Matrix decompositions over the double numbers

Ran Gutin
Department of Computer Science, Imperial College London

December 7, 2021

Abstract

We take matrix decompositions that are usually applied to matrices over the real numbers or complex numbers, and extend them to matrices over an algebra called the double numbers. In doing so, we unify some matrix decompositions: For instance, we reduce the LU decomposition of real matrices to LDL decomposition of double matrices. We similarly reduce eigendecomposition of real matrices to singular value decomposition of double matrices. Notably, these are opposite to the usual reductions. This provides insight into linear algebra over the familiar real numbers and complex numbers. We also show that algorithms that are valid for complex matrices are often equally valid for double matrices. We finish by proposing a new matrix decomposition called the Jordan SVD, which we use to challenge a claim made in Yaglom’s book Complex Numbers In Geometry concerning Linear Fractional Transformations over the double numbers.

1 Introduction

In this paper, we consider extending numerical linear algebra to a hypercomplex number system called the double numbers (and related number systems like the double-complex numbers / tessarines). Specifically, we are interested in the subject of matrix decompositions.

The process we follow is as follows:

1. Take a matrix decomposition over the complex numbers (such as LDL, SVD, QR).
2. Extend it to double matrices via analogy.
3. Break it into real components.
4. Observe what decompositions of real matrices emerge out.

We often find that the outcome of step 4 is interesting, in that the decomposition of real matrices that emerges from step 4 is often something recognisable. Some examples include: If in step 1, we start with the LDL decomposition (which only applies to Hermitian matrices), then in step 4 we derive the LU decomposition.
(which applies to most other matrices). Another example is that if we start with the Singular Value Decomposition, which is often seen as a special case of eigendecomposition, then in step 4 we derive precisely eigendecomposition.

We also observe that algorithms that are valid for the decomposition in step 1 can be extended to the decomposition in step 4. Sometimes this extension of algorithms can be done automatically.

The third step (breaking into real components) leads us to a minor novelty: We write a double matrix using the notation \[A, B\], which we define to mean \(A \frac{1 + j}{2} + B^T \frac{1 - j}{2}\) where \(A\) and \(B\) are real matrices. Crucially, note that \(B\) is transposed. The symbol \(j\) is defined below.

## 2 Definitions

### 2.1 Double numbers and double matrices

The *double numbers*, called split-complex numbers on Wikipedia, and *perplex numbers* in some other sources) are the hypercomplex number system in which every number is of the form

\[a + bj, a \in \mathbb{R}, b \in \mathbb{R}\]

where \(j\) is a number that satisfies \(j^2 = 1\) without \(j\) being either 1 or \(-1\).

The definition of the arithmetic operations should be clear from the above description. Define \((a + bj)^* = a - bj\) (similar to complex conjugation).

Consider the basis \(e = \frac{1 + j}{2}\) and \(e^* = \frac{1 - j}{2}\). Observe that \(e^2 = e\), \((e^*)^2 = e^*\) and \(ee^* = e^*e = 0\). Thus \((ae + be^*)(ce + de^*) = (ac)e + (bd)e^*\). In other words, multiplication of double numbers is componentwise in this basis. Thanks to this, we see that the double numbers can be defined as the algebra \(\mathbb{R} \oplus \mathbb{R}\), where \(\oplus\) denotes the direct sum of algebras. Observe that \((ae + be^*)^* = be + ae^*\).

We shall consider matrices of double numbers. A matrix \(M\) of double numbers can be written in the form \(A \frac{1 + j}{2} + B^T \frac{1 - j}{2}\) where \(A\) and \(B\) are real matrices. Write \([A, B]\) for \(A \frac{1 + j}{2} + B^T \frac{1 - j}{2}\). Observe the following identities:

\[
[A, B] + [C, D] = [A + C, B + D] \\
[A, B] \times [C, D] = [AC, DB] \\
[A, B]^* = [B, A]
\]

In the above \([A, B]^*\) refers to the conjugate-transpose operation. Notice that the second component of \([A, B] \times [C, D]\) is \(DB\), not \(BD\).
2.2 Families of double matrices

We say that a double matrix $M$ is Hermitian if it satisfies $M^* = M$. Observe then that a Hermitian matrix is precisely of the form $[A, A]$.

We say that a double matrix $M$ is unitary if it satisfies $M^* M = I$. Observe then that a unitary matrix is precisely of the form $[A, A^{-1}]$.

We say that a double matrix $M$ is lower triangular if it is of the form $[L, U]$ where $L$ is a lower triangular real matrix and $U$ is an upper triangular real matrix. Similarly, a matrix of the form $[U, L]$ is called upper triangular.

A double matrix is called diagonal if it is of the form $[D, E]$ where $D$ and $E$ are diagonal real matrices.

A double matrix is real precisely when it is of the form $[A, A^T]$.

2.3 Double-complex numbers and matrices

A double-complex number ([1]) is a number of the form $w + zj$ where $w$ and $z$ are complex numbers, and $j^2 = +1$. Multiplication is the same as over the double numbers, namely: $(w + zj)(w' + z'j) = ww' + zz' + j(wz' + zw')$. The product is a commutative one. The definition of conjugation is $(w + zj)^* = w - zj$. Sometimes these numbers are called tessarines.

By a double-complex matrix, we mean a matrix whose entries are double-complex numbers. A double-complex matrix can always be expressed in the form $[A, B]$, which we define to mean $A \frac{1+j}{2} + B^T \frac{1-j}{2}$ where $A$ and $B$ are complex matrices. Observe that we use the transpose of $B$ rather than the conjugate-transpose of $B$.

Observe then the following identities:

\[
\begin{align*}
[A, B] + [C, D] &= [A + C, B + D] \\
[A, B] \times [C, D] &= [AC, DB] \\
[A, B]^* &= [B, A]
\end{align*}
\]

The various families of matrices (Hermitian, unitary, triangular, diagonal) have identical descriptions over the double-complex numbers as they do over the double numbers. A double-complex matrix $M$ is a complex matrix precisely when it is of the form $M = [A, A^T]$ where $A$ is a complex matrix.

We need these numbers in order to study Singular Value Decomposition over double or double-complex matrices.

2.4 Computing with double numbers

In order to present the algorithms in this paper, we use the Python library Sympy (version 1.6, [2]). We have also implemented some of the algorithms
below using the C++ library Eigen ([3]). We have not included the Eigen code here. Eigen is significantly (likely many orders of magnitude) faster than Sympy for the purposes of computing with double matrices. The slow speed of Sympy is caused by the fact that it’s a symbolic computation library, and when using it, we represent a double number as a symbolic expression in the form $a + b \cdot j$.

We need to implement the following functions to be able to do experiments with double numbers:

```python
from sympy import *

J = var('J')

def simplify_split(e):
    real_part = (e.subs(J, 1) + e.subs(J, -1))/2
    imag_part = (e.subs(J, 1) - e.subs(J, -1))/2
    return simplify(real_part + J*imag_part)

def first_part(m):
    return simplify_split(((1 + J) * m).subs(J, 0))

def second_part(m):
    return simplify_split(((1 - J) * m).subs(J, 0))

def scabs(x):
    """Absolute value for split-complex numbers.""
    return sqrt(abs(first_part(x)) * abs(second_part(x)))
```

The function `simplify_split` is particularly useful for simplifying expressions involving double numbers. It is based on the identity $f(a + bj) = \frac{f(a+b) + f(a-b)}{2} + j\frac{f(a+b) - f(a-b)}{2}$ where $f$ is any analytic function over $\mathbb{R}$. We end up using it a lot in our algorithms, though it’s completely unnecessary if one doesn’t use symbolic algebra the way we have. The functions `first_part` and `second_part` are useful for extracting multiples of the basis vectors $e = \frac{1+j}{2} = \frac{1}{2} + \frac{1}{2}j$ and $e^* = \frac{1-j}{2}$ respectively. To extract $B$ from $M = [A, B]$, one needs to compute `second_part(M.T)` (where $M.T$ means the transpose of $M$). The expression $M^*$ can be expressed as $M.T.subs(J,-J)$.

It is straightforward to adapt the code to other Computer Algebra Systems like Sagemath ([4]).

### 3 Simple matrix decompositions over the double numbers

In this section, we will consider analogues of various matrix decompositions over the double numbers. We will observe by taking components (in other words, by extracting $A$ and $B$ from $[A, B]$) that these decompositions are equivalent to
well-known decompositions of real matrices.

3.1 LDL decomposition (a variant of Cholesky)

Recall the LDL decomposition ([5, 6]): Let $M$ be a Hermitian complex matrix; we have that $M$ can almost always be expressed in the form

$$M = LDL^*$$

where $L$ is a lower triangular matrix and $D$ is a diagonal Hermitian matrix. What is the analogue of this over the double numbers? If we interpret the above decomposition over the double numbers using the definitions in section 2.2, we get

$$[A, A] = [L, U][D, D][L, U]^*,$$

or in other words

$$A = LDU,$$

which is the familiar LDU decomposition.

Notice that while it’s known that LDL trivially reduces to an instance of LU (a more general decomposition), it is not as obvious that a reduction in the opposite direction is possible. We have demonstrated that such a thing is possible.

3.2 Singular Value Decomposition

In the following, we work with square matrices. It is possible to generalise to non-square matrices.

Consider the singular value decomposition ([7]):

$$M = USV^*$$

where $U$ and $V$ are unitary, and $S$ is diagonal and Hermitian. If we interpret this over the double numbers (using section 2.2), we get

$$[A, B] = [P, P^{-1}][D, D][Q^{-1}, Q]$$

We break into components and get

$$A = PDQ^{-1}, B = QDP^{-1}.$$ 

Finally, observe that $AB = PD^2P^{-1}$ and $BA = QD^2Q^{-1}$. In other words, we get the familiar eigendecompositions of $AB$ and $BA$ (which it turns out have the same eigenvalues).

Take note though that in order for a double matrix $[A, B]$ to have an SVD, it may be necessary to work with a complexification of the double numbers. This complexification is sometimes called the double-complex numbers or the tessarines.
Finally, notice that while it’s obvious that SVD is reducible to eigendecomposition (namely, the eigendecompositions of $A^*A$ and $AA^*$) it is less obvious that a reduction in the opposite direction is possible. We have provided such a reduction here.

### 3.3 QR decomposition

Recall that the QR decomposition ([6]) of a complex matrix $M$ is of the form

$$M = QR$$

where $Q$ is a unitary matrix and $R$ is an upper triangular matrix.

Extend the above to double matrices to arrive at

$$[A, B] = [C, C^{-1}][U, L].$$

We get

$$A = CU, B = LC^{-1},$$

and therefore that $BA = LU$. In other words, the analogue of QR decomposition over double matrices yields the LU decomposition of the product of two real matrices.

### 4 Algorithms

In this section, we demonstrate how the standard algorithms for LDL decomposition, QR decomposition, and Singular Value Decomposition generalise to double matrices.

#### 4.1 LDL decomposition

The following relations ([6]) can be used to compute the LDL decomposition of a complex matrix:

$$D_j = A_{jj} - \sum_{k=1}^{j-1} L_{jk}L_{jk}^* D_k,$$

$$L_{ij} = \frac{1}{D_j} \left( A_{ij} - \sum_{k=1}^{j-1} L_{ik}L_{jk}^* D_k \right) \text{ for } i > j$$

These relations generalise to double matrices, and enable one to compute the LDL decomposition of a double matrix in the same way. In doing so, we derive an algorithm (equivalent to the standard one) for computing the LDU decomposition of a real matrix.
4.2 Singular Value Decomposition

There are multiple algorithms for computing the singular values of a matrix, which can usually be adapted for computing the singular vectors as well. Here, for simplicity’s sake, we will focus on only computing the singular values. The simplest such algorithm is the following ([8]):

```python
def svd(A, iters=20):
    M = A.H * A
    for _ in range(iters):
        L, D = Matrix.LDLdecomposition(M)
        M = D ** Rational(1,2) * L.H * L * D ** Rational(1,2)
    return D ** Rational(1,2)
```

This algorithm converges for all real matrices ([9]). A nearly verbatim generalisation to double-complex matrices is provided by figure 1.

It turns out that the resulting algorithm is equivalent to the classic LR algorithm ([9, 6]) for eigendecomposition. In that sense, we’ve started with a special case of the LR algorithm (for SVD) and automatically derived the general LR algorithm (for eigendecomposition of the product of two matrices). The general LR algorithm is rarely used in practice due to its numerical instability.

4.3 QR decomposition

There are multiple ways of computing the QR decomposition of a complex matrix. Two such ways are via the Gram-Schmidt process ([6]) or Householder reflections ([6]). We have implemented both procedures and observed that they generalise to double matrices. In doing this, we have found some new algorithms for computing the LU decomposition of the product of two matrices. First though, we begin by presenting a third way to compute the QR decomposition of a double matrix.

4.3.1 QR algorithm via decomposition into real components

To compute the QR decomposition of a double matrix \([A, B]\), one can break it down into the problem of finding:

- \(L\) and \(U\) such that \(BA = LU\)
- \(C\) such that \(A = CU\) and \(B = LC^{-1}\).
- Explicitly finding \(C^{-1}\).

The output would then be the two matrices \(R = [U, L]\) and \(Q = [C, C^{-1}]\).

This can be done by carrying out the following steps:

1. Compute \(BA\). This costs time \(n^3\). ¹
2. Find the LU decomposition of \(BA\): \(LU = BA\). This costs time \(2n^3/3\).

¹We measure algorithm complexity by counting flops, which are the number of multiplications, additions, subtractions, divisions and square roots.
3. Find $C$ via the equation $A = CU$ using forward substitution. This costs time $n^3/2$.

4. Do the same to find $C^{-1}$ using the equation $B = LC^{-1}$. This costs time $n^3/2$.

The overall running time is thus $8n^3/3$. This is the same as that of the Gram-Schmidt process and Householder method.

It bears pointing out that finding the double matrix $Q = [C, C^{-1}]$ is optional in the above algorithm.

### 4.3.2 Gram-Schmidt

Here, we show how to implement the Gram-Schmidt process for double matrices.

The running time of this algorithm is simply double that of the Gram-Schmidt process on real matrices. That is, it is $8n^3/3$.

See figure 2 for an implementation.

It bears pointing out that finding the matrix $R = [U, L]$ is optional in the above algorithm.

### 4.3.3 Householder method

The QR decomposition can also be computed using Householder reflections, suitably generalised to double vectors. The running time is twice that of the usual algorithm over real matrices, and is therefore equal to $8n^3/3$.

See figure 3 for an implementation.

### 5 The role of involutions in the double-complex numbers

The automorphism group of the double-complex algebra (understood as an algebra over the real numbers) has eight elements. Six of those have order 2, and can therefore be called involutions. Involutions are central to the study of double-complex matrices because they can be used as a substitute or even generalisation of complex conjugation. Linear algebra essentially reduces to just five operations: Addition, subtraction, multiplication, division, and conjugation.

We now observe that four of those involutions result in a trivial theory. They are:

- $(w, z) \mapsto (w, z)$
- $(w, z) \mapsto (\overline{w}, z)$
- $(w, z) \mapsto (w, \overline{z})$
- $(w, z) \mapsto (\overline{w}, \overline{z})$
To illustrate why they result in a trivial theory, we pick one of them, and hope that the reader understands that a similar issue is present in the other three. By \((w, z)\), we mean \(w \frac{1 + j}{2} + z \frac{1 - j}{2}\) where \(w\) and \(z\) are complex numbers.

For the sake of argument, we will define \((w, z)^*\) to mean \((\overline{w}, \overline{z})\). We can define \((A, B)\) to mean \((1, 0)A + (0, 1)B\) where \(A\) and \(B\) are arbitrary complex matrices. What we observe is that if \(f(M)\) is some matrix operation defined on complex matrices \(M\), we can generalise its definition to double-complex matrices by interpreting any complex conjugations in its definition to mean the operation \((w, z)^* = (\overline{w}, \overline{z})\). We will then immediately have that \(f((A, B)) = (f(A), f(B))\).

For instance, if \(f\) is the pseudoinverse of a matrix (its verbatim generalisation that uses this particular involution) then we have that this operation always distributes over the components \(A\) and \(B\). The same thing is true if \(f\) is the SVD or the LU decomposition, or we would argue *anything*. There is therefore nothing to study.

We now give the only two involutions which are of interest. They are:

- \((w, z) \mapsto (z, w)\)
- \((w, z) \mapsto (\overline{w}, \overline{z})\)

We claim that these result in an identical theory to each other. To understand why, we endeavour to define an operation \((A, B) \mapsto [A, B]\) on pairs of complex matrices, mapping to the double-complex matrices bijectively, that satisfies:

\[
\begin{align*}
+[C, D] &= [A + C, B + D] \\
[A, B] \times [C, D] &= [AC, DB] \\
[A, B]^* &= [B, A]
\end{align*}
\]

The third identity involves an involution, so our definition of \([A, B]\) will change along with the involution. If the involution is defined as \((w, z) \mapsto (z, w)\), then we can define \([A, B]\) to mean \(A \frac{1 + j}{2} + B^T \frac{1 - j}{2}\) as we have chosen to do in this paper. This then causes the above three identities to all be satisfied. Likewise, if we define our involution to be \((w, z) \mapsto (\overline{w}, \overline{z})\), then we can define \([A, B]\) to mean \(A \frac{1 + j}{2} + B^T \frac{1 - j}{2}\). Once again, the above three identities get satisfied.

So what’s the difference between these two involutions? It is about which double-complex matrices \([A, B]\) are understood to be the complex ones. One approach makes it so that the double-complex matrices of the form \([A, A^T]\) are taken to be the complex ones. The other approach makes it so that double-complex matrices of the form \([A, A^*]\) are taken to be the complex ones. Since in this paper, we treat double-complex matrices as pairs of complex matrices, we don’t ultimately care which ones are taken to be complex. By accident, we made it so that the complex matrices are those of the form \([A, A^T]\).

Observe that given either definition of \([A, B]\), the claim that a double-complex matrix is Hermitian precisely when it satisfies \(A = B\) is still true. All the other
characterisations of the different families of matrices given in section 2.2 stay the same.

The convention that matrices of the form \([A, A^T]\) are the complex one (which is the one we didn’t pick) is arguably slightly better. It results in the notions of unitary or Hermitian double-complex matrices being direct generalisations of their complex counterparts. For that reason, in future papers, this convention might be a slightly better one to adopt.

Given that it turns out that our notion of Hermitian matrix is not a direct generalisation of the one over complex matrices, what does it generalise? It turns out that it includes all the complex symmetric matrices, which are those complex matrices which satisfy \(A = A^T\). Our notion of unitary matrix generalises the set of complex orthogonal matrices, which are those that satisfy \(A^T A = A A^T = I\). This all follows from section 2.2. Perhaps this suggests that complex orthogonality and unitarity are on equal footings with each other.

## 6 Jordan SVD

We will begin by doing what we did in section 3, but for the polar decomposition. The polar decomposition of a square complex matrix \(M\) is a decomposition of the form \(M = UP\) where \(U\) is unitary and \(P\) is Hermitian.

Generalising the polar decomposition to the double-complex matrices gives us:

\[ [A, B] = [P, P^{-1}] [C, C] \]

which expands into:

\[ A = PC \]
\[ B = CP^{-1} \]

This implies that:

\[ BA = C^2 \]
\[ AB = P C^2 P^{-1} \]

We see that the double-complex polar decomposition of \([A, B]\) gives us a complex matrix \(\sqrt{BA}\) and a matrix \(P\) such that \(P \sqrt{BAP^{-1}} = \sqrt{AB}\).

In this section, we will show that such a decomposition of \([A, B]\) exists whenever \(A\) and \(B\) are non-singular. The problem of determining precisely when a polar decomposition exists (in the singular case) remains an open problem.

Recall that over the complex numbers, SVD is equivalent to polar decomposition. Given the derivation of double-complex SVD in section 3, it follows that there are non-singular double-complex matrices which don’t have SVDs. This follows from the fact that we need \(AB\) and \(BA\) to be diagonalisable. Due to this, the polar decomposition is not equivalent to SVD over double-complex matrices. We restore the equivalence of polar decomposition and SVD by defining a more sophisticated generalisation of the SVD to double-complex matrices called the
Jordan SVD, which borrows features from the Jordan Normal Form. We proceed to show that it has uses in non-Euclidean geometry (section 7), the computation of Moore-Penrose pseudoinverses of double-complex matrices (section 8), and in generalising some well-known or obscure matrix decompositions (section 6.1).

**Definition 6.1** (Jordan SVD). The Jordan SVD of a double-complex matrix $M$ is a factorisation of the form $M = U[J, J]V^*$ where $J$ is a complex Jordan matrix, and $U$ and $V$ are unitary double-complex matrices.

**Definition 6.2** (Polar decomposition). The polar decomposition of a double-complex matrix $M$ is a factorisation of the form $M = UP$ where $U$ is unitary and $P$ is Hermitian.

**Theorem 6.1.** A double-complex matrix $M$ has a Jordan SVD if and only if it has a polar decomposition.

**Proof.** Assume that $M$ has a Jordan SVD. Write it as $W[J, J]V^*$. Let $U = WV^*$ and $P = V[J, J]V^*$. Observe that $UP$ is a polar decomposition of $M$.

We now prove the converse. Assume that $M$ has a polar decomposition $UP$. Write $P$ as $[A, A]$. $A$ has a Jordan decomposition $QJQ^{-1}$. We thus have that $M = U[Q, Q^{-1}][J, J][Q, Q^{-1}]^*$, which is a Jordan SVD where the three factors are $U[Q, Q^{-1}]$, $[J, J]$ and $[Q, Q^{-1}]^*$.

Now we show that unlike the SVD, the Jordan SVD / polar decomposition always exists for non-singular matrices.

**Theorem 6.2** (Existence of Jordan SVD). If a double-complex matrix $M = [A, B]$ is invertible, then it has a Jordan SVD.

**Proof.** Since $M$ is invertible, so are $A$ and $B$. Since $A$ and $B$ are invertible, we have that $\sqrt{AB}$ exists. Consider the Jordan decomposition of $\sqrt{AB}$, which we will write as $PJ\tilde{P}^{-1}$. We have that $AB = PJ\tilde{P}^{-1}$. Let $Q = A^{-1}PJ$. We can therefore express $A$ in two ways: $A = P\tilde{Q}^{-1} = PJ\tilde{P}^{-1}B^{-1}$. Cancelling $P$ and $J$ gives $Q^{-1} = J\tilde{P}^{-1}B^{-1}$. Rearranging gives $B = QJP^{-1}$.

We can therefore express $A$ and $B$ as $P\tilde{Q}^{-1}$ and $QJP^{-1}$ respectively. Let $U = [P, P^{-1}]$ and $V = [Q, Q^{-1}]$. We have that $M = U[J, J]V^*$, as claimed.

By the half-plane, we mean the set of all those complex numbers which have real part greater than zero, together with all complex numbers whose real part equals zero but whose imaginary part is non-negative. Denote the half-plane as $H$. In symbols, we have that $H = \{z \in \mathbb{C} : \Re(z) > 0\} \cup \{ix : x \in \mathbb{R}, x \geq 0\}$.

We aim to show now that as long as a Jordan SVD of a matrix exists, then there exists a unique Jordan SVD in which all the eigenvalues of $J$ are on the half-plane (up to permutation of the Jordan blocks of $J$).
Lemma 6.1. If a matrix $M = [A, B]$ has a Jordan SVD $U[J, J]V^*$, then it has a Jordan SVD $U'[J', J'](V')^*$ where each element on the diagonal of $J'$ belongs to the half-plane.

Proof. Write $J = J_1 \oplus J_2 \oplus \ldots \oplus J_k$ where each $J_i$ denotes some Jordan block of $J$. Call the corresponding eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_k$. Assume for the sake of illustration that $\Re(\lambda_1) < 0$ and for the rest we have $\Re(\lambda_i) > 0$. Furthermore, assume that $J_1$ has dimensions $m \times m$ for some $m$. We observe that $(-I_m) \oplus I_{n-m}$ multiplied by $J$ changes $J_1$ to a non-negative block. We also observe that $U((-I_m) \oplus I_{n-m})$ is unitary. We thus have that $M$ is equal to $U'[J', J']V^*$ where $U' = U((-I_m) \oplus I_{n-m})$ and $J' = ((-I_m) \oplus I_{n-m})J$. We have that $J'$ is not necessarily a Jordan matrix, but is still similar to a Jordan matrix which has the same eigenvalues as it. We take the Jordan decomposition of $J'$ to get $J' = P J'' P^{-1}$. We thus have that $[J', J'] = [P J'' P^{-1}, P J'' P^{-1}] = [P, P^{-1}][J'', J''][P, P^{-1}]^*$. Putting it all together we have that $M$ can be expressed as $U''[J'', J''](V'')^*$ where $U'' = U'[P^{-1}, P]$, and $J''$ is defined as earlier, and $V' = V[P, P^{-1}]$. \hfill $\Box$

The following lemma characterises the possible Jordan Normal Forms of the square roots of any invertible complex matrix.

Lemma 6.2. Let $M$ be an invertible complex matrix. Consider its Jordan Normal Form $J = J_{k_1}(\lambda_1) \oplus J_{k_2}(\lambda_2) \oplus \cdots \oplus J_{k_n}(\lambda_n)$, where $n$ is the number of Jordan blocks in $J$, and $k_i$ denotes the size of the $i$th Jordan block. The Jordan Normal Form of any square root of $M$ is of the form $J_{\sigma(1)}(\pm \sqrt{\sigma(1)}) \oplus J_{\sigma(2)}(\pm \sqrt{\sigma(2)}) \oplus \cdots \oplus J_{\sigma(n)}(\pm \sqrt{\sigma(n)})$ where $\sigma$ is some permutation on $\{1, 2, \ldots, n\}$.

Proof. We have that $M = PJP^{-1}$ where $J$ is some Jordan matrix. We have that either $J = J_{k_1}(\lambda_1) \oplus J_{k_2}(\lambda_2) \oplus \cdots \oplus J_{k_n}(\lambda_n)$.

Consider some arbitrary square root of $M$, and denote it as $\sqrt{M}$. Consider the Jordan decomposition of $\sqrt{M}$, which we shall write as $Q K Q^{-1}$. We have that $K = J_{\ell_1}(\mu_1) \oplus J_{\ell_2}(\mu_2) \oplus \cdots \oplus J_{\ell_m}(\mu_m)$ where $m$ is the number of Jordan blocks in $K$, and $\ell_i$ denotes the size of the $i$th Jordan block of $K$. We have that $M = Q K^2 Q^{-1}$. Consider the Jordan decomposition of $K^2$, which we will write as $R K' R^{-1}$. We get that $M = (QR) K'(QR)^{-1}$. Since the Jordan Normal Form of $J_{\ell_i}(\mu_i)$ is $J_{\ell_i}(\mu_i^2)$, we have that $K' = J_{\ell_1}(\mu_1^2) \oplus J_{\ell_2}(\mu_2^2) \oplus \cdots \oplus J_{\ell_m}(\mu_m^2)$. Since both $K'$ and $J$ are the Jordan Normal Forms of $M$, we have that $m = n$, and there must exist a permutation $\sigma$ such that $\mu_i^2 = \lambda_{\sigma(i)}$ and $\ell_i = k_{\sigma(i)}$. The conclusion follows. \hfill $\Box$

We now prove that the Jordan SVD is unique as long as the eigenvalues of $J$ are all on the half-plane.

Theorem 6.3. An invertible double-complex matrix $M = [A, B]$ always has a Jordan SVD $U[J, J]V^*$ where the eigenvalues of $J$ are on the half-plane, and this is unique.
Proof. By theorem 6.2, $M$ has a Jordan SVD $U[J, J^*]V^*$.

Let $J = J_{k_1}(\lambda_1) \oplus J_{k_2}(\lambda_2) \oplus \cdots \oplus J_{k_n}(\lambda_n)$. We have that $AB = PJ^2P^{-1}$. Let $K$ be the Jordan Normal Form of $AB$. $K$ must also be the Jordan Normal Form of $J^2$, so we get that $K = J_{k_1}(\lambda_1^2) \oplus J_{k_2}(\lambda_2^2) \oplus \cdots \oplus J_{k_n}(\lambda_n^2)$. Consider any other Jordan SVD of $M$, which we will write as $[P', (P')^{-1}] [J', J']^*$. We have that $J'$ must be the Jordan Normal Form of some square root of $AB$, namely the one that’s equal to $P'J'(P')^{-1}$. It then follows from lemma 6.2 and the form of $K$ that $J' = J_{k_{\sigma(1)}}(\pm\lambda_{\sigma(1)}) \oplus J_{k_{\sigma(2)}}(\pm\lambda_{\sigma(2)}) \oplus \cdots \oplus J_{k_{\sigma(n)}}(\pm\lambda_{\sigma(n)})$ (where $\sigma$ is some permutation on $\{1, 2, \ldots, n\}$). For each $i$, our choice of plus or minus is fully determined by our desire for $\pm\lambda_{\sigma(i)}$ to land on the half-plane. In detail: If the real part of $\lambda_{\sigma(i)}$ is negative, then we pick the minus option to make the real part positive. If the real part of $\lambda_{\sigma(i)}$ is positive, then we pick the plus option to make the real part positive. The remaining case is where $\Re(\lambda_{\sigma(i)}) = 0$; in that case, pick plus or minus depending on whichever one lands in the half-plane.

From the above paragraph, we conclude that there exists at most one value of $J'$ (up to permutation of the Jordan blocks) in which all the eigenvalues are on the half-plane. By lemma 6.1, this value of $J'$ exists. We are done. \[\Box\]

We finish this section by giving the Sympy code for computing the Jordan SVD under the assumption that a double-complex matrix is invertible:

```python
def jordan_svd(M):
    A, B = first_part(M), second_part(M.T)
    P, J = Matrix.jordan_form((A * B) ** Rational(1, 2))
    Q = A ** -1 * P * J
    U = P * (1 + j)/2 + P.T ** -1 * (1 - j)/2
    V = Q * (1 + j)/2 + Q.T ** -1 * (1 - j)/2
    S = J * (1 + j)/2 + J.T * (1 - j)/2
    return U, S, V
```

### 6.1 As a generalisation of other matrix decompositions

The Jordan SVD generalises the Jordan decomposition. Let $PJP^{-1}$ be the Jordan decomposition of a complex matrix $A$. We then have that $[A, A] = U[J, J]V^*$ where $U = V = [P, P^{-1}]$. Note that $[A, A]$ is the general form of a double-complex Hermitian matrix.

Observe that the Jordan SVD generalises the ordinary SVD of a complex matrix. Let $UDV^*$ be the SVD of a complex matrix $A$. A Jordan SVD of $[A, \overline{A}^T]$ is then $[U, \overline{U}^T][D, D][V, \overline{V}^T]^*$. 

Observe that the double-complex polar decomposition generalises the so-called *Algebraic Polar Decomposition* ([10, 11]) of complex matrices. The Algebraic Polar Decomposition of a complex matrix $A$ is a factorisation of the form $A = UP$ where $U$ satisfies $UU^T = I$ and $P$ satisfies $P = PT$. Given $M = [A, AT^T]$, we have that a polar decomposition of $M$ is of the form $[U, U^T][P, P]$. 

13
6.2 Double-number Jordan SVD as opposed to double-complex Jordan SVD

Recall that there is an alternative to the Jordan Normal Form over real matrices where Jordan blocks of the form \[
\begin{pmatrix}
a + bi & 0 \\
0 & a - bi
\end{pmatrix}
\] are changed to blocks of the form \[
\begin{pmatrix}
a & -b \\
b & a
\end{pmatrix}
\]. This change results in a variant of the Jordan Normal Form that uses only real numbers and not complex numbers. Using this variant of the JNF, it’s possible to define a variant of the Jordan SVD for double matrices without using double-complex numbers. For the sake of completeness, we define this variant now but don’t use it later on.

**Definition 6.3** (Double-number Jordan SVD). _The double-number Jordan SVD of a double matrix \(M\) is a factorisation of the form \(M = U[J, J]V^*\) where \(J\) is a real Jordan matrix, and \(U\) and \(V\) are unitary double matrices._

7 Using the Jordan SVD to challenge a claim made in Yaglom’s _Complex Numbers in Geometry_

7.1 Introduction to _Complex Numbers in Geometry_

We work with the English translation of Yaglom’s _Complex Numbers in Geometry_, published in 1968. We will abbreviate this to CNG.

The topic of CNG is _linear fractional transformations_ over different number systems. A linear fractional transformation (from now on, abbreviated to LFT) is a function of the form \(z \mapsto \frac{az + b}{cz + d}\) where all variables \(a, b, c, d\) and \(z\) belong to some number system (formally, some commutative ring). Any such transformation can be represented using the \(2 \times 2\) matrix \[
\begin{pmatrix}
a & b \\
c & d
\end{pmatrix}
\]. If a non-singular \(2 \times 2\) matrix \(M\) represents an LFT, then any scalar multiple \(\lambda M\) of \(M\) represents the same LFT, as long as \(\lambda\) is not a zero-divisor.

CNG considers three number systems: The complex numbers, dual numbers and the double numbers. LFTs over complex numbers are well known. They are usually called _Moebius transformations_.

CNG provides some motivation for studying LFTs in the case of dual numbers and double numbers, a topic which some might otherwise find esoteric. To do this, it presents a geometric interpretation of what elements in the dual numbers, double numbers and complex numbers represent.

We will begin with the dual numbers, because it is the easiest and most fundamental case. Naively, one might think that a dual number represents a point on a plane; that is, the dual number \(x + \epsilon y\) represents the point on the Cartesian...
plane with Cartesian coordinate \((x, y)\). While this of course can work, CNG presents another interpretation of what a dual number represents. In CNG, a dual number is understood to represent a line on the plane. Additionally, this line is more than just a line, in that it is also oriented, meaning that the line is pointing in one of two opposite directions. This way, every line on the plane is represented by two dual numbers, depending on which of two orientations is given to the line.

A dual number of the form \(\tan(\theta/2)(1 + es)\) is understood to represent a line whose \(x\)-intercept is \(s\) and which makes an angle \(\theta\) with the \(x\) axis. Notice that if \(\pi\) gets added to \(\theta\), then the resulting dual number is not the same, even though the line is the same. This shows that the lines are oriented.

Strictly speaking, the set of dual numbers doesn’t quite suffice to express every line on the plane (because a line may not intersect the \(x\) axis). This is a problem which can be fixed by extending the set of dual numbers with the addition of some infinite dual numbers. Those additional numbers are of the form \(\frac{1}{x}\) where \(x\) can be an arbitrary real number. To understand this topic in complete generality, we suggest looking up homogeneous coordinates and projective lines over rings. Having made this change, as CNG does, we can now represent every line on the plane. Using homogeneous coordinates, a line which makes an angle \(\theta\) with the \(x\) axis, and which has a perpendicular distance of \(R\) from the origin, can be represented by the point on the dual-number projective line with homogeneous coordinate \([\sin(\frac{\theta + iR}{2}) : \cos(\frac{\theta + iR}{2})]\).

The elements \(a, b, c, d\) in \(z \mapsto az + bcz + d\) are all ordinary dual numbers, while \(z\) can also take any value of the form \(\frac{1}{x}\).

CNG shows that the LFTs over the dual numbers contain all Euclidean isometries. What this means is that all translations, rotations and reflections can be expressed as LFTs over the dual numbers. Strictly speaking, the reflections are always followed by orientation reversals, implying that while there is a subgroup of dual-number LFTs that’s isomorphic to the Euclidean group, the geometric interpretation of this subgroup is somewhat inconsistent with how one might normally imagine the Euclidean group. This subgroup is represented by the dual-number equivalent of the unitary matrices.

The LFTs over the dual numbers contain some transformations which are not Euclidean isometries. Because of this, Yaglom is motivated to classify all possible LFTs over the dual numbers. In a paper by Gutin, this classification is interpreted as a matrix decomposition of dual number matrices. Understood geometrically, matrix decompositions endeavour to express arbitrary matrices as products of simpler matrices. This is equivalent to describing a geometric transformation as a sequence of simpler and sometimes more familiar transformations. Gutin shows that Yaglom’s decomposition is of the form \(USV^*\) where \(U\) and \(V\) are unitary matrices over the dual numbers. All of the exotic behaviours of the dual number LFTs are due to the matrix \(S\). Gutin shows that \(S\) need only be of the form:
• \[
\begin{bmatrix}
a & 0 \\
0 & b
\end{bmatrix}
\]
where \(a\) and \(b\) are arbitrary real numbers.

• \[
\begin{bmatrix}
a & -be \\
be & a
\end{bmatrix}
\]
where \(a\) and \(b\) are arbitrary real numbers.

A similarity to Singular Value Decomposition is now obvious. Gutin goes on to show that all matrices over the dual numbers can be expressed in a similar way to the above (but generalised appropriately to \(n \times n\) matrices for arbitrary \(n\), and to singular matrices as well). The above special case is only for \(2 \times 2\) matrices, and only for invertible matrices. This special case is enough for CNG.

Note: If \(S\) is of the form \[
\begin{bmatrix}
a & -be \\
be & a
\end{bmatrix}
\]
(which is the second case), then \(S\) represents a transformation called an axial dilation. Axial dilations commute with unitary matrices. As a result, we have that \(USV^* = UV^*S\). In other words, if \(S\) is an axial dilation then we can simplify the decomposition to \(US\) instead of \(USV^*\).

In the axial dilation case, the book states the decomposition in this simpler way.

A question arises: What happens if the role of the dual numbers above is changed to the complex numbers? In that case, the complex numbers represent oriented lines in the elliptic plane (the plane which elliptic geometry takes places over). This is in contrast to the dual numbers, which represent oriented lines in the Euclidean plane. The elliptic plane is essentially a sphere (but where antipodal points are identified), and the lines are thus great circles. An arbitrary great circle can be chosen to be the equator. The oriented great circle which intersects the equator at longitude \(s\), and makes an angle \(\theta\) with the equator at the point of intersection, can be represented by the complex number \(\tan(\theta/2)(\cos(s) + i\sin(s))\). In the case where \(\theta = \pi\) (where the line is the same as the equator, but oriented in the opposite direction to when \(\theta = 0\)) the oriented line is represented as \(\infty\). Similar to the case of the dual numbers, the unitary matrices act as isometries of the elliptic plane. The set of transformations of the elliptic plane expressible as complex-number LFTs can be decomposed using Singular Value Decomposition of complex matrices, in a similar way to how we decomposed transformations of the Euclidean plane (expressible as dual-number LFTs) using an analogue of Singular Value Decomposition for dual-number matrices.

It’s worth pointing out that CNG does not interpret complex LFTs as transformations of the elliptic plane. We won’t speculate on why the book omitted this. Instead, the book proposes some other interpretations of complex LFTs. In one such interpretation, the complex numbers are understood to represent oriented points (oriented either clockwise or anti-clockwise) in the hyperbolic plane (in the sense of hyperbolic geometry). We won’t say any more about this.

Before we describe the double case, we must mention the fact that there are three non-Euclidean metric geometries over the plane: Euclidean geometry, elliptic geometry and hyperbolic geometry. The significance of these geometries lies in the fact that they satisfy four of the five postulates in Euclid’s Elements (once they are formulated in a suitable manner) while the fifth postulate is
contradicted by two of the three geometries. We’ve related the dual numbers to Euclidean geometry and the complex numbers to elliptic geometry. It’s clear what comes next.

The double numbers represent oriented lines on the hyperbolic plane. Strictly speaking, one must projectively extend the double numbers to represent all such lines, but the preceding claim is mostly correct. The LFTs over the double numbers can be seen to contain a subgroup isomorphic to the group of all isometries of the hyperbolic plane.

7.2 A challenge to Yaglom’s classification of LFTs over the double numbers

Like in the dual number case, CNG endeavours to classify all possible LFTs over the double numbers. We suspect that this classification is not entirely correct.

Below, we provide the relevant quote from CNG, and do our best to faithfully interpret it:

Each axial circular transformation of the Lobachevskii plane represents a motion, or a motion together with an axial symmetry with respect to some cycle $S_1$ (axial inversion of the first, second, or third kind), or a motion together with an axial symmetry with respect to an equidistant curve and a reversion (that is, together with an axial inversion of the fourth kind).

In our copy of the book, this can be found at the end of Section III, just before the Appendix (page 194). By an axial circular transformation of the Lobachevskii plane, CNG means an LFT over the double numbers. This is the term that the book uses in place of double LFT. The term Lobachevskii plane is used in some books in place of hyperbolic plane.

A motion is represented precisely by a unitary matrix over the double numbers. CNG uses this to mean a hyperbolic isometry. We make the daring choice to interpret CNG’s classification as a decomposition of the form $USV^*$ where $U$ and $V$ are unitary matrices, and $S$ is some matrix that CNG confines to only a few sets of possibilities. This seems to be what CNG meant, because the resulting set of possibilities for $S$ is nearly exhaustive. Perhaps the choice of $U$ and $V$ could be restricted depending on $S$, but CNG’s classification has the best chance of being exhaustive if we allow $U$ and $V$ to be chosen arbitrarily. Here are the possibilities CNG allows for $S$:

1. By an axial inversion of the first kind, CNG means a matrix of the form $\begin{bmatrix} 0 & -k \\ 1 & 0 \end{bmatrix}$ where $k$ is an arbitrary real number.

2. By an axial inversion of the second kind, CNG means a matrix of the
Oddly enough, this is a single matrix, and not an infinite family.

3. By an axial inversion of the third kind, CNG means a matrix of the form
\[
\begin{bmatrix}
(1 - \alpha)j & 1 + \alpha \\
-(1 + \alpha) & (1 - \alpha)j
\end{bmatrix},
\]
where \(\alpha\) is an arbitrary non-negative real number.

4. By an axial inversion of the fourth kind, CNG means a matrix of the form
\[
\begin{bmatrix}
0 & kj \\
1 & 0
\end{bmatrix}.
\]

In this paper, we provide an alternative, and we would argue fairly similar, decomposition of double matrices called the Jordan SVD. We now proceed to use it.

If the above set of values for \(S\) is indeed exhaustive, then we can check that claim by using the Jordan SVD. Recall that the Jordan SVD of a double-complex matrix of the form \([A, B]\) is the decomposition \([A, B] = U[J, J]V^*\) where \(U\) and \(V\) are unitary matrices over the double-complex numbers and \(J\) is a complex Jordan matrix (the same as in the Jordan Normal Form of a complex matrix).

To check exhaustiveness, we only need to check that all possible values of \(J\) are exhausted, where \(J\) is a possible Jordan Normal Form of a real matrix. Here are the matrices \(J\) for the above four possible sets of values of \(S\):

1. \[
\begin{bmatrix}
k & 0 \\
0 & 1
\end{bmatrix}
\]
for some \(k \in \mathbb{R}\).

2. \[
\begin{bmatrix}
2 & 1 \\
0 & 2
\end{bmatrix}.
\]

3. \[
\begin{bmatrix}
\sqrt{-2i\alpha^2 + 4\alpha + 2i} & 0 \\
0 & \sqrt{2i\alpha^2 + 4\alpha - 2i}
\end{bmatrix}.
\]

4. \[
\begin{bmatrix}
1 & 0 \\
0 & ik
\end{bmatrix}.
\]

Recall that if a matrix \(M\) represents an LFT, then \(\lambda M\) represents the same LFT as long as \(\lambda\) is not a zero divisor. Taking this into account, we can generalise the above possibilities to:

1. \[
\begin{bmatrix}
a & 0 \\
0 & b
\end{bmatrix}
\]
where \(a\) and \(b\) are arbitrary real numbers. This is obtained by taking case 1, substituting \(k = a/b\) and multiplying by \(\lambda = b\).

2. \[
\begin{bmatrix}
2 & 1 \\
0 & 2
\end{bmatrix}.
\]
 Taking case 2 and scaling the matrix by some \(\lambda\) does not result in a Jordan matrix unless \(\lambda = 1\). This appears to be the downfall of CNG’s claim that this is an exhaustive classification.

3. \[
\begin{bmatrix}
z & 0 \\
0 & z^*
\end{bmatrix}.
\]
This can be obtained by taking case 3, substituting in the appropriate value of \(\alpha\) and multiplying by the appropriate \(\lambda\).

4. \[
\begin{bmatrix}
\frac{1}{\sqrt{2}} & 0 \\
0 & \frac{1}{\sqrt{2}} - i
\end{bmatrix},
\]
which is obtained by taking case 4, substituting \(k = 1\) and
multiplying by \( \frac{1-i}{\sqrt{2}} \).

The LFT of the form \( \frac{2z + (1+j)}{(1-j)z + 2} \) doesn't appear to be covered by the above classification. The corresponding matrix is \([J, J]\) where \( J = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \). This matrix isn’t covered by any of the four cases above. This suffices to show that the classification in CNG is incomplete.

One proposal for completing CNG’s classification is to replace case 2, which CNG calls an axial inversion of the second kind (of which there is only one), with the infinite family of matrices of the form \( \begin{bmatrix} k & 1+j \\ 1-j & k \end{bmatrix} \) where \( k \) is an arbitrary real number. The only objection to this might be that CNG’s classification is of a geometric nature, and it’s unclear what geometric meaning such matrices have. Further work might be needed to find a geometrically meaningful extension to case 2.

We must point out that we have slightly simplified matters above. This was done for the sake of clarity, but not in a way that we think undermines CNG’s claim. When CNG studies double LFTs, it allows the use of the conjugation operation: \( z \mapsto z^* \). Therefore the group that it studies consists of transformations of the form \( z \mapsto \frac{az+b}{cz+d} \) and \( z \mapsto \frac{az^*+b}{cz^*+d} \). The latter transformations don’t admit a matrix representation. Furthermore, the four different types of axial inversions (as described in the book) are all transformations of the latter type, and therefore don’t have matrix representations. This changes none of our conclusions. Why?

We interpret CNG’s claim to be that every “axial circular transformation of the Lobachevskii plane” \( T \) is of the form \( U \circ S \circ V \) where \( U \) is a motion, \( S \) is an axial inversion of one of the four types, and \( V \) is another motion. Let \( C \) denote conjugation. If \( T \) reverses orientations and \( T = U \circ S \circ V \), we have that \( T = U' \circ S' \circ V' \circ C \) where \( U' \) corresponds to a unitary matrix, \( S' \) corresponds to one of the four types of matrices we associated with axial inversions, and \( V' \) corresponds to another unitary matrix. Our conclusions are therefore unchanged.

8 Pseudoinverse, and its relation to Jordan SVD

It is natural to consider an analogue of the Moore-Penrose pseudoinverse over double-complex matrices. Over the real numbers and complex numbers, the pseudoinverse has a connection to the SVD: Namely, that \((US^*)^+ = VS^+U^*\), where \( M^+ \) means the pseudoinverse of a matrix \( M \). This connection is not as useful over double or double-complex matrices, because a pseudoinverse might exist while a (naive) SVD might not. The Jordan SVD remedies this, and re-establishes the connection between SVD and pseudoinverse.

In a twin paper to this one, by Gutin, entitled An analogue of the relationship between SVD and pseudoinverse over double-complex matrices, it is shown that
whenever a double-complex matrix has a pseudoinverse then it also has a Jordan SVD. This means that the Jordan SVD can, in principle, always be used to find the pseudoinverse of a double-complex matrix: Let the double-complex matrix $M$ have Jordan SVD $U[J, J]V^*$. Assuming that $J$ has no non-trivial nilpotent Jordan blocks, we have that $M^+ = V[J^+, J^+]U^*$.

In the same paper, it is also shown that a sufficient condition for $[A, B]$ to have a Jordan SVD is for $\text{rank}(A) = \text{rank}(B) = \text{rank}(AB) = \text{rank}(BA)$. This is also a necessary and sufficient condition for $[A, B]$ to have a pseudoinverse.

9 Extending the set of permutation matrices

In order for a decomposition to exist in general (or be computable in a stable manner) it is often necessary to introduce pivoting. In this section, we investigate a hypothesis on how to extend such decompositions to double matrices.

First we must ask what the analogue is of a permutation matrix over the double numbers. It is tempting to say it is of the form $[P, P^{-1}]$ where $P$ is a permutation matrix over the real numbers. The justification for this claim is that such a matrix is necessarily a real matrix (because $P^{-1} = P^T$). Now we argue that this is undesirable, and that a much nicer analogue is the set of pairs of permutation matrices $[P, Q]$. We will first assume, for the sake of argument, that a permutation matrix should be of the form $[P, P^{-1}]$:

Consider the LUP decomposition: Every complex matrix $M$ can be written in the form

$$M = LU P$$

where $L$ is a lower triangular matrix, $U$ is an upper triangular matrix, and $P$ is a permutation matrix. We would like to extend the above claim to double matrices as well. So we get:

$$[A, B] = [L_1, U_1][U_2, L_2][P, P^{-1}]$$

which therefore results in the claim that we can always solve

$$A = L_1U_2P, B = P^{-1}L_2U_1.$$ 

We now demonstrate that this is not the case: Consider $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ and $B = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. By the existence criteria for LU decomposition ([6]), the above factorisation of $A$ and $B$ is impossible.

To remedy this problem, we extend the set of permutation matrices to the double numbers in a different way. We define a double permutation matrix to be of the form $[P, Q]$ where $P$ and $Q$ are arbitrary permutation matrices. The analogue of LUP decomposition reduces to the trivial but correct claim that we can take the LUP decomposition separately of two real matrices.
10 Decompositions that feature permutation matrices

In this section, we study double analogues of matrix decompositions that feature permutation matrices. We observe that once we take components, we arrive at familiar decompositions of real matrices.

10.1 BKP decomposition

We use the name BKP decomposition (where BKP stands for Bunch-Kaufman-Parlett) for a decomposition first introduced in a paper by Bunch and Parlett ([12]), and then further studied in a paper by Bunch and Kaufman ([13]).

The BKP decomposition ([6]) is a decomposition of Hermitian matrices $M$ that takes the form

$$PMP^* = LDL^*$$

where $P$ is a permutation matrix, $L$ is a lower triangular matrix, and perhaps surprisingly, $D$ is a block-diagonal Hermitian matrix where the blocks are of size $1 \times 1$ or $2 \times 2$.

The analogue of the above for double matrices is the decomposition

$$[P, Q][A, A][Q, P] = [L, U][D, D][U, L].$$

We take components to get

$$PAQ = LDU$$

which is a variant of the LUP decomposition where the matrix $D$ is block-diagonal with $1 \times 1$ or $2 \times 2$ blocks.

10.2 RRQR decomposition

The RRQR decomposition (for rank-revealing QR decomposition, [6, 14]) is a decomposition of real matrices $M$ that takes the form

$$M\Pi = QR$$

where $\Pi$ is a permutation matrix, $Q$ is a unitary matrix, and $R$ is an upper triangular matrix.

Consider the double analogue of the above

$$[A, B][\Pi_1, \Pi_2] = [C, C^{-1}][U, L]$$

which breaks apart into

$$A\Pi_1 = CU, \Pi_2 B = LC^{-1}.$$
11 Difficulties automatically extending numerical algorithms to matrix decompositions featuring permutation matrices

We briefly note that most algorithms that compute decompositions featuring permutation matrices do not straightforwardly generalise to our decompositions over double matrices. This includes all algorithms based on variants of Gaussian elimination with pivoting. It appears that those algorithms make rigid assumptions about the set of permutation matrices. We have attempted to find a method for carrying out such a generalisation, but are not satisfied with our results.

12 Open problems

12.1 Jordan SVD

We finish this paper with two open problems concerning the Jordan SVD. The Jordan SVD can be defined without using double or double-complex numbers, allowing the broader linear algebra community to suggest solutions to these problems. We say that a pair of $n \times n$ complex matrices $A$ and $B$, written $[A, B]$, has a Jordan SVD if there exist invertible matrices $P$ and $Q$, and a Jordan matrix $J$, such that:

- $A = PJQ^{-1}$.
- $B = QJP^{-1}$.

The two open problems are:

1. Find a necessary and sufficient condition for the Jordan SVD to exist.

2. Prove that the Jordan matrix $J$ is unique in all cases, subject to the restriction that all the eigenvalues of $J$ belong to the half-plane. We defined the half-plane to consist of those complex numbers which either have positive real part, or which have zero real part but which have non-negative imaginary part.

In a companion paper ([15]), we have made progress on problem 1 by showing that a sufficient condition for the Jordan SVD to exist is that $\text{rank}(A) = \text{rank}(B) = \text{rank}(AB) = \text{rank}(BA)$. Still, there are matrix pairs $[A, B]$ which have Jordan SVDs but which don’t satisfy this condition. The paper [11] may be relevant, and taking inspiration from it one might conjecture that a necessary and sufficient condition is for $AB$ to be similar to $BA$.

In this paper, we made progress on problem 2 by showing that the Jordan SVD is unique whenever $A$ and $B$ are invertible.
12.2 Insight into numerical linear algebra over real and complex matrices?

The main insight of this paper is that some matrix decompositions which appear to be special cases of other matrix decompositions are actually equivalent to them. An example is SVD, which is equivalent to eigendecomposition. This equivalence is revealed only through the use of the double or double-complex numbers. This insight is made more interesting when we observe that this means that a numerical algorithm for computing the SVD can successfully find the eigendecomposition of a matrix simply by generalising the algorithm verbatim to double-complex numbers. In practice though, the resulting eigendecomposition algorithm might fail to converge for some matrices, while the original SVD algorithm might converge for every matrix. The fact that it sometimes works is still intriguing.

But can it ever be useful? The decompositions in section 3, which serve to demonstrate this phenomenon, have some drawbacks. For some matrices, they don’t exist, and when they do exist, they may not be computable in a numerically stable manner. While a matrix decomposition over the complex numbers may be stable, its generalisation to the double or double-complex numbers may not be stable (for instance, see the QR decomposition).

Sometimes, a matrix decomposition can be made stable by introducing an additional factor which is a permutation matrix. For instance, compare the LU decomposition $M = LU$ with the PLU decomposition $M = PLU$. It turns out that the naive generalisation of the PLU decomposition to double matrices doesn’t exist for all double matrices. We propose therefore to make $P$ an arbitrary pair of permutation matrices $[P, Q]$ (understood as a single double matrix). We find that this fixes many existential issues, and not just with the PLU decomposition (which is a fairly trivial example). For instance, this can be done with some stable variants of the LDL decomposition (like the Bunch-Kaufmann-Parlett decomposition) with the result being a stable variant of LU. Unfortunately, it also results in a situation where algorithms for computing matrix decompositions don’t generalise in the right way. Algorithms for computing matrix decompositions like the PLU assume that the set of permutation matrices is the usual one. In this paper, we’ve tried to ensure that the generalisation of an algorithm to double matrices would be (nearly) verbatim, but this may no longer be possible once permutation matrices get involved.

In studying the Jordan SVD, we also studied the polar decomposition (which it is equivalent to). From an existential viewpoint, the polar decomposition is fairly pleasant, because it exists for all invertible double-complex matrices. Can the polar decomposition for double-complex matrices be computed in a numerically stable manner? Also, we’re not sure how relevant the double-complex polar decomposition is for people working in practical applications, but we remain hopeful.

Returning to what’s been established, we still think the qualitative insight that
SVD and eigendecomposition are more tightly linked than previously thought could become useful eventually. The other examples in section 3 show tighter links between different decompositions than what one would expect.

Acknowledgments. I would like to thank Gregory Gutin for many useful comments and suggestions.

References

[1] G. Baley Price. *An Introduction to Multicomplex Spaces and Functions*. Chapman & Hall/CRC Pure and Applied Mathematics. Taylor & Francis, 1990.

[2] Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Ratnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, Matthew J. Curry, Andy R. Terrel, Štěpán Roučka, Ashutosh Saboo, Isuru Fernando, Sumith Kulal, Robert Cimrman, and Anthony Scopatz. Sympy: symbolic computing in python. *PeerJ Computer Science*, 3:e103, January 2017.

[3] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[4] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.2)*, 2020. https://www.sagemath.org.

[5] Nicholas J. Higham. Cholesky factorization. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(2):251–254, September 2009.

[6] Gene H. Golub and Charles F. van Loan. *Matrix Computations*. JHU Press, fourth edition, 2013.

[7] Gilbert Strang. *Introduction to Linear Algebra*. Wellesley-Cambridge Press, Wellesley, MA, fourth edition, 2009.

[8] K. Vince Fernando and Beresford N. Parlett. Implicit Cholesky algorithms for singular values and vectors of triangular matrices. *Numerical Linear Algebra with Applications*, 2(6):507–531, 1995.

[9] Heinz Rutishauser. Solution of eigenvalue problems with the lr-transformation. *National Bureau of Standards, Applied Mathematics Series*, 49:47–81, 1958.

[10] Dipa Choudhury and Roger A. Horn. A complex orthogonal-symmetric analog of the polar decomposition. *SIAM Journal on Algebraic Discrete Methods*, 8(2):219–225, 1987.
[11] Irving Kaplansky. Algebraic polar decomposition. *SIAM J. Matrix Anal. Appl.*, 11(2):213–217, March 1990.

[12] J. R. Bunch and B. N. Parlett. Direct methods for solving symmetric indefinite systems of linear equations. *SIAM Journal on Numerical Analysis*, 8(4):639–655, 1971.

[13] James R Bunch and Linda Kaufman. Some stable methods for calculating inertia and solving symmetric linear systems. *Mathematics of computation*, pages 163–179, 1977.

[14] Peter Businger and Gene H. Golub. Linear least squares solutions by householder transformations. *Numer. Math.*, 7(3):269–276, June 1965.

[15] Ran Gutin. An analogue of the relationship between svd and pseudoinverse over double-complex matrices, 2021.
Figure 1: LR algorithm for computing SVD over double-complex matrices.
(Note that Sympy’s inbuilt LDL decomposition only works for positive-definite
matrices, so we needed to make our own version from scratch).

    from sympy import *

    J = var('J')

def simplify_split(e):
    real_part = (e.subs(J, 1) + e.subs(J, -1))/2
    imag_part = (e.subs(J, 1) - e.subs(J, -1))/2
    return simplify(real_part + J*imag_part)

def LDL(A):
    """Implements LDL decomposition for some (most?) Hermitian matrices,
    including some indefinite ones."""
    n = A.rows
    D = zeros(n, n)
    L = eye(n)
    for i in range(n):
        for j in range(n):
            if i == j:
                D[j,j] = A[j,j] - sum(L[j,k] * L[j,k].subs(J,-J) * D[k,k]  
                        for k in range(0, j))
                D[j,j] = simplify_split(D[j,j])
            if i > j:
                L[i,j] = 1/D[j,j] * (  
                        A[i,j] - sum(L[i,k] * L[j,k].subs(J,-J) * D[k,k]  
                        for k in range(0, j)))
                L[i,j] = simplify_split(L[i,j])
    return L, D

def svd(A, iters=20):
    M = A.T.subs(J,-J) * A
    for _ in range(iters):
        L, D = LDL(M)
        M = D ** Rational(1,2) * L.T.subs(J,-J) * L * D ** Rational(1,2)
    return D ** Rational(1,2)
from sympy import *

J = var('J')

def simplify_split(e):
    real_part = (e.subs(J, 1) + e.subs(J, -1))/2
    imag_part = (e.subs(J, 1) - e.subs(J, -1))/2
    return simplify(real_part + J*imag_part)

def gram_schmidt_qr(m):
    u = zeros(m.rows, m.cols)
    for k in range(m.cols):
        u[:, k] = m[:, k] - sum((proj(u[:, j], m[:, k]) for j in range(k)),
                                   start=zeros(m.rows, 1))
    Q = simplify_split(u * split_dot(u, u) ** Rational(-1, 2))
    R = zeros(m.rows, m.cols)
    for i in range(m.cols):
        for j in range(m.cols):
            if i <= j:
                R[i, j] = split_dot(Q[:, i], m[:, j])
    R = simplify_split(R)
    return Q, R

def split_dot(u, v):
    return u.T.subs(J, -J) * v

def proj(u, a):
    return u * split_dot(u, a) * split_dot(u, u).inv()
Figure 3: Householder QR implementation for double matrices.

```python
def householder_reflection(x):
    """Computes a householder reflection from a split-complex vector."""
    for i in range(x.rows):
        if scabs(x[i]) != 0:
            break
        if i == x.rows - 1:
            return eye(x.rows)
        if i != 0:
            initial_rotation = eye(x.rows)
            initial_rotation[0,0] = initial_rotation[i,i] = 0
            initial_rotation[i,0] = -1
            initial_rotation[0,i] = 1
            return householder_reflection(initial_rotation * x) * initial_rotation
    alpha = simplify_split(-x[0] / vec_norm(Matrix([x[0]])) * vec_norm(x))
    e1 = simplify_split(Matrix([1] + [0] * (x.rows - 1)))
    u = simplify_split(x - e1 * alpha)
    v = simplify_split(u / vec_norm(u))
    return simplify_split(expand(eye(x.rows) - 2 * v * v.T.subs(J,-J)))

def householderQR(m):
    """Computes the Q matrix in the QR decomposition using Householder reflections"""
    if m.cols == 1:
        return householder_reflection(m)
    Q1 = simplify_split(householder_reflection(m[:,0]))
    Qrest = householderQR(simplify_split((Q1 * m)[1:,1:]))
    return simplify_split(Q1.T.subs(J,-J) * diag(eye(1),Qrest))
```

28