UNIVERSIDADE ESTADUAL DE CAMPINAS INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E COMPUTAÇÃO CIENTÍFICA
DEPARTAMENTO DE MATEMÁTICA APLICADA

Renan Das Chagas Palma

Decomposição Polar De Matrizes

Monografia apresentada ao Instituto de Matemática, Estatística e Computação Científica da Universidade Estadual de Campinas como parte dos requisitos para obtenção de créditos na disciplina Projeto Supervisionado, sob a orientação do Prof. Alberto Saa.

Campinas
24/06/2024

Contents

1 Introduction

The polar decomposition of a given complex or real matrix A is a factorization of the form $A = UP$ where U is a unitary e P is a positive semi-definite hermitian matrix. In fact, as we will see later P has the same rank as A, we will also see that A doesn't have to be square. If $A \in \mathbb{C}^{n\times m}$ with $n \geq m$ there will be $A = UP$ with the only difference $U \in \mathbb{C}^{n\times m}$ is a matrix with columns forming an orthonormal set this matrix that has very similar properties of a unitary one. The case A is real is completely analogous with only difference U is orthogonal. The polar decomposition can be found in [1, 3, 4].

It will be shown that the existence of this decomposition is a direct consequence of the Singular Value Decomposition (SVD) of a given matrix, a well-known result that will be shown later in this work, exists for any matrix A. Furthermore, we will explore a straightforward method to compute the Polar decomposition of A using its SVD.

The primary objective of this work is to introduce an alternative algorithm for calculating the polar decomposition, distinct from the SVD-based approach. This algorithm called Newton-Schulz cubic iteration involves a series of iterative steps applied to A to derive U. We will test this algorithm and compare its performance with the conventional SVD-based method. You can find a detailed description of the method in [3].

2 SVD

First some definitions from [2].

Definition 1 (Positive Semi-definite matrix)**.** *A square symmetric matrix $A \in \mathbb{C}^{n\times n}$ is called positive semi-definite if for all vectors $z \in \mathbb{C}^n$, the following condition holds:*

$$z^* Az \geq 0,$$

where z^* denotes the conjugate transpose of z.

Definition 2 (Rank)**.** *Let $A \in \mathbb{C}^{n\times m}$. Consider Range$(A) = \{y \in \mathbb{C}^n \mid Ax = y\}$, which is a subspace of $\mathbb{C}^n$. We call rank(A) the dimension of this subspace.*

Definition 3 (Unitary Matrix)**.** *A square matrix A is called unitary iff $AA^* = I$ this set of matrices are very important, they are length and angle preserving transformations. there is a caracterization will be used later on that can be easily proved. the columns or rows of A are an orthonormal set.*

The next theorem is an exercise left for the reader from [2].

Theorem 2.1 (Singular Value Decomposition)**.** *Given $A \in \mathbb{C}^{n\times m}$ a non-zero complex matrix with rank r. A can be expressed as $A = U\Sigma V^*$ where $U \in \mathbb{C}^{n\times n}$ and $V \in \mathbb{C}^{m\times m}$ are a unitary matrices and $\Sigma \in \mathbb{C}^{n\times m}$ is a particularly interesting form, $\Sigma_{ii} = \sigma_i$ for $i \in \{1, \ldots, r\}$ with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$ and 0 everywhere else. σ_i are called the singular values of the matrix A and are unique.*

Proof. This exercise is based on induction. First, let's consider the case when $\text{rank}(A) = 1$. Let v be a basis for the range of A. This means that $Ae_i \in \text{span}(\{v\})$ for each i, where e_i is the i-th standard basis vector. Therefore, we can write $a_i = \gamma_i v$ for some scalars γ_i, where a_i is the i-th column of A. In other words, we have:

$$\frac{a_i}{\gamma_i} = \frac{a_j}{\gamma_j} \quad \text{for all } i, j \in \{1, \ldots, m\}. \tag{2.1}$$

So, every column of A is a multiple of any other column. Therefore, A can be written as:

$$A = \begin{bmatrix} a_1 & \tilde{\gamma}_2 a_1 & \cdots & \tilde{\gamma}_m a_1 \end{bmatrix} = a_1 v^* \tag{2.2}$$

where $v^* = \begin{bmatrix} 1 & \tilde{\gamma}_2 & \cdots & \tilde{\gamma}_m \end{bmatrix}$. Let $\sigma_1 = \|a_1\|_2 \|v^*\|_2$. This implies that $A = \sigma_1 a_1 v^*$, where both a_1 and v^* are vectors with unit norm.

We can easily construct a Householder transformations $U \in \mathbb{C}^{n \times n}$ and $V \in \mathbb{C}^{m \times m}$ that map e_1 to a_1 and e_1 to v, respectively. Specifically, $Ue_1 = a_1$ and $Ve_1 = v$. Consequently, the first columns of U and V are a_1 and v, respectively.

Finally, it is straightforward to see that:

$$A = \sigma_1 a_1 v^* = U \begin{bmatrix} \sigma_1 & \dots & 0 \\ 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{bmatrix} V^*. \tag{2.3}$$

Thus, every rank-1 matrix has a singular value decomposition (SVD).

Now, let's consider A with rank $r > 1$. Take v_1 as a unit vector in the direction of maximum magnification of A. In other words, let $\sigma_1 = \|Av_1\|_2 = \|A\|_2$. Define $u_1 = \sigma_1^{-1} A v_1$.

Let $\tilde{U} \in \mathbb{C}^{n \times n}$ and $\tilde{V} \in \mathbb{C}^{m \times m}$ be unitary matrices where u_1 and v_1 are their respective first columns. Consider the matrix $\tilde{A} = \tilde{U}^* A \tilde{V}$. Then, $\tilde{A}$ has an interesting form:

$$\tilde{A} = \tilde{U}^* [A\tilde{v}_1 \ A\tilde{v}_2 \ \dots \ A\tilde{v}_m] = \tilde{U}^* [\sigma_1 u_1 \ Av_2 \ \dots \ Av_m] = [\tilde{U}^* \sigma_1 u_1 \quad \tilde{U}^* A v_2 \quad \dots \quad \tilde{U}^* A v_m]. \tag{2.4}$$

The first column of $\tilde{A}$ is

$$\tilde{U}^* \sigma_1 u_1 = \begin{bmatrix} u_1^* \\ u_2^* \\ \vdots \\ u_n^* \end{bmatrix} \sigma_1 u_1 = \sigma_1 \begin{bmatrix} \langle u_1, u_1 \rangle \\ \langle u_1, u_2 \rangle \\ \vdots \\ \langle u_1, u_n \rangle \end{bmatrix} = \sigma_1 \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

So,

$$\tilde{A} = \begin{bmatrix} \sigma_1 & z^* \\ 0 & \hat{A} \end{bmatrix}. \tag{2.5}$$

where $z \in \mathbb{C}^{m-1}$ and $\hat{A} \in \mathbb{C}^{(n-1) \times (m-1)}$.

Also, if

$$w = \begin{bmatrix} \sigma_1 \\ z \end{bmatrix} \in \mathbb{C}^m$$

then,

$$\frac{\|Aw\|_2}{\|w\|_2} = \frac{\sqrt{(\sigma_1^2 + z^* z)^2 + \sum_i (\hat{A} z)_i^2}}{\sqrt{\sigma_1^2 + z^T z}} \geq \frac{\sqrt{(\sigma_1^2 + z^* z)^2}}{\sqrt{\sigma_1^2 + z^* z}} = \sqrt{\sigma_1^2 + z^* z}. \tag{2.6}$$

But

$$\sigma_1 \geq \frac{\|Aw\|_2}{\|w\|_2} \geq \sqrt{\sigma_1^2 + z^* z}, \tag{2.7}$$

which means $z = 0$.

Then A has this form:

$$\tilde{A} = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \hat{A} \end{bmatrix}. \tag{2.8}$$

Now we clearly have $\tilde{A}$ with rank $r-1$, and so by the induction hypothesis, we have an SVD for $\hat{A}$, $\hat{A} = \hat{U}\hat{\Sigma}\hat{V}^*$. Note that we also have:

$$\tilde{A} = \begin{bmatrix} 1 & 0 \\ 0 & \hat{U} \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 \\ 0 & \hat{\Sigma} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \hat{V}^* \end{bmatrix}. \tag{2.9}$$

Thus, since $A = \tilde{U}\tilde{A}\tilde{V}^*$, we have:

$$A = \tilde{U} \begin{bmatrix} 1 & 0 \\ 0 & \hat{U} \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 \\ 0 & \hat{\Sigma} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \hat{V}^* \end{bmatrix} \tilde{V}^* = U\Sigma V^*, \tag{2.10}$$

where U and V are unitary.

□

An important corollary that is going to be used to guarantee the convergence in the Newton-Schulz algorithm is this:

$$\|A\|_F = \left(\sum_{i=1}^{n} \sum_{j=1}^{m} |a_{ij}|^2 \right)^{1/2} \tag{2.11}$$

which is the Frobenius norm. It will be shown that if A has rank r, then,

$$\|A\|_F = \left(\sum_{i=1}^{r} \sigma_i^2 \right)^{1/2}. \tag{2.12}$$

This is a straightforward result if we consider another true result:

$$\|UA\|_F = \|A\|_F \tag{2.13}$$

if U is unitary. For this, just notice that

$$\|A\|_F = \sqrt{\mathrm{tr}(A^*A)}$$

because $(A^*A)_{ij} = \sum_{k=1}^{n} \overline{a_{ik}} \cdot a_{jk}$, and so $(A^*A)_{ii} = \sum_{k=1}^{n} |a_{ik}|^2$. Hence,

$$\mathrm{tr}(A^*A) = \sum_{i=1}^{n} \sum_{k=1}^{n} |a_{ik}|^2 = \|A\|_F^2. \tag{2.14}$$

Then if $A = BC$ where B is unitary,

$$\|A\|_F^2 = \|BC\|_F^2 = \mathrm{tr}((BC)^*BC) = \mathrm{tr}(C^*(B^*B)C) = \mathrm{tr}(C^*C) = \|C\|_F^2. \tag{2.15}$$

In light of this result and the fact that the trace is cyclic ($\mathrm{tr}(XYZ) = \mathrm{tr}(ZXY) = \mathrm{tr}(YZX)$) [6], we have:

$$\|A\|_F = \|U\Sigma V^*\|_F = \|\Sigma\|_F = \left(\sum_{i=1}^{r} \sigma_i^2 \right)^{1/2} \tag{2.16}$$

3 Polar Decomposition

Now the existence and uniqueness of the polar decomposition will be shown: Given $A \in \mathbb{C}^{n\times m}$ let $A = S\Sigma V^*$ be it's SVD decomposition. According to the steps outlined in [4]:

$$\begin{aligned} A = S\Sigma V^* = SI_n\Sigma V^* &= \left[\begin{array}{c|c} S_1 & S_2 \end{array}\right] \left[\begin{array}{c|c} V^*V & 0 \\ \hline 0 & I_{n-m} \end{array}\right] \Sigma V^* \\ &= \left[\begin{array}{c|c} S_1V^*V & S_2 \end{array}\right] \left[\begin{array}{c} \Sigma_m \\ \hline 0 \end{array}\right] V^* \\ &= \left[\begin{array}{c|c} S_1V^*V\Sigma_m + S_2 \cdot 0 \end{array}\right] V^* = S_1V^*V\Sigma_mV^* = (S_1V^*)(V\Sigma_mV^*) \end{aligned} \tag{3.1}$$

where $S \in \mathbb{C}^{n\times n}$, $S_1 \in \mathbb{C}^{n\times m}$, $S_2 \in \mathbb{C}^{n\times n-m}$ and $V \in \mathbb{C}^{m\times m}$.

Now we can easily show that

$$U = S_1V^* \in \mathbb{C}^{n\times m} \tag{3.2}$$

has columns forming an orthonormal set, and that

$$P = V\Sigma_mV^* \in \mathbb{C}^{m\times m} \tag{3.3}$$

is a semidefinite-positive matrix.

First, is straight forward to see that $P = P^*$ and since the diagonal entries of Σ_m are all non-negative numbers, we can write $\Sigma_m = \Sigma_m^{\frac{1}{2}}\Sigma_m^{\frac{1}{2}}$ where $\Sigma_{ii}^{\frac{1}{2}} = \sigma_i^{\frac{1}{2}}$. Thus, we can easily prove that P is semi-definite positive. If $x \in \mathbb{C}^m$, then

$$x^*Px = x^*(V\Sigma V^*)x = x^*(V\Sigma^{\frac{*}{2}}\Sigma^{\frac{1}{2}}V^*)x = (\Sigma^{\frac{1}{2}}V^*x)^*(\Sigma^{\frac{1}{2}}V^*x) = y^*y = \|y\|_2^2 \geq 0,$$

where $y = \Sigma^{\frac{1}{2}}V^*x$.

Second, By definition S_1 has columns forming an orthonormal set and V is unitary. The columns of U are $S_1\overline{v}_i$ where v_i are the rows of V and so

$$\langle S_1\overline{v}_i, S_1\overline{v}_j\rangle = (S_1\overline{v}_j)^*(S_1\overline{v}_i) = \overline{v}_j^*S_1^*S_1\overline{v}_i = v_j^T\overline{v}_i = (v_j^T\overline{v}_i)^T = v_i^*v_j = \langle v_j, v_i\rangle = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

And the columns of U are orthonormal.

For the uniqueness, P is always unique because $A^*A = P^*U^*UP = P^*P$. But since P is Hermitian, $A^*A = P^2$. Now, a well-known result is that every complex matrix is unitarily similar to an upper triangular matrix (Schur decomposition). It's straightforward to show that if a matrix is normal (it commutes with its adjoint), then the upper triangular matrix in the Schur decomposition is, in fact, a diagonal one. So, since A^*A is Hermitian, and a Hermitian matrix commutes with its adjoint, a Hermitian matrix is always diagonalizable. These statements are well known and can be found in [1] and [2].

Then, $P^2 = EDE^{-1}$, where $D = \text{diag}(\lambda_1, \ldots, \lambda_m)$ contains the eigenvalues, and E consists of the eigenvectors of A^*A. Consequently, $P = ED^{1/2}E^{-1}$, and for each non-zero element in D, we have two possibilities: $\pm\sqrt{\lambda_i}$ for the elements of $D^{1/2}$.

If A^*A is full rank, it implies that there are 2^m square roots! P is unique, though, because first, $\lambda_i \geq 0$. If $\lambda_i < 0$, for the eigenvector v_i associated with λ_i, we would have:

$$v_i^*(A^*A)v_i = v_i^*(\lambda_iv_i) = \lambda_i\|v_i\|_2^2 < 0$$

This would contradict the fact that A^*A is positive semidefinite. Since P is also positive semidefinite, the only possible choice for the diagonal elements of $D^{1/2}$ are the positive square roots of the elements of D.

3.1 Newton-Schulz

The Newton-Schulz cubic iteration is an algorithm used to iteratively compute the matrix U in the decomposition $A = UP$. Once we have U, we can easily determine P using the relation $P = U^*A$.

The iterative method is based on the following iteration:

$$X_{k+1} = \frac{1}{2} X_k \left(3I - X_k^* X_k\right), \tag{3.4}$$

where $X_0 = A$. Note here X_k keeps the size of X_0 for each iteration.

Let's see why this always converges to U. Let's consider $X_k \in \mathbb{C}^{n\times m}$ with $n \geq m$. The SVD of X_k, which we know always exists:

$$X_k = \tilde{U}\Lambda_k \tilde{V}^*. \tag{3.5}$$

After one iteration, we will have:

$$\begin{aligned} X_{k+1} &= \frac{1}{2}\tilde{U}\Lambda_k\tilde{V}^* \left(3I - (\tilde{U}\Lambda_k\tilde{V}^*)^*\tilde{U}\Lambda_k\tilde{V}^*\right) \\ &= \frac{1}{2}\tilde{U}\Lambda_k\tilde{V}^* \left(3I - \tilde{V}\Lambda_k^*(\tilde{U}^*\tilde{U})\Lambda_k\tilde{V}^*\right) \\ &= \frac{3}{2}\tilde{U}\Lambda_k\tilde{V}^* - \frac{1}{2}\tilde{U}\Lambda_k\Lambda_k^*\Lambda_k\tilde{V}^* \\ &= \tilde{U}\left(\frac{3}{2}\Lambda_k - \frac{1}{2}\Lambda_k\Lambda_k^*\Lambda_k\right)\tilde{V}^* \\ &= \tilde{U}\Lambda_{k+1}\tilde{V}^*, \end{aligned} \tag{3.6}$$

where the diagonal entries of Λ_{k+1} are given by

$$\lambda_{k+1}^{(i)} = \frac{1}{2}(3\lambda_k^{(i)} - (\lambda_k^{(i)})^3). \tag{3.7}$$

where $\lambda_k^{(i)} = (\Lambda_k)_{ii}$.

In other words, this iteration maintains the unitary factors of the preceding matrix. Consequently, it retains the unitary factors of the initial matrix A. Thus, we have $\tilde{U} = S$ and $\tilde{V}^* = V^*$, where S and V are the unitary matrices in the SVD of A. The values of Λ_{k+1} evolve according to the simple cubic polynomial $p(x) = \frac{1}{2}(3x - x^3)$, and we know that if $x_0 \in (0, 1)$ the sequence x_k define by $x_k = p^k(x_0)$ is increasing and bounded by 1, because

$$x_k \in (0,1) \to (1 - x_k^2) > 0 \to x_k(1 - x_k^2) > 0 \to x_k(3 - x_k^2) > 2x_k \to \frac{1}{2}(3x_k - x_k^3) > x_k \to p(x_k) > x_k.$$

Therefore $\{p^k(x_0)\}$ converges to L and because p is continuous:

$$L = \lim_{k\to\infty} p^k(x_0) = p\left(\lim_{k\to\infty} p^{k-1}(x_0)\right) = p(L). \tag{3.8}$$

and L is a fixed point of p. Then $L = 0$ or $L = 1$ since $x_0 \neq 0$ and p is monotonically increasing in $(0, 1)$ it must be 1. If A is full rank, then:

$$\lim_{k\to\infty} \Lambda_k = \begin{bmatrix} I_m \\ \hline 0 \end{bmatrix} \tag{3.9}$$

and

$$\lim_{k\to\infty} X_k = \hat{S}V^* \tag{3.10}$$

where $\hat{S}$ is the first m columns of S. Thus, in light of the equation 3.2 from last section, X_k converges to the unitary factor in the polar decomposition. Unfortunately if A is not full rank, the newton-schulz algorithm fails to converge for the right unitary matrix. If A has rank r<m.

$$\lim_{k\to\infty} \Lambda_k = \left[\begin{array}{c|c} I_r & 0 \\ \hline 0 & 0 \end{array}\right]$$

and

$$X = \lim_{k\to\infty} X_k = S\left[\begin{array}{c|c} I_r & 0 \\ \hline 0 & 0 \end{array}\right] V^* \tag{3.11}$$

note here $S \in \mathbb{C}^{n\times m}$ and $X \in \mathbb{C}^{n\times m}$ with $n \geq m$ which has columns forming an orthormal set if $X^*X = I_m$ and so,

$$X^*X = V\left[\begin{array}{c|c} I_r & 0 \\ \hline 0 & 0 \end{array}\right] S^*S \left[\begin{array}{c|c} I_r & 0 \\ \hline 0 & 0 \end{array}\right] V^* = V\left[\begin{array}{c|c} I_r & 0 \\ \hline 0 & 0 \end{array}\right]\left[\begin{array}{c|c} I_r & 0 \\ \hline 0 & 0 \end{array}\right] V^*. \tag{3.12}$$

Where the second and third matrices are $m \times n$ and $n \times m$ respectively. Note if A was full rank, those matrices would have been

$$\left[\begin{array}{c|c} I_m & 0 \end{array}\right]\left[\begin{array}{c} I_m \\ \hline 0 \end{array}\right] = I_m.$$

and then, from (3.11), we have

$$X^*X = (v_1 \dots v_r\ 0 \dots 0)V^* \neq I_m. \tag{3.13}$$

An interesting thing happens in the case A is not full rank. even X_k not converging for U, the algorithm still converges to the correct matrix P. This is because

$$X^*A = \left(S\left[\begin{array}{c|c} I_r & 0 \\ \hline 0 & 0 \end{array}\right] V^*\right)^* S\Sigma V^* = V\left[\begin{array}{c|c} \Sigma_r & 0 \\ \hline 0 & 0 \end{array}\right] V^* \tag{3.14}$$

Where the middle matrix is an $m \times m$ matrix, which, according to the results of the last section 3.3, is in fact the matrix P.

To ensure the algorithm works, we need to start with a full-rank matrix that retains the same unitary factors as in the SVD of A and has singular values within the interval $(0, 1]$. One suitable choice is to normalize A by its Frobenius norm:

$$\frac{A}{\|A\|_F}$$

This normalized matrix has the same unitary factors as A, and the maximum singular is within $(0, 1]$.

$$\frac{\sigma_j}{\sqrt{\sum_{i=1}^m \sigma_i^2}} \leq \frac{\sigma_j}{\sqrt{\sigma_1^2}} \leq \frac{\sigma_1}{\sqrt{\sigma_1^2}} = 1.$$

Algorithm 1 Newton-Schulz cubic iteration

Input: $A \in \mathbb{C}^{n\times m}$, N number of iterations, ϵ tolerance
Output: Y, Y^*A

$X \leftarrow \frac{A}{\|A\|_F}$
for $k = 0, 1, \ldots, N-1$ **do**
  $Y \leftarrow \frac{1}{2}X(3I - X^*X)$
  if $|\|Y\|_F - \sqrt{m}| < \epsilon \cdot n \cdot m$ **then**
    return Y, Y^*A
  else
    $X \leftarrow Y$
  end if
end for
return Y, Y^*A

Algorithm 2 SVD Approach

Input: $A \in \mathbb{C}^{n\times m}$
Output: U, P

$S, \Sigma, V \leftarrow \text{svd}(A)$
return SV^*, $V\Sigma V^*$

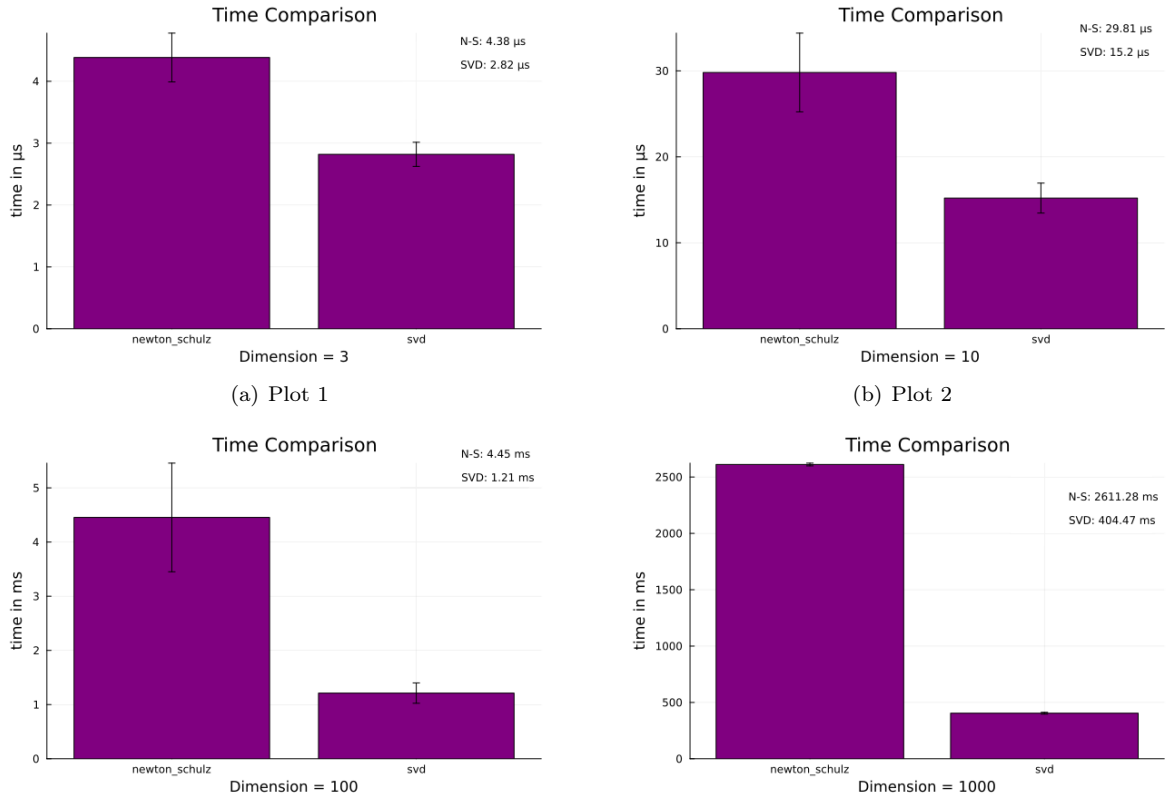
4 Computational Tests

All the tests were conducted using Julia notebooks in Pluto, focusing on matrices with entries of type `Float64` and `ComplexF64`. The computational setup consisted of a desktop equipped with an AMD Ryzen 5 4th generation CPU and 16 GB of RAM. The tests were performed under the Windows 10 operating system using Julia version 1.10.3, and Julia was configured to run with a single thread.

It is reasonable to expect that the performance of the Newton-Schulz algorithm will be better when considering matrices with singular values uniformly distributed in $(\alpha, 1)$ with $\alpha > 0$ sufficiently far from 0. In fact, what matters is the relative distance between the singular values. The closer $\dfrac{\sigma_{\min}}{\sigma_{\max}}$ is to 1, the fewer iterations will be needed to converge.

The first aspect I examined was the performance of the Newton-Schulz algorithm compared to SVD with matrices with random entries uniformly distributed in $(0, 1)$.

The graph below illustrates median time comparisons using the @benchmark function from the BenchmarkTools library in Julia to run Newton-Schulz and SVD for square matrices $A \in \mathbb{R}^{n\times n}$ with $n \in \{3, 10, 100, 1000\}$. For more details how algorithms, tests and graphs were made check this link [5].



(a) Plot 1

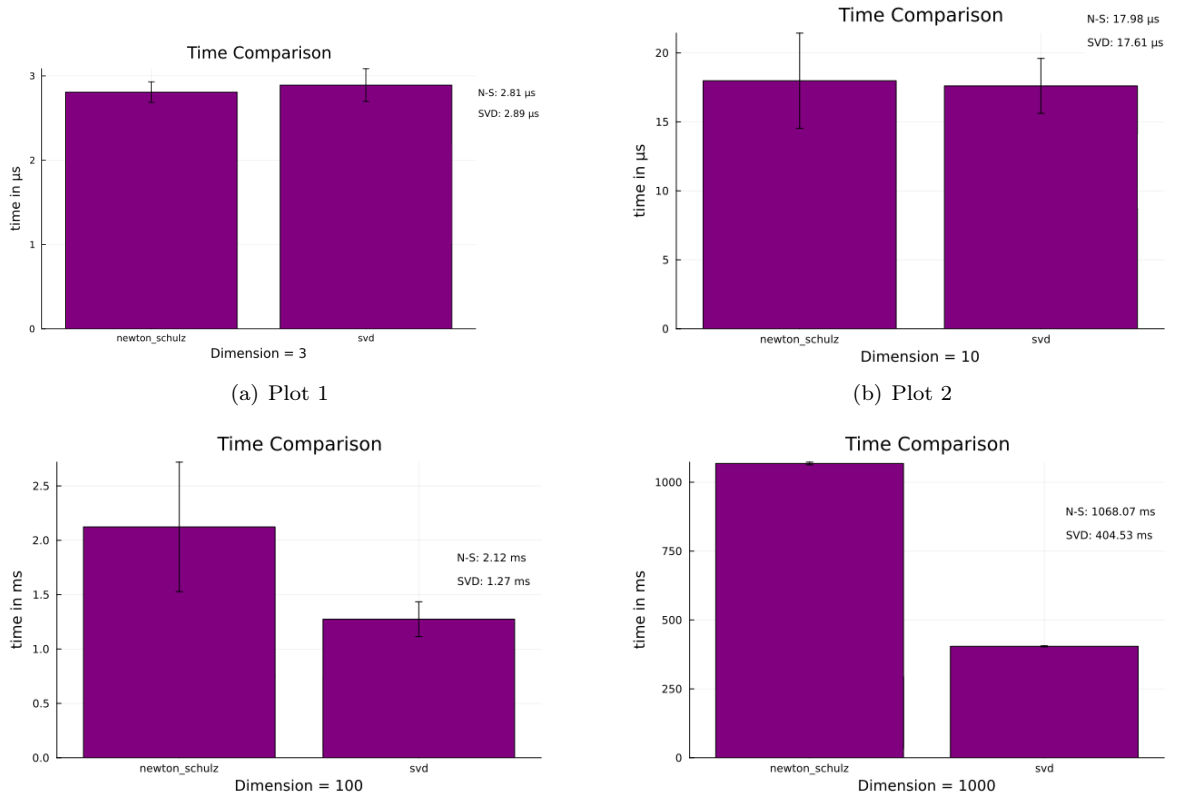
(b) Plot 2

(c) Plot 3

(d) Plot 4

Figure 1: Mean and stdeviation using @benchmark function from the benchmarkTools library in Julia

The same test mentioned above was conducted, considering matrices with singular values uniformly distributed in the $(\frac{1}{2}, 1)$ interval. Here, as mentioned above, since the minimum singular value is closer to 1, it needs fewer iterations to converge for the unitary matrix U that's why the performance of newton-schulz is better.



(a) Plot 1

(b) Plot 2

(c) Plot 3

(d) Plot 4

Figure 2: Mean and stdeviation using @benchmark function from the benchmarkTools library in Julia

Lastly, larger matrices require more computation time for two main reasons. Firstly, the number of calculations increases proportionally with matrix size when considering the FLOP count. Counting only the basic arithmetic operations involved we have:

$$X_k^* X_k$$

is an n^2 dot product with vectors of length m, each one accounting for m multiplications and $m-1$ additions, totaling $(2m-1)n^2$ operations. The expression $3I - X_k^* X_k$ introduces additional m additions and $2m$ multiplications. The final multiplication, $\frac{1}{2}X_k(3I - X_k^* X_k)$, involves nm dot products with vectors of length m, contributing $(2m-1)nm + m$ operations. Therefore, the total flop count per iteration is $5m + (2m-1)(n^2 + nm)$, for each iteration. The second reason is, of course, that as the matrix size increases, the number of singular values also increases, and the probability of having smaller singular values increases as well.

The next 3 figures illustrate the number of iterations needed for convergence. All figures used matrices of sizes 3, 10, 100, and 1000. In Figure 3, the matrices were real with entries uniformly distributed in $(0, 1)$. In Figure 4, complex matrices were used, with real and imaginary parts uniformly distributed in $(0, 1)$. Figure 5 shows matrices with singular values uniformly distributed in $(\frac{1}{2}, 1)$.

The graphs do not show any difference between the complex and real matrices. However, as discussed earlier, significantly fewer iterations are needed for the matrices in Figure 5. This is illustrated in the earlier plots as well, for example, in the times for matrices 1000 in figure 1 versus figure 2.

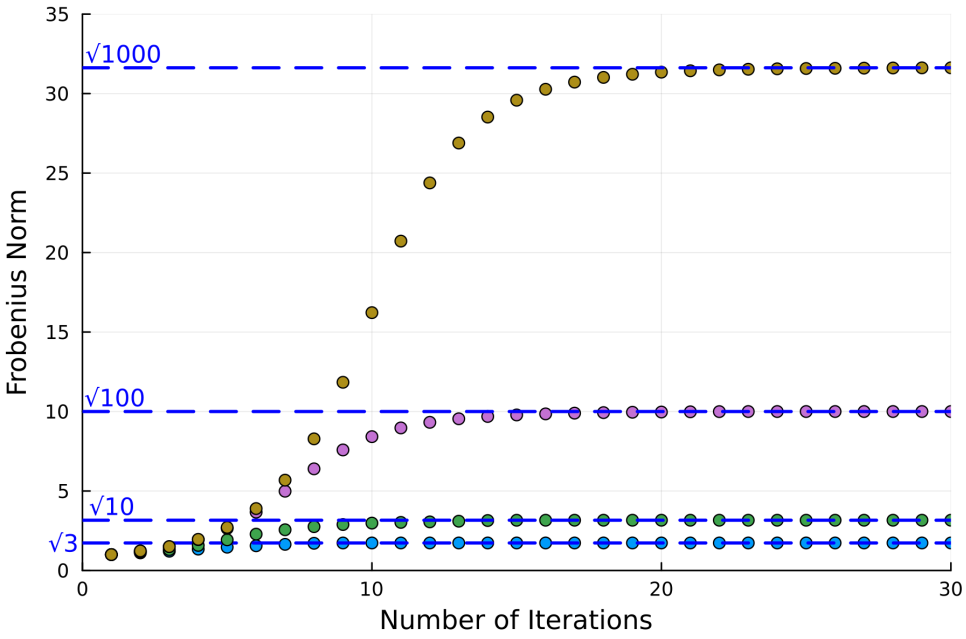


Figure 3:

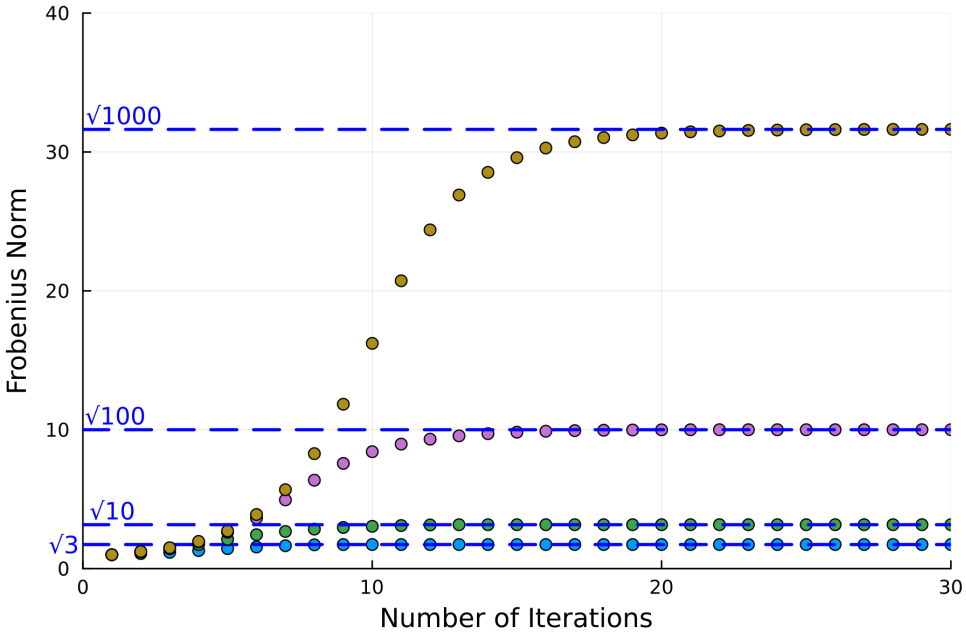


Figure 4:

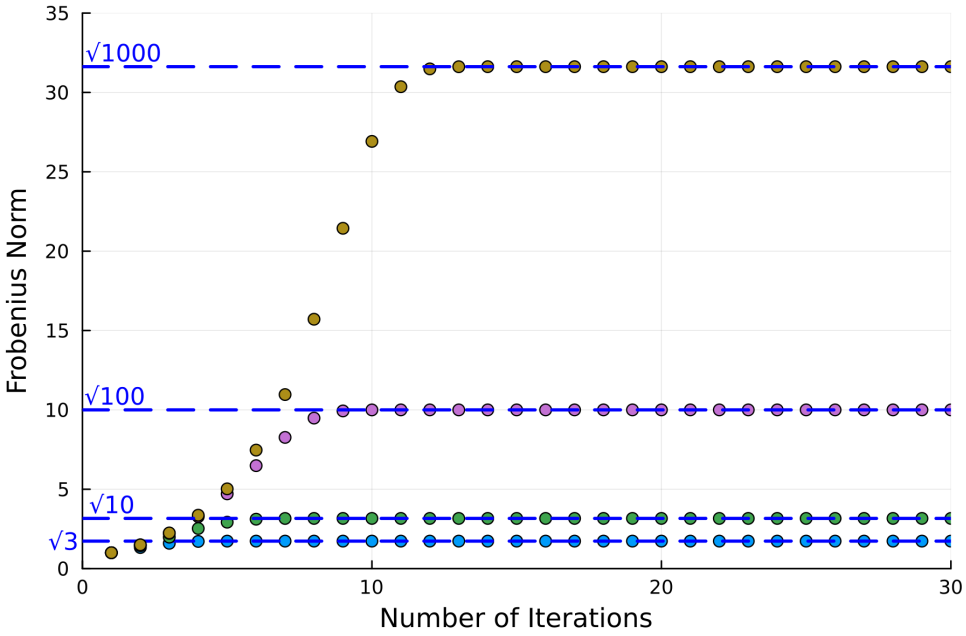


Figure 5:

5 Conclusion

In this study, I compared two distinct algorithms for computing the polar decomposition of a matrix: the Newton-Schulz cubic iteration and a method based on Singular Value Decomposition (SVD). Through a series of experiments and theoretical analysis, I evaluated the performance of each algorithm in various scenarios. Although I couldn't find how the SVD algorithm from the 'LinearAlgebra' library in Julia is implemented, I used as a proxy the algorithm described in [1]. This algorithm, known as the Golub-Reinsch SVD has a computational complexity $4m^2n + 8m^2n + 9n^3$ operations to calculate Σ, U, and V. This number makes sense when we look at the results: for the matrices in (a) Plot 2, the times are very similar, and in that case, the flop count is also similar, approximately $24n^2m = 24n^3$ for Newton-Schulz.

References

[1] Gene H. Golub, Charles F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Fourth Edition, 2013.

[2] David S. Watkins, *Fundamentals of Matrix Computations*, John Wiley & Sons, Third Edition, 2010.

[3] Li, W., & Sun, W. (2002). Perturbation Bounds of Unitary and Subunitary Polar Factors. SIAM Journal on Matrix Analysis and Applications, 23(4), 1183–1193.

[4] Higham, N. J. (1986). Computing the polar decomposition—with applications. *SIAM Journal on Scientific and Statistical Computing.*

[5] GitHub Repository for Polar Decomposition, https://github.com/rcpalma/ms777---polar-decomposition/blob/main/notebook_used.

[6] "Polar Decomposition". Wikipedia, https://en.wikipedia.org/wiki/Polar_decomposition.

[7] "Matrix Norm". Wikipedia, https://en.wikipedia.org/wiki/Matrix_norm