Comp. Appl. Math.
DOI 10.1007/s40314-017-0507-0

# Factorizations of hyperpower family of iterative methods via least squares approach

**Ashim Kumar**[1] · **Predrag S. Stanimirović**[2] · **Fazlollah Soleymani**[3] · **Mihailo Krstić**[2] · **Kostadin Rajković**[2]

Received: 26 May 2017 / Revised: 9 August 2017 / Accepted: 28 August 2017


**Abstract** A new method for finding proper factorizations of the hyperpower family of iterative methods for computing generalized inverses is proposed. The method is based on numerical optimization of an appropriate least squares problem. An iterative method in a non-normalized form for finding the generalized outer inverses with prescribed range and null space of any given complex matrix is proposed and considered. Computational complexity and convergence of the method are studied. Several numerical experiments are also presented.

**Keywords** Generalized inverse · Least squares · Moore–Penrose inverse · Hyperpower method · Convergence analysis

**Mathematics Subject Classification** 65F30 · 15A09

## 1 Introduction and preliminaries

The traditional generalized inverses, such as the pseudo-inverses $A^+$, the weighted Moore–Penrose inverses $A^+_{MN}$, the Drazin inverses $A^D$, the group inverses $A^\#$, the Bott–Duffin inverses $A^{-1}_L$, the generalized Bott–Duffin inverses $A^+_{(L)}$, and so forth are all of special interest in matrix theory Yanai et al. (2011, chapter 3). They are extensively used in statistics, control theory, denumerable Markov-chains, nonlinear equations, optimization techniques, numerical linear algebra, see, e.g., Hsuan et al. (1985), Li et al. (2011), Srivastava et al.

Communicated by Jinyun Yuan.

✉ Fazlollah Soleymani
soleymani@iasbs.ac.ir; fazl_soley_bsb@yahoo.com

[1] Panjab University Constituent College, Nihal Singh Wala 142046, India

[2] Faculty of Sciences and Mathematics, University of Niš, Visegradska 33, Nis 18000, Serbia

[3] Department of Mathematics, Institute for Advanced Studies in Basic Sciences, Zanjan, Iran

Published online: 09 September 2017

(2017) and Nashed (1976, chapter IV). Most of these generalized inverses are special outer generalized inverses $X=A^{(2)}_{T,S}$, (for more details one may consult the seminal book (Ben-Israel 2003, chapter 2)).

Let $\mathbb{C}^{m\times n}$ and $\mathbb{C}^{m\times n}_r$ denote the set of all complex $m \times n$ matrices and all complex $m \times n$ matrices of rank $r$. Also, $I_m$ denotes the unit matrix of order $m$. By $A^*$, $\mathcal{R}(A)$, and $\mathcal{N}(A)$, we mean conjugate transpose, the range, and the null space of a matrix $A \in \mathbb{C}^{m\times n}$, respectively. For $A \in \mathbb{C}^{m\times n}$, the set of outer generalized inverses or $\{2\}$-inverses is defined by $A\{2\} = \{X \in \mathbb{C}^{n\times m} : XAX = X\}$. The generalized inverse $A^{(2)}_{T,S} \in A\{2\}$ of a complex matrix $A \in \mathbb{C}^{m\times n}$ is defined as the matrix $X \in \mathbb{C}^{n\times m}$ satisfying $XAX = X$, and $\mathcal{R}(X) = T$, $\mathcal{N}(X) = S$. A fundamental Lemma 1 reveals conditions for the existence of $A^{(2)}_{T,S}$ of a complex matrix.

**Lemma 1** Ben-Israel and Greville (2003) *Let $A \in \mathbb{C}^{m\times n}_r$, and let $T$ and $S$ be subspaces of $\mathbb{C}^n$ and $\mathbb{C}^m$, respectively, with $\dim T = \dim S^{\perp} = t \le r$. Then, $A$ has a $\{2\}$-inverse $X$ such that $\mathcal{R}(X) = T$ and $\mathcal{N}(X) = S$ if and only if*

$$AT \bigoplus S = \mathbb{C}^m, \tag{1.1}$$

*where $\bigoplus$ indicates a direct-sum of two subspaces. Furthermore, $X$ is unique and is denoted by $A^{(2)}_{T,S}$.*

Generalized inverse matrices are simply matrix representations of the inverse transformation with the domain extended to a space. However, there are infinitely many ways in which the generalization can be made, and thus there are infinitely many corresponding generalized inverses Yanai et al. (2011, page vi).

Techniques for computing the generalized outer inverse $A^{(2)}_{T,S}$ are a subject of current research (e.g., see Borle and Lomonaco 2016; Liu and Zuo 2014; Srivastava and Gupta 2016 and the references therein). Direct solvers such as Gaussian elimination with partial pivoting cannot be extended easily for finding generalized inverses. On the other hand, iterative methods in the class of Schulz-type methods (e.g., see Pan et al. 2006; Pan 2010; Stanimirović and Soleymani 2014) are the methods of choice, in which an approximation of $A^{(2)}_{T,S}$ can be found per cycle up to a desired accuracy.

The well-known quadratically convergent Newton–Schulz iterative method (Hotelling 1943; Schulz 1933) (sometimes known as Schultz matrix scheme) is given by the iterative rule

$$V_{k+1} = V_k(2I - AV_k), \quad k = 0, 1, 2, \ldots, \tag{1.2}$$

where $I$ is an identity matrix of appropriate order and by a proper initial value $V_0$. This scheme has interesting features of being based exclusively on matrix–matrix operations. A general background and usefulness of such a method is described in Higham (2002, chapter 14.5). An important difficulty in the implementation of this method lies in the fact that this iteration is slow, even converges linearly at the beginning of the process and might take many iterations to achieve its final quadratic convergence order. This would increase the computational load of the whole algorithm used for finding $A^{(2)}_{T,S}$. To remedy this drawback, higher order iterative methods which are economic in terms of computational efficiency should be constructed.

Let us review some of the higher order methods. The cubically convergent scheme of Chebyshev (Sen and Prabhu 1976) is expressed by

$$V_{k+1} = V_k(3I - AV_k(3I - AV_k)), \quad k = 0, 1, 2, \ldots. \tag{1.3}$$

Recently, authors in Ghorbanzadeh et al. (2016) presented a 10th-order iterative method of the form below

$$V_{k+1} = V_k\,(I + R_k)\left[(I + \alpha R_k^2 + R_k^4)(I + \beta R_k^2 + R_k^4)\right],\ R_k = I - AV_k, \tag{1.4}$$

with only six matrix–matrix multiplications (mmms shortly), whereas $\alpha = \frac{1}{2}\left(1 - \sqrt{5}\right)$, and $\beta = \frac{1}{2}\left(1 + \sqrt{5}\right)$. This scheme is efficient in terms of a balance between the number of mmms and the convergence rate.

New iterative schemes of orders 5 and 9 for computing outer inverse, which consider the possibility of reducing the total number of matrix multiplications per each cycle by means of proper factorizations, were considered in Petković and Petković (2015). Several other novel methods via proper factorizing of the hyperpower family of iterative methods with arbitrary orders were presented in Soleymani et al. (2015).

The hyperpower matrix scheme (Climent et al. 2001) of 13th order is stated as

$$V_{k+1} = V_k(I + R_k + R_k^2 + R_k^3 + \cdots + R_k^{12}),\quad R_k = I - AV_k,\quad k = 0, 1, 2, \ldots. \tag{1.5}$$

The algorithm requires 13 mmms to obtain 13th order of convergence in (1.5), which is not that economic in contrast to three mmms of (1.3) to hit the convergence rate three. It would be beneficial, if one can design a high-order scheme by reducing its operational cost, that is to say, the number of mmms, since the most important cost in implementing the Schulz-type methods is the cost of mmms. This would be our motivation which will be discussed in subsequent sections.

For some factorizations of the hyperpower method it is impossible to find coefficients of these factorizations explicitly as real numbers. For this purpose, our intention is to define an approach which is applicable on all factorizations. Particularly, we intend to develop a new matrix iteration scheme of 13th order of convergence by reducing its cost expressed in terms of mmms with respect to the seminal iterative scheme (1.5). It would make the method quite efficient in terms of involved number of mmms in comparison to the existing iterative methods of the same type for finding $A^{(2)}_{T,S}$.

The most important novelty of this paper is a different approach to find the unknown coefficients in the introduced factorization of the hyperpower iterative method that reduces its complexity. The defined algorithm in this way gives an answer to the query that what should be done when the numerical values of the coefficients in a factorization of the hyperpower iteration cannot be determined explicitly? As a matter of fact, the numerical optimization which solves the least squares problem is proposed to find unknown coefficients in a newly introduced structure. However, the main drawback of the method is high number of required scalar to matrix multiplications (smms shortly) per cycle. For this purpose, the use of such an approach is justified in the case when it is not possible to find coefficients of the factorization in the form of predefined numerical expressions representing real numbers.

The structure of this paper is as follows. A new matrix iterative scheme of order 13, for computing outer inverses is defined in Sect. 2. Also, an algorithm for defining coefficients of the factorization by means of the numerical optimization is presented. Convergence analysis along with the error bounds of the proposed method are established therein as well. A discussion in terms of computational efficiency index is presented in Sect. 3. A number of numerical examples, including the rectangular case, and Toeplitz matrices are worked out in Sect. 4. It is observed that our method gives improved results in terms of both computational speed and accuracy. A short conclusion will finally be provided in Sect. 5.

## 2 Construction of a new method

We consider a class of iterative methods of the form

$$V_{k+1} = V_k\Bigg(\prod_{i=1}^{r}\left(a_{0i} + a_{1i}R_k + a_{2i}R_k^2 + \left(R_k^2 + \frac{R_k}{r}\right)R_k^2\right) + b_0 I + b_1 R_k + b_2 R_k^2\Bigg), \quad R_k = I - AV_k, \tag{2.1}$$

where the coefficients $a_{0i}, a_{1i}, a_{2i}, i = 1, \ldots, r$ and $b_0, b_1, b_2$ are unknowns which we need to determine from the constraint

$$\prod_{i=1}^{r}\left(a_{0i} + a_{1i}R_k + a_{2i}R_k^2 + \left(R_k^2 + \frac{R_k}{r}\right)R_k^2\right) + b_0 I + b_1 R_k + b_2 R_k^2 = \sum_{i=0}^{4r} R_k^i. \tag{2.2}$$

Each fixed value of $r \in \mathbb{N}$ in (2.1) produces an iterative method of the order $4r + 1$ which requires $3 + r$ mmms. Lemma 2 determines the method from the class (2.1) with the highest computational efficiency index.

**Lemma 2** *Function* $f : \mathbb{N} \to \mathbb{R}$ *defined as*

$$f(r) = (1 + 4r)^{\frac{1}{3+r}}, \tag{2.3}$$

*reaches a maximal value* 1.5518 *on the set* $\mathbb{N}$ *for* $r = 2$.

*Proof* We first prove that the function $f$ is decreasing for $r \geq 3$. For this purpose, we will define function $g : \mathbb{R} \to \mathbb{R}$ as follows

$$g(x) = (1 + 4x)^{\frac{1}{3+x}}, \tag{2.4}$$

which is an extension of the function $f$ on the set $\mathbb{R}$. We will prove that $g$ declines on set $[3, +\infty)$. The derivative of $g$ is equal to

$$g'(x) = (1 + 4x)^{\frac{1}{x+3}}\left(\frac{4}{(3 + x)(4x + 1)} - \frac{\ln(4x + 1)}{(3 + x)^2}\right). \tag{2.5}$$

Now, taking into account the assumption $x \geq 3$, the inequality $g'(x) < 0$ is equivalent with the following:

$$g'(x) < 0 \iff \frac{4}{(3 + x)(4x + 1)} - \frac{\ln(4x + 1)}{(x + 3)^2} < 0, \tag{2.6}$$

which further implies

$$\begin{aligned} g'(x) < 0 &\iff \frac{4}{4x + 1} - \frac{\ln(4x + 1)}{3 + x} < 0 \\ &\iff 4(3 + x) - (4x + 1)\ln(4x + 1) < 0 \\ &\iff \ln(4x + 1) > \frac{4(3 + x)}{4x + 1}. \end{aligned} \tag{2.7}$$

Now, we will define a function $h : \mathbb{R}^+ \to \mathbb{R}$ as follows $h(x) = \ln(4x + 1) - \frac{4(3+x)}{4x+1}$. Clearly, $h'(x) = \frac{16(3+x)}{(4x+1)^2} > 0$, because of the assumption $x \geq 3$. Therefore, the function $h$ is increasing for $x \geq 3$. On the other hand, since $h(3) = \ln(13) - \frac{24}{13} \approx 0.7187 > 0$, one can verify $h(x) > 0$ for any $x \geq 3$. We already have $h(2) = \ln(9) - \frac{20}{9} \approx -0.0249 < 0$.

Therefore, $g$ is a declining function on the interval $[3, +\infty)$ and the function $g$ reaches the highest value on the interval $[3, +\infty)$ at the point $x = 3$.

With direct checking, we have that $g(2) = \max\{g(1), g(2), g(3)\}$, and therefore the maximum of the function $f$ on the set of natural numbers is achieved just in $r = 2$ (because $f$ is a restriction of $g$ on $\mathbb{N}$). □

According to Lemma 2, the case $r = 2$ is the most promising from the factorizations (2.1). However, this case was considered as ad hoc method without convergence analysis in Petković and Petković (2015).

After several attempts, it would be obvious that the unknown parameters in the case $r = 3$ cannot be obtained theoretically! Anyhow, it is possible to use an alternative strategy of numerical *optimization* to obtain the unknown parameters. To avoid numerical instability (Du Croz and Higham 1992), it is necessary to use minimization with a large number of floating point arithmetic or based on their rational approximations.

### 2.1 The novel method

Here our goal is the case $r = 3$. Accordingly, we derive a 13th-order matrix iterative method with the help of (1.5) for finding the generalized outer inverse $A^{(2)}_{T,S}$, in which the number of mmms are less than the ones required in (1.5). Bearing this in mind and using the same methodology as in Ghorbanzadeh et al. (2016) and Soleymani et al. (2015), we can simplify (1.5) by proper factorization and obtain the following non-normalized iterative scheme

$$V_{k+1} = V_k\,[e \times P_k], \quad k = 0, 1, 2, \ldots, \tag{2.8}$$

where

$$\begin{aligned} P_k = &\left(a_{01} I + a_{02} R_k + a_{03} R_k^2 + R_k^2 (R_k/3 + R_k^2)\right) \\ &\times \left(a_{11} I + a_{12} R_k + a_{13} R_k^2 + R_k^2 (R_k/3 + R_k^2)\right) \\ &\times \left(a_{21} I + a_{22} R_k + a_{23} R_k^2 + R_k^2 (R_k/3 + R_k^2)\right) \\ &+ b_0 I + b_1 R_k + b_2 R_k^2, \end{aligned} \tag{2.9}$$

whereas $e, a_{0i}, a_{1i}, a_{2i}$, and $b_i$, $i = 0, 1, 2$ are real parameters that have to be found.

Note that the structure (2.8)–(2.9) requires only six mmms, which are far lower than that of the corresponding scheme (1.5) to achieve the equal convergence rate 13. It is remarked that the term non-normalized (Soleymani et al. 2015) refers to a matrix method at which the coefficients are not anymore integers with values $\pm 1$. Clearly, if one is able to find the unknowns of (2.9), then an efficient matrix scheme will be derived.

The remaining part of this section contains the contribution of this study to find out the above parameters so that the new iterative method (2.8)–(2.9) with 13th order could be applied in computing outer inverse $A^{(2)}_{T,S}$.

To obtain the above unknown parameters, we will use the least squares approach (Lawson and Hanson 1974) and consider the matrix expression

$$Q_k = e \times P_k. \tag{2.10}$$

The scalar coefficient corresponding to $R_k^{i-1}$ in $Q_k$ is denoted by $L_i$. Accordingly, by equating the coefficients $L_i$ to 1, we obtain a system of 13 nonlinear equations $L_i = 1$ for $1 \le i \le 13$ with 13 unknown parameters. Then, it is necessary to minimize

$$\sum_{i=1}^{13} (L_i - 1)^2, \tag{2.11}$$

which could be handled in some computer algebra systems such as Mathematica (Trott 2006). A step by step procedure to find the unknown parameters using the least squares approach can be found in Algorithm 1.

**Algorithm 1** Computing the unknown coefficients in the proposed method (2.8)–(2.9).

1: Define the symbolic expression $W$ which represents the proposed factorization. In our case,
```
W=(a0+a1 Rk+a2 Rk^2+(Rk^2+Rk/3) Rk^2) (b0+b1 Rk+b2 Rk^2+(Rk^2+Rk/3) Rk^2)
   (c0+c1 Rk+c2 Rk^2+(Rk^2+Rk/3) Rk^2)+d0+d1 Rk+d2 Rk^2.
```
2: Find the list of expressions corresponding to $R_k^i, i = 0, \ldots, 12$:
```
L= CoefficientList[Collect[Expand[e*W], Rk], Rk].
```
3: Define the minimization problem:
```
eqs1 = Sum[(L[[i]] - 1)^2, {i, 1, Length[L]}]; vars = Variables[eqs1];.
```
4: Solve the minimization problem:
```
{mn,l} = NMinimize[eqs1, vars, WorkingPrecision -> digits];.
```
5: Obtain the unknown parameters:
```
{e,a2,b2,c2,a1,b1,c1,a0,b0,c0,d0,d1,d2}
={e,a2,b2,c2,a1,b1,c1,a0,b0,c0,d0,d1,d2}/.l;.
```

In the above algorithm, $R_k = I - AV_k$. Applying 40 decimals working precision, we may find the unknowns in floating point arithmetic and their rational approximations as follows:

$$\begin{aligned}
a_{01} &= \frac{5149850384}{3709496951}, & a_{02} &= \frac{1215660179}{3758230826}, & a_{03} &= \frac{8509855554}{4332860201},\\
a_{11} &= \frac{25345430979}{20099801969}, & a_{12} &= -\frac{631412965}{4325488242}, & a_{13} &= -\frac{1399913270}{2224514759},\\
a_{21} &= \frac{183704129}{3133797841}, & a_{22} &= \frac{1231128136}{3610056361}, & a_{23} &= -\frac{1778627640}{2662421539},\\
b_0 &= \frac{605347770}{674572997}, & b_1 &= \frac{3476846045}{8892960773}, & b_2 &= \frac{1410126973}{702148560},
\end{aligned} \tag{2.12}$$

where $e = 1$. A simple implementation of this algorithm in the programming package Mathematica is brought forward in Appendix.

The above approximations of the unknown parameters are enough in all practical examples, since we work in double precision arithmetic. If one wishes to find generalized inverses in high precision computing, a higher working precision, such as 1000 must be chosen just once to compute much more accurate values for the parameters.

It is necessary to remind that an initial approximation (initial value) $V_0$ must be taken into account according to the rule (2.13) to observe the convergence. That is, it is sufficient only to choose the initial matrix in the form $V_0 = \alpha Y$, $\mathcal{R}(Y) = T$, $\mathcal{N}(Y) = S$, whereas

$$\left\| AA_{T,S}^{(2)} - AV_0 \right\| < 1. \tag{2.13}$$

A sharp initial value could be constructed for outer inverses by generalizing the idea of Pan and Schreiber (1991) of the form $V_0 = \alpha Y$, where $\alpha = \frac{2}{\sigma_1^2 + \sigma_r^2}$ and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$ are the nonzero eigenvalues of $YA$.

## 2.2 Convergence and its rate

**Theorem 1** *Let $A \in \mathbb{C}_r^{m\times n}$, $T$ be a subspace of $\mathbb{C}^n$ of dimension $s \leq r$, and $S$ be the subspace of $\mathbb{C}^m$ of dimension $m - s$. Also, suppose that an initial approximation $V_0 = \alpha Y$, $\mathcal{R}(Y) = T$, $\mathcal{N}(Y) = S$, is taken in accordance with (2.13). Then, the sequence $\{V_k\}$ generated by (2.8)–(2.9) converges with 13th order to $A_{T,S}^{(2)}$ by ignoring the roundoff errors in double precision arithmetic.*

*Proof* Working in double precision arithmetic, the iterative method (2.8)–(2.9) becomes equivalent to (1.5), i.e.,

$$V_{k+1} \cong V_k(I + R_k + R_k^2 + R_k^3 + \cdots + R_k^{12}), \quad k = 0, 1, 2, \ldots.$$

And subsequently, we have

$$\mathcal{F}_{k+1} = AA_{T,S}^{(2)} - I + \left(I - AA_{T,S}^{(2)} + \mathcal{F}_k^{13}\right) = \mathcal{F}_k^{13}, \tag{2.14}$$

whereas $\mathcal{F}_k = AA_{T,S}^{(2)} - AV_k$, is the residual matrix in the $k$th-iterate of performing our proposed scheme. To complete showing the convergence behavior, we now take into account that $E_k = A_{\mathcal{R}(Y),\mathcal{N}(Y)}^{(2)} - V_k$, is the error matrix for finding the generalized outer inverse $A_{T,S}^{(2)}$. Thus, we attain

$$AE_{k+1} = AA_{\mathcal{R}(Y),\mathcal{N}(Y)}^{(2)} - AV_{k+1} = \mathcal{F}_{k+1} = \mathcal{F}_k^{13}. \tag{2.15}$$

Using the relation (2.15) and some calculations, we get that

$$\|AE_{k+1}\| \leq \|\mathcal{F}_k\|^{13} = \|AE_k\|^{13} \leq \|A\|^{13}\|E_k\|^{13}. \tag{2.16}$$

Using the inequality (2.16), we could find the rate of convergence for enough large $k$ in what follows:

$$\begin{aligned}
\|E_{k+1}\| &= \left\|V_{k+1} - A_{R(Y),N(Y)}^{(2)}\right\| \\
&= \left\|A_{\mathcal{R}(Y),\mathcal{N}(Y)}^{(2)} AV_{k+1} - A_{R(Y),N(Y)}^{(2)} AA_{R(Y),N(Y)}^{(2)}\right\| \\
&= \left\|A_{\mathcal{R}(Y),\mathcal{N}(Y)}^{(2)} \left(AV_{k+1} - AA_{R(Y),N(Y)}^{(2)}\right)\right\| \\
&\leq \|A_{\mathcal{R}(Y),\mathcal{N}(Y)}^{(2)}\| \, \|AE_{k+1}\| \\
&\leq \left\|A_{T,S}^{(2)}\right\| \, \|A\|^{13} \, \|E_k\|^{13}.
\end{aligned} \tag{2.17}$$

Hence, one can verify that the sequence $\{V_k\}$ generated by (2.8)–(2.9) satisfies $\|A_{T,S}^{(2)} - V_k\| \to 0$ as $k \to \infty$ with 13th order. The proof is complete. □

## 3 Computational complexity

The computational efficiency of a matrix iterative method for finding $A_{T,S}^{(2)}$, must be measured by an index relating to the cost of an iteration process and its convergence order. The computational efficiency index is given by Ehrmann (1959)

$$\text{CEI} = p^{\frac{1}{c}}, \tag{3.1}$$

whereas $c$ stands for the total computational cost of an algorithm and $p$ is the local convergence order.

It is clear that the governing cost per cycle of each Schulz-type method is determined by the mmms. Let us suppose that the cost of mmms is unity (as Traub made in Traub (1964)).

Notice that Söderström and Stewart (1974) illustrated that the approximate number of iterations that the Schulz scheme (1.2) requires in a machine precision to converge is given by

$$s \approx 2\log_2 \kappa_2(A), \tag{3.2}$$

where $\kappa_2$ denotes the condition number of the matrix $A$ in the norm 2. Hence, similar to (3.2) under the same conditions, the estimated number of iterations for our iterative method to converge is given by Soleymani (2014)

$$s \approx 2\log_{13}\kappa_2(A). \tag{3.3}$$

Note that here

$$\kappa_2(A) = \|A\|_2\|A^{(2)}_{T,S}\|_2. \tag{3.4}$$

The proposed iterative method (2.8)–(2.9) reaches 13th-order convergence using only six mmms. Therefore, if one applies the definition of (computational) efficiency index as (3.1), then the reported method (2.8)–(2.9) achieves the efficiency

$$CEI_{(2.8)-(2.9)} = 13^{\frac{1}{6}} \approx 1.53341, \tag{3.5}$$

which is greater than the efficiency indices of different methods given below

$$CEI_{(1.2)} \approx 1.41421,\ CEI_{(1.3)} \approx 1.44225,\ CEI_{(1.5)} \approx 1.21811,\ CEI_{(1.4)} \approx 1.4678. \tag{3.6}$$

This reveals that the iterative scheme (2.8)–(2.9) is computationally economic in theoretical findings and will be valuable for practical problems with an acceptable efficiency index.

## 4 Numerical experiences

In this section, some experiments are executed to check the usefulness and capability of the newly proposed method. Mathematica 10 has been employed in our calculations (Trott 2006). We have carried out the demonstrations on a computer with specifications: Microsoft Windows 8.1 Pro, Intel(R), Core(TM) i7, CPU @ 2.50 GHz, with 16 GB of RAM.

The methods (1.2), (1.3), (1.4), (1.5) and (2.8)–(2.9) were used in numerical comparisons and denoted by NM, CM, HM10, HM13 and IHM13, respectively. As the programs were running, we calculated the running time using the command `AbsoluteTiming[]` to report the elapsed computational time (in seconds) for the experiments. Here, $\|E_k\| = \|V_k - A^{(2)}_{T,S}\|$ is the norm of the error matrix.

**Definition 1** Soleymani (2014) For any subordinate matrix norm $\|\cdot\|_*$, the numerical local convergence order for iterative methods could be defined as

$$\varrho = \frac{\ln\left[\|V_{k+1} - A^{(2)}_{T,S}\|\,\|V_k - A^{(2)}_{T,S}\|^{-1}\right]}{\ln\left[\|V_k - A^{(2)}_{T,S}\|\,\|V_{k-1} - A^{(2)}_{T,S}\|^{-1}\right]}, \tag{4.1}$$

wherein the last three approximations $V_{k-1}$, $V_k$ and $V_{k+1}$ in the convergence phase are used.

This definition is useful for our numerical comparisons, when applying an iterative method in high precision computing environment and to observe numerically the local convergence order of different methods for solving academic problems.

Figure 1 reveals the estimate number of iterates for different iterative methods by increasing the condition number for finding the outer inverses based on Sect. 3.

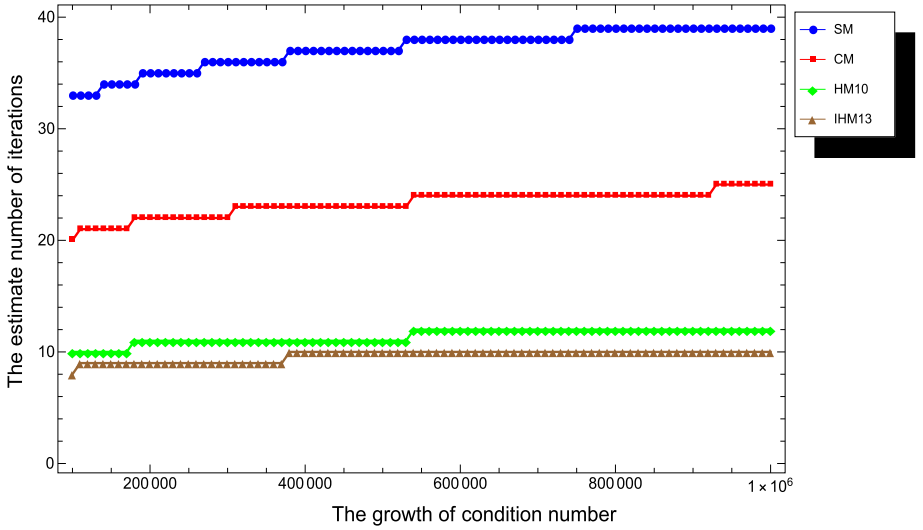


**Fig. 1** A comparison of the estimate number of iterations for different methods

**Table 1** Results of comparisons in Experiment 1

| | NM | CM | HM13 | IHM13 |
|---|---|---|---|---|
| $\varrho$ | 2.00 | 3.00 | 13.00 | 13.00 |
| Time | 0.01424 | 0.01053 | 0.01565 | 0.01266 |
| CEI | 1.414 | 1.442 | 1.218 | 1.533 |
| IT | 10 | 6 | 3 | 3 |

**Experiment 1** (Academic Test) *Here, we used 2000 precision digits when the maximum number of iterations is appointed to 100. Let us consider the matrix*

$$A = \begin{pmatrix} 3 & 3/5 & 3/5 & 3/5 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \tag{4.2}$$

*The exact pseudo-inverse of A is equal to*

$$A^{\dagger} = \begin{pmatrix} 1/3 & -1/15 & -1/15 & -1/15 & 0 \\ 0 & 1/3 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 0 & 0 \\ 0 & 0 & 0 & 1/3 & 0. \end{pmatrix}.$$

*A comparison of numerical results derived by applying different methods are reported in Table 1. These results are derived using the stopping criterion* $\|E_k\| = \|V_k - A^{\dagger}\|_2 \leq 10^{-150}$. *The initial approximation in this example is taken as* $V_0 = \frac{1}{\sigma^2}A^*$, *where* $\sigma$ *is taken as the largest singular value of A.*

Here IT stands for the number of iterations. It is necessary to mention that the methods HM13 and IHM13 produce almost identical values during the iterations. In the

rest of this example, it is shown that computation with very high precision is required to ensure successful application of defined iterations (2.8)–(2.9). It is assumed that `WorkingPrecision->2000` is specified during the application of all considered iterative methods. If the unknown parameters are generated applying the function `NMinimize` under the assumption `WorkingPrecision->50`, then the iterative method (2.8)–(2.9) does not finishes convergence successfully and generates the following residuals: $4.0739 \times 10^{-6}$, $5.0112 \times 10^{-26}$, $5.0113 \times 10^{-26}$, $5.0113 \times 10^{-26}$. After that, the last value becomes invariant in all iterations. If the unknown parameters are generated applying the function `NMinimize` with `WorkingPrecision->2000`, then the iterative method (2.8)–(2.9) converges very quickly: $4.6038 \times 10^{-5}$, $2.8038 \times 10^{-52}$, $4.4474 \times 10^{-666}$. The application of the function `NMinimize` in 2000 digits precision requires an amount of time to compute unknown coefficients. However, after that, further applications of the method are easy to handle. Note again that the coefficients of the iterative scheme should be computed only once!

**Experiment 2** *A polynomial matrix is given by*

$$S_5(t) = \begin{bmatrix} t+1 & t & t & t & t+1 \\ t & t-1 & t & t & t \\ t & t & t+1 & t & t \\ t & t & t & t-1 & t \\ t+1 & t & t & t & t+1 \end{bmatrix}.$$

*In conjunction with $A = S_5(t)$, consider the matrices*

$$U = \begin{bmatrix} 2t+1 & t & t \\ t & 2t-1 & t \\ t & t & 2t+1 \\ t & t & t \\ 2t+1 & t & t \end{bmatrix}, \; V = \begin{bmatrix} t^2+1 & t^2 & t^2 & t^2 & t^2+1 \\ t^2 & t^2-1 & t^2 & t^2 & t^2 \\ t^2 & t^2 & t^2+1 & t^2 & t^2 \end{bmatrix},$$

*and $G = UV$. The exact outer inverse of $A$ corresponding to $G$ is equal to*

$$A^{(2)}_{\mathcal{R}(U),\mathcal{N}(V)} = U(VAU)^{-1}V =$$

$$\begin{bmatrix} \frac{-10t^5+6t^4+3t^3-t^2+t+1}{-63t^5+42t^4-t^3+2t^2+8t+4} & \frac{t(15t^4+t^3+6t^2-2)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{t(-15t^4+13t^3-8t^2+4t+2)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{t^2(-20t^3+6t^2+3t-1)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{-10t^5+6t^4+3t^3-t^2+t+1}{-63t^5+42t^4-t^3+2t^2+8t+4} \\ \frac{t(-2t^4+3t^3+8t^2-3t-2)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{-66t^5+13t^4-25t^3+2t^2+12t+4}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{t(3t^4+t^3+16t^2-8t-4)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{t^2(67t^3-9t^2-18t-4)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{t(-2t^4+3t^3+8t^2-3t-2)}{63t^5-42t^4+t^3-2t^2-8t-4} \\ \frac{t(-16t^4+13t^3-6t^2+3t+2)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{t(39t^4-7t^3+8t^2-4)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{-24t^5+7t^4+15t^3-6t^2+4t+4}{-63t^5+42t^4-t^3+2t^2+8t+4} & \frac{t^2(-31t^3+13t^2+2t-4)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{t(-16t^4+13t^3-6t^2+3t+2)}{63t^5-42t^4+t^3-2t^2-8t-4} \\ \frac{t(2t^4-3t^3+t^2-5t+1)}{-63t^5+42t^4-t^3+2t^2+8t+4} & -\frac{t(3t^4+29t^3+6t^2+18t+4)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{t(3t^4+t^3-2t^2+10t-4)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{t^3(4t^2+33t-1)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{t(2t^4-3t^3+t^2-5t+1)}{-63t^5+42t^4-t^3+2t^2+8t+4} \\ \frac{-10t^5+6t^4+3t^3-t^2+t+1}{-63t^5+42t^4-t^3+2t^2+8t+4} & \frac{t(15t^4+t^3+6t^2-2)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{t(-15t^4+13t^3-8t^2+4t+2)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{t^2(-20t^3+6t^2+3t-1)}{63t^5-42t^4+t^3-2t^2-8t-4} & \frac{-10t^5+6t^4+3t^3-t^2+t+1}{-63t^5+42t^4-t^3+2t^2+8t+4} \end{bmatrix}.$$

*Our goal in this example is to perform numerical experiments for various values of the parameter $t$. The initial approximation in this example is taken as $V_0 = \frac{1}{\sigma^2}G$, where $\sigma$ is the largest singular value of $G$.*

The following concluding remarks can be stated according to results arranged in Table 2. First, IHM13 is similar to HM13 in terms of accuracy. Second, numerical stability is similar for both of these methods. And finally, only efficient variants of hyperpower methods of higher order show better performances for larger values of $t$ with respect to hyperpower methods of smaller order.

**Table 2** Results of comparisons in Experiment 2

| | NM | CM | HM13 | IHM13 |
|---|---|---|---|---|
| $t = 1$ | | | | |
| IT | 24 | 15 | 7 | 7 |
| Time | 0.105512 | 0.079368 | 0.085585 | 0.061719 |
| $\|E_k\|$ | $1.24623 \times 10^{-296}$ | $1.23272 \times 10^{-253}$ | $1.363953 \times 10^{-1110}$ | $1.481026 \times 10^{-999}$ |
| $t = 10$ | | | | |
| IT | 25 | 23 | 10 | 10 |
| Time | 0.146670 | 0.131086 | 0.127636 | 0.119472 |
| $\|E_k\|$ | 1.41975 | $8.750818 \times 10^{-340}$ | $2.365318 \times 10^{-497}$ | $2.365318 \times 10^{-497}$ |
| $t = 100$ | | | | |
| IT | 25 | 25 | 14 | 14 |
| Time | 0.155957 | 0.168545 | 0.254167 | 0.156128 |
| $\|E_k\|$ | 1.78698 | 1.65318 | $1.16148 \times 10^{-157}$ | $1.16148 \times 10^{-157}$ |

**Table 3** Results of comparisons in terms of the total number of mmms in Experiment 3

| Matrices | NM | CM | HM10 | IHM13 |
|---|---|---|---|---|
| $T_{100\times100}$ | 60 | 60 | 60 | 54 |
| $T_{200\times200}$ | 68 | 66 | 66 | 60 |
| $T_{300\times300}$ | 74 | 72 | 72 | 66 |
| $T_{400\times400}$ | 76 | 75 | 78 | 66 |
| $T_{500\times500}$ | 80 | 78 | 78 | 72 |
| $T_{600\times600}$ | 82 | 78 | 78 | 72 |
| $T_{700\times700}$ | 84 | 81 | 84 | 72 |
| $T_{800\times800}$ | 84 | 81 | 84 | 78 |
| $T_{900\times900}$ | 86 | 84 | 84 | 78 |
| $T_{1000\times1000}$ | 88 | 84 | 84 | 78 |

**Experiment 3** *In this experiment, the computational time of different schemes are compared for computing the regular inverse of ten Toeplitz matrices of different sizes with the stopping termination* $\|V_{k+1} - V_k\|_F \leq 10^{-8}$ *by applying the initial approximation* $V_0 = \frac{2}{\sigma_1^2+\sigma_r^2}A^T$. *Here, we applied the usual coefficients* (2.12)*, because we are working in double precision arithmetic. Results are displayed in* Tables 3 *and* 4 *in terms of the required number of mmms and the computational CPU time (in s).* $T_{n\times n}$ *denotes a Toeplitz matrix of order n.*

From the numerical results arranged in Tables 3 and 4, we can observe that, like the existing methods, the presented method shows a consistent convergence behavior. According to results arranged in Table 3, IHM13 is the best with respect to the total number of mmms. On the other hand, according to results arranged in Table 4, IHM13 requires greater CPU time with respect to CM and HM10 methods. This observation can be explained by the fact that it requires many smms per cycle. Let us mention that both the CM and HM10 methods do not require many smms and the IHM13 method requires 14 smms.

**Table 4** Results of comparisons in terms of the elapsed time in Experiment 3

| Matrices | NM | CM | HM10 | IHM13 |
|---|---|---|---|---|
| $T_{100\times100}$ | 0.020000 | 0.010000 | 0.010000 | 0.030000 |
| $T_{200\times200}$ | 0.100000 | 0.090000 | 0.080000 | 0.100000 |
| $T_{300\times300}$ | 0.350000 | 0.250000 | 0.250000 | 0.310000 |
| $T_{400\times400}$ | 0.680001 | 0.660001 | 0.670001 | 0.740001 |
| $T_{500\times500}$ | 1.380002 | 1.130002 | 1.050001 | 1.336003 |
| $T_{600\times600}$ | 2.030003 | 1.830002 | 1.700002 | 2.040003 |
| $T_{700\times700}$ | 2.990004 | 2.910004 | 2.690004 | 3.100004 |
| $T_{800\times800}$ | 4.209006 | 4.010006 | 3.920005 | 4.630007 |
| $T_{900\times900}$ | 5.990008 | 5.617008 | 5.447011 | 6.070009 |
| $T_{1000\times1000}$ | 8.252013 | 7.532011 | 7.130010 | 7.895011 |

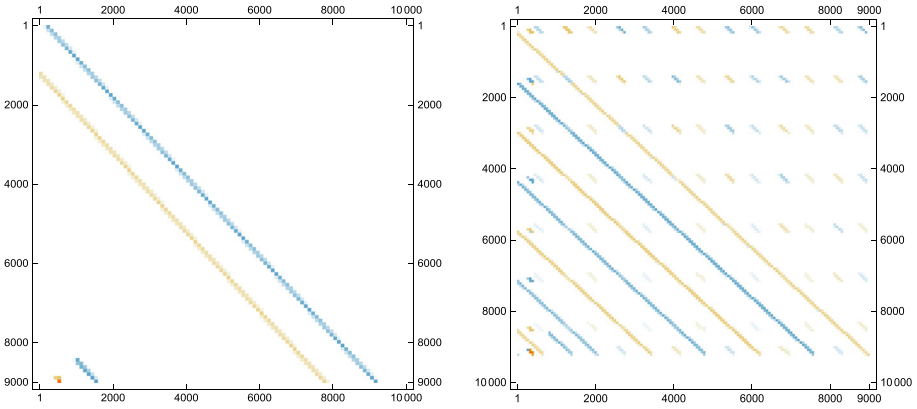


**Fig. 2** The sparsity pattern of the random matrices and the structure of their Moore–Penrose inverses in Experiment 4

Unlike these discussions for the full matrices, the following test reveal that how the new method is an interesting choice for computing outer generalized inverse when the input matrix and its generalized inverse are sparse.

**Experiment 4** *This experiment evaluates the applicability of the new method for finding the Moore–Penrose inverse of ten random sparse complex matrices (possessing sparse pseudo-inverses). These matrices are of the order* $m \times n = 9000 \times 10000$ *and defined as follows (in the programming package Mathematica):*

```
SeedRandom[1234];
number=10; m = 9000; n = 10000;
Table[
A[l] = SparseArray[{
Band[{1200, 1}, {m, n}] -> RandomComplex[],
Band[{10, 200}, {m, n}] -> {-2., RandomComplex[],
RandomComplex[{-2 - 3 I, 2 + 3 I}]},
Band[{-600, 1000}] -> -1.1,
Band[{-100, 500}] -> 600.1}, {m, n},
0.];
, {l, number}];
```

**Table 5** Results of comparisons in terms of the total number of mmms in Experiment 4

| Matrices | NM | CM | HM10 | IHM13 |
|---|---|---|---|---|
| # 1 | 90 | 87 | 90 | 78 |
| # 2 | 84 | 81 | 84 | 72 |
| # 3 | 86 | 84 | 84 | 78 |
| # 4 | 84 | 81 | 84 | 72 |
| # 5 | 86 | 84 | 84 | 78 |
| # 6 | 86 | 84 | 84 | 78 |
| # 7 | 84 | 81 | 84 | 78 |
| # 8 | 84 | 81 | 84 | 72 |
| # 9 | 82 | 81 | 78 | 72 |
| # 10 | 82 | 81 | 78 | 72 |

**Table 6** Results of comparisons in terms of the elapsed time in Experiment 4

| Matrices | NM | CM | HM10 | IHM13 |
|---|---|---|---|---|
| # 1 | 5.53313 | 5.7306 | 6.78076 | 2.79493 |
| # 2 | 4.11156 | 3.63804 | 3.04203 | 2.6071 |
| # 3 | 5.8704 | 4.79304 | 3.1237 | 8.44314 |
| # 4 | 3.50096 | 3.83185 | 2.90456 | 2.59475 |
| # 5 | 4.54084 | 4.94808 | 4.2539 | 4.32605 |
| # 6 | 4.83896 | 3.68288 | 5.42394 | 2.82721 |
| # 7 | 4.77746 | 3.28114 | 4.51089 | 4.00957 |
| # 8 | 5.06606 | 4.48237 | 2.41829 | 2.58207 |
| # 9 | 4.75548 | 3.71723 | 5.44153 | 3.77723 |
| # 10 | 5.32507 | 4.32103 | 5.53298 | 2.59711 |

*To save memory space and obtain acceptable computational times, there is no need in full saving of the matrix entries, and we must apply the command* `SparseArray[]`*, when working with sparse matrices. Herein,* $I = \sqrt{-1}$.

In this experiment, the initial approximation for the Moore–Penrose inverse is constructed by $V_0 =$ `Conjugate Transpose [A[j]]` $*$ `(1./((SingularValueList[A[j], 1][[1]])^2))` while the stopping termination is $\|V_{k+1} - V_k\|_\infty \leq 10^{-8}$. The sparsity pattern of the random matrices in this test and the sparse structure of their pseudo-inverse are brought forward in Fig. 2.

The results for the total number of mmms and the computational time are compared in Tables 5 and 6, respectively. It shows a clear advantage of the new scheme (2.8)–(2.9) in finding the Moore–Penrose inverse. In a nutshell, the new scheme (2.8)–(2.9) with 13th order of convergence based on least squares approach provides a fast way to compute the outer inverses when both the input matrix and its inverse are sparse.

## 5 Concluding summary

In this paper, we have proposed a new 13th-order iterative algorithm for finding the generalized inverse $A^{(2)}_{T,S}$. This algorithm is obtained by factorizing the hyperpower method of

order $p = 13$. Then the least squares technique and unconstrained minimization is applied to obtain the unknown coefficients in non-normalized form. Therefore, our proposed method became a new configuration for the original Hyperpower method of the order 13. Also, the method promotes new approach in finding the coefficients in arbitrary factorization of the hyperpower family. Convergence analysis of the proposed technique had also been studied.

We also noted that our proposed algorithm consumes only 6 mmms to reach 13th order, and hence achieves a leading efficiency index 1.53341. Numerical experiments support the theoretical details illustrated in this work. According to our experience, any non-normalized form suffers from a potential risk: increased number of smms, which eventually could disrupt the overall efficiency of the algorithm. However, such matrix multiplication rich algorithms are quite good choices for sparse matrices possessing sparse inverse.

At this moment, it would be interesting if one could generalize the idea of this paper for constructing a general family of iterations for matrix generalized inversion or to propose efficient structures with a proper and efficient factorization. Answering such queries could be considered for future works in this field of research.

**Acknowledgements** P. S. Stanimirović gratefully acknowledges support from the Research Project 174013 of the Serbian Ministry of Science. The authors are also thankful to referees for several interesting comments on an earlier version of this manuscript.

## Appendix

A piece of Mathematica code indicating how to obtain the unknown parameters in Algorithm 1 to achieve the 13th order of convergence.

```
ClearAll["Global`*"];
unknowns = {a0, a1, a2, e, b0, b1, b2, c0, c1, c2, d0, d1, d2};
fact1 = (a0 + a1 R^1 + a2 R^2 + R^2 (1/3 R  + R^2));
fact2 = (b0 + b1 R^1 + b2 R^2 + R^2 (1/3 R + R^2));
fact3 =  (c0 + c1 R^1 + c2 R^2 + R^2 (1/3 R + R^2));
sol = e*(fact1*fact2*fact3  + (d0 + d1 R^1 + d2 R^2)) // Expand;
S = Table[s[i] = Simplify@Coefficient[sol, R^i], {i, 1, 12, 1}];
s[0] = a0 b0  c0 e + d0 e;
solution = NMinimize[Sum[(s[j] - 1)^2, {j, 0, 12}],  unknowns,
WorkingPrecision -> 40, MaxIterations -> 150]; sol2 =
 (unknowns /.
solution[[2]]) FullSimplify@Rationalize[sol2, 10^-19]
```

## References

Ben-Israel A, Greville TNE (2003) Generalized inverses, 2nd edn. Springer, New York

Borle A, Lomonaco S (2016) Convergence acceleration of iterative methods for inverting real matrices using Frobenius norm minimization. In: 18th international symposium on SYNASC, Romania

Climent J-J, Thome N, Wei Y (2001) A geometrical approach on generalized inverses by Neumann-type series. Linear Algebra Appl 332–334:533–540

Du Croz JJ, Higham NJ (1992) Stability of methods for matrix inversion. IMA J Numer Anal 12:1–19

Ehrmann H (1959) Konstruktion und Durchführung von Iterationsverfahren höherer Ordnung. Arch Ration Mech Anal 4:65–88

Ghorbanzadeh M, Mahdiani K, Soleymani F, Lotfi T (2016) A class of Kung-Traub-type iterative algorithms for matrix inversion. Int J Appl Comput Math 2:641–648

Higham NJ (2002) Accuracy and stability of numerical algorithms. Society for Industrial and Applied Mathematics, 2nd ed
Hotelling H (1943) Some new methods in matrix calculation. Ann Math Stat 14:1–34
Hsuan F, Langenberg P, Getson A (1985) The {2}-inverse with applications in statistics. Linear Algebra Appl 70:241–248
Lawson C, Hanson R (1974) Solving least squares problems. Prentice-Hall, Englewood Cliffs
Li H-B, Huang T-Z, Zhang Y, Liu X-P, Gu T-X (2011) Chebyshev-type methods and preconditioning techniques. Appl Math Comput 218:260–270
Liu X, Zuo Z (2014) A higher-order iterative method for computing $A^{(2)}_{T,S}$. J Appl Math:7 (article ID 741368)
Nashed MZ (1976) Generalized inverses and applications. Academic Press, New York
Pan VY (2010) Newton's iteration for matrix inversion, advances and extensions. In: Matrix methods: theory, algorithms and applications. World Scientific, Singapore
Pan VY, Schreiber R (1991) An improved Newton iteration for the generalized inverse of a matrix with applications. SIAM J Sci Stat Comput 12:1109–1131
Pan VY, Kunin M, Rosholt R, Kodal H (2006) Homotopic residual correction processes. Math Comput 75:345–368
Petković MD, Petković MS (2015) Hyper-power methods for the computation of outer inverses. J Comput Appl Math 278:110–118
Sen SK, Prabhu SS (1976) Optimal iterative schemes for computing Moore–Penrose matrix inverse. Int J Syst Sci 8:748–753
Schulz G (1933) Iterative Berechnung der Reziproken matrix. Z Angrew Math Mech 13:57–59
Söderström T, Stewart GW (1974) On the numerical properties of an iterative method for comuting the Moore–Penrose generalized inverse. SIAM J Numer Anal 11:61–74
Soleymani F, Stanimirović PS, Haghani FK (2015) On hyperpower family of iterations for computing outer inverses possessing high efficiencies. Linear Algebra Appl 484:477–495
Soleymani F (2014) On finding robust approximate inverses for large sparse matrices. Linear Multilinear Algebra 62:1314–1334
Srivastava S, Gupta DK (2016) A two-step iterative method and its acceleration for outer inverses. Sādhanā 41:1179–1188
Srivastava S, Gupta DK, Stanimirović P, Singh S, Roy F (2017) A hyperpower iterative method for computing the generalized Drazin inverse of Banach algebra element. Sādhanā 42:625–630
Stanimirović PS, Soleymani F (2014) A class of numerical algorithms for computing outer inverses. J Comput Appl Math 263:236–245
Traub JF (1964) Iterative methods for solution of equation. Prentice-Hall, Englewood Cliffs
Trott M (2006) The mathematica guidebook for numerics. Springer, New York
Yanai H, Takeuchi K, Takane Y (2011) Projection matrices, generalized inverse matrices, and singular value decomposition. Springer, New York