RESEARCH

Open Access

CrossMark

Several numerical methods for computing unitary polar factor of a matrix

Fazlollah Soleymani[1], Farhad Khaksar Haghani[2] and Stanford Shateyi[3*]

*Correspondence:
stanford.shateyi@univen.ac.za
[3]Department of Mathematics and
Applied Mathematics, University of
Venda, Thohoyandou, 0950, South
Africa
Full list of author information is
available at the end of the article

Abstract

We present several numerical schemes for computing the unitary polar factor of rectangular complex matrices. Error analysis shows high orders of convergence. Many experiments in terms of number of iterations and elapsed times are reported to show the efficiency of the new methods in contrast to the existing ones.

MSC: 65F30

Keywords: iterative methods; polar decomposition; numerical methods; polar factor; Hermitian; order of convergence

1 Preliminaries

Let $\mathbb{C}^{m \times n}$ ($m \geq n$) denote the linear space of all $m \times n$ complex matrices. The polar decomposition of a complex matrix $A \in \mathbb{C}^{m \times n}$ could be defined as

$$A = UH, \qquad U^*U = I_r, \qquad \text{rank}(U) = r = \text{rank}(A), \qquad (1)$$

where H is a Hermitian positive semi-definite matrix of order n and $U \in \mathbb{C}^{m \times n}$ is a sub-unitary matrix [1]. A matrix U is sub-unitary if $\|Ux\|_2 = \|x\|_2$ for any $x \in \mathcal{R}(U^H) = \mathcal{N}(U)^\perp$, where $\mathcal{R}$ and $\mathcal{N}$ denote the linear space spanned by columns of matrix X (range of X) and the null space of matrix X, respectively. Note that if $\text{rank}(A) = n$ then $U^*U = I_n$, and U is an orthonormal Stiefel matrix.

The Hermitian factor H is always unique and can be written as $(A^*A)^{\frac{1}{2}}$, while the unitary factor U is unique if A is nonsingular; see for more [2].

It is required to remark that the polar and matrix sign decompositions are intimately connected [3]. For example, Roberts' integral formula [4],

$$\text{sign}(A) = \frac{2}{\pi} \int_0^\infty \left(t^2 I + A^2\right)^{-1} dt, \qquad (2)$$

has an analog in

$$U = \frac{2}{\pi} \int_0^\infty \left(t^2 I + A^*A\right)^{-1} dt. \qquad (3)$$

These integral formulas reveal that any property or iterative method involving the matrix sign function can be transformed into one for the polar decomposition by replacing A^2 via A^*A, and *vice versa*.

Practical interest in the polar decomposition stems mainly from the fact that the unitary polar factor of A is the nearest unitary matrix to A in any unitarily invariant norm. The polar decomposition is therefore of interest whenever it is required to orthogonalize a matrix [5]. To obtain more background in this topic, one may refer to [6–9].

Now we briefly review some of the most important iterative matrix methods for computing polar decomposition. Among many iterations (see *e.g.* [10] and the references therein) available for finding U, the most practically useful one is the Newton iteration. The method of Newton introduced for polar decomposition in [5] is as follows:

$$U_{k+1} = \frac{1}{2}\left(U_k + U_k^{-*}\right), \tag{4}$$

for the square nonsingular cases and the following alternative for general rectangular cases [11]:

$$U_{k+1} = \frac{1}{2}\left(U_k + U_k^{\dagger*}\right), \tag{5}$$

wherein $U^{\dagger}$ stands for the Moore-Penrose generalized inverse. Note that, throughout this work, U_k^{-*} stands for $(U_k^{-1})^*$. Similar notations are used as well.

Remark 1.1 We point out that here we focus mainly on computing the unitary polar factor of rectangular matrices, since the high-order methods discussed in this work will not require the computation of pseudo-inverse and is better than the corresponding Newton's version (5), which requires the computation of one pseudo-inverse per computing cycle.

Recently, an efficient cubically convergent method has been introduced in [12] as follows:

$$U_{k+1} = U_k[38I + 42Y_k][9I + 60Y_k + 11Z_k]^{-1}, \tag{6}$$

where $Y_k = U_k^* U_k$, $Z_k = Y_k Y_k$.

An (enough close) initial matrix U_0 must be employed in such matrix fixed-point type methods to ensure convergence. Such an approximation/guess for the unitary factor of the rectangular complex matrices can be constructed by

$$U_0 = \frac{1}{\alpha}A, \tag{7}$$

where $\alpha > 0$ is an estimate of $\|A\|_2$. This is known as one of the good ways in the literature for constructing an initial value to ensure the convergence of iterative Newton-type methods for finding the unitary polar factor of A.

The other sections of this paper are organized as follows. In Section 2, we derive an iteration function for polar decomposition. Next, Section 3 discusses the convergence properties of this method. It is revealed that the rate of convergence is six since the proposed formulation transforms the singular values of the approximated matrices produced per cycle with a sixth rate to unity (one). This discloses that our method is quite rapid. Several other new iterative methods are constructed in Section 4. Many numerical experiments are provided to support the theoretical aspects of the paper in Section 5. Finally, conclusions are drawn in Section 6.

2 A numerical method

The procedure of constructing a new iterative method for U, is to apply a zero-finder on a particular map [13]. That is, solving the following nonlinear (matrix) equation:

$$F(U) := U^* U - I = 0, \tag{8}$$

where I is the identity matrix, by an appropriate root-finding method could yield novel schemes.

To that end, we first introduce the following iterative expression for finding the simple zeros of nonlinear equations:

$$\begin{cases} y_k = u_k - \frac{20 - 9L(u_k)}{20 - 19L(u_k)} \frac{f(u_k)}{f'(u_k)}, \\ u_{k+1} = y_k - \frac{f(y_k)}{f'(y_k)}, \end{cases} \tag{9}$$

with $L(u_k) = \frac{f''(u_k)f(u_k)}{f'(u_k)^2}$. This is a combination of the cubical method proposed in [12] and the quadratically convergent Newton's method.

Theorem 2.1 *Let $\alpha \in D$ be a simple zero of a sufficiently differentiable function $f : D \subseteq \mathbb{C} \to \mathbb{C}$ for an open interval D, which contains x_0 as an initial approximation of α. Then the iterative expression* (9) *has sixth order of convergence.*

Proof The proof is based on Taylor expansions of the function f around the appropriate points and would be similar to those taken in [14]. As a consequence, it is skipped over. □

Here using (9) for solving $u^2 - 1 = 0$, we have the following iteration in the reciprocal form:

$$u_{k+1} = \frac{684u_k + 5{,}316u_k^3 + 5{,}876u_k^5 + 924u_k^7}{81 + 2{,}524u_k^2 + 6{,}990u_k^4 + 3{,}084u_k^6 + 121u_k^8}, \quad k = 0, 1, \dots. \tag{10}$$
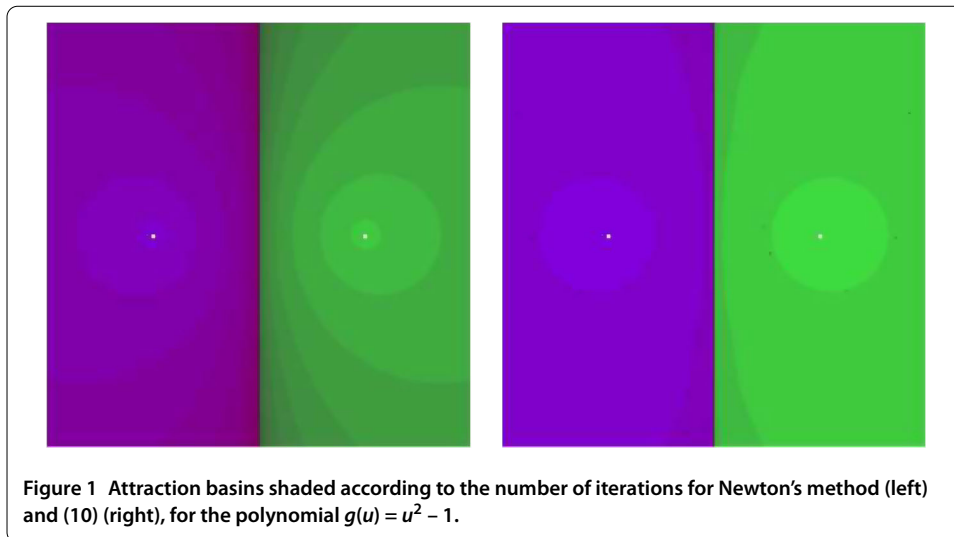
The iteration obtained after applying a nonlinear equation solver on the mapping (8) and its reciprocal, could be used for polar decomposition. But here, the experimental results show that the reciprocal form (10) is more stable in the presence of round-off errors.

Drawing the attraction basins [15] of (10) for finding the solution of the polynomial equation $u^2 - 1 = 0$ in the complex plane reveals that the application of (9) for finding matrix sign function and consequently the unitary polar factor has global convergence. This is done in Figure 1 on the rectangle $[-2, 2] \times [-2, 2]$.

By taking into account this global convergence behavior, we extend (10) as follows:

$$U_{k+1} = U_k[684I + 5{,}316Y_k + 5{,}876Z_k + 924W_k]$$
$$\times [81I + 2{,}524Y_k + 6{,}990Z_k + 3{,}084W_k + 121L_k]^{-1}, \tag{11}$$

where U_0 is chosen by (7) (or its simplest form $U_0 = A$) and $Y_k = U_k^* U_k$, $Z_k = Y_k Y_k$, $W_k = Y_k Z_k$, and $L_k = Y_k W_k$. The iteration algorithm (11) converges to the unitary polar factor under some conditions. These discussions will be presented in the next section.

Figure 1 Attraction basins shaded according to the number of iterations for Newton's method (left) and (10) (right), for the polynomial $g(u) = u^2 - 1$.

3 Convergence properties

This section is dedicated to the convergence properties of (11) for finding the unitary polar factor of A.

Theorem 3.1 *Assume that $A \in \mathbb{C}^{m \times n}$ is an arbitrary matrix. Then the matrix iterates $\{U_k\}_{k=0}^{k=\infty}$ of (11) converge to U.*

Proof The proof of this theorem follows the lines of the proofs given in [16]. As such, it is skipped over. □

Theorem 3.2 *Let $A \in \mathbb{C}^{m \times n}$ be an arbitrary matrix. Then the new method (11) is of sixth order to find the unitary polar factor of A.*

Proof The proposed scheme (11) transforms the singular values of U_k according to the following map:

$$\sigma_i^{(k+1)} = \sigma_i^{(k)}\left[684 + 5{,}316\sigma_i^{(k)^2} + 5{,}876\sigma_i^{(k)^4} + 924\sigma_i^{(k)^6}\right]$$
$$\times \left[81 + 2{,}524\sigma_i^{(k)^2} + 6{,}990\sigma_i^{(k)^4} + 3{,}084\sigma_i^{(k)^6} + 121\sigma_i^{(k)^8}\right]^{-1}, \tag{12}$$

and it leaves the singular vectors invariant. From equation (12), it is enough to show that convergence of the singular values to unity possesses a sixth order of convergence for $k \geq 1$. Thus, we arrive at

$$\frac{\sigma_i^{(k+1)} - 1}{\sigma_i^{(k+1)} + 1} = -\frac{(-1 + \sigma_i^{(k)})^6(-9 + 11\sigma_i^{(k)})^2}{(1 + \sigma_i^{(k)})^6(9 + 11\sigma_i^{(k)})^2}. \tag{13}$$

Taking absolute values from both sides of (13), one gets the following:

$$\left|\frac{\sigma_i^{(k+1)} - 1}{\sigma_i^{(k+1)} + 1}\right| \leq \left(\frac{-9 + 11\sigma_i^{(k)}}{9 + 11\sigma_i^{(k)}}\right)^2 \left|\frac{\sigma_i^{(k)} - 1}{\sigma_i^{(k)} + 1}\right|^6. \tag{14}$$

This demonstrates the sixth rate of convergence for the proposed numerical algorithm (11). Consequently, the proof is complete.   □

Remark 3.1 The presented method is not a member of the Padé family of iterations given in [17] (and discussed deeply in [18]), with global convergence. As a result, it is interesting from both theoretical and computational point of views.

The new formulation (11) is quite rapid, but there is still a way for speeding up the whole process via an acceleration technique given for Newton's method in [5], known as scaling. Some important scaling approaches were derived in different norms as comes next; we have

$$\theta_k = \left(\frac{\|U_k^\dagger\|_2}{\|U_k\|_2} \right)^{\frac{1}{2}}, \tag{15}$$

where $\| \cdot \|_2$ is the spectral norm. This scale factor is optimal in the given U_k, since (15) minimizes the next error $\|U_{k+1} - U\|_2$. Unfortunately, to determine the scale factor (15), one needs to compute two extreme singular values of U_k at each iteration. To save the cost of computing the extreme singular values, one might approximate the scaling parameter as in the following [19]:

$$\theta_k = \left(\frac{\|U_k^\dagger\|_F}{\|U_k\|_F} \right)^{\frac{1}{2}} \tag{16}$$

or

$$\theta_k = \left(\frac{\|U_k^{-1}\|_1 \|U_k^{-1}\|_\infty}{\|U_k\|_1 \|U_k\|_\infty} \right)^{\frac{1}{4}}. \tag{17}$$

Another relatively inexpensive scaling factor is [20]

$$\theta_k = \left| \det(U_k) \right|^{-1/n}. \tag{18}$$

The complex modulus of the determinant in this choice is inexpensively obtained from the same matrix factorization used to calculate U_k^{-1}.

Finally in this section, the new scheme can be expressed in the following accelerated form as well:

$$\begin{cases} \text{Compute } \theta_k \text{ (for example) by (16),} \quad k \geq 0, \\ M_k = 81I + 2{,}524\theta_k^2 Y_k + 6{,}990\theta_k^4 Z_k + 3{,}084\theta_k^6 W_k + 121\theta_k^8 L_k, \\ U_{k+1} = \theta_k U_k [684I + 5{,}316\theta_k^2 Y_k + 5{,}876\theta_k^4 Z_k + 924\theta_k^6 W_k]M_k^{-1}. \end{cases} \tag{19}$$

4 Some other iterative methods

As discussed in the preceding sections, the construction of the iterative methods for finding the unitary polar factor of a matrix mainly relies on the nonlinear equation solver which is going to be applied on the mapping (8).

Now, some may question that the construction (9) is straightforward, since it is the combination of two already known methods. It is here stated that the main goal is to attain a

new scheme for a polar decomposition which has *global convergence behavior* and is *new*, *i.e.*, it is not a member of the Padé family of iterations (or its reciprocal). So, the novelty and usefulness of (9) in terms of solving nonlinear equations is not of main interest here and the importance is focused on providing a novel and useful scheme for finding the unitary polar factor.

To construct some other new and useful iterative methods for finding the unitary polar factor of a matrix, we could again use the first sub-step of (9) along with different kinds of approximation for the newly appearing first derivative in the second sub-step. As such, we could derive the following nonlinear equation solver:

$$
\begin{cases}
y_k = u_k - \frac{20-9L(u_k)}{20-19L(u_k)} \frac{f(u_k)}{f'(u_k)}, \\
u_{k+1} = y_k - \frac{f(y_k)}{f[u_k, y_k]},
\end{cases}
\tag{20}
$$

wherein $f[x_k, y_k]$ is the two-point divided difference. Note again that pursuing the optimality conjecture of Kung-Traub or usefulness of the iterative method in terms of solving nonlinear equation is not the only cutting-edge factor, since the most eminent factor is in designing a new scheme for unitary polar factor with global convergence behavior. An application of (20) to equation (8) results in the following *fourth-order* scheme for the unitary polar factor:

$$
U_{k+1} = U_k[47I + 102Y_k + 11Z_k][9I + 98Y_k + 53Z_k]^{-1}.
\tag{21}
$$

At this moment, by applying a similar secant-like strategy in a third sub-step after (20), one may design the following *seventh-order* scheme:

$$
\begin{cases}
y_k = u_k - \frac{20-9L(u_k)}{20-19L(u_k)} \frac{f(u_k)}{f'(u_k)}, \\
z_k = y_k - \frac{f(y_k)}{f[u_k, y_k]}, \\
u_{k+1} = z_k - \frac{f(z_k)}{f[z_k, y_k]},
\end{cases}
\tag{22}
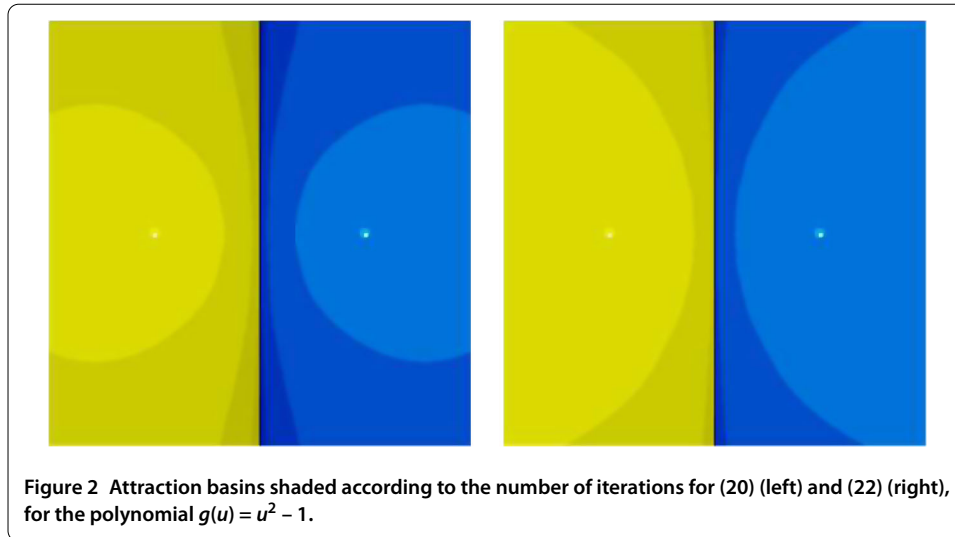$$

and subsequently the following iterative method:

$$
\begin{aligned}
U_{k+1} = U_k[765I + 7{,}840Y_k + 12{,}866Z_k + 4{,}008W_k + 121L_k] \\
\times [81I + 3{,}208Y_k + 12{,}306Z_k + 8{,}960W_k + 1{,}045L_k]^{-1}.
\end{aligned}
\tag{23}
$$

The attraction basins of these two new iterative methods are provided in Figure 2, which manifest their global convergence behavior. Note that a theoretical discussion for proving this global behavior is also possible using a similar strategy as given in [16].

The error analysis of the new schemes (21) and (23) are similar to the case given in Section 3. As a result, they are not included here.

5 Numerical results
We have tested the contributed methods (11), (21), (23) denoted by PM1, PM2, and PM3, respectively, using the programming package Mathematica 10 in double precision [21]. Apart from this scheme, several iterative methods, such as (5) denoted by NM, and (6)

Figure 2 Attraction basins shaded according to the number of iterations for (20) (left) and (22) (right), for the polynomial $g(u) = u^2 - 1$.

denoted by KHM, and the scaled Newton method (denoted by ANM) are given by

$$\begin{cases} \text{Compute } \theta_k \text{ by (16)}, \quad k \geq 0, \\ U_{k+1} = \frac{1}{2}[\theta_k U_k + \theta_k^{-1} U_k^{\dagger*}], \end{cases} \tag{24}$$

have been tested and compared. We used the following stopping criterion: $R_{k+1} = \frac{\|U_{k+1} - U_k\|_\infty}{\|U_k\|_\infty} \leq \epsilon$, wherein $\epsilon = 10^{-10}$ is the tolerance.

We now apply different numerical methods for finding the unitary polar factors of many randomly generated rectangular matrices with complex entries. In order to help the readers to re-run the experiments we used `SeedRandom[12345]` for producing pseudo-random (complex) numbers.

The random matrices for different dimensions of $m \times n$ are constructed by the following piece of Mathematica code ($I = \sqrt{-1}$):

```
SeedRandom[12345]; number = 15;
Table[A[l] = RandomComplex[{-10 - 10 I,
                            10 + 10 I}, {m, n}];, {l, number}];
```

We have gathered up the numerical results for the experiments in Tables 1-6. The initial approximation is constructed as $U_0 = \frac{1}{\|A\|_2} A$. Only for the cases $m \times n = 110 \times 100$ and $m \times n = 510 \times 500$, the comparisons of the required number of iterations have been reported and we mainly focused on the elapsed CPU time (in seconds) to clearly reveal that our proposed scheme is quite efficient in most cases. The results of comparison for the square nonsingular cases of $m \times n = 600 \times 600$ are included in Table 7. This shows that the efficient results are in complete agreement with the CPU time utilized in the execution of program(s) for PM2.

To give an answer to the key question: whether the increasing order convergence is worth in view of increasing the matrix multiplications in each iteration, it is requisite to incorporate the notion of efficiency index, $p^{1/\theta}$, whereas p and θ stand for the rate of convergence and the computational cots per cycle, respectively. This is achieved by assuming that each matrix-matrix multiplication cost 1-unit while the cost for one regular matrix inverse is 1.5-unit and one matrix Moore-Penrose inverse is 3-unit. Consequently, the effi-

Table 1 Results of comparison for the dimension $m \times n = 110 \times 100$ in terms of the number of iterations

Matrix No.	NM	ANM	KHM	PM1	PM2	PM3
1	10	8	6	4	5	4
2	10	7	6	4	5	4
3	10	8	6	4	5	4
4	10	7	6	4	5	4
5	10	8	6	4	5	4
6	10	8	6	4	5	4
7	10	8	6	4	5	4
8	10	8	6	4	5	4
9	10	8	6	4	5	4
10	10	7	6	4	5	4
11	10	8	6	4	5	4
12	10	8	6	4	5	4
13	10	8	6	4	5	4
14	10	8	6	4	5	4
15	10	8	6	4	5	4

Table 2 Results of comparison for the dimension $m \times n = 110 \times 100$ in terms of the elapsed time

Matrix No.	NM	ANM	KHM	PM1	PM2	PM3
1	0.040002	0.041002	0.020001	0.018001	0.019001	0.020001
2	0.039002	0.044002	0.029002	0.022001	0.018001	0.019001
3	0.042002	0.042002	0.020001	0.018001	0.018001	0.020001
4	0.039002	0.041002	0.020001	0.019001	0.018001	0.022001
5	0.040002	0.042002	0.020001	0.019001	0.018001	0.020001
6	0.041002	0.041002	0.020001	0.019001	0.018001	0.020001
7	0.039002	0.045003	0.023001	0.018001	0.027002	0.026002
8	0.050003	0.041002	0.020001	0.022001	0.019001	0.020001
9	0.039002	0.045003	0.020001	0.018001	0.021001	0.021001
10	0.043002	0.041002	0.020001	0.019001	0.018001	0.020001
11	0.040002	0.042002	0.020001	0.019001	0.019001	0.020001
12	0.040002	0.049003	0.022001	0.019001	0.018001	0.019001
13	0.040002	0.041002	0.020001	0.021001	0.018001	0.020001
14	0.041002	0.045003	0.020001	0.019001	0.018001	0.019001
15	0.040002	0.042002	0.020001	0.019001	0.019001	0.019001

Table 3 Results of comparison for the dimension $m \times n = 210 \times 200$ in terms of the elapsed time

Matrix No.	NM	ANM	KHM	PM1	PM2	PM3
1	0.208012	0.240014	0.092005	0.088005	0.093005	0.091005
2	0.207012	0.240014	0.090005	0.086005	0.079005	0.090005
3	0.209012	0.221013	0.092005	0.088005	0.081005	0.094005
4	0.209012	0.243014	0.097005	0.106006	0.093005	0.090005
5	0.216012	0.240014	0.091005	0.107006	0.094005	0.089005
6	0.211012	0.216012	0.090005	0.088005	0.084005	0.090005
7	0.208012	0.243014	0.091005	0.105006	0.093005	0.092005
8	0.211012	0.225013	0.095005	0.091005	0.078005	0.088005
9	0.210012	0.238014	0.092005	0.089005	0.092005	0.088005
10	0.217012	0.239014	0.091005	0.094005	0.093005	0.089005
11	0.208012	0.218012	0.090005	0.086005	0.078004	0.089005
12	0.209012	0.240014	0.102006	0.087005	0.080005	0.089005
13	0.209012	0.244014	0.092005	0.086005	0.078005	0.089005
14	0.210012	0.239014	0.091005	0.086005	0.079005	0.094005
15	0.216012	0.239014	0.097006	0.105006	0.093005	0.088005

Table 4 Results of comparison for the dimension *m* × *n* = 410 × 400 in terms of the elapsed time

Matrix No.	NM	ANM	KHM	PM1	PM2	PM3
1	1.131065	1.151066	0.581033	0.619035	0.542031	0.607035
2	1.144065	1.150066	0.587034	0.597034	0.522030	0.610035
3	1.190068	1.144065	0.578033	0.587034	0.532030	0.632036
4	1.153066	1.144065	0.589034	0.591034	0.524030	0.598034
5	1.135065	1.147066	0.581033	0.586033	0.538031	0.607035
6	1.145066	1.148066	0.588034	0.599034	0.527030	0.602034
7	1.134065	1.152066	0.587034	0.593034	0.532030	0.599034
8	1.123064	1.157066	0.577033	0.594034	0.518030	0.617035
9	1.137065	1.149066	0.589034	0.593034	0.520030	0.604035
10	1.127064	1.140065	0.503029	0.593034	0.521030	0.614035
11	1.129065	1.144065	0.577033	0.591034	0.524030	0.596034
12	1.119064	1.147066	0.593034	0.592034	0.522030	0.600034
13	1.139065	1.152066	0.591034	0.590034	0.527030	0.600034
14	1.124064	1.142065	0.587033	0.593034	0.522030	0.595034
15	1.126064	1.155066	0.597034	0.605035	0.522030	0.619035

Table 5 Results of comparison for the dimension *m* × *n* = 510 × 500 in terms of the number of iterations

Matrix No.	NM	ANM	KHM	PM1	PM2	PM3
1	12	9	7	5	6	5
2	12	9	7	5	6	5
3	12	9	7	5	6	5
4	12	9	7	5	6	5
5	12	9	7	5	6	5
6	12	9	7	5	6	5
7	12	9	7	5	6	5
8	12	9	7	5	6	5
9	12	9	7	5	6	5
10	12	9	7	5	6	5

Table 6 Results of comparison for the dimension *m* × *n* = 510 × 500 in terms of the elapsed time

Matrix No.	NM	ANM	KHM	PM1	PM2	PM3
1	2.160124	2.222127	1.067061	1.116064	0.926053	1.074061
2	2.185125	2.181125	1.041060	1.129065	0.924053	1.102063
3	2.142123	2.199126	1.029059	1.174067	0.957055	1.121064
4	2.154123	2.111121	1.113064	1.121064	0.955055	1.096063
5	2.139122	2.131122	1.086062	1.049060	0.961055	1.077062
6	2.130122	2.195126	1.084062	1.050060	0.939054	1.075061
7	2.126122	3.235185	1.077062	1.134065	0.938054	1.076062
8	2.126122	2.098120	1.108063	1.083062	0.957055	1.076062
9	2.100120	2.147123	1.054060	1.084062	0.994057	1.069061
10	2.157123	2.210126	1.052060	1.076062	0.966055	1.073061

Table 7 Results of comparison for the dimension *m* × *n* = 510 × 500 in terms of the elapsed time

Matrix No.	NM	ANM	KHM	PM1	PM2	PM3
1	2.006115	1.940111	0.972056	0.984056	0.878050	1.003057
2	1.967113	1.937111	0.969055	0.983056	0.876050	1.013058
3	1.967113	1.918110	0.972056	0.994057	0.878050	1.003057
4	1.982113	1.912109	0.980056	0.996057	0.876050	1.012058
5	2.099120	1.932111	0.968055	0.992057	0.886051	1.011058
6	1.969113	1.919110	0.977056	0.984056	0.889051	1.003057
7	1.974113	1.919110	0.975056	0.983056	0.881050	1.015058
8	1.967113	1.920110	0.969055	0.999057	0.877050	1.011058
9	1.976113	1.920110	0.992057	1.003057	0.875050	1.004057
10	1.970113	1.932111	0.975056	0.990057	0.876050	1.012058

Table 8 Results of comparison for the dimension $m \times n = 600 \times 600$ in terms of the elapsed time

Matrix No.	NM	ANM	KHM	PM1	PM2	PM3
1	1.281073	1.320075	1.687096	1.866107	1.568090	1.870107
2	1.301074	1.322076	1.695097	1.853106	1.554089	1.878107
3	1.216070	1.333076	1.706098	1.867107	1.562089	1.578090
4	1.286074	1.324076	1.703097	1.847106	1.561089	1.895108
5	1.296074	1.325076	1.716098	1.858106	1.557089	1.858106
6	1.370078	1.319076	1.905109	1.830105	1.768101	1.873107
7	1.286074	1.320075	1.708098	1.857106	1.566090	1.583091
8	1.531088	1.317075	1.906109	2.123121	1.766101	1.868107
9	1.375079	1.314075	1.917110	1.851106	1.762101	1.888108
10	1.288074	1.321076	1.711098	1.841105	1.562089	1.855106

ciency indices for the discussed methods are: $E(4) \simeq 1.2599$, $E(6) \simeq 1.2210$, $E(11) \simeq 1.2698$, $E(21) \simeq 1.2866$, and $E(23) \simeq 1.2962$.

However, it is also required to state that for square cases and as could be seen in Table 8, the NM and ANM are better choices since they are using the regular inverses in their iterative structures, unlike their structures in the rectangular cases. Furthermore, it sounds as if the computation of the scaling factor for the proposed method will not be attractive, due to the computation of an extra pseudo-inverse per cycle.

The acquired numerical results agree with the theoretical discussions given in Sections 2 and 3, overwhelmingly. As a result, we can state that PM1-PM3 reduce the number of iterations and time in finding the polar decomposition.

6 Concluding remarks

In this paper, we developed high-order methods for matrix polar decomposition. It has been shown that the convergence is global. Many numerical tests (of various dimensions) have been provided to show the performance of the new method.

In 1991, Kenney and Laub [17] proposed a family of rational iterative methods for sign (subsequently for polar decomposition), based on Padé approximation. Their principal Padé iterations are convergent globally. Thus, we have convergent methods of arbitrary orders for sign (subsequently for polar decomposition). However, here we tried to propose new methods, which are interesting from theoretical point of view and are not members of Padé family. Numerical results have demonstrated the behavior of the new algorithms.

Author details
[1]Department of Applied Mathematics, Ferdowsi University of Mashhad, Mashhad, Iran. [2]Department of Mathematics, Faculty of Basic Science, Shahrekord Branch, Islamic Azad University, Shahrekord, Iran. [3]Department of Mathematics and Applied Mathematics, University of Venda, Thohoyandou, 0950, South Africa.

References
 1. Higham, NJ: Functions of Matrices: Theory and Computation. SIAM, Philadelphia (2008)

2. Laszkiewicz, B, Ziętak, K: Approximation of matrices and family of Gander methods for polar decomposition. BIT Numer. Math. **46**, 345-366 (2006)
3. Higham, NJ: The matrix sign decomposition and its relation to the polar decomposition. Linear Algebra Appl. **212/213**, 3-20 (1994)
4. Roberts, JD: Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. Int. J. Control **32**, 677-687 (1980)
5. Higham, NJ: Computing the polar decomposition - with applications. SIAM J. Sci. Stat. Comput. **7**, 1160-1174 (1986)
6. Byers, R: Solving the algebraic Riccati equation with the matrix sign function. Linear Algebra Appl. **85**, 267-279 (1987)
7. Gander, W: Algorithms for the polar decomposition. SIAM J. Sci. Stat. Comput. **11**, 1102-1115 (1990)
8. Soheili, AR, Toutounian, F, Soleymani, F: A fast convergent numerical method for matrix sign function with application in SDEs. J. Comput. Appl. Math. **282**, 167-178 (2015)
9. Soleymani, F, Stanimirović, PS, Stojanović, I: A novel iterative method for polar decomposition and matrix sign function. Discrete Dyn. Nat. Soc. **2015**, Article ID 649423 (2015)
10. Nakatsukasa, Y, Bai, Z, Gygi, F: Optimizing Halley's iteration for computing the matrix polar decomposition. SIAM J. Matrix Anal. Appl. **31**, 2700-2720 (2010)
11. Du, K: The iterative methods for computing the polar decomposition of rank-deficient matrix. Appl. Math. Comput. **162**, 95-102 (2005)
12. Khaksar Haghani, F: A third-order Newton-type method for finding polar decomposition. Adv. Numer. Anal. **2014**, Article ID 576325 (2014)
13. Soleymani, F, Stanimirović, PS, Shateyi, S, Haghani, FK: Approximating the matrix sign function using a novel iterative method. Abstr. Appl. Anal. **2014**, Article ID 105301 (2014)
14. Soleymani, F: Some high-order iterative methods for finding all the real zeros. Thai J. Math. **12**, 313-327 (2014)
15. Cordero, A, Soleymani, F, Torregrosa, JR, Shateyi, S: Basins of attraction for various Steffensen-type methods. J. Appl. Math. **2014**, Article ID 539707 (2014)
16. Khaksar Haghani, F, Soleymani, F: On a fourth-order matrix method for computing polar decomposition. Comput. Appl. Math. **34**, 389-399 (2015)
17. Kenney, C, Laub, AJ: Rational iterative methods for the matrix sign function. SIAM J. Matrix Anal. Appl. **12**, 273-291 (1991)
18. Kielbasiński, A, Zieliński, P, Ziętak, K: On iterative algorithms for the polar decomposition of a matrix. Appl. Math. Comput. **270**, 483-495 (2015)
19. Dubrulle, AA: Frobenius iteration for the matrix polar decomposition. Technical report HPL-94-117, Hewlett-Packard Company (1994)
20. Byers, R, Xu, H: A new scaling for Newton's iteration for the polar decomposition and its backward stability. SIAM J. Matrix Anal. Appl. **30**, 822-843 (2008)
21. Wolfram Research, Inc., Mathematica, Version 10.0, Champaign, IL (2015)