A sixth-order iterative method for approximating the polar decomposition of an arbitrary matrix ☆

Alicia Cordero[a], Juan R. Torregrosa[a,*]

[a] *Instituto Universitario de Matemática Multidisciplinar, Universitat Politècnica de València, València, Spain*

Abstract

A new iterative method for computing the polar decomposition of any rectangular complex matrix is presented and analyzed. The study of the convergence shows that this method has order of convergence six. Some numerical tests confirm the theoretical results and allow us to compare the proposed iterative scheme with other known ones.

Keywords: Polar decomposition, singular value decomposition, matrix iteration, unitary factor, Hermitian matrix, iterative method
AMS Subject Classification: 65F25, 65F30.

1. Introduction

The *polar decomposition* is a generalization to complex matrices of the trigonometric representation of a complex number. Specifically, let A be a complex matrix of size $m \times n$, $m \geq n$ (in other case, we work with the transpose matrix). Then there exist a matrix $U \in \mathbb{C}^{m\times n}$, with orthonormal columns and a Hermitian positive semi-definite $H \in \mathbb{C}^{n\times n}$ such that

$$A = UH, \qquad U^*U = I_n, \tag{1}$$

where U^* denotes the conjugate transpose of U and I_n is the identity matrix of size $n \times n$. The Hermitian factor H is always unique and can be expressed as $H = (A^*A)^{1/2}$. If matrix A has full rank, then H is positive definite and the unitary factor U is uniquely determined.

Let us observe that, once the unitary factor U is calculated, the other factor is obtained in a simple way, $H = U^*A$. So, our goal in this work is to obtain factor U.

The polar decomposition is well known and can be found in many textbooks, for example, [1] and [2]. An early reference is [3]. This decomposition has many applications in several fields. In [4] the author describes different applications of the polar decomposition to factor analysis, aerospace computations and optimization. For example, the optimization method called Conjugate Gradient, for the minimization of $F(x)$, $F : \mathbb{R}^n \to \mathbb{R}$, is more stable when the Hessian matrix is replaced by the Hermitian factor of its polar decomposition. On the other hand, the square root of a positive definite matrix A is the Hermitian factor of the polar decomposition of L^T, where $A = LL^T$ is the Choleski decomposition of A. In addition, polar decomposition has many advantages in front of other decompositions in the context of computer graphics.

It is well known that the unitary factor U possesses a best approximation property and in [4] the author describes, under some conditions, good approximation properties of the Hermitian factor H. Other interesting

☆This research was partially supported by Ministerio de Economía y Competitividad MTM2014-52016-C02-02.

*Corresponding author

Email addresses: acordero@mat.upv.es (Alicia Cordero), jrtorre@mat.upv.es (Juan R. Torregrosa)

properties of the polar decomposition appear in the literature (see, for instance, [1] and [2]) . For example, the eigenvalues of matrix H and the singular values of matrix A are the same, as well as the 2-norm condition number. These matrices also have a common set of eigenvectors.

The polar decomposition of A can be computed using the singular value decomposition (SVD) [1]. Let us suppose that matrix $A \in \mathbb{C}^{m\times n}$, $m \geq n$, has the singular value decomposition

$$A = P\begin{pmatrix} \Sigma \\ 0 \end{pmatrix} Q^*, \tag{2}$$

where $P \in \mathbb{C}^{m\times m}$ and $Q \in \mathbb{C}^{n\times n}$ are unitary matrices and $\Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_n)$, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$.

If we partition matrix P in the form $P = [P_1, P_2]$, where P_1 is an $m \times n$ matrix such that $P_1^* P_1 = I_n$, it follows that A has the polar decomposition $A = UH$, where $U = P_1 Q^*$ and $H = Q\Sigma Q^*$.

On the other hand, let A be an $m \times n$ complex matrix with full rank and $A = QR$ its QR-factorization, where Q is an $m \times n$ matrix with orthonormal columns and R is an $n \times n$ upper triangular nonsingular matrix. The polar decomposition of A is given in terms of that of R by

$$A = QR = Q(U_R H_R) = (QU_R)H_R = UH.$$

As some authors show in their works, these approaches are not always the most efficient or the most convenient. So, we are going to present iterative schemes for approximating the polar decomposition of a rectangular complex matrix.

In this paper, we are interested in computing the polar decomposition by means of an iterative method of the fixed-point form $U_{k+1} = G(U_k)$, provided that the initial guess matrix U_0 is given. Let us remember that from the unitary factor U the other factor of the polar decomposition is obtained easily. In [4] Higham proposed a fixed-point algorithm based on Newton's method to compute the square root of a number, for obtaining the unitary factor U of a nonsingular $n \times n$ matrix A. Starting with $U_0 = A$, the sequence U_k is computed by

$$U_{k+1} = \frac{1}{2}(U_k + U_k^{-*}), \tag{3}$$

where U_k^{-*} denotes $(U_k^{-1})^*$. The quadratic convergence of sequence $\{U_k\}_{k\geq 0}$ was proved.

Fifteen years later, Du in [5] generalized Higham's algorithm for rectangular matrices by means of the iterative expression

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

where $U_k^{\dagger}$ denotes the Moore-Penrose pseudoinverse of U_k. This scheme keeps the order of convergence of the previous one.

In a similar way as Hihgam with Newton's method, Gander in [6] used Halley's scheme for scalar equations for designing the following algorithm that converges to the unitary factor with order of convergence three for nonsingular matrices.

$$U_{k+1} = [U_k(3I + U_k^* U_k)]\left[I + 3U_k^* U_k\right]^{-1}. \tag{5}$$

Recently, a fourth-order iterative method for computing the polar decomposition was developed by Khaksar and Soleymani in [7] from a fourth-order method for solving nonlinear equations. The iterative expression of this scheme is

$$U_{k+1} = [U_k(7I + Y_k)(I + 3Y_k)]\left[I + 18Y_k + 13Z_k\right]^{-1}, \tag{6}$$

where $Y_k = U_k^* U_k$, $Z_k = Y_k Y_k$ and $U_0 = A$.

The rest of the paper is organized as follows: in Section 2, a new root-finding scheme for scalar equations is designed and, from it, an iterative scheme for computing the polar decomposition of any rectangular matrix is derived. Section 3 is devoted to analyze the convergence of the proposed scheme under some conditions. Moreover, we prove that the new scheme has order six for a proper initial matrix. Some numerical test are presented in Section 4 to confirm the theoretical results and for comparing our scheme with other known ones. This paper finishes with several conclusions and some of the references used in it.

2. The proposed iterative scheme

Many multi-point iterative schemes for solving nonlinear scalar equations $f(x) = 0$ have been designed in the last years. The main interest of these fixed-point methods is the possibility to reach any order of convergence. The connection between the matrix iterations for computing polar decomposition, or in general for approximating the solution of a nonlinear matrix equation, and iterative methods for nonlinear scalar equations were described by Iannazzo in [8].

In fact, the matrix form of fixed-point type methods for polar decomposition is the generalization of applying the nonlinear equation solvers to the matrix equation

$$F(U) := U^* U - I = 0,$$

where I is the identity matrix of the appropriate size. This reveals the relation between the polar decomposition and matrix sign function (see, for example, the papers of Higham [9], Kenney and Laub [10], Sharifi et al. [11] and the references therein).

Let us consider the following iterative expression for finding the simple zeros of a nonlinear equation $f(x) = 0$

$$\begin{aligned} y_k &= x_k - \frac{10 - 4L_f(x_k)}{10 - 9L_f(x_k)} \frac{f(x_k)}{f'(x_k)}, \\ x_{k+1} &= y_k - \frac{f(y_k)}{f'(y_k)}, \end{aligned} \tag{7}$$

where $L_f(x_k) = \dfrac{f(x_k) f''(x_k)}{f'(x_k)^2}$ is the degree of logarithmic convexity. This scheme is the composition of the scheme proposed in [11] for finding matrix sign functions and Newton's method. By using Taylor expansion of the different elements of the iterative expression (7), we can prove the following result.

Theorem 1. *Let $\alpha \in I$ be a simple zero of a sufficiently differentiable function $f : I \subseteq \mathbb{R} \to \mathbb{R}$ for an open interval I, and $x_0 \in I$ an initial guess close enough to α. Then, iterative expression (7) converge to α with order of convergence six, being its error equation*

$$e_{k+1} = \frac{1}{25} c_2 (c_2^2 - 5c_3)^2 e_k^6 + O(e_k^7),$$

where $c_j = \dfrac{1}{j!} \dfrac{f^{(j)}(\alpha)}{f'(\alpha)}$, $j = 2, 3, \ldots$, and $e_k = x_k - \alpha$.

Proof. Expanding $f(x_k)$, $f'(x_k)$ and $f''(x_k)$ about $x = \alpha$ by Taylor series, we have

$$f(x_k) = f'(\alpha) \left[e_k + c_2 e_k^2 + c_3 e_k^3 + c_4 e_k^4 + c_5 e_k^5 + c_6 e_k^6 + O(e_k^7) \right],$$

$$f'(x_k) = f'(\alpha)\left[1 + 2c_2 e_k + 3c_3 e_k^2 + 4c_4 e_k^3 + 5c_5 e_k^4 + 6c_6 e_k^5 + O(e_k^6)\right]$$

and

$$f''(x_k) = f'(\alpha)\left[2c_2 + 6c_3 e_k + 12c_4 e_k^2 + 20c_5 e_k^3 + 30c_6 e_k^4 + O(e_k^5)\right].$$

From these expressions, we get

$$\begin{aligned} L_f(x_k) &= 2c_2 e_k + (-6c_2^2 + 6c_3)e_k^2 + 4(4c_2^3 - 7c_2c_3 + 3c_4)e_k^3 - 10(4c_2^4 - 10c_2^2c_3 + 3c_3^2 + 5c_2c_4 - 2c_5)e_k^4 \\ &\quad + 6(16c_2^5 - 52c_2^3c_3 + 28c_2^2c_4 - 17c_3c_4 + c_2(33c_3^2 - 13c_5) + 5c_6)e_k^5 + O(e_k^6). \end{aligned}$$

Therefore, Taylor expansion of y_k gives us the error of the first step of the multipoint method (7),

$$\begin{aligned} y_k &= \frac{1}{5}(c_2^2 - 5c_3)e_k^3 + \left(\frac{9c_2^3}{25} + \frac{6c_2c_3}{5} - 3c_4\right)e_k^4 \\ &\quad - \frac{6}{125}\left(59c_2^4 - 105c_2^2c_3 - 50c_2c_4 + 25(c_3^2 + 5c_5)\right)e_k^5 \\ &\quad + \left(\frac{1}{625}(5399c_2^5 - 17815c_2^3c_3 + 12200c_2c_3^2 + 7175c_2^2c_4 - 6125c_3c_4) + 4c_2c_5 - 10c_6\right)e_k^6 + O(e_k^7). \end{aligned}$$

In a similar way, expanding $f(y_k)$ and $f'(y_k)$ about $x = \alpha$ and by replacing in the second step of (7), we obtain

$$e_{k+1} = \frac{1}{25}c_2(c_2^2 - 5c_3)^2 e_k^6 + O(e_k^7)$$

and the proof is finished. ■

Solving the equation $u^2 - 1 = 0$ by (7), we have the iterative expression

$$u_{k+1} = \frac{4 + 141u_k^2 + 435u_k^4 + 211u_k^6 + 9u_k^8}{36u_k + 314u_k^3 + 384u_k^5 + 66u_k^7}, \quad k = 0, 1, \ldots \tag{8}$$

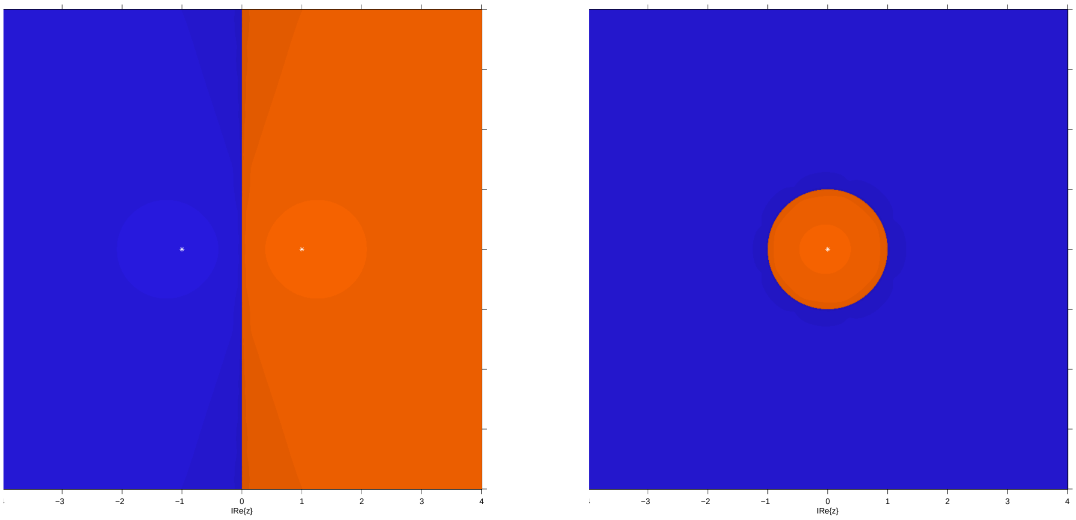
and the associated fixed point operator is denoted by T, $u_{k+1} = T(u_k)$. We are going to analyze some properties of this operator.

The unique attracting fixed points of operator T are the roots of the quadratic polynomial, that is, $u = 1$ and $u = -1$. Drawing the dynamical plane of operator T in the complex plane, shows a global convergence of our scheme on the quadratic polynomial $u^2 - 1 = 0$ (see Figure 1a). For the representation of this dynamical planes we have used the software described in [12]. We draw a mesh with four hundred points per axis; each point of the mesh is a different initial estimation which we introduce in the procedure. When the method reaches one root of the equation in less than two hundred iterations, this point is drawn in a different color for each root (with a tolerance of 10^{-3}). The color will be more intense when the number of iterations is lower. The roots of the polynomial will be represented by white stars in the picture. Finally, if there is no convergence to any root, after a maximum of 200 iterations, then the point of the mesh used as initial estimation is painted in black.

On the other hand, by applying Möbius transformation $h(u) = \dfrac{u+1}{u-1}$, which sends one root to zero and the other to infinity, operator T is conjugated to operator O

$$O(u) = (h \circ T \circ h^{-1})(u) = u^6\frac{(1 + 5u)^2}{(5 + u)^2},$$

whose dynamical plane we can see in Figure 1b, being the orange area the basin of attraction of zero and the blue area the basin of infinity.



(a) Dynamical plane of operator T

(b) Dynamical plane of parameter O

Figure 1: Dynamical planes before and after using Möbius transformation

Taking into account that operator O is independent of the quadratic polynomial, we can assure that the iterative scheme (7) has global convergence on any quadratic polynomial. In addition, operator O satisfies the following property that we call *reciprocal property*

$$O\left(\frac{1}{u}\right) = \frac{1}{O(u)}.$$

From the reciprocal property and this global convergence behavior, we can use the iterative scheme (8) in the reciprocal form

$$u_{k+1} = \frac{36u_k + 314u_k^3 + 384u_k^5 + 66u_k^7}{4 + 141u_k^2 + 435u_k^4 + 211u_k^6 + 9u_k^8}, \quad k = 0, 1, \ldots$$

and extend it in the matrix context

$$U_{k+1} = \left[U_k(36I + 314Y_k + 384Z_k + 66X_k)\right]\left[4 + 141Y_k + 435Z_k + 211X_k + 9W_k\right]^{-1}, \tag{9}$$

where $Y_k = U_k^* U_k$, $Z_k = Y_k Y_k$, $X_k = Y_k Z_k$ and $W_k = Z_k Z_k$.

Expression (9) is a new iterative fixed-point scheme for finding the polar decomposition via calculating the unitary matrix U. This method is not a member of Padé family of iterations given in [10]. In the next section we are going to analyze the convergence of sequence $\{U_k\}_{k\geq 0}$ generated by the iterative scheme (9).

3. Convergence analysis

The next results establish the order of convergence of the method described by (9).

Theorem 2. *Let A be an arbitrary $m \times n$ complex matrix of rank r. Then, the sequence of matrix iterates $\{U_k\}_{k\geq 0}$ obtained from (9) converges to the unitary factor U, for $U_0 = A$.*

Proof. In order to prove this, we make use of the singular value decomposition of matrix A,

$$A = P \begin{pmatrix} \Sigma \\ 0 \end{pmatrix} Q^*,$$

where $P \in \mathbb{C}^{m\times m}$ and $Q \in \mathbb{C}^{n\times n}$ are unitary and $\Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_n)$, being $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$ the singular values of matrix A. Now, we define the following sequence of matrices

$$\bar{D}_k = \begin{pmatrix} D_k \\ 0 \end{pmatrix} = P^* U_k Q.$$

Afterwards, from (9), we have

$$\begin{array}{rcl} D_0 & = & \Sigma \\ D_{k+1} & = & \left[D_k(36I + 314D_k^2 + 384D_k^4 + 66D_k^6)\right]\left[4I + 141D_k^2 + 435D_k^4 + 211D_k^6 + 9D_k^8\right]^{-1}. \end{array} \tag{10}$$

As D_0 is a diagonal matrix with nonnegative diagonal entries, it follows by induction that sequence $\{D_k\}_{k\geq 0}$ is defined as

$$D_k = diag(d_1^{(k)}, d_2^{(k)}, \ldots, d_r^{(k)}, 0, \ldots, 0),$$

where $r = rank(A)$. Note that (10) represents k uncoupled scalar iterations:

$$\begin{array}{rcl} d_i^{(0)} & = & \sigma_i, \;\; 1 \leq i \leq r \\ d_i^{(k+1)} & = & \left[36d_i^{(k)} + 314{d_i^{(k)}}^3 + 384{d_i^{(k)}}^5 + 66{d_i^{(k)}}^7\right]\left[4 + 141{d_i^{(k)}}^2 + 435{d_i^{(k)}}^4 + 211{d_i^{(k)}}^6 + 9{d_i^{(k)}}^8\right]^{-1}. \end{array}$$

Some algebraic manipulations give the relation between $d_i^{(k+1)}$ and $d_i^{(k)}$

$$\frac{d_i^{(k+1)} - 1}{d_i^{(k+1)} + 1} = -\frac{(d_i^{(k)} - 1)^6}{(d_i^{(k)} + 1)^6}\frac{(2 - 3d_i^{(k)})^2}{(2 + 3d_i^{(k)})^2}.$$

If we repeat this step until $d_i^{(0)}$, we have

$$\frac{d_i^{(k+1)} - 1}{d_i^{(k+1)} + 1} = -\left(\frac{(d_i^{(0)} - 1)}{(d_i^{(0)} + 1)}\right)^{6^{k+1}} H(d_i^{(k)}, d_i^{(k-1)}, \ldots, d_i^{(0)}).$$

Taking into account that $d_i^{(0)} > 0$, $i = 1, 2, \ldots, r$, and $H(d_i^{(k)}, d_i^{(k-1)}, \ldots, d_i^{(0)}) < 1$ we conclude

$$\left|\frac{d_i^{(k+1)} - 1}{d_i^{(k+1)} + 1}\right| \to 0, \;\; \text{as} \;\; k \to +\infty,$$

for each i, that is

$$D_k \to I_r \;\; \text{and} \;\; \bar{D}_k \to \begin{pmatrix} I_r & 0 \\ 0 & 0 \end{pmatrix}.$$

Therefore, as $k \to +\infty$, $U_k \to U$ and subsequently $H = U^*A$. This finishes the proof. ■

Theorem 3. *Let A be an arbitrary $m \times n$ complex matrix of rank r. Then, the iterative scheme described by (9) has sixth order of convergence for finding the unitary factor U.*

Proof. By using the previous theorem, scheme (9) transforms the singular values of U_k according to

$$\sigma_i^{(k+1)} = \frac{36\sigma_i^{(k)} + 314{\sigma_i^{(k)}}^3 + 384{\sigma_i^{(k)}}^5 + 66{\sigma_i^{(k)}}^7)}{4 + 141{\sigma_i^{(k)}}^2 + 435{\sigma_i^{(k)}}^4 + 211{\sigma_i^{(k)}}^6 + 9{\sigma_i^{(k)}}^8}, \ \ 1 \leq i \leq r,$$

and leaves the singular vectors invariant. As in the previous theorem, algebraic manipulations of this expression allow us to assure that

$$\frac{\sigma_i^{(k+1)} - 1}{\sigma_i^{(k+1)+1}} = -\frac{(\sigma_i^{(k)} - 1)^6}{(\sigma_i^{(k)} + 1)^6} \frac{(2 - 3\sigma_i^{(k)})^2}{(2 + 3\sigma_i^{(k)})^2}.$$

Therefore,

$$\left| \frac{\sigma_i^{(k+1)} - 1}{\sigma_i^{(k+1)} + 1} \right| \leq \left| \frac{\sigma_i^{(k)} - 1}{\sigma_i^{(k)} + 1} \right|^6 \left| \frac{2 - 3\sigma_i^{(k)}}{2 + 3\sigma_i^{(k)}} \right|^2 .$$

This confirms the sixth order of convergence of iterative scheme (9). The proof is completed. ■

In spite of this sixth-order, the speed of convergence can be slow at the beginning of the process, so it is appropriate to scale matrix U_k before each iteration. There are different practical ways for producing the scaling parameter, we can see two of them below. One is due to Du [5] and its expression is

$$\gamma_k = \left(\frac{\|U_k^\dagger\|_1 \|U_k^\dagger\|_\infty}{\|U_k\|_1 \|U_k\|_\infty} \right)^{1/4} \tag{11}$$

and the other was developed by Kenney and Laub in [10] as

$$\gamma_k = \left(\frac{\|U_k^\dagger\|_F}{\|U_k\|_F} \right)^{1/2} . \tag{12}$$

By using each of them, our algorithm can be written the following form. From an initial guess, we calculate iterates until a stopping criterion is satisfied.

$$\left\{ \begin{array}{l} \text{Compute } \gamma_k, \\ M_k = 4I + 141\gamma_k^2 Y_k + 435\gamma_k^4 Z_k + 211\gamma_k^6 X_k + 9\gamma_k^8 W_k, \\ U_{k+1} = \left[\gamma_k U_k (36I + 314\gamma_k^2 Y_k + 384\gamma_k^4 Z_k + 66\gamma_k^6 X_k)\right] M_k^{-1}, \ k = 0, 1, 2, \ldots \end{array} \right. \tag{13}$$

4. Numerical results

In this section we are going to present the numerical results. The numerical tests have been made in Matlab in double precision or variable precision arithmetics, with 50 digits of mantissa, depending on the size of the matrix. The computer specifications are Intel(R) Core(TM), i5-2500, CPU 3.30 GHz, with 16 GB of RAM. We compare our scheme, denoted as CTM, with several known iterative methods such as (4) denoted by NM, (5) denoted by HM and (6) denoted by KSM, of orders of convergence two, three and four, respectively, which also require one inverse per iteration. We use two different stopping criterium: the difference between the last iterates

$$\|U_{k+1} - U_k\|_M < tol$$

and this other one

$$\left\|U_{k+1}^* U_{k+1} - I\right\|_M < tol,$$

where tol is the tolerance and $\|\cdot\|_M$ is a proper matrix norm. We also calculate the approximated computational order of convergence (ACOC), (see [13]), according to

$$p \approx ACOC = \frac{\ln(\|U_{k+1}-U_k\|/\|U_k-U_{k-1}\|)}{\ln(\|U_k-U_{k-1}\|/\|U_{k-1}-U_{k-2}\|)},$$

which is an approach of the theoretical order of convergence p.

Let us remark that the value of ACOC that is presented in the different tables is the last coordinate of vector ACOC when the variation between its values is small. In other case, it is denoted by $-$. We present four examples, in each of them matrix A has particular characteristics.

Example 1. In this experiment, we analyze the behavior of the different methods on a random rectangular matrix of size 510×500 generated in Matlab by $A = rand(510, 500)$, working in double precision arithmetics.

In Table 1 we show the results obtained by applying the different methods on matrix A, with tolerance $tol = 10^{-10}$ and initial estimation $U_0 = A$. Specifically, we present the values of $\|U_{k+1} - U_k\|_2$ and $\|U_{k+1}^T U_{k+1} - I_{500}\|_2$ in the last iteration, where I_{500} is the identity matrix of order 500, the number of iterations, the ACOC and the elapsed time, in seconds, being the mean execution time for 50 performances of each method (the command cputime of Matlab has been used).

	Newton	Halley	KSM	CTM
iterations	13	9	7	6
$\|U_{k+1}-U_k\|_2$	1.70e-14	1.12e-15	1.64e-14	1.31e-15
$\|U_{k+1}^T U_{k+1}-I_{500}\|_2$	3.34e-15	1.50e-15	1.51e-15	1.31e-15
e-time	5.67	1.93	1.84	1.78
ACOC	-	-	-	-

Table 1: Numerical results for a random matrix of size 510×500 in double precision aritmethics

Let us note that, in this case, ACOC is not stable, so it does not give us any information.

Now, we study the behavior of the different methods on the matrix A, using the acceleration via scaling (12) and variable precision arithmetics for all of them. We consider again $tol = 10^{-10}$ in $\|\cdot\|_2$ and $U_0 = A$. We present the numerical results of this experiment in Table 2. We can observe a reduction in the number of iterations, but the computational elapsed time increases, probably by the cost of variable precision arithmetics. In this case, the ACOC is stable in many cases because we use a scaling parameter and variable precision arithmetics.

	Newton	Halley	KSM	CTM
iterations	10	8	6	5
$\|U_{k+1}-U_k\|_2$	4.82e-14	1.16e-15	3.18e-15	1.12e-15
$\|U_{k+1}^T U_{k+1}-I_{500}\|_2$	3.87e-15	1.83e-15	1.44e-15	1.55e-15
e-time	8.65	4.37	3.70	3.12
ACOC	2.0018	2.9090	3.3436	5.6323

Table 2: Numerical results for a random matrix of size 510×500 with scaling parameter and variable precision arithmetics

Example 2. In this example, we analyze the behavior of the different methods on Hilbert matrix of size 75×75 generated in Matlab by $B = hilb(75)$, working in double precision arithmetics and under the same conditions as Example 1. Let us remember that it is an example of ill-conditioned matrix.

	Newton	Halley	KSM	CTM
iterations	> 1000	46	26	22
$\|U_{k+1} - U_k\|_2$	-	8.88e-16	2.49e-11	4.26e-15
$\|U_{k+1}^T U_{k+1} - I_{75}\|_2$	-	1.11e-15	1.30e-15	8.41e-16
e-time	-	0.16	0.11	0.10
ACOC	-	-	-	-

Table 3: Numerical results for a Hilbert's matrix of size 75×75

In this case, the number of iterations increases significantly, in fact Newton's method is not convergent. However, the elapsed time decreases because matrix B has a small size. ACOC is again unstable.

When we use the methods with scaling parameter, due to the numerical singularity of matrix B, we get several warnings from Matlab and bad results in all cases. The stopping criterium is satisfied in a reasonable number of iterations, but the last obtained matrix U does not satisfies $\|U^T U - I_{75}\|_2 < 10^{-10}$. In addition, by using variable precision arithmetics the results have not improved.

Example 3. In this experiment, we analyze the behavior of the different methods on a random rectangular matrix of size 60×50 generated in Matlab by $C = rand(60, 50)$, working in variable precision arithmetics and under the same conditions as in the previous examples.

	Newton	Halley	KSM	CTM
iterations	10	7	6	4
$\|U_{k+1} - U_k\|_2$	1.31e-15	5.42e-23	7.61e-42	3.05e-14
$\|U_{k+1}^T U_{k+1} - I_{50}\|_2$	1.91e-15	3.28e-57	1.81e-57	2.68e-57
e-time	10.06	86.05	77.56	54.56
ACOC	1.8320	2.9999	4.0000	7.2160

Table 4: Numerical results for a random matrix of size 60×50 in variable precision arithmetics

Variable precision arithmetics gives us the most stable results, but with high computational time. In fact, for matrix A of Example 1 this precision does not work, in a reasonable time of computation. In this case, all methods work perfectly and the values shown in Table 4 confirm the theoretical results.

Example 4. This example is devoted to a tridiagonal matrix D of size 200×200, with $v = [2, 2, \ldots, 2]$ as main diagonal and $u = [-1, -1, \ldots, -1]$ as sub and super-diagonal. This matrix appears frequently when we discretize boundary problems by means of finite differences. Now, we use double precision arithmetics, $tol = 10^{-3}$ and $U_0 = A$ as initial guess. The obtained results are shown in Table 5.

Let us observe that ACOC is again unstable. For big matrices, this always happens if we do not use variable precision arithmetics.

	Newton	Halley	KSM	CTM
iterations	15	10	7	6
$\|U_{k+1} - U_k\|_2$	3.43e-14	2.89e-15	2.22e-16	5.00e-15
$\|U_{k+1}^T U_{k+1} - I_{50}\|_2$	2.14e-15	4.49e-16	4.44e-16	6.66e-16
e-time	0.1881	0.0830	0.0996	0.0749
ACOC	1.9993	2.5421	-	-

Table 5: Numerical results for a tridiagonal matrix of size 200 × 200 in double precision arithmetics

	Newton	Halley	KSM	CTM
iterations	17	12	8	7
$\|U_{k+1} - U_k\|_2$	3.39e-14	4.44e-16	4.44e-16	2.22e-16
$\|U_{k+1}^T U_{k+1} - I_{50}\|_2$	3.12e-15	4.44e-16	4.45e-16	9.89e-31
e-time	0.2044	0.0977	0.0999	0.0861
ACOC	1.9993	-	-	-

Table 6: Numerical results for a tridiagonal matrix of size 200 × 200 and $U_0 = D/\|D\|_2$, in double precision arithmetics

Although in the convergence theorem we assume matrix A as initial guess, we want to analyze what does happen if we use other initial estimation. The results that we present in Table 6 correspond to the tridiagonal matrix D, with initial guess $U_0 = D/\|D\|_2$ in double precision arithmetics.

For many multiples of matrix A, we obtain similar results, but for any multiple of the identity matrix all methods are not convergent.

The numerical tests shown in Tables 1 to 6 confirm the theoretical results and we can state that the new algorithm reduces the number of iterations and the computational time in finding the polar decomposition, respect to the rest of schemes.

5. Conclusions

The polar decomposition of a matrix is an important theoretical and computational tool, so techniques for computing it are of interest. In this paper, we have proposed and iterative method, for approximating the polar decomposition of an arbitrary complex matrix, with order of convergence six. Many numerical tests (with matrices of different dimensions) have been presented to show the performance of the new method and to confirm the theoretical results. The proposed new method is competitive in relation to the known ones and it is not a member of Padé family.

References

[1] G. Golub, C. Van Loan, Matrix Computations, Johns Hopkins University Press, Baltimore, MD, 1983.

[2] R.A. Horn, C.A. Johnson, Matrix Analysis, Cambridge University Press, London, 1985.

[3] L. Autonne, Sur les groupes lineaires, reels et orthogonaux, Bull. Sot. Math. France 30 (1902) 121–134.

[4] N.J. Higham, Computing the polar decomposition - With applications, SIAM J. Sci. Statist. Comput. 7 (1986) 1160–1174.

[5] K. Du, The iterative methods for computing the polar decomposition of rank-deficient matrix, Appl. Math. Comput. 162 (2005) 95–102.

[6] W. Gander, Algorithms for the polar decomposition, SIAM J. Sci. Statist. Comput. 11(6) (1990) 1102–1115.

[7] F. Khaksar, F. Soleymani, On a fourth-order matrix method for computing polar decomposition, Comp. Appl. Math. 34 (2015) 389–399.

[8] B. Iannazzo, A family of rational iterations and its application to the computation of the matrix Pth root, SIAM J. Matrix Analysis and Applications 30 (2008) 1445–1462.

[9] N.J. Higham, The matrix sign decomposition and its relation to the polar decomposition, Linear Algebra and Appl. 212/213 (1994) 3–20.

[10] C. Kenney, A.J. Laub, On scaling Newton's method for polar decomposition and the matrix sign function, SIAM J. Matrix Analysis and Applications 13 (1992) 688–706.

[11] M. Sharifi, S. Karimi, F. Khaksar, M. Arab, S. Shateyi, On a cubically convergent iterative method for matrix sign, The Scientific World Journal, Volume 2015 (2015), Article ID 964257, 6 pages.

[12] F. Chicharro, A. Cordero, J.R. Torregrosa, Drawing dynamical and parameter planes of iterative families and methods, The Scientific World Journal, Volume 2013 (2013), Article ID 780153.

[13] A. Cordero, J.R. Torregrosa, Variants of Newton's method using fifth-order quadrature formulas, Applied Mathematics and Computation 190 (2007) 686–698.