Decomposition of nD-rotations: Classification, properties and algorithm

Aurélie Richard, Laurent Fuchs, Eric Andres, Gaëlle Largeteau-Skapin

Decomposition of nD-rotations: classification, properties and algorithm

A. Richard[a,*], L. Fuchs[a], E. Andres[a], G. Largeteau-Skapin[a]

[a] *Laboratoire XLIM-SIC, Université de Poitiers, 86360 Chasseneuil, France*

Abstract

In this paper, the decomposition of nD-rotations is studied. Using this decomposition, nD-rotations are classified and properties are underlined. For each class, an algorithm previously presented by the authors to decompose nD-rotation into planar rotations is proposed and a generalized algorithm is presented. Moreover, an alternate algorithm based on the Schur decomposition is investigated. A comparison between these different algorithms is finally provided.

Keywords: nD-dimensional rotations, planar rotations, Cartan-Dieudonné theorem, Schur decomposition, classification of nD-rotation.

1. Introduction

Rotation characterization is very useful in many application domains such as in computer vision [1], in medicine [2, 3], in Procrustes analysis [4] or in global visibility computation [5] for example. In many cases, estimating 3D rotations is sufficient but higher dimensions are needed in some cases (color image analysis, global visibility computation using Plücker space for example).

rajouter une phrase sur le modele conforme

From a set of points and their images, our aim is to characterize the rotation parameters (determination of the angles and planes) in nD. Since, our framework is experimental, the method must be somewhat robust to noise. This work is supposed to be a basis for the description of nD discrete rotations by extension of hinge angles for 3D discrete rotations [6]. In this paper, we suppose without loss of generality that the rotations are centred on the origin of the frame and that the angles are in $[0; \pi]$.

Unfortunately, as explained in [7], in the available literature there are not many methods that determine the characteristics of a nD-rotation. Some methods can be used only in 3D [2, 3, 8]. Other methods [9, 10] estimate the rotor which describes the rotation. In [9], the rotation planes cannot be computed easily.

Moreover, in [10], the versor can be determined only with exact data. The constructive proof of the Cartan-Dieudonné theorem proposed in [11] could

*Corresponding author

provide an algorithm to characterize nD rotations by their decomposition into planar rotations. Unfortunately it has been shown that this solution is not practicable [7].

In [12], Fontijne et al. proposed a method to reconstruct versors using vector correspondences. This method is based on the construction of a reflection sequence which aligns the points with their correspondences one by one. As explained by its authors, this method is not well suited for noisy data [13]. In this case, retrieving the reflexion axis could be very difficult. Even if the found axis is in accordance with the data, due to the multiple necessary decisions, the obtained versor could be significantly different from the correct one.

Another method can be deduced from the Schur decomposition [14, 15] which factorizes a matrix A into two matrices (an orthogonal matrix U and a block-diagonal matrice T such that $A = UTU^t$). As far as we know, the Schur decomposition has not been used to decompose nD-rotations into planar rotations. Since few methods exist, the authors have proposed a new algorithm to determine nD rotations from n points and their images [7]. From the Cartan-Dieudonné theorem [16], we know that a nD rotation can be decomposed into an even number of planar rotations. In [7], some of the authors have proposed an algorithm that finds these planar rotations and the angle for each of them. This fully characterizes the rotation.

The paper is organized as follow. In Section 2, our method presented in [7] for determining rotation parameters is summarized and its limits are underlined. In Section 3, nD-rotations are classified and an improvement of our algorithm is proposed. Another method based on the Schur decomposition is investigated in the next section. Finally, a comparison between our algorithm, its improvements and the Schur decomposition is provided.

2. The nD rotation decomposition algorithm presented in [7]

In [7], the authors have proposed an algorithm to decompose the nD-rotations. It works in the Geometric Algebra framework [9, 17, 18, 19, 20]. However, there are particular classes of rotations that are not handled by the proposed algorithm: the isoclinic rotations and what we called the pseudo-isoclinic rotations. We will therefore extend this algorithm to integrate those particular cases. We briefly present the useful notions for the understanding; for more details the reader may refer to [7].

In the following, the n dimensional space E (usually $\mathbb{R}^n$) is considered with a basis (not necessarily an orthogonal one) denoted by $(\mathbf{x_i})_{i=1\ldots n}$. The space E must be equipped with a non-degenerated quadratic form and the associated bilinear form is denoted by $\mathcal{B}$. In standard cases, $\mathcal{B}$ corresponds to the usual scalar product. Thus, the space E is a metric space upon which isometries can be defined.

2.1. Some results about nD-rotations

Let R be a rotation in $SO(E)$, the rotation group of E. The fixed-point subspace of R is denoted by F. Its orthogonal subspace in E is denoted by P

or $F^{\perp}$. In other words,

$$F = \{\mathbf{x} \in E | R(\mathbf{x}) = \mathbf{x}\} \text{ and } F^{\perp} = \{\mathbf{x} \in E \mid \forall \mathbf{y} \in F,\ \mathcal{B}(\mathbf{x}, \mathbf{y}) = 0\}.$$

The respective images of $(\mathbf{x_i})_{i=1\ldots n}$ by R are denoted by $(\mathbf{y_i})_{i=1\ldots n}$. Since $\mathcal{B}$ is non-degenerate, the space E can be decomposed into $E = P \oplus F$.

In the following of this paper, the following notations are used:

* the orthogonal subspace to F is denoted by P
* the rotation planes are denoted by P_i
* $\mathcal{P}_i$ is used in the R-isogonal subspace decomposition provided in Section 3.3

Moreover, E can also be decomposed into another direct sum given by the following well-known proposition [16]:

Proposition 1. *Let f be a linear isometry of E. The vector space E is an orthogonal direct sum :*

$$E = V \oplus W \oplus P_1 \oplus \cdots \oplus P_r$$

where the subspaces V, W and P_i are stable under f, and $f\mid_V = Id_V$, $f\mid_W = -Id_W$ and every P_i for $i = 1, \ldots, r$ is a plane such that $f\mid_{P_i}$ is a rotation.

In term of matrices, Proposition 1 shows us that there exists an orthonormal basis of E such that the matrix of f can be written as

$$\mathcal{F} = \begin{pmatrix} I_p & & & & \\ & -I_q & & & \\ & & R_{\theta_1} & & \\ & & & \ddots & \\ & & & & R_{\theta_r} \end{pmatrix}$$

where I_p denotes the identity matrix of order p and R_{θ_i} $(i = 1, \ldots, r)$ are 2×2 planar rotation matrices. As we deal with rotations, W has an even dimension q. Thus, we can consider the submatrix $-I_q$ as the concatenation of $q/2$ rotations of angle π. So, we can set $V = F$, $W = 0$ and $P = P_1 \oplus \cdots \oplus P_r$.

2.2. First version of our algorithm [7]

The method we proposed in [7] consists of three steps. We begin by the determination of P, the orthogonal subspace to the rotation fixed-points set. From it, we can construct the rotation planes and finally we can retrieve the rotation angles.

Determination of P.

The invariant subspace F of a transformation R and its orthogonal P are in direct sum. Moreover, the null space (denoted by Ker) and the range (denoted by Im) of an application are in direct sum too. From the two equalities $E = P \oplus F$ and $E = Ker(R - Id) \oplus Im(R - Id)$ we deduce that P is generated by $(\mathbf{y_i} - \mathbf{x_i})_{i=1\ldots n}$. The complete proof is provided in [7].

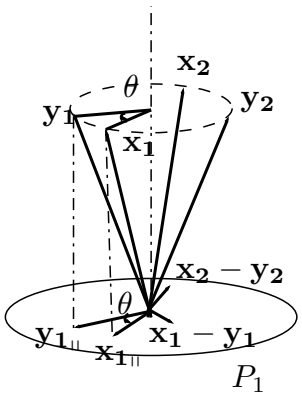

Figure 1: Graphical illustration of rotation plane and angle. $\mathbf{x_1}, \mathbf{y_1}$ (resp. $\mathbf{x_2}, \mathbf{y_2}$) represent a vector and its image by a rotation. The rotation plane P_1 is generated by $\mathbf{x_1} - \mathbf{y_1}$ and $\mathbf{x_2} - \mathbf{y_2}$. The angle between $\mathbf{x_1}$ and $\mathbf{y_1}$ is the same than those between $\mathbf{x_1}_{\shortparallel}$ and $\mathbf{y_1}_{\shortparallel}$ in P_1.

Determination of the rotation planes.

The goal here is to build a basis of P in order to be able to determine the rotation planes. Using Proposition 1, the subspace P can be decomposed as

$$P = P_1 \oplus \cdots \oplus P_r, i = 1, \ldots, r.$$

Then, if we compute an orthogonal basis of P, two elements of basis will determine P_i. Actually, let $(\mathbf{b_1}, \mathbf{b_2}, \cdots, \mathbf{b_{2r-1}}, \mathbf{b_{2r}})$ be an orthogonal basis of P. In other words,

$$P = \mathbf{b_1} \wedge \mathbf{b_2} \wedge \cdots \wedge \mathbf{b_{2r-1}} \wedge \mathbf{b_{2r}}.$$

The plane P_i is generated by $\mathbf{b_{2i-1}} \wedge \mathbf{b_{2i}}$ for $i = 1...r$.

The most obvious choice to compute it would be to use the Gram-Schmidt orthogonalization process over the vectors $\mathbf{y_i} - \mathbf{x_i}$. However, in case of noisy data, it is difficult to obtain a matrix with correct rank[1]. This is very important in case of real data. It is one of the main reasons why we worked on this method rather than simply using the Schur method as it can be seen in Section 4. We decided to use an alternative which is to use a SVD decomposition over the matrix formed by the vectors $\mathbf{y_i} - \mathbf{x_i}$. The approximation property of the SVD [14] ensures that we obtain a matrix with rank $2r$ that is the dimension of P even in case of slightly noisy data. Hence, following the usual presentation of the SVD decomposition, the first $2r$ columns form the desired basis.

Determination of rotation angles.

The last step of the algorithm finds the rotation angle of each planar rotation. For each rotation plane P_i, the rotation angle between $\mathbf{x}$ and $\mathbf{y}$ is the same than

[1]The $(\mathbf{y_i} - \mathbf{x_i})_{(i=1...n)}$ family generates P. So, the obtained matrix must be of rank $2r \leq n$ that is at most $2r = n - 1$ if n is odd.

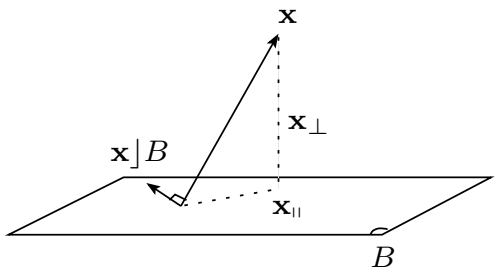

Figure 2: Contraction of a vector $\mathbf{x}$ by a plane B.

the angle between their projections $\mathbf{x}_{\shortparallel}$ and $\mathbf{y}_{\shortparallel}$ in P_i (cf Figure 1).

Then, for each P_i we have to determine these two vectors $\mathbf{x}_{\shortparallel}$ and $\mathbf{y}_{\shortparallel}$. To do so, we suppose that the vectors $\mathbf{x}$ and $\mathbf{y}$ are chosen such that their projections onto P_i are not null.

A collinear vector to $\mathbf{x}_{\shortparallel}$ is given by $(\mathbf{x} \rfloor P_i) \rfloor P_i$ where $(. \rfloor .)$ denotes the orthogonal contraction[2] (see [7]). A graphical illustration of the orthogonal contraction of a vector into a plane is provided in Figure 2.

Then, the rotation angle θ in P_i is the angle between $\mathbf{x}$ and $\mathbf{y}$ and is given by:

$$\cos(\theta) = \frac{\mathbf{x}_{\shortparallel} . \mathbf{y}_{\shortparallel}}{||\mathbf{x}_{\shortparallel}||||\mathbf{y}_{\shortparallel}||} \text{ and } \sin(\theta) = \frac{\mathbf{x}_{\shortparallel} \wedge \mathbf{y}_{\shortparallel}}{||\mathbf{x}_{\shortparallel}||||\mathbf{y}_{\shortparallel}||P_i}.$$

The computation is applied to all rotation planes in order to find all the rotation angles. Unfortunately, as shown in [7], this algorithm is not adapted in some particular cases.

2.3. Limits of this algorithm

In $4D$, a rotation can be decomposed into one or two planar rotations. These 4D-rotations are said to be respectively simple and double rotations. They leave fixed respectively a plane and one point (the rotation center). If the angles θ_1 and θ_2 of a double rotation have the same magnitude ($|\theta_1| = |\theta_2| \in [0, \pi]$) then the 4D rotation is said to be isoclinic. This is also called a Clifford displacement [21].

More generally, we have the following definition:

Definition 1 (Isoclinic rotations). In a n-dimensional space, with n even, a rotation is said isoclinic if all its $\frac{n}{2}$ angles are equal (up to the sign).

These particular rotations are only defined in an even dimension space. Indeed, in an odd dimension space, the unique isoclinic rotation is the identity transformation. In [7], the term pseudo-isoclinic was introduced as follows:

[2] If A is an a-blade and B a b-blade then $< A, B >_\perp$ is the subspace of dimension $(b - a)$ in B which is orthogonal to A

Definition 2 (Pseudo-isoclinic rotations). A nD-rotation with at least two equal angles is called pseudo-isoclinic.

This definition works both in an even and in an odd dimension space.

In [7], we showed that our algorithm is not adapted to these particular cases. Actually, if the data represents these two kinds of rotations, the rotation parameters are not well determined by the algorithm such as it is presented in [7]. In [7], numerical experiments have been conducted with dimension strictly smaller than 7. In this case, four groups of pseudo-isoclinic rotations can be distinguished:

(a) In 4D, the rotation has 2 equal angles.

(b) In 6D, the rotation has 3 equal angles.

$\Rightarrow$ the rotation is said *isoclinic.*

(c) In 5D, the rotation has 2 equal angles.

(d) In 6D, the rotation has exactly 2 equal angles.

An adaptation of the algorithm to these cases is proposed in Section 3. There exist previous works on isoclinic rotations [21]: the rotation planes are given by orthogonal planes generated by a vector and its image. Our algorithm uses this to handle cases (a,b). To the best of the knowledge of the authors case (c) has not yet been addressed. Nevertheless, some rotations in an odd dimension space (see Section 3.1.1) can be handled like isoclinic rotations in an even dimension space. An enhancement is also provided in case (d). It is based on the projections onto the range. These improvements are detailed case-by-case in the following section.

This list is only exhaustive for rotations in a space of dimension strictly smaller than 7. Actually, in higher dimension, other cases have to be considered. For example, a $9D$-rotation can be decomposed into four rotations having four angles $\theta_1, \theta_2, \theta_3, \theta_4$ with $\theta_1 = \theta_2 \neq \theta_3 = \theta_4$. This pseudo-isoclinic rotation is composed of a couple of isoclinic rotations. This underlines that the term pseudo-isoclinic, as it is defined in [7], gathered various cases. A classification of these pseudo-isoclinic rotations can thus be proposed. In Section 3.3 a general method is proposed.

3. Extension to the pseudo-isoclinic rotations

In this section, a classification of the nD-rotations as well as some new definitions are proposed. According to the best of the knowledge of the authors, it is the first time that such a classification is proposed. For each class of rotations, a decomposition algorithm is presented. Finally, a generalized decomposition algorithm which can be applied to all pseudo-isoclinic rotation is supplied.

3.1. Classification and properties of nD-rotations

In the following, R denotes a nD-rotation which is not the identity transform.

3.1.1. Classification of nD-rotations

The nD-rotations can be classified with respect to their decomposition. Let us recall or even introduce some new terms useful for the proposed classification.

The isoclinic notion, explained in the previous section, is only defined in an even dimension space. In an odd dimension space, a similar notion can be defined.

Definition 3 (Pure pseudo-isoclinic rotations). In an odd dimension space, the nD-rotations, with all its $\frac{n-1}{2}$ angles equal are called *pure pseudo-isoclinic* rotation.

The (pseudo) isocliny property is defined for angle equality. Since the data is noisy, it is necessary to decide when two angles are equal. This problem is a recurrent one when working with real numbers. In order to have a partition of the finite set of the angles we introduce the following relation.

Let θ_1 and θ_2 be two angles. Let β be the approximation parameter and let $\doteq_\beta$ be the following relation:

$$\theta_1 \doteq_\beta \theta_2 \Leftrightarrow \big|\ |\theta_1| - |\theta_2|\ \big| \leq \beta$$

For our experiments we arbitrarily fix the value of β to $\frac{\pi}{12}$. Moreover, since β is given and to avoid cumbersome indexes, we simply set $\doteq$ in order to denote $\doteq_\beta$ in the following.

Let us introduce the new term *pseudo-isogonal* to characterize the relation $\doteq$. This relation is reflexive and symmetric but not transitive so $\doteq$ is not an equivalence relation. The idea is to use this relation to form a partition of the angle set. The following definition is required:

Definition 4 (Pseudo-isogonal component). The *pseudo isogonal component* (PIC) of an angle θ denoted by $[\theta]$ is defined as the set of the angles $\alpha \leq \theta$ in relation with θ. In other words :

$$[\theta] = \{\alpha \in [0; \theta] \mid \alpha \doteq \theta\}.$$

The pseudo-isogonal component is not an empty set. Actually, from the reflexivity property, an angle belongs always to its pseudo-isogonal component.

Let $\mathcal{S}$ be the ordered set of the angles (in decreasing order).

Definition 5 (Pseudo-isogonal partition). A *pseudo-isogonal partition* (PIP) is a set of pseudo-isogonal components such that it forms a partition of the angle set; that means:

$$\forall \theta_i \in \mathcal{S}, \exists ![\theta_j] \in PIP \mid \theta_i \in [\theta_j].$$

Algorithm 1 Compute the pseudo-isogonal partition of a rotation

Input: $\mathcal{S}$ (ordered set of angles of the rotation R)
Output: $\mathcal{C}$ (set of the pseudo-isogonal component of R)

```
C = ∅
while S ≠ ∅ do
    Θ : first element of S
    Compute [Θ]
    C ← C ∪ [Θ]
    S ← S\[Θ]
end while
```

Remark 1. *The following property holds: $PIP \subset PIC$. There is no equality between both sets because of the non transitivity of the relation $\doteq$.*

The method to compute a pseudo-isogonal partition of a rotation R is summarized by the algorithm 1.

Definition 6 (Length of the pseudo-isogonal component). The *length of the pseudo-isogonal component* denoted by $l([\theta])$ is defined as its cardinal number.

This length is included between 1 and $\lfloor \frac{n}{2} \rfloor$. For a given rotation, the component number is as well included in the interval $[1; \lfloor \frac{n}{2} \rfloor]$.

Proposition 2. *The pseudo-isogonal component of an isoclinic or a pure pseudo-isoclinic rotation is unique and its length is maximal.*

Definition 7 (Isogonal space). An even dimensional subspace of E (of dimension at least 4) is said *R-isogonal* if the restriction of the rotation R to this subspace is an isoclinic rotation.

A nD-rotation R (for $n \geq 8$) can have many R-isogonal subspaces. For example, a $9D$-rotation with two pseudo-isogonal components of length 2 has two R-isogonal subspaces. An isoclinic rotation R (resp. pure pseudo-isoclinic rotation) has only one R-isogonal subspace. Its dimension is n (respectively $\frac{n-1}{2}$). The R-isogonal subspace is the maximal subspace in P (the orthogonal subspace to the invariant subspace).

Example 1. *Let us consider a 11D-rotation which can be decomposed into 5 rotations with angles $(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) = (130^o, 130^o, 72^o, 68^o, 60^o)$. With these new definitions, the following relations and equalities are true: $\theta_1 \mathcal{R} \theta_2$, $\theta_3 \mathcal{R} \theta_4$, $\theta_3 \mathcal{R} \theta_5$, $\theta_4 \mathcal{R} \theta_5$, $[\theta_1] = \{\theta_1, \theta_2\} = [\theta_2]$, $[\theta_3] = \{\theta_3, \theta_4, \theta_5\} = [\theta_4] = [\theta_5]$, $l([\theta_1]) = l([\theta_2]) = 2$, $l([\theta_3]) = l([\theta_4]) = l([\theta_5]) = 3$.*

Remark 2. *The partition based on the pseudo-isogonal components can be calculated from the input data of our algorithm. Actually for a given matrix A,*

its SVD decomposes A as USV^t where S is diagonal and its elements[3]*, called singular values, are the square root of the eigenvalues of AA^t. Here, the SVD is applied to the matrix $(A - Id)$. An elementary calculation shows that the eigenvalues of $(A-I)(A-I)^t$ with A be a rotation matrix are given by $(2-2\cos\theta_i)$. Each eigenvalue is of order two. Thereby, the number of angles is the half of the number of non-null singular value number (see Section 3.2.1).*

The nD-rotations can be classified according to the number of angles $\mathcal{N}_a$ and to the number of pseudo-isogonal components $\mathcal{N}_c$. This classification is given by the following theorem.

Theorem 1 (Classification of nD rotations). *Let R be a nD-rotation. If all the angles of R are different then R is a* ***general rotation*** *else R is a* ***pseudo-isoclinic rotation****. In this case, if all the $\lfloor \frac{n}{2} \rfloor$ angles of R are equal and n is even (resp. odd) then R is an* ***isoclinic rotation*** *(resp.* ***pure pseudo-isoclinic rotation****) in other cases R is an* ***ordinary pseudo-isoclinic rotation****.*

3.1.2. Properties of nD-rotations

In [21], Mebius defines two particular 4D-isoclinic rotations. The 4D-space is equipped with an arbitrary standard orientation. A 4D-isoclinic rotation has two angles of same magnitude (the angles have the same or an opposite sign). If the two angles have the same orientation then the rotation is said to be *left-isoclinic* else it is said to be *right-isoclinic*. In other words:

$$Rot_{4D} = Rot_{2D}(\theta_1)Rot_{2D}(\theta_2) \text{ with } \theta_1 = \theta_2 \text{ or } \theta_1 = -\theta_2.$$

Using these definition, we have the following theorem:

Theorem 2 (Isoclinic decomposition theorem [21]). *Any 4D rotation can be decomposed into two 4D-rotations: one left- and one right-isoclinic rotation. This decomposition can be processed in two ways, differing by a central reversion.*

review 1: since a $4D$ rotation stands in two orthogonal planes P_1 and P_2 we have:

In other words,

$$\begin{aligned} Rot_{4D} &= [Rot_{2D}(\theta_1)Rot_{2D}(\theta_2)]_{P_1}[Rot_{2D}(\theta_3)Rot_{2D}(\theta_4)]_{P_2} \\ &= \underbrace{[Rot_{2D}(\theta_1)Rot_{2D}(\theta_3)]}_{\text{Left-isoclinic}}\underbrace{[Rot_{2D}(\theta_2)Rot_{2D}(\theta_4)]}_{\text{Right-isoclinic}} \end{aligned}$$

with $\theta_1 = \theta_3$ and $\theta_2 = -\theta_4$.

The proof of this theorem is provided in [21].

Moreover, Mebius decomposes the 3D rotation into the same way:

Theorem 3. [21] *Any 3D-rotation of angle 2θ can be decomposed in a right- and a left-isoclinic rotation of angle θ.*

review 1: a $3D$-rotation can be seen as the composition of rotation in two planes P_1 and P_2

In other words,

[3] By convention, singular values are in decreasing order.

$$Rot_{3D}(2\theta) = [Rot_{2D}(\theta)Rot_{2D}(\theta)]_{P_1}[Rot_{2D}(\theta)Rot_{2D}(-\theta)]_{P_2} = R_{P_1}(2\theta).$$

The proof of this second theorem is also provided in [21].

The notion of left- and right-isoclinic rotations make no sense in higher dimensions. Actually, two angles can be said to be equal or opposite but relations between three angles can not be characterized in this way. In order to by-pass the problem, another notion needs to be introduced:

Definition 8 (Standard rotations). A nD-rotation with all its angles of same magnitude is said to be a *standard rotation* (n not necessarily even).

To decompose a rotation into standard rotations, one needs to order the rotation in a convenient way. In 3D, rotations do not commute but there is no isoclinic rotation. Whereas, in 4D, under some special condition on orthogonality, rotations can be permuted and thus placed in the convenient order. Let us recall the notion of completely orthogonal planes:

Definition 9 (Completely orthogonal planes). Two planes P_1 and P_2 are said to be completely orthogonal if each line in P_1 is orthogonal to any line in P_2.

The isoclinic decomposition theorem of Mebius can be extended to any nD-rotations in completely orthogonal planes:

Theorem 4 (Decomposition of the nD-rotations). *Any nD-rotation in planes completely orthogonal can be decomposed into standard rotations. The decomposition is not unique.*

Let us introduce the notion of simple, double and k-rotations used in the proof of Theorem 4.

Definition 10 (Simple and double rotations). A nD-rotation with only one (resp. two) angle is called *simple* (resp. *double*) rotation.

Definition 11 (k-rotation). A nD-rotation which can be decomposed into k planar rotations with k angles ($3 \leq k \leq \lfloor \frac{n}{2} \rfloor$) is called by k-rotations.

More precisely, Theorem 4 can be reformulated as : a simple rotation can be decomposed into two standard rotations, a k-rotation can be decomposed into k standard rotations. Let us prove Theorem 4.

Proof. In the proof, the notation $R_P(\alpha)$ denotes the rotation of angle α in the plane P. The proof is in two parts: for $k = 1$ (simple rotation) and for $k \geq 2$ (k-rotation).

* Simple rotations:

Let R be the nD-rotation of an angle θ in the rotation plane P. The rotation R is the composition of the rotation R_1 of angle θ in the plane P and the rotation R_2 of angle 0 in $\overline{P}$ with P and $\overline{P}$ completely orthogonal. The rotation R_1 can be written as:

$$R_P(\theta) = R_P\left(\frac{\theta+0}{2}\right) R_P\left(\frac{\theta-0}{2}\right).$$

In the same way, the rotation R_2 can be written as:

$$R_{\overline{P}}(0) = R_{\overline{P}}\left(\frac{\theta+0}{2}\right) R_{\overline{P}}\left(\frac{0-\theta}{2}\right).$$

With these decompositions, the rotation R can be written:

$$R = R_P(\theta)R_{\overline{P}}(0) = R_P\left(\frac{\theta+0}{2}\right) R_P\left(\frac{\theta-0}{2}\right) R_{\overline{P}}\left(\frac{\theta+0}{2}\right) R_{\overline{P}}\left(\frac{0-\theta}{2}\right).$$

Since the rotation planes are completely orthogonal, the factors commute:

$$R = R_P\left(\frac{\theta+0}{2}\right) R_{\overline{P}}\left(\frac{\theta+0}{2}\right) R_P\left(\frac{\theta-0}{2}\right) R_{\overline{P}}\left(\frac{0-\theta}{2}\right).$$

The rotation R is the composition of two standard double rotations of angle of magnitude $|\theta|/2$ and $|-\theta|/2$.

* the k-rotations:

Let R be a k-rotation with k angles $\theta_1, \theta_2, ..., \theta_k$. The rotation planes are supposed to be completely orthogonal. In each plane, the rotation can be expressed as:

$R_{P_1}(\theta_1) = R_{P_1}\left(\frac{\theta_1+\theta_2}{2}\right) ... R_{P_1}\left(\frac{\theta_1+\theta_k}{2}\right) R_{P_1}\left(\theta_1 - \frac{(\theta_1+\theta_2)}{2} - ... - \frac{(\theta_1+\theta_k)}{2}\right)$

$R_{P_2}(\theta_1) = \Big[R_{P_2}\left(\frac{\theta_1+\theta_2}{2}\right) R_{P_2}\left(\frac{-(\theta_1+\theta_3)}{2}\right) ... R_{P_2}\left(\frac{-(\theta_1+\theta_k)}{2}\right)$

$R_{P_2}\left(-\theta_1 + \frac{(\theta_1+\theta_2)}{2} + ... + \frac{(\theta_1+\theta_k)}{2}\right)\Big]$

and for $j = 2, ..., k-1$

$R_{P_{j+1}}(\theta_{j+1}) = \Big[R_{P_{j+1}}\left(\frac{-(\theta_1+\theta_2)}{2}\right) ... R_{P_{j+1}}\left(\frac{-(\theta_1+\theta_{j-1})}{2}\right) R_{P_{j+1}}\left(\frac{\theta_1+\theta_j}{2}\right)$

$R_{P_{j+1}}\left(\frac{-(\theta_1+\theta_{j+1})}{2}\right) ... R_{P_{j+1}}\left(\frac{-(\theta_1+\theta_k)}{2}\right) R_{P_{j+1}}\left(-\theta_1 + \frac{(\theta_1+\theta_2)}{2} + ... + \frac{(\theta_1+\theta_k)}{2}\right)\Big]$

If we denote by $\Theta_1, \Theta_2, ..., \Theta_k$ the k angles of the standard double rotations then the angles are related by the following relations:

$$\begin{cases} \Theta_i = \left|\frac{\theta_1+\theta_i}{2}\right| & i = 2, ..., k. \\ \Theta_k = \left|\theta_1 - \sum_{i=1}^{k-1} \Theta_i\right| \end{cases}$$

The first $(k-1)^{\text{th}}$ standard rotations are of angle of magnitude $\left|\frac{\theta_1+\theta_j}{2}\right|$ for $j = 2, ..., k$. The last one is of angle of magnitude $\left|\theta_1 - \frac{(\theta_1+\theta_2)}{2} - ... - \frac{(\theta_1+\theta_k)}{2}\right|$. The common factor is θ_1 but we can choose another one so the decomposition is not unique.□

<table>
<tr><th colspan="2">Rotations</th><th>$\mathcal{N}_a$</th><th>$\mathcal{N}_c$</th><th>Parity of n</th><th colspan="3">Algo</th></tr>
<tr><td colspan="2">General</td><td>$\leq \lfloor \frac{n}{2} \rfloor$</td><td>$\mathcal{N}_a$ (of length 1)</td><td>E & O</td><td>Algo of [7]</td><td></td><td rowspan="4">Algo 5</td></tr>
<tr><td rowspan="3">Pseudo-isoclinic</td><td>Isoclinic</td><td>$\frac{n}{2}$</td><td>1 (of length $\frac{n}{2}$)</td><td>E</td><td>Algo 2</td><td rowspan="3">Algo 4</td></tr>
<tr><td>Pure pseudo-isoclinic</td><td>$\lfloor \frac{n}{2} \rfloor$</td><td>1 (of length $\lfloor \frac{n}{2} \rfloor$)</td><td>O</td><td>Algo 3</td></tr>
<tr><td>Ordinary pseudo-isoclinic</td><td>x</td><td>$y < x$</td><td>E & O</td><td>Algo from section 3.2.3</td></tr>
</table>

Table 1: The nD-rotation classification and the associated decomposition algorithms. The letter E (resp. O) denotes the even (resp. odd) dimension, $\mathcal{N}_a$, the number of angles and $\mathcal{N}_c$ the number of pseudo-isogonal components.

Table 1 sums up all the different nD-rotations according to these criteria. The algorithms which permit their decomposition are also provided. They are developed in the following sections.

3.2. Extension of the algorithm from [7]

In this section, for each pseudo-isoclinic rotations, a method to extend the algorithm provided in [7] is proposed. A generalized method available for all pseudo-isoclinic is then provided.

First, we extend our algorithm to the isoclinic rotations, the pure pseudo-isoclinic rotations and the ordinary pseudo-isoclinic rotations only in 6D with two equal angles. These cases are particular cases of the generalized method which deals with all the pseudo-isoclinic rotations provided in Section 3.3.

3.2.1. Isoclinic rotations

The first improvement of our algorithm presented in [7] is the extension of the isoclinic rotations.

This particular case can be identified from the SVD. If the rotation is isoclinic then the eigenvalue is unique and is of order n. Thus, isoclinic rotations can be easily detected (cf Remark 2) and treated separately. In case of noisy data, the eigenvalues are compared using an error margin ϵ ($|v_1 - v_2| \leq \epsilon \Rightarrow v_1 =_\epsilon v_2$).

In this case, all the half-lines from the origin point are rotated by the same angle and all planes generated by a vector and its image are invariant. In other words, a 4D-isoclinic rotation is completely characterized by any point of the hypersphere centered in 0 and its image which belongs to this hypersphere too [21]. In 4D, it is sufficient to find two couples $(\mathbf{x_i}, \mathbf{y_i})$ and $(\mathbf{x_j}, \mathbf{y_j})$ with $\mathbf{y_i} = R(\mathbf{x_i})$ and $\mathbf{y_j} = R(\mathbf{x_j})$ such that $P_i = \mathbf{x_i} \wedge \mathbf{y_i}$ and $P_j = \mathbf{x_j} \wedge \mathbf{y_j}$ are orthogonal. The planes P_i and P_j can be used as rotation planes. In a general way, two planes generated by two vectors and their images are not mutually orthogonal. Consequently, it is not judicious to randomly choose vectors among our input data. In order to correctly build those planes using only our input

data (n points and their images) we apply the following method (based on the Gram-Schmidt orthogonalization). The first plane P_1 is arbitrarily defined by $\mathbf{x_1} \wedge \mathbf{y_1}$. Then a couple $(\mathbf{x_i}, \mathbf{y_i})$ which does not belongs to P_1 can be chosen. The projection of $(\mathbf{x_i},\mathbf{y_i})$ on P_1 is given by $(\mathbf{x_{i_{|\!|}}}, \mathbf{y_{i_{|\!|}}})$. The vectors $\mathbf{x_{i\perp}} = \mathbf{x_i} - \mathbf{x_{i_{|\!|}}}$ and $\mathbf{y_{i\perp}} = \mathbf{y_i} - \mathbf{y_{i_{|\!|}}}$ can be computed. They define a plane P_2 generated by a vector and its image [4]. Moreover the plane P_2 is orthogonal to P_1 (by construction). So we can choose P_1 and P_2 as rotation planes.

In nD, $\frac{n}{2}$ planes are required. In the same way, P_1 and P_2 are built. The third plane is generated by $\mathbf{x_k} - proj_{P_1}(\mathbf{x_k}) - proj_{P_2}(\mathbf{x_k})$ and $\mathbf{y_k} - proj_{P_1}(\mathbf{y_k}) - proj_{P_2}(\mathbf{y_k})$ with $(\mathbf{x_k}, \mathbf{y_k})$ which belongs to neither P_1 nor P_2. In a general way, the $(i+1)^{\text{th}}$ plane is generated by $\mathbf{x_i} - \sum_{j=1}^{i-1} proj_{P_j}(\mathbf{x_i})$ and $\mathbf{y_i} - \sum_{j=1}^{i-1} proj_{P_j}(\mathbf{y_i})$ with $(\mathbf{x_i}, \mathbf{y_i})$ not belongs to $P_1, ..., P_i$. This method can be used in any even dimension space to characterize the isoclinic rotations.

Algorithm 2 Find P_i (rotation plane) for isoclinic rotation in all even dimension space

Input: $(\mathbf{x_i})_{i=1,...,n}$, $(\mathbf{y_i})_{i=1...n}$
Output: P_i
  $P_1 = \mathbf{x_1} \wedge \mathbf{y_1}$
  for $i = 2$ to $dim/2$ **do**
    Choose $(\mathbf{x_i}, \mathbf{y_i}) \notin P_1, ..., P_{i-1}$
    $\mathbf{x_{i_{|\!|}}} = \sum_{j=1}^{i-1} proj_{P_j}(\mathbf{x_i})$
    $\mathbf{y_{i_{|\!|}}} = \sum_{j=1}^{i-1} proj_{P_j}(\mathbf{y_i})$
    $P_i = (\mathbf{x_i} - \mathbf{x_{i_{|\!|}}}) \wedge (\mathbf{y_i} - \mathbf{y_{i_{|\!|}}})$
  end for

The algorithm of [7] can be extended to the isoclinic rotations. It is sufficient to detect these rotations from the SVD and then to apply Algorithm 2.

3.2.2. Pure pseudo-isoclinic rotations

A nD-rotation R (with n odd) is said to be a *pure isoclinic rotation* if its $\lfloor \frac{n}{2} \rfloor$ angles are equal. A pure pseudo-isoclinic rotation has an invariant subspace F of dimension 1 and the restriction of R to P (of even dimension $n-1$) is an isoclinic rotation. Therefore, the subspace P is R-isogonal.

Let the matrix A be the rotation matrix. Let the matrix U be the matrix which comes from the SVD of the matrix $(A - Id)$. The subspace P is generated by the $(n-1)$ first column vectors of the matrix U and the subspace F is generated by its last vector.

One way to determine the rotation planes is to project all the n vectors and their images onto the subspace P using orthogonal contractions. Then, Algorithm 2 is applied. Naturally, its input data are these projections.

[4] From [7]: $\mathbf{y_{i\perp}} + \mathbf{y_{i_{|\!|}}} = R(\mathbf{x_{i\perp}}) + R(\mathbf{x_{i_{|\!|}}})$ and $\mathbf{y_{i\perp}} = R(\mathbf{x_{i\perp}}) \in F, \mathbf{y_{i_{|\!|}}} = R(\mathbf{x_{i_{|\!|}}}) \in P$

This method can be extended to all the pure pseudo-isoclinic rotations in any dimension.

The rotation parameter determination for pure pseudo-isoclinic rotation in any dimension space is summarized in Algorithm 3.

Algorithm 3 Find P_i (rotation plane) for pure pseudo-isoclinic rotation in all odd dimension space

Input: $(\mathbf{x_i})_{i=1...n}$, $(\mathbf{y_i})_{i=1...n}$
Output: P_i
  $\mathbf{z_i} = \mathbf{y_i} - \mathbf{x_i}$
  $(S, U, V) = SVD[\mathbf{z_1}...\mathbf{z_n}]$
  $P = \bigwedge_{i=1}^{n-1} U(:, i)$
  for $i = 1$ to n **do**
    $\overline{\mathbf{x_i}} = proj_P(\mathbf{x_i})$
    $\overline{\mathbf{y_i}} = proj_P(\mathbf{y_i})$
  end for
  Choose $(\overline{\mathbf{x_k}}, \overline{\mathbf{y_k}}) \neq (0, 0)$
  $P_1 = \overline{\mathbf{x_k}} \wedge \overline{\mathbf{y_k}}$
  for $i = 2$ to $(dim - 1)/2$ **do**
    Choose $(\overline{\mathbf{x_i}}, \overline{\mathbf{y_i}}) \notin P_1...P_{i-1}$
    $\mathbf{x_{i_{||}}} = \sum_{j=1}^{i-1} proj_{P_j}(\overline{\mathbf{x_i}})$
    $\mathbf{y_{i_{||}}} = \sum_{j=1}^{i-1} proj_{P_j}(\overline{\mathbf{y_i}})$
    $P_i = (\overline{\mathbf{x_i}} - \mathbf{x_{i_{||}}}) \wedge (\overline{\mathbf{y_i}} - \mathbf{y_{i_{||}}})$
  end for

3.2.3. Ordinary pseudo-isoclinic rotations in 6D with exactly 2 equal angles

To better understand the following, we start to detail the 6D-rotation with three angles θ_1, θ_2 and θ_3 where exactly two are equal. In the following, the angles θ_1 and θ_2 are supposed equal.[5] The rotation R has got two pseudo-isogonal components. This is an example which can be generalized to any dimension and any pseudo-isogonal component number (see Section 3.3).

The space E can be decomposed as $E = P \oplus F$. The space F is reduced to 0. The space P can be decomposed as $P = P_1 \oplus P_2 \oplus P_3$. We suppose that the rotation angle θ_i belongs to P_i, for $i = 1, 2, 3$.

Let A be the rotation matrix. The SVD of the matrix $(A - Id)$ gives two distinct singular values: s_1 of order 4 and s_2 is of order 2. Let $(j, j+1)$ with j odd be the indexes of the matrix S such that $S(j, j) = S(j+1, j+1) = s_2$. The j^{th} and $(j+1)^{\text{th}}$ columns of the matrix U (denoted by $U(:, j)$ and $U(:, j+1)$ in the following) span P_3 and the four other columns span $P_1 \oplus P_2$.

In other words,

$$P_3 = U(:, j) \wedge U(:, j+1).$$

[5] In other words, $\theta_1 \doteq \theta_2$, $\theta_3 \neq \theta_2$, $\theta_3 \neq \theta_1$.

$$P_1 \oplus P_2 = \bigwedge_{k=1}^{6} U(:,k), \quad k \neq j, j+1. \tag{1}$$

The restriction of the rotation R to $P_1 \oplus P_2$ is an isoclinic rotation, that means the space $P_1 \oplus P_2$ is R-isogonal. In this space all planes generated by a vector and its image are invariant. Thus, the spaces P_1 and P_2 are generated by a vector and its images.

If the six vectors and their images are projected onto the space $P_1 \oplus P_2$, these projections can be used as input data (instead of the $\mathbf{x_i}$ and $\mathbf{y_i}$) of Algorithm 2.

The algorithm provided in [7] can be extended using these three previous methods as shown by Algorithm 4. Nevertheless, a generalized method can be now proposed which includes the previous three algorithms. This method works with all the pseudo-isoclinic rotations.

3.3. General method

The aim of this section is to provide an algorithm which can be applied to the three pseudo-isoclinic rotations previously stated in any dimension space. It is a generalization of the three algorithms presented in Sections 3.2.1, 3.2.2, 3.2.3. In particular, it is based on the identification of the R-isogonal subspaces of the nD-rotation R. Actually, any pseudo-isoclinic rotations have got at least one R-isogonal subspace.

Let R be a nD-rotation in E and let A be its matrix in the canonical basis of E. The space E can be decomposed into $E = P \oplus F$ where F is the invariant subspace and P its orthogonal. The space F is eventually reduced to 0.

Let us introduce some notations:

* Let $\theta_1, ..., \theta_k$ for $1 \leq k \leq \lfloor \frac{n}{2} \rfloor$ be the k rotation angles of R.
* Let j be the number of distinct pseudo-isogonal components of the partition.
* Let $\mathcal{C}_i^{\alpha_i}$ for $1 \leq i \leq j$ be the i-th component of the partition with length α_i.

  By convention, since singular values are in decreasing order, the components are in angle value decreasing order.

  Since, the pseudo-isogonal components form a partition of the angle set, we have $\sum_{i=1}^{j} \alpha_i = k$.

Consequently, the space P can be decomposed into j subspaces as

$$P = \mathcal{P}_1 \oplus ... \oplus \mathcal{P}_j$$

where each $\mathcal{P}_i$ is of dimension $2\alpha_i$ with $1 \leq \alpha_i \leq \lfloor \frac{n}{2} \rfloor$. This decomposition is called *R-isogonal subspace decomposition of the rotation.*[6]

[6] If for all i, $\alpha_i = 1$, then each $\mathcal{P}_i$ are planes and $P = P_1 \oplus ... \oplus P_j$

Obviously, if $\alpha_i \geq 2$, the subspace $\mathcal{P}_i$ is R-isogonal.

These subspaces can be calculated from the SVD of the matrix $A - Id$. Actually, the matrix A can be decomposed into USV^t. The matrix S is diagonal and its diagonal elements s_i for $i = 1, ..., n$ are in decreasing order. The components are in decreasing order too. Thus, the first $(2\alpha_1)^{\text{th}}$ singular values are equal, the next $(2\alpha_2)^{\text{th}}$ are equal too, and so on. Consequently the first $(2\alpha_1)^{\text{th}}$ columns of the matrix U span $\mathcal{P}_1$, the next $(2\alpha_2)^{\text{th}}$ span $\mathcal{P}_2$ and so on. In a general way, we have the following equality:

$$\mathcal{P}_k = \bigwedge_{i=2\alpha_{k-1}+1}^{2\alpha_k} U(:, i). \tag{2}$$

By convention, $\alpha_0 = 0$. The formula (2) extends the formula (1) given in the previous part. Using the formula (2), rotation planes are easy to retrieve. Actually, for each $\mathcal{P}_k$, two cases are possible:

* Either $2\alpha_k = 2$ then $\mathcal{P}_k$ is a plane. It is determined by the formula (2).
* Or $2\alpha_k > 2$ then $\mathcal{P}_k$ is not a plane but a direct sum of planes. All the n vectors and their images are projected on R-isogonal subspace $\mathcal{P}_k$. The planes are determined by application of Algorithm 2.

This can be summarized by Algorithm 5.

Algorithm 4 Find P_i (rotation plane) in the pseudo-isoclinic rotation case.

Input: $(\mathbf{x_i})_{i=1...n}$, $(\mathbf{y_i})_{i=1...n}$
Output: P_i

```
z_i = y_i − x_i
(S, U, V) = SVD[z_1 ... z_n]
Calculate the pseudo-isogonal partition composed with the C_i^{α_i} for 1 ≤ i ≤ j
(i-th component which length α_i of the partition, in conventional order).
P = P_1 ⊕ ... ⊕ P_j
P_k = ⋀_{i=2α_{k-1}+1}^{2α_k} U(:, i) with α_0 = 0
for all P_k do
  if 2α_k = 2 then
    P_k = ⋀_{i=2α_{k-1}+1}^{2α_k} U(:, i)
  else
    Calculate the projections into P_k of (x_i)_{i=1...n}, (y_i)_{i=1...n}
    P_k=Output of Algorithm 2 with these projections as input data
  end if
end for
```

4. An alternative: the Schur decomposition

It is well-known that every rotation matrix can be block-diagonalized in an orthogonal basis [16]. One way to block-diagonalize a matrix is to compute

Algorithm 5 General algorithm to find P_i (rotation plane) and θ_i (rotation angle). The lines 3-9 can be replaced by the algorithm 4.

Input: $(\mathbf{x_i})_{i=1...n}$, $(\mathbf{y_i})_{i=1...n}$
Output: P_i, θ_i

// Compute rotation planes
1: $\mathbf{z_i} = \mathbf{y_i} - \mathbf{x_i}$
2: $(S, U, V) = SVD[\mathbf{z_1} \dots \mathbf{z_n}]$
3: **match** case **with**
4: case [(n even) & (One singular value (contained in S) of order n)]
// Isoclinic rotations
5: $|P_i$= Output of Algorithm 2
6: case [(n odd)& (One non-null singular value (contained in S) of order $n-1$)]
//Pure pseudo-isoclinic rotations
7: $|P_i$= Output of Algorithm 3
8: case [at least one singular value of order at least 4]
//Ordinary pseudo-isoclinic rotations
9: $|P_i$=Output of the Algorithm of section 3.2.3
10: case: other cases
//General case
11: **{**$k = rank(U)$
12: **for all** i such that $1 \leq i \leq \frac{k}{2}$ **do**
13: $P_i = U(:, 2i-1) \wedge U(:, 2i)$
14: **end for}**
15: **end match**
// Compute rotation angles
16: **for all** P_i **do**
17: Find $\mathbf{x_{j''}} = \ll \mathbf{x_j}, P_i >_\perp, P_i >_\perp$ and $\mathbf{y_{j''}} = \ll \mathbf{y_j}, P_i >_\perp, P_i >_\perp$ non-null
18: Calculate $\cos(\theta_j) = \frac{\mathbf{x_{j''}}.\mathbf{y_{j''}}}{||\mathbf{x_{j''}}||||\mathbf{y_{j''}}||}$ and $\sin(\theta_j) = \frac{\mathbf{x_{j''}} \wedge \mathbf{y_{j''}}}{||\mathbf{x_{j''}}||||\mathbf{y_{j''}}||P_i}$.
19: **end for**

the Schur decomposition [14, 15]. This decomposition is very used for example for the resolution of Riccati equations [22], for the definition of the matrix sign fonction [23] and in control theory [24]. However, according to the best knowledge of the authors, the Schur decomposition has never been used to characterize rotations that means to find rotation angles and rotation planes. In the following, we explain how to do this.

The principle of the Schur decomposition is to factorize a given matrix A into $A = UTU^t$ where U is orthogonal and T is block-diagonal with blocks of size 1×1 or of size 2×2. In order to become block-diagonal, a rotation matrix A undergoes orthogonal transformations. These transformations are stored into the matrix U (which is a basis change matrix). The blocks 2×2 of the matrix T represent 2×2 planar rotation matrices and the blocks 1×1 represent the identity matrices of order 1. From T, the rotation angles are thus retrieved

and from the corresponding columns of U the rotation planes can be computed. This method is summarized by Algorithm 6.

Algorithm 6 Find P_i (rotation plane) and θ_i (rotation angle)

Input: $(\mathbf{x_i})_{i=1...n}$, $(\mathbf{y_i})_{i=1...n}$
Output: P_i, θ_i
  $A = [\mathbf{y_1} ... \mathbf{y_n}]$
  $(U, T) =$ Schur-decomposition(A)
  // U is orthogonal, T is block-diagonal
  for each 2×2-block in T **do**
    $(j, j+1) =$ indexes of columns which compose the block
    $\theta_i = acos(T_{jj})$
    $P_i = T(:, j) \wedge T(:, j+1)$
  end for

5. Comparison of the different algorithms

5.1. Numerical experimentations

Our algorithm, its improvement and the Schur decomposition have been implemented and compared in higher dimensions with the library developed in [20] using the OCaml language [25].

The following numerical experiments have been conducted. The rotation planes and angles are randomly chosen and the rotation matrices A_i are then generated. To simulate the noise, the obtained matrices are biased that means the second or the third decimal of each floating number value is randomly changed. The biased matrices are denoted by B_i.

Afterward, the different methods are applied to these matrices. All the methods are experimented with the same matrices. The estimated rotation planes and angles are thus estimated (resp. P_i and θ_i). In [7], in order to estimate the error of the method, we successively compute the rotor R corresponding to these parameters and the image of each basis vector $\mathbf{x_k}$ using $R\mathbf{x_k}R^{-1}$. The rotor R is given by $R = \cos(\theta_i) - P_i \sin(\theta_i)$. The matrix such that the k^{th} column is the vector $R\mathbf{x_k}R^{-1}$ is denoted by E_i and is compared with the initial matrix A_i. In [7], we have (arbitrarily) considered that the plane P_i is generated by the $(2i-1)^{\text{th}}$ and the $2i^{\text{th}}$ columns of the matrix U which results from the SVD. Since there exists $\frac{k!}{2^{\frac{k}{2}}\,\frac{k}{2}!}$ possibility of permutations of the vectors of U (where k is the number of columns of U), it is not obvious to retrieve the same matrix. Here, we evaluate the transformation rather than to calculate the images of basis vectors. To do so, the rotor R is calculated, then a vector $\mathbf{v}$ is randomly generated. Its image $\mathbf{v_{theo}}$ is calculated using the matrix of $\mathbf{y_i}$ and is compared to $\mathbf{v_{prat}} = R\mathbf{v}R^{-1}$. The error is thus equal to the norm of $(\mathbf{v_{theo}} - \mathbf{v_{prat}})$. As in [7], the tests have been conducted in 4D, 5D and 6D (not in 3D because there exists only one rotation plane so the change of evaluation do not modify the result) with 500 matrices in each case.

Dim.	Prec.	Algo [7] (**1**)	New eval (**2**)	Algo 2 (**3**)	Algo 6 (**4**)	Algo 5 (**5**)
4D	0.001	0.027	0.013	0.011	0.023	0.011
	0.01	0.24	0.12	0.11	0.20	0.11
5D	0.001	0.040	0.017	-	0.032	0.013
	0.01	0.32	0.14	-	0.24	0.13
6D	0.001	0.069	0.028	0.028	0.041	0.019
	0.01	0.49	0.20	0.19	0.29	0.16

Table 2: Result of the various algorithms: (**1**): algorithm presented in [7] without modification; (**2**) algorithm presented in [7] with the new evaluation; (**3**) adaptation to the isoclinic case with new evaluation; (**4**) the Schur decomposition; (**5**) adaptation to all the pseudo-isoclinic cases.

Dim.	*Prec.*	(**1**)*vs*(**2**)	(**2**)*vs*(**3**)	(**2**)*vs*(**4**)	(**2**)*vs*(**5**)	(**4**)*vs*(**5**)
4D	0.001	52%	15%	43%	15%	52%
	0.01	50%	8%	40%	8%	45%
5D	0.001	58%	–	47%	24%	59%
	0.01	56%	–	42%	7%	46%
6D	0.001	59%	0%	32%	32%	54%
	0.1	59%	5%	31%	20%	45%
Mean		**55.6%**	**7.0%**	**39.2%**	**17.7%**	**50.2%**

Table 3: Comparative study of the different algorithms: (**1**): algorithm presented in [7] without modification; (**2**) algorithm presented in [7] with the new evaluation; (**3**) adaptation to the isoclinic case with new evaluation; (**4**) the Schur decomposition, (**5**) adaptation to all the pseudo-isoclinic cases.

5.2. Result analysis

The results are given in Tables 2, 3, 4 and 5. As in [7], experiments have been conducted with a precision 10^{-2} and 10^{-3} for each dimension (4D, 5D and 6D).

Table 2 gives for each dimension and each precision the mean error for each algorithm. The quantities have no unit since it is a norm. The error is calculated for the algorithm presented in [7] with the previous (**1**) and the new evaluation (**2**), its extension to the isoclinic rotations only (**3**)(no result in odd dimension), its extension to all the pseudo-isoclinic rotations (**5**) and for the algorithm deduced of the Schur decomposition (**4**).

The comparative study between all the algorithms is given by Table 3. Percentages indicate the improvements between the various algorithms. The mean improvement is given in the last row.

The error is smaller using the new evaluation than the previous one. Since parameter rotations are the same in both cases, this proves that *our evaluation used in [7] is not a good evaluation.* The rotation parameters are well estimated but due to their non-unicity, the error is very high. The evaluation method used in [7] must therefore be used only if the parameters are unique.

Dimension	Precision	Isoclinic rotations	Pseudo-isoclinic rotations
4D	0.001	31	-
	0.01	17	-
5D	0.001	-	35
	0.01	-	24
6D	0.001	4	97
	0.01	2	57

Table 4: Number of isoclinic and pseudo-isoclinic rotations detected by the comparison of the singular values.

Dim.	Prec.	All matrices			Iso. & Pseudo-iso. cases		
		Before	After	Comp	Before	After	Comp
4D	0.001	0.013	0.011	(**15%**)	0.054	0.018	(**67%**)
	0.01	0.12	0.11	(**8%**)	0.37	0.10	(**73%**)
5D	0.001	0.017	0.013	(**24%**)	0.085	0.026	(**69%**)
	0.01	0.14	0.13	(**7%**)	0.40	0.09	(**78%**)
6D	0.001	0.028	0.019	(**32%**)	0.078	0.032	(**59%**)
	0.01	0.20	0.16	(**20%**)	0.35	0.12	(**66%**)

Table 5: Mean of the error before (algorithm presented in [7] using the new evaluation) and after improvement (in isoclinic and pseudo-isoclinic case), using algorithm 5 for all the matrices and only those corresponding to an isoclinic or pseudo-isoclinic rotations.

The algorithm extended to the isoclinic case have an edge (7%). This can be explained by the number of isoclinic rotations. These numbers are given in the table 4. *In this case, the mean over all the 500 matrices is not representative* when only 2 matrices are concerned as for example in 6D. A better comparison in thus provided by the table 5 where the mean over the 500 matrices and only matrices which correspond to isoclinic rotations and pseudo-isoclinic rotations is calculated. By considering only these matrices, the mean is improved by almost 69%. In a general way, *our generalized algorithm is very efficient when data represent isoclinic or pseudo-isoclinic rotations. The improvement between the initial version proposed in [7] and the extended version is very significant.* The improvement is almost 63%. Moreover, *our new algorithm is better than those deduced of the Schur decomposition* (+50%).

5.3. Numerical example in 3D

Let us consider the space $E = \mathbb{R}^3$ equipped with the canonical basis $(\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3})$.

We consider the composition of a rotation of angle $\pi/4$ around the X-axis and a rotation of angle $\pi/6$ around the Y-axis. The vectors $\mathbf{y_i}$ are thus given by: $\mathbf{y_1} = (\sqrt{3}/2, 0, -1/2), \mathbf{y_2} = (\sqrt{2}/4, \sqrt{2}/2, \sqrt{6}/4)$ and $\mathbf{y_3} = (\sqrt{2}/4, -\sqrt{2}/2, \sqrt{6}/4)$.

From our algorithm, it is easy to show that the rotation plane (which is unique in 3D) is represented by the 2-blade $P = \mathbf{x'_{j1}} \wedge \mathbf{x'_{j2}}$ where $\mathbf{x'_{j1}} = (\sqrt{3}/2 - 1, 0, -1/2)$ and $\mathbf{x'_{j2}} = (\sqrt{2}/8, \sqrt{2}/2 - 1, -(4\sqrt{6} - 7\sqrt{2})/(8\sqrt{3} - 16))$. The angle

rotation is given by the angle between $\mathbf{x_{1_{||}}}$ and $\mathbf{y_{1_{||}}}$, the projection of $\mathbf{x_1}$ and $\mathbf{y_1}$ in P. For more numerical examples in particular in 4D, the reader can refer to [7].

6. Conclusion

In [7], we have proposed an algorithm that computes the decomposition into planar rotations of an nD rotation. The input data are n points and their images. Using the geometric algebra framework this algorithm is easy to implement.

If the data do not represent a pseudo-isoclinic rotation, the calculated planes and angles are very close to the exact ones even if the input data are biased. However, the algorithm was not adapted to the pseudo-isoclinic rotations.

In this paper, we have proposed an improvement which takes into account this particular case of rotations. The evaluation method is changed. The new method is more adapted since the parameters are not unique.

Our generalized algorithm is very efficient when the data represent isoclinic or pseudo-isoclinic rotations. The improvement between the initial version proposed in [7] and the extended version is significant. Moreover, our new algorithm is better than those deduced of the Schur decomposition.

A classification of nD-rotations is also presented. Moreover, some properties for decomposition of these rotations are also provided.

As an application of this work, we plan to use it in order to describe nD discrete rotations by extending the hinge angle notion developed for 3D rotations in [6].

References

[1] L. G. Shapiro and G C. Stockman. *Computer Vision*. Prentice Hall, 2001.

[2] PL. Cheng, AC. Nicol, and JP. Paul. Determination of axial rotation angles of limb segments - a new method. *Journal of Biomechanics*, Volume 33, 2000.

[3] PL. Cheng. Joint rotation between two attitudes in the spherical rotation coordinate system. *Journal of Biomechanics*, Volume 37, 2004.

[4] J. C. Gower and G. B. Dijksterhuis. *Procrustes problems*. Oxford University Press, 2004.

[5] S. Charneau, L. Aveneau, and L. Fuchs. Exact, robust and efficient full visibility computation in the Plücker space. *The Visual Computer journal*, Volume 23:Pages 773–782, September 2007.

[6] Y. Thibault, Y. Kenmochi, and A. Sugimoto. Computing upper and lower bounds of rotation angles from digital images. *Pattern Recognition*, 2009.

[7] A. Richard, L. Fuchs, and S. Charneau. An algorithm to decompose n-dimensional rotations into planar rotations. LNCS 6026, 2010.

[8] GA. Watson. Computing Helmert transformations. *Journal of Computational and Applied Mathematics*, Volume 197, 2006.

[9] C. Perwass and G Sommer. Numerical evaluation of versors with Clifford Algebra. *Proceeding of the AGACSE 2001*, 2001.

[10] L. Dorst. Determining a versor in nD geometric algebra from known transformation of n vectors. In *Proc. GraVisMa 2009 Conference*, 2009.

[11] G. Aragon-Gonzalez, JL. Aragon, MA. Rodriguez-Andrade, and L. Verde-Star. Reflections, rotations, and pythagorean numbers. *Advances in Applied Clifford Algebra*, Volume 19, 2009.

[12] D. Fontijne and L. Dorst. Reconstructing rotations and rigid body motions from points correspondences as a sequence of reflections. *AGACSE 2010*, 2010.

[13] A. Richard, G. Largeteau, M. Rodriguez, E. Andres, L. Fuchs, and JSD Ouattara. Properties and applications of the simplified generalized perpendicular bisector. 2011.

[14] G. Golub and C. Van Loan. *Matrix computations*. Johns Hopkins Studies in Mathematical Sciences (3rd edition), 1996.

[15] http://planetmath.org/encyclopedia/DecompositionOfOrthogonalOperators AsRotationsAndReflections.html. web site.

[16] M. Audin. *Geometry*. Springer, 2003.

[17] L. Dorst, D. Fontijne, and S. Mann. *Geometric algebra for computer science: an object oriented approach to geometry*. Morgan Kauffmann Publishers, 2007.

[18] D. Hildebrand, D. Fontijne, C. Perwass, and L. Dorst. *Geometric algebra and its application to computer graphics*. 25th annual conference of the European Association for Computer Graphics - Interacting with Virtual worlds, 2004.

[19] G. Sommer. *Geometric computing with Clifford Algebra*. Springer-Verlag, 2001.

[20] S. Charneau. *Étude et application des algèbres géométriques pour le calcul de la visibilité globale dans un espace projectif de dimension $n \geq 2$*. PhD thesis, Université de Poitiers, France, 2007.

[21] J.E. Mebius. *Applications of quaternions by dynamical simulation, computer graphics and biomechanics*. PhD thesis, Delft University of Technology, The Netherlands, 1994.

[22] A. Laub. A Schur method for solving algebraic Riccati equations. *Automatic Control, IEEE Transactions on*, Volume 24(Number 6):Pages 913–921, 2002.

[23] J-G. Sun. Perturbation analysis of the matrix sign function. *Linear algebra and its applications*, Volume 250:177–206, 1997.

[24] A. Bojanczyk, G. Golub, and P. Van Dooren. The periodic Schur decomposition. algorithms and applications. In *Proc. SPIE Conference*, volume 250, pages 31–42, 1992.

[25] http://www.ocaml.org. web site.