unified Superfast Algorithms*, Springer, Birkhäauser, Boston, Mass, USA, 2001.

[28] M. Miladinović, S. Miljković, and P. Stanimirović, "Modified SMS method for computing outer inverses of Toeplitz matrices," *Applied Mathematics and Computation*, vol. 218, no. 7, pp. 3131–3143, 2011.

[29] H. Chen and Y. Wang, "A family of higher-order convergent iterative methods for computing the Moore-Penrose inverse," *Applied Mathematics and Computation*, vol. 218, no. 8, pp. 4012–4016, 2011.

[30] A. R. Soheili, F. Soleymani, and M. D. Petković, "On the computation of weighted Moore-Penrose inverse using a high-order matrix method," *Computers & Mathematics with Applications*, vol. 66, no. 11, pp. 2344–2351, 2013.

[31] F. Soleymani and P. S. Stanimirović, "A note on the stability of a pth order iteration for finding generalized inverses," *Applied Mathematics Letters*, vol. 28, pp. 77–81, 2014.

[32] P. S. Stanimirović and F. Soleymani, "A class of numerical algorithms for computing outer inverses," *Journal of Computational and Applied Mathematics*, vol. 263, pp. 236–245, 2014.

[33] S. Wolfram, *The Mathematica Book*, Wolfram Media, 5th edition, 2003.

[34] M. P. Knapp and C. Xenophontos, "Numerical analysis meets number theory: using rootfinding methods to calculate inverses mod p^n," *Applicable Analysis and Discrete Mathematics*, vol. 4, no. 1, pp. 23–31, 2010.