Int. J. Appl. Comput. Math (2016) 2:641–648
DOI 10.1007/s40819-015-0083-1

ORIGINAL PAPER

A Class of Kung–Traub-Type Iterative Algorithms for Matrix Inversion

Mohammad Ghorbanzadeh[1] · **Katayoun Mahdiani**[2] · **Fazlollah Soleymani**[2] · **Taher Lotfi**[2]

Published online: 12 August 2015

Abstract Kung–Traub (J ACM 21:643–651, 1974) constructed two optimal general iterative methods without memory for finding solution of nonlinear equations. In this work, we are going to show that one of them can be applied for matrix inversion. It is observed that the convergence order 2^m can be attained using $2m$ matrix–matrix multiplications. Moreover, a method with the efficiency index $10^{1/6} \approx 1.4677$ will be furnished. To justify that our procedure works efficiently, some numerical problems are included.

Keywords Matrix inversion · Initial matrix · Schulz method · Kung–Traub

Mathematics Subject Classification 15A09 · 65F30

Introduction

Recently, several one or multipoint iterative methods of various convergence orders have been extended to approximate the inverse of a given matrix [1,2]. Note that throughout this paper and as for prerequisite, the reader is expected to be familiar with iterative methods for solving nonlinear equations [3].

Based on our best knowledge, Schulz and Hotelling are pioneer in the application of iterative methods for computing matrix inverses [4,5]. They managed to extend Newton's method to compute matrix inverses of regular matrices. Some further discussions, extensions and modifications have then been done for computing inner inverses in [6], Drazin inverse in [7], Moore–Penrose inverse in [8], weighted Moore–Penrose inverse in [9] and interval versions in [10].

✉ Fazlollah Soleymani
fazl_soley_bsb@yahoo.com; fazlollah.soleymani@gmail.com

[1] Department of Mathematics, Imam Reza International University, Mashhad, Iran

[2] Department of Applied Mathematics, Islamic Azad University, Hamedan Branch, Hamedan, Iran

In this work and motivated by the above-mentioned works, we investigate the applicability of Kung–Traub's iterative class of methods [11], given in what follows:

$$\begin{cases} \Theta_{1,n}(F)(V_n) = V_n - \frac{F(V_n)}{F'(V_n)}, \quad n = 0, 1, 2, \ldots, \\ \qquad\qquad \vdots \\ \Theta_{j,n}(F)(V_n) = Q_{j-1}(0), \end{cases} \tag{1}$$

for $j = 1, \ldots, m$ for matrix inversion, whereas $Q_j(U)$ is the inverse Hermite interpolatory polynomial of degree at most j such that

$$\begin{cases} Q_j(F(V_n)) = V_n, \\ Q_j'(F(V_n)) = \frac{1}{F'(V_n)}, \\ Q_j\Big(F(\Theta_{k,n}(F)(V_n))\Big) = \Theta_{k,n}(V_n), \quad k = 1, \ldots, j-1. \end{cases} \tag{2}$$

As an instance, the three-step eighth order method extracted from (1) could be written as

$$\begin{cases} \Theta_{1,n} = V_n - \frac{F(V_n)}{F'(V_n)}, \quad n = 0, 1, 2, \ldots, \\ \Theta_{2,n} = \Theta_{1,n} - \frac{F(\Theta_{1,n})}{\Big(F(V_n)-F(\Theta_{1,n})\Big)^2}\frac{F(V_n)}{F'(V_n)}, \\ \Theta_{3,n} = \Theta_{2,n} - \left(\frac{1}{F(V_n)-F(\Theta_{2,n})}\left(\frac{V_n-\Theta_{2,n}}{F(V_n)-F(\Theta_{2,n})} - \frac{1}{F'(V_n)}\right) - \frac{F(\Theta_{1,n})}{F'(V_n)(F(V_n)-F(\Theta_{2,n}))^2}\right) \\ \qquad \times \frac{F(\Theta_{1,n})F^2(V_n)}{F(\Theta_{1,n})-F(\Theta_{2,n})}. \end{cases} \tag{3}$$

Kung–Traub have developed two general methods without memory for finding simple roots of nonlinear equations in [11]. However, both of them cannot be applied in matrix inversion. Here, we deal with one of their classes, i.e. (1). In fact by using (1), we are able to construct a general fast class of algorithms with a simple but efficient structure.

The rest of this work is organized as follows. "Derivation" section concerns applying general Kung–Traub's class (1) for approximating matrix inverses. The extension of the proposed iterative class for Moore–Penrose inverse will be discussed in "Further Discussions" section along with a new efficient formulation. Some numerical performances are illustrated in "Numerical Performances" section. Finally, the last section concludes the paper.

Derivation

We now wish to apply the general method (1) for computing the approximate inverses of a given nonsingular matrix A. To this end, we consider the nonlinear matrix equation

$$F(V) := V^{-1} - A. \tag{4}$$

It is clear that $F(A^{-1}) = 0$. Considering (4), we first apply (3) to produce the matrix sequence $\{V_n\}$, as follows:

$$\begin{cases} V_0 \text{ is given}, \\ \Theta_{1,n} = V_n\,(2I - AV_n), \\ \Theta_{2,n} = \Theta_{1,n}\Big(2I - A\Theta_{1,n}\Big), \\ V_{n+1} = \Theta_{3,n} = \Theta_{2,n}\Big(2I - A\Theta_{2,n}\Big), \end{cases} \tag{5}$$

which later be shown that it converges to A^{-1}. To be more precise, we obtain the following second-, fourth-, and eighth-order iterative methods for computing the inverse of the matrix A, respectively:

$$V_{n+1} = V_n\,(2I - AV_n), \tag{6}$$

which is known as the Schulz quadratically convergent method, and

$$V_{n+1} = V_n\,(2I - AV_n)\Big(2I - AV_n(2I - AV_n)\Big), \tag{7}$$

which is known as the double Schulz method, and

$$\begin{aligned} V_{n+1} = {} & V_n\,(2I - AV_n)\Big(2I - AV_n(2I - AV_n)\Big) \\ & \times\Big(2I - AV_n(2I - AV_n)\big(2I - AV_n(2I - AV_n)\big)\Big), \end{aligned} \tag{8}$$

which is known as the method of Householder [12]. Inductively, with $\Theta_{1,n} = V_n\,(2I - AV_n)$, for an m-step method extracted from the class (1), we have

$$V_{n+1} = \Theta_{m,n} = \Theta_{m-1,n}\Big(2I - A\Theta_{m-1,n}\Big), \quad m = 2, 3, \ldots. \tag{9}$$

The general class of algorithms (9) could be re-written after proper factorizations in the following form

$$V_{n+1} = V_n(I + Y_n)(I + Y_n^2)\cdots(I + Y_n^{2^m}), \quad m = 1, 2, \ldots, \tag{10}$$

wherein $Y_n = I - AV_n$, to provide a general class of matrix methods of order $q = 2^m$, at the cost of $2m$ matrix multiplication per cycle whose efficiency is an increasing function of m.

Now, it must be discussed theoretically that the new scheme is computationally efficient. It is well-known that the efficiency index could be expressed by

$$EI = p^{\frac{1}{\theta}}, \tag{11}$$

which is recently developed in [13] for such matrix methods and it has been proved that an iteration of order p will *almost* require the following number of iterations to converge in floating point arithmetic

$$s \approx 2\log_p \kappa_2(A). \tag{12}$$

In (12), $\kappa_2(A)$ is the condition number of an input matrix in l_2 and defined generally as $\kappa_2(A) = \|A\|_2\|A^{(2)}_{T,S}\|_2$, where $A^{(2)}_{T,S}$ is the generalized outer inverse.

θ in (11) is the whole computational cost of an algorithm including matrix–matrix products and the stopping criterion. On the other hand, a reliable termination for finding approximate inverses of regular matrices is given by

$$\|I - V_nA\|_* \le \epsilon. \tag{13}$$

Note that V_nA should be computed further per computing cycle. This adds one more multiplication per cycle. Accordingly, we now compare the computational efficiency indices of different methods with various orders extracted from (10) by choosing (13). This is illustrated in Fig. 1. It shows that the growth in the number of steps (m), which ends in higher convergence order, will produce higher efficiency indices for computing inverses.

Hence, the general class of methods in this work is efficient and consists of several existing well-known schemes of the literature.

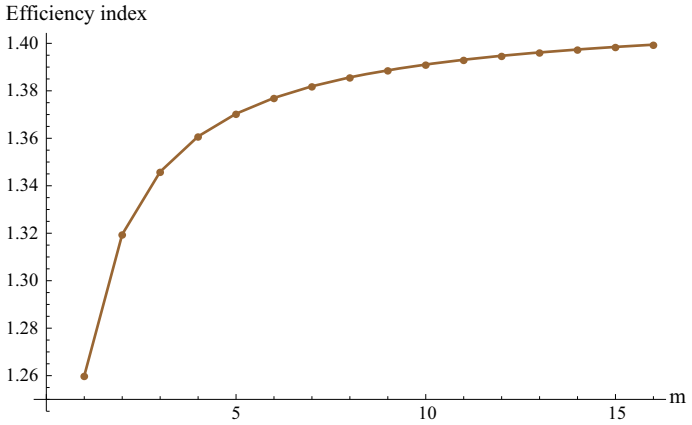

Fig. 1 The comparison of efficiency indices for different methods of various orders extracted from (10)

The general class of methods (9) or (10) needs a good initial approximation, namely V_0, ideally close to A^{-1} which is addressed in the following theorem.

Theorem 2.1 *If*

$$\|I - AV_0\| < 1, \tag{14}$$

then the class of iterative methods given by (9) *or* (10) *produce a convergent sequence of matrices* $\{V_n\}$ *to* A^{-1}. *Moreover, it is of order* 2^m.

Proof The proof is similar to those taken in [13]. □

An interesting initial matrix was introduced and discussed in [14] is as follows

$$V_0 = \alpha A^*, \tag{15}$$

where $0 < \alpha < \frac{2}{||A||_2^2}$. Because of the fact that, the computation of matrix norm $\|\cdot\|_2$, is difficult for large matrices, an alternative bound for α could be considered in what follows $0 < \alpha < \frac{2}{\sigma_{max}^2}$. Another easy to compute initial approximation which is also useful even in the case of finding Moore–Penrose inverses is

$$V_0 = \frac{1}{\|A\|_F} A^*. \tag{16}$$

Remark 2.2 The matrix iterations extracted from (10) of any arbitrary order are matrix–matrix multiplication based iterative algorithms. The most important application of such iterative schemes is in finding the (pseudo) inverses of $\mathcal{H}$-matrices, i.e. the large sparse matrices possessing sparse inverses emerging in scientific problems, see e.g. [15].

Further Discussions

Let us now extend the contributed method (10) for calculating the pseudo-inverse $A^\dagger$. That is, we must analytically reveal that the sequence $\{V_n\}_{n=0}^{n=\infty}$ generated by the iterative Schulz-type class (10), for any $n \geq 0$, tends to the Moore–Penrose inverse as well. Note that for the

rectangular complex matrix $A \in \mathbb{C}^{l\times k}$ and the sequence $\{V_n\}_{n=0}^{n=\infty}$ generated by (9), for any $n \geq 0$, V_n reads

$$(AV_n)^* = AV_n, \quad (V_nA)^* = V_nA, \quad V_nAA^\dagger = V_n, \quad A^\dagger AV_n = V_n. \tag{17}$$

Theorem 3.1 *For the complex matrix $A \in \mathbb{C}^{l\times k}$, and the sequence $\{V_n\}_{n=0}^{n=\infty}$ generated by (10), for any $n \geq 0$, using the initial approximation (15), the sequence is converged to the pseudo-inverse $A^\dagger$ with 2^m order of convergence.*

Proof If we consider $E_n^\dagger = V_n - A^\dagger$ as the error matrix, then the steps of the proof would be similar to those taken in [13]. Hence, we only state that it reads the following error bound

$$\|E_{n+1}^\dagger\| \leq \|A\|^{2^m}\|A^\dagger\|\|E_n^\dagger\|^{2^m}. \tag{18}$$

That is, $\|V_{n+1} - A^\dagger\| \to 0$. The proof is complete. □

It should be commented that although the families of Kung–Traub are optimal in solving nonlinear equations, their counterparts in matrix inversion are not any more optimal and does not follow the optimality condition of Kung–Truab [11]. To be more precise, matrix methods of higher order with the same number of matrix–matrix multiplications can be constructed.

Mathematically speaking, the eighth order scheme of Kung–Traub requires 6 matrix–matrix multiplications. Now, we point out that a much more efficient method of order 10 requiring the same number of multiplications can be derived from proper factorization of the hyperpower scheme of order 10 [16]. That is to say, it would not be tough to derive that the hyperpower method of order 10:

$$V_{n+1} = V_n\left(I + R_n + \cdots + R_n^9\right), \quad R_n = I - AV_n, \tag{19}$$

can be re-written in a much simpler form as follows

$$V_{n+1} = V_n\left(I + R_n\right)\left[(I + \alpha R_n^2 + R_n^4)(I + \beta R_n^2 + R_n^4)\right], \tag{20}$$

with only 6 matrix–matrix multiplications whereas $\alpha = \frac{1}{2}\left(1 - \sqrt{5}\right)$) and $\beta = \frac{1}{2}\left(1 + \sqrt{5}\right)$. We name this method as MHP10. This shows that the new formulation hits the computational efficiency index $10^{1/6} \approx 1.4677$ even without considering any burden on the stopping criterion.

We also remark that the extension and use of the above methods for generalized outer inverses would be simple and could be done according to the recent work [17].

Numerical Performances

This section investigates problems related to the numerical precision of the matrix inverse-finding, using Mathematica 10 built-in precision, [18]. For numerical comparisons in this section, we have used the class (10) with various convergence rates, i.e., the second order scheme denoted by M2, the fourth order scheme denoted by M4, the eighth order scheme denoted by M8 and the sixteenth order scheme denoted by M16. As the programs were running, we found the running time using the command `AbsoluteTiming[]` to report the elapsed CPU time (in seconds) for the experiments. The computer specifications are Microsoft 7 Ultimate, 64-bit Operating System, with 8GB of RAM.

Example 4.1 The aim of this example is to compare different methods for finding approximate inverses of a randomly generated dense matrix of the size 1000×1000 as follows:

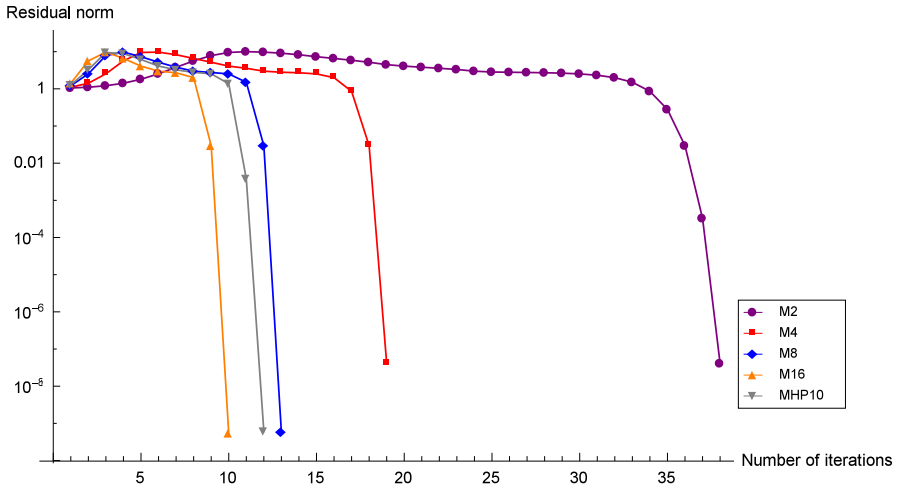

Fig. 2 The comparison of different methods in terms of the number of iterations for Example 4.1

Table 1 The results of comparisons for Example 4.1 in terms of the elapsed time

Methods	M2	M4	M8	M16	MHP10
Time (in seconds)	4.805809	4.173609	3.868807	3.588006	3.619206

```
d = 1000; SeedRandom[123]; A = RandomReal[{-10, 10}, {d, d}];
```

In this test, the prescribed tolerance is $\epsilon = 10^{-4}$ and the maximum number of iterations set to 100 using the stopping termination (13) in the l_∞.

We used the initial matrix (16) for starting the methods. The results of comparisons in terms of the number of iterations along with the residual norm has been illustrated in Fig. 2, which shows the applicability of the iterative scheme (10), while the computational times are provided in Table 1. They all agree with the discussions in "Derivation" and "Further Discussions" sections.

Example 4.2 This test compares the behavior of different methods for finding the Moore–Penrose inverse of a large sparse matrix of the size $10,000 \times 11,000$ possessing a sparse inverse as follows ($I = \sqrt{-1}$):

```
l = 10,000; k = 11,000; SeedRandom[123];
A = SparseArray[{Band[{2000, 1200}, {l, k}] -> Random[] - I,
    Band[{100, 6000}, {m, n}] -> {1.1, -Random[]},
    Band[{-200, 4000}] -> -0.02, Band[{4500, -6000}]
    -> 0.1}, {l, k}, 0.];
```

Here, the initial matrix is $V_0 = \frac{2}{\|A\|_F} A^*$ and the same conditions as in Example 4.1 but the different stopping criterion as $\frac{\|V_{n+1} - V_n\|_F}{\|V_{n+1}\|_F} \leq \epsilon$. The results are given in Table 2 and Fig. 3.

It is observed that for larger matrices the presented method (M16) of the class (10) have an edge over similar existing methods. Numerical examples in general confirmed the discussions of the paper.

Table 2 The results of comparisons for Example 4.2 in terms of the elapsed time

Methods	M2	M4	M8	M16	MHP10
Time (in seconds)	2.995205	2.277604	2.227205	2.012404	2.028004

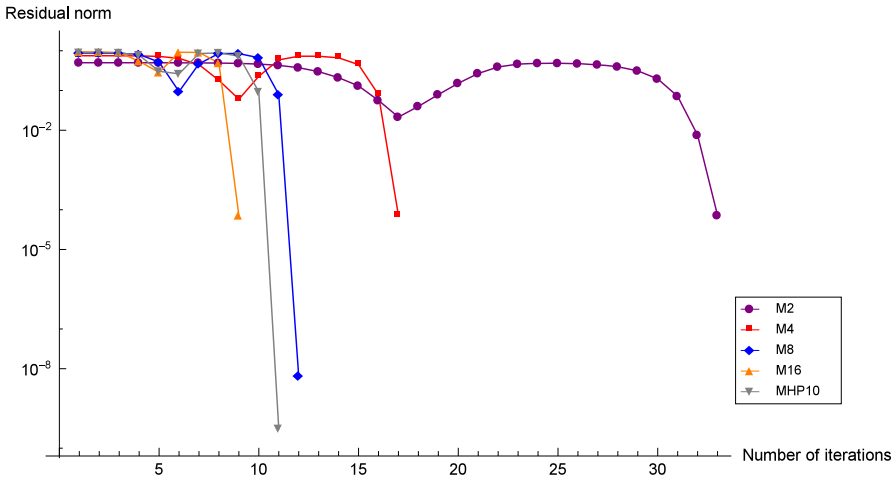

Fig. 3 The comparison of different methods in terms of the number of iterations for Example 4.2

Conclusion and Outlook

We have introduced a new class of methods for matrix inversion with arbitrary convergence rate according to the well-known derivative-involved class of Kung–Traub. It has been shown that the new methods possess higher efficiency indices in contrast to the existing methods of the same type. Theoretical analysis showed that this class has the 2^m rate of convergence using only $2m$ matrix–matrix products per cycle.

It is interesting to note that the new Schulz-type iterations are advocated in case of the $\mathcal{H}$-matrices or structured matrices, since their structure admits cheap matrix operations.

The further outlook is to accelerate this general family of iteration methods via scaling. That is to say, to speed up the whole procedure (specially at the beginning of the process) without further imposing matrix–matrix products.

Acknowledgments This research was supported by the Islamic Azad University, Hamedan Branch, Hamedan, Iran. The authors are thankful to referee for some interesting comments on an earlier version of this manuscript.

References

1. Weiguo, L., Juan, L., Tiantian, Q.: A family of iterative methods for computing Moore-Penrose inverse of a matrix. Linear Algebra Appl. **438**, 47–56 (2013)
2. Soleymani, F.: On a fast iterative method for approximate inverse of matrices. Commun. Korean Math. Soc. **28**, 407–418 (2013)
3. Traub, J.F.: Iterative Methods for the Solution of Equations. Prentice Hall, New York (1964)

4. Schulz, G.: Iterative berechnung der reziproken matrix. Zeitschrift fr Angewandte Mathematik und Mechanik **13**, 57–59 (1933)
5. Hotelling, H.: Analysis of a complerx statistical variable into principal components. J. Educ. Psychol. **24**, 498–520 (1933)
6. Li, W., Li, Z.: A family of iterative methods for computing the approximate inverse of a square matrix and inner inverse of a non-square matrix. Appl. Math. Comput. **215**, 3433–3442 (2010)
7. Soleymani, F., Stanimirović, P.S.: A higher order iterative method for computing the Drazin inverse. Sci. World J., vol. **2013**, Article ID 708647, 11 p
8. Soleymani, F.: A fast convergent iterative solver for approximate inverse of matrices. Numer. Linear Algebra Appl. **21**, 439–452 (2014)
9. Soleymani, F., Stanimirović, P.S., Zaka Ullah, M.: On an accelerated iterative method for weighted Moore-Penrose inverse. Appl. Math. Comput. **222**, 365–371 (2013)
10. Zhang, X., Cai, J., Wei, Y.: Interval iterative methods for computing Moore-Penrose inverse. Appl. Math. Comput. **183**, 522–532 (2006)
11. Kung, H.T., Traub, J.F.: Optimal order of one-point and multipoint iteration. J. ACM **21**, 643–651 (1974)
12. Householder, A.S.: The Theory of Matrices in Numerical Analysis. Dover Publications, New York (1964)
13. Soleymani, F.: On finding robust approximate inverses for large sparse matrices. Linear Multilinear Algebra **62**, 1314–1334 (2014)
14. Ben-Israel, A., Cohen, D.: On iterative computation of generalized inverses and associated projections. SIAM J. Numer. Anal. **3**, 410–419 (1966)
15. Alpert, B., Beylkin, G., Coifman, R., Rokhlin, V.: Wavelet-like bases for the fast solution of second-kind integral equations. SIAM J. Sci. Comput. **14**, 159–184 (1993)
16. Climent, J.-J., Thome, N., Wei, Y.: A geometrical approach on generalized inverses by Neumann-type series. Linear Algebra Appl. **332–334**, 533–540 (2001)
17. Stanimirović, P.S., Soleymani, F.: A class of numerical algorithms for computing outer inverses. J. Comput. Appl. Math. **263**, 236–245 (2014)
18. Trott, M.: The Mathematica Guide-Book for Numerics. Springer, New York (2006)