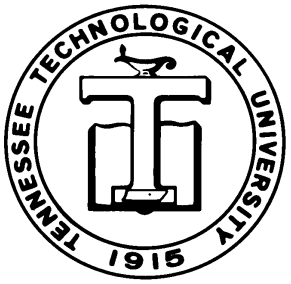
DEPARTMENT OF MATHEMATICS
TECHNICAL REPORT

HECKE ALGEBRAS, SVD, AND OTHER COMPUTATIONAL EXAMPLES WITH CLIFFORD

Rafal Ablamowicz

September 1999

No. 1999-9



TENNESSEE TECHNOLOGICAL UNIVERSITY
Cookeville, TN 38505

Hecke Algebras, SVD, and Other Computational Examples with CLIFFORD*

Rafał Abłamowicz
Department of Mathematics,
Tennessee Technological University
Cookeville, TN 38505
rablamowicz@tntech.edu

September 30

Abstract: CLIFFORD is a Maple package for computations in Clifford algebras $C\ell(B)$ of an arbitrary symbolic or numeric bilinear form B. In particular, B may have a non-trivial antisymmetric part. It is well known that the symmetric part g of B determines a unique (up to an isomorphism) Clifford structure on $C\ell(B)$ while the antisymmetric part of B changes the multilinear structure of $C\ell(B)$. As an example, we verify Helmstetter's formula which relates Clifford product in $C\ell(g)$ to the Clifford product in $C\ell(B)$. Experimentation with Clifford algebras $C\ell(B)$ of a general form B is highly desirable for physical reasons and can be easily done with CLIFFORD. One such application includes a derivation of a representation of Hecke algebras in ideals generated by q-Young operators. Any element (multivector) of $C\ell(B)$ is represented in Maple as a multivariate Clifford polynomial in the Grassmann basis monomials although other bases, such as the Clifford basis, may also be used. Using the well-known isomorphism between simple Clifford algebras $C\ell(Q)$ of a quadratic form Q and matrix algebras through a faithful spinor representation, one can translate standard matrix algebra problems into the Clifford algebra language. We show how the Singular Value Decomposition of a matrix can be performed in a Clifford algebra. Clifford algebras of a degenerate quadratic form provide a convenient tool with which to study groups of rigid motions in robotics. With the help from CLIFFORD we can actually describe all elements of **Pin**(3) and **Spin**(3). Rotations in R^3 can then be generated by unit quaternions realized as even elements in $C\ell^+_{0,3}$. Throughout this work all symbolic computations are performed with CLIFFORD and its extensions.

Keywords: Contraction, reversion, Hecke algebra, Young operator, spinor representation, exterior algebra, multilinear structure, quaternions, Singular Value Decomposition, rigid motions.

*Extended version of a talk presented at "ACACSE'99:Applied Clifford Algebra in Cybernetics, Robotics, Image Processing and Engineering", International Workshop as a Special Parallel Session of the 5th International Conference on Clifford Algebras and their Applications in Mathematical Physics, June 27-July 4, 1999, Ixtapa, Zihuatanejo, Mexico

Contents

1 Introduction

A first working version of a Maple package CLIFFORD was presented in Banff in 1995 [1]. From a modest program capable of symbolic computations in Clifford algebras of an arbitrary bilinear form, CLIFFORD has grown to include 96 main procedures, 21 new Maple types, close to 4,000 lines of code written in the Maple programming language, and an extensive on-line documentation. There is a number of special-purpose extensions available to CLIFFORD such as suppl and asvd used in this paper [6]. In fact, anyone who uses Maple can easily write additional procedures to tackle specific problems.

There are major advantages in using CLIFFORD on a Computer Algebra System. One is an ability to solve equations and find the most general elements in the Clifford algebra satisfying given conditions. This approach has been presented in Sections 3 and 5. In Section 3 Young operators in the Hecke algebra $H_{\mathsf{F}}(3, q)$ are eventually found by systematically solving three equations that define them. Computations in this section were first reported in [7] along with a physical motivation.

There, experimentation with CLIFFORD led to finding Young operators realized as idempotents in the Hecke algebra which in turn had been embedded into the even part of a Clifford algebra $C\ell(B)$ of a suitable bilinear form B. Such embedding was first given in [17] where the bilinear form B was found so that the defining relations on Hecke generators were satisfied. Furthermore, it was shown in [7] that Young operators corresponding to conjugate tableaux were related through the operation of reversion in the Clifford algebra, an idea that was first proposed in [17]. Here, we only show the mechanics of the search for such operators, Garnir elements, and bases in the representation spaces as they were performed with CLIFFORD.

In the same spirit, in Section 5, we describe a search for the elements in $\mathbf{Pin}(3)$ considered as a subgroup of the group of units in the Clifford algebra $C\ell_{0,3}$. Seven general types, not entirely exclusive, are eventually found through a systematic search and analysis. Then, the elements of $\mathbf{Spin}(3)$ are computed and related to unit quaternions. Rotations in coordinate planes and in a plane orthogonal to an arbitrary non-zero axis vector are described using quaternions realized as elements of $C\ell^+_{0,3}$. A symbolic formula describing the most general rotation is derived. Finally, using the ability of CLIFFORD to compute in Clifford algebras of a degenerate quadratic form, the semi-direct product $\mathbf{Spin}(3)$ o R^3 is shown to generate rigid motions on a suitable subspace of the Clifford algebra $C\ell_{0,3,1}$.

The second advantage of using symbolic program like CLIFFORD is its ability to compute with expressions containing totally undefined symbolic coefficients. It is possible, of course, like in Section 5 to impose additional conditions on these coefficients when needed (by defining aliases for roots of polynomial equations). In Section 2 we verify one of Helmstetter's formulas [19] that relates Clifford product in $C\ell(B)$, the Clifford algebra of an arbitrary bilinear form B, to the Clifford product in $C\ell(g)$ where $g = g^T$ is the symmetric part of B. A re-wording of the Helmstetter formula presented to the Author by Pertti Lounesto [23] proved to be suitable for symbolic verification with CLIFFORD. In fact, while this problem turned up to be a challenge for CLIFFORD in view of its complexity, it also has helped to fine-tune the program to make such computations feasible. We will only illustrate computations in dimension 3; however, computations in dimension up to 9 have been successfully completed.

The third major advantage of using CLIFFORD shows up in Section 4: here, we perform within the same workspace not only symbolic computations with a Clifford algebra, but also with a linear algebra package built into Maple. This way we can illustrate in two low-dimensional examples two parallel approaches to the Singular Value Decomposition (SVD) of a matrix: one through the matrix algebra, and one through the Clifford algebra. In Section 4.3 we comment on SVD based entirely on the Clifford algebra approach.

2 Verification of the Helmstetter formula

In his paper [19] Helmstetter studies canonical isomorphisms between Clifford algebras $C\ell(Q)$ and $C\ell(Q')$ of two quadratic forms Q and Q' defined on the same (real or complex) vector space V. The forms are related via the identity $Q'(\mathbf{x}) = Q(\mathbf{x}) + B(\mathbf{x}, \mathbf{x})$ for every $\mathbf{x} \in V$ and some bilinear form B on V (that is, B is an element in $(V \otimes V)^*$). Helmstetter constructs a *deformed* Clifford product $*$ on $C\ell(Q)$ by extending the Clifford product $\mathbf{xy}$ of two elements $\mathbf{x}$ and $\mathbf{y}$ in $V \hookrightarrow C\ell(Q)$

$$\mathbf{x} * \mathbf{y} = \mathbf{xy} + B(\mathbf{x}, \mathbf{y})$$

to all elements in $C\ell(Q)$. Together with the new product $*$, the Clifford algebra $C\ell(Q)$ becomes a deformed Clifford algebra $C\ell(Q,B)$. Given now two different bilinear forms B and B' on the quadratic space (V,Q) such that $B(\mathbf{x},\mathbf{x}) = B'(\mathbf{x},\mathbf{x})$ for every $\mathbf{x} \in V$, Helmstetter proves that there exists $F \in \bigwedge^2 V^*$ such that

$$B'(\mathbf{x},\mathbf{y}) - B(\mathbf{x},\mathbf{y}) =< F, \mathbf{x} \wedge \mathbf{y} >$$

and the mapping

$$\phi : C\ell(Q,B) \to C\ell(Q,B'), \qquad u \mapsto e^{\wedge F} \lrcorner u \tag{1}$$

gives an isomorphism from $C\ell(Q,B)$ to $C\ell(Q,B')$ which acts as an identity on V. In the above, $e^{\wedge F} \lrcorner u$ denotes the *left contraction* of u by the exterior exponential of F (see [9], [22]). A special case of (1) occurs when B is symmetric, that is, $B = g = g^T$ and $B' = g + A$ for some antisymmetric form A.

Thus, with a slight change of notation, let $B = g + A$, $g^T = g$, $A^T = -A$ and let us consider two Clifford algebras $C\ell(g)$ and $C\ell(B)$ on the same vector space V. We have therefore three contractions: $\mathbf{x} \underset{B}{\lrcorner} \mathbf{y} = B(\mathbf{x},\mathbf{y})$, $\mathbf{x} \underset{A}{\lrcorner} \mathbf{y} = A(\mathbf{x},\mathbf{y})$, and $\mathbf{x} \underset{g}{\lrcorner} \mathbf{y} = g(\mathbf{x},\mathbf{y})$.[1] Then, the B-dependent Clifford product $\underset{B}{uv}$ of any two elements u and v in $C\ell(B)$ can be written [23] solely in terms of the operations in $C\ell(g)$ as

$$\underset{B}{uv} = ((u \underset{g}{\llcorner} e^{\wedge F})(v \underset{g}{\llcorner} e^{\wedge F})) \underset{g}{\llcorner} e^{\wedge(-F)} \tag{2}$$

where $u \underset{g}{\llcorner} e^{\wedge F}$ denotes the *right contraction* of u by $e^{\wedge F}$ with respect to g. The product of $u \underset{g}{\llcorner} e^{\wedge F}$ and $v \underset{g}{\llcorner} e^{\wedge F}$ in (2) is taken in $C\ell(g)$. Let $\dim_{\mathsf{R}}(V) = n$. Then, the element $F \in \bigwedge^2 V^*$ is defined as

$$F = \sum_{K,L} (-1)^{|\pi(K,L)|} A_K \mathbf{e}_L \mathbf{j}^{-1} \tag{3}$$

where $\mathbf{j}^{-1}$ denotes the inverse of the unit pseudoscalar $\mathbf{j} = \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \cdots \wedge \mathbf{e}_n$ in $C\ell(g)$, the product $\mathbf{e}_L \mathbf{j}^{-1}$ is taken in $C\ell(g)$, and the summation is taken over all multi indices $K = [k_1, k_2]$ and $L = [l_1, l_2, \ldots, l_s]$ satisfying the following relations:

$$K \cap L = \emptyset,\ K \cup L = \{1, 2, \ldots, n\},\ n = 2 + s,\ K \text{ and } L \text{ are ordered by } <.$$

$\pi(K,L)$ denotes a permutation which puts the list $[k_1, k_2, l_1, l_2, \ldots, l_s]$ in the standard order $[1, 2, \ldots, n]$ and $|\pi(K,L)|$ equals 0 or 1 depending whether $\pi(K,L)$ is an even or odd element of S_n. In (3) we have also adopted notation $\mathbf{e}_L = \mathbf{e}_{l_1 l_2 \ldots l_s} = \mathbf{e}_{l_1} \wedge \mathbf{e}_{l_2} \wedge \cdots \wedge \mathbf{e}_{l_s}$.

Before we proceed to verify formula (2) with CLIFFORD [6], let's observe the following properties of the left and right contraction:

$$u \underset{g}{\llcorner} v = \mathbf{j}^{-1}((\mathbf{j}u) \wedge v), \quad u \underset{g}{\lrcorner} v = (u \wedge (v\mathbf{j}))\mathbf{j}^{-1}, \quad \mathbf{j}^{-1} = \frac{\tilde{\mathbf{j}}}{\det(g)}, \tag{4}$$

[1] The symbols $\mathsf{x} \underset{B}{\lrcorner} \mathsf{y}$, $\mathsf{x} \underset{A}{\lrcorner} \mathsf{y}$, and $\mathsf{x} \underset{g}{\lrcorner} \mathsf{y}$ denote the left contraction of y by x with respect to B, A, and g respectively.

where $\tilde{}$ is the g-dependent reversion in $C\ell(g)$.[2] Observe also that since $F \in \bigwedge^2 V^*$, we have

$$e^{\wedge F} = \sum_{k=0}^{N} \frac{F^{\wedge k}}{k!}, \quad N = \lfloor n/2 \rfloor \tag{5}$$

where $F^{\wedge k} = F \wedge F \wedge \cdots \wedge F$ is the exterior product of F computed k-times and $\lfloor \cdot \rfloor$ denotes the floor function. For example, for different values of n, F has the following form:

$$n = 2,\ F = -A_{12}\,\mathbf{e}_{12}^{-1}, \tag{6}$$

$$n = 3,\ F = -(A_{12}\,\mathbf{e}_3 - A_{13}\,\mathbf{e}_2 + A_{23}\,\mathbf{e}_1)\,\mathbf{e}_{123}^{-1}, \tag{7}$$

$$n = 4,\ F = -(A_{12}\,\mathbf{e}_{34} - A_{13}\,\mathbf{e}_{24} + A_{14}\,\mathbf{e}_{23} + A_{23}\,\mathbf{e}_{14} + A_{34}\,\mathbf{e}_{12} - A_{24}\,\mathbf{e}_{13})\,\mathbf{e}_{1234}^{-1}, \tag{8}$$

and so on. We will verify the validity of (2) in a numeric and a symbolic case. In the Maple symbolic language, formula (2) becomes:[3]

$$\mathrm{cmul}(u, v) = \mathrm{RCg}(\mathrm{cmulg}(\mathrm{RCg}(u, \mathrm{wexp}(F, N)), \mathrm{RCg}(v, \mathrm{wexp}(F, N)), \mathrm{wexp}(-F, N)))$$

with the CLIFFORD procedures `cmulg` and `RCg` representing the Clifford product and the right contraction in $C\ell(g)$ and `wexp` giving the exterior exponential in $\bigwedge V$. We limit our two examples to dimension $n = 3$. Computations presented in the following two sections can be extended with CLIFFORD to higher dimensions.

2.1 Numeric example when n = 3

Let's first assign an arbitrary matrix to B, split B into its symmetric and antisymmetric parts g and A, and compute the bivector F with a procedure `makeF`:

```
> dim:=3:eval(makealiases(dim,'ordered')):B:=matrix(dim,dim,[4,8,3,0,9,5,-2,1,7]);
```

$$B := \begin{bmatrix} 4 & 8 & 3 \\ 0 & 9 & 5 \\ -2 & 1 & 7 \end{bmatrix}$$

```
> g,A:=splitB(B);
```

$$g,\, A := \begin{bmatrix} 4 & 4 & \frac{1}{2} \\ 4 & 9 & 3 \\ \frac{1}{2} & 3 & 7 \end{bmatrix}, \begin{bmatrix} 0 & 4 & \frac{5}{2} \\ -4 & 0 & 2 \\ \frac{-5}{2} & -2 & 0 \end{bmatrix}$$

```
> F:=makeF(dim);
```

$$F := \frac{2}{91}\, e13 + \frac{86}{455}\, e12$$

Next, we find the exterior exponentials of F and $-F$ which we assign to Maple variables F_1 and F_2 respectively.

```
> N:=floor(dim/2):F1:=wexp(F,N);F2:=wexp(-F,N);
```

[2] From now on in this section we assume that $\det(g) \neq 0$.

[3] In CLIFFORD, the Clifford product uv of two elements u and v can be entered as `u &c v` (the infix form) or as `cmul(u,v)`.

$$F1 := 1 + \frac{2}{91}\, e13 + \frac{86}{455}\, e12, \quad F2 := 1 - \frac{2}{91}\, e13 - \frac{86}{455}\, e12.$$

Let u and v be two arbitrary elements in $C\ell(B)$:

```
> u:=2+e1-e23+e123;v:=3-e3+e12+e23;
```

$$u := 2 + e1 - e23 + e123, \quad v := 3 - e3 + e12 + e23.$$

The Clifford product $\underset{B}{uv}$ of u and v in $C\ell(B)$, the left hand side of (2), is then equal to

```
> cmul(u,v);
```

$$-48\, e3 + 79\, Id + 13\, e13 - 6\, e12 + 81\, e2 - 8\, e123 - 81\, e1$$

while the right hand side of (2) gives the same result:

```
> RCg(cmulg(RCg(u,F1),RCg(v,F1)),F2);
```

$$-48\, e3 + 79\, Id + 13\, e13 - 6\, e12 + 81\, e2 - 8\, e123 - 81\, e1$$

2.2 Symbolic computations when n = 3

A purely symbolic computation when $n = 3$ will look as follows. Matrix B is now defined as an arbitrary symbolic 3×3 matrix with a symmetric part g and an antisymmetric part A, and F is again computed using the procedure `makeF`. The exterior exponentials of F and $-F$ are again denoted respectively by F_1 and F_2. All symbolic parameters in B are assumed to be real or complex.

```
> dim:=3:eval(makealiases(dim)):
> B:=matrix(dim,dim,[g11,g12+A12,g13+A13,g12-A12,g22,g23+A23,g13-A13,g23-A23,g33]);
```

$$B := \begin{bmatrix} g11 & g12 + A12 & g13 + A13 \\ g12 - A12 & g22 & g23 + A23 \\ g13 - A13 & g23 - A23 & g33 \end{bmatrix}$$

```
> g,A:=splitB(B);
```

$$g,\, A := \begin{bmatrix} g11 & g12 & g13 \\ g12 & g22 & g23 \\ g13 & g23 & g33 \end{bmatrix}, \begin{bmatrix} 0 & A12 & A13 \\ -A12 & 0 & A23 \\ -A13 & -A23 & 0 \end{bmatrix}$$

```
> F:=map(normal,clicollect(makeF(dim)));
```

$$F := -\frac{(A23\, g11 + A12\, g13 - A13\, g12)\, e23}{\%1} + \frac{(-A23\, g13 + A13\, g23 - A12\, g33)\, e12}{\%1} + \frac{(A23\, g12 - A13\, g22 + A12\, g23)\, e13}{\%1}$$

$$\%1 := -g33\, g22\, g11 + g22\, g13^2 - 2\, g13\, g23\, g12 + g33\, g12^2 + g23^2\, g11$$

```
> N:=floor(dim/2):F1:=wexp(F,N):F2:=wexp(-F,N):
```

We will now define two general elements u and v in $C\ell(B)$ by decomposing them over a Grassmann basis (provided by a procedure `cbasis`). Coefficients in these two expansions are assumed to be real or complex.[4]

```
> cbasis(dim);
```

[4] For technical reasons, in Maple these coefficients cannot be defined as u_i and v_i; that is why we use here uu_i and vv_i.

$$[Id,\ e1,\ e2,\ e3,\ e12,\ e13,\ e23,\ e123]$$

```
> u:=add(uu[k]*cbasis(dim)[k],k=1..2^dim);
```

$$u := uu_1\, Id + uu_2\, e1 + uu_3\, e2 + uu_4\, e3 + uu_5\, e12 + uu_6\, e13 + uu_7\, e23 + uu_8\, e123$$

```
> v:=add(vv[k]*cbasis(dim)[k],k=1..2^dim);
```

$$v := vv_1\, Id + vv_2\, e1 + vv_3\, e2 + vv_4\, e3 + vv_5\, e12 + vv_6\, e13 + vv_7\, e23 + vv_8\, e123$$

The Clifford product of u and v in $C\ell(B)$ is then collected and assigned to a constant res_1 which we won't display due to its length.

```
> res1:=clicollect(cmul(u,v)):
```

As before, we finish by computing the right hand side of (2). By assigning it to res_2, we can then easily find that $res_1 - res_2 = 0$ as expected.

```
> res2:=clicollect(RCg(cmulg(RCg(u,F1),RCg(v,F1)),F2)):res1-res2;
```

$$0$$

3 Hecke algebra computations

In [7] it was shown that the symmetric group S_n and its group deformation, the Hecke algebra $H_{\mathsf{F}}(n,q)$ could be constructed as a subalgebra of a Clifford algebra $C\ell(B)$ for a suitably chosen q-dependent non-symmetric bilinear form B. q-Young operators were constructed as Clifford idempotents and the Hecke algebra representations in ideals generated by these idempotents were computed. Appropriate q-Young diagrams and tableaux representing symmetrizers, antisymmetrizers, and operators of mixed symmetries were realized inside the Hecke algebra, while the ordinary case of the symmetric group was obtained in the limit $q \to 1$.

The Hecke algebra is the generalization of the group algebra of the symmetric group S_n by adding the requirement that transpositions of adjacent elements $i, i+1$ are no longer involutions. Following [7] we set $t_i^2 = (1-q)t_i + q$ which reduces to $s_i^2 = 1$ in the limit $q \to 1$. The defining relations of the Hecke algebra will be given according to Bourbaki [11]. Let $\{\mathbf{1}, t_1, \ldots, t_n\}$ be a set of generators which fulfill these relations:

$$t_i^2 = (1-q)t_i + q, \tag{9}$$

$$t_i t_j = t_j t_i, \quad |i-j| \geq 2, \tag{10}$$

$$t_i t_{i+1} t_i = t_{i+1} t_i t_{i+1}, \tag{11}$$

then their algebraic span is the Hecke algebra $H_{\mathsf{F}}(n,q)$. The algebra morphism ρ which maps the Hecke algebra into the even part of an appropriate Clifford algebra was found in [17]. In particular, $\rho(t_i) = b_i := \mathbf{e}_i \wedge \mathbf{e}_{i+n}$, $i = 1, \ldots, n$, where $\mathbf{e}_1, \ldots, \mathbf{e}_{2n}$ are the generators of the Clifford algebra $C\ell(B,V)$, $V = \operatorname{span}\{\mathbf{e}_i\}$, with the following non-symmetric bilinear form B :

```
> dim:=8:n:=dim/2:eval(makealiases(dim,'ordered')):B:=defB(dim);
```

$$B := \begin{bmatrix} 0 & 0 & 0 & 0 & q & -1-q & 1 & 1 \\ 0 & 0 & 0 & 0 & -1-q & q & -1-q & 1 \\ 0 & 0 & 0 & 0 & 1 & -1-q & q & -1-q \\ 0 & 0 & 0 & 0 & 1 & 1 & -1-q & q \\ 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ q & 1 & 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & q & 1 & 1 & 0 & 0 & 0 & 0 \\ -1 & -1 & q & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Then, the form of B guarantees that the following relations hold:

$$b_i^2 = (1-q)b_i + q, \tag{12}$$

$$b_i b_j = b_j b_i, \quad |i-j| \geq 2, \tag{13}$$

$$b_i b_{i+1} b_i = b_{i+1} b_i b_{i+1}. \tag{14}$$

Following [17], we define the Hecke generators b_i as *balanced* basis Grassmann monomials of order 2 :

```
> for i from 1 to n do b.i:=(e.i) &w (e.(n+i)) od:
```

Using procedure `cliexpand` we can expand these generators in terms of the unevaluated Clifford product which is denoted in Maple as $\&C$:

```
> seq(cat('b',i)=cliexpand(b.i),i=1..4);
```

$b1 = (e1 \,\&\mathrm{C}\, e5) - q\,Id,\ b2 = (e2 \,\&\mathrm{C}\, e6) - q\,Id,\ b3 = (e3 \,\&\mathrm{C}\, e7) - q\,Id,\ b4 = (e4 \,\&\mathrm{C}\, e8) - q\,Id.$

Checking if the Hecke generators satisfy the defining relations can be done as follows:

```
> map(evalb,[seq(simplify(CS(cmul(b.i,b.i)-(1-q)*b.i-q))=0,i=1..n)]);
```

$$[true,\ true,\ true,\ true]$$

```
> map(evalb,[seq(seq(CS(cmul(b.i,b.j)-cmul(b.j,b.i))=0,i=j+2..n),j=1..n)]);
```

$$[true,\ true,\ true]$$

```
> map(evalb@factor@normal,[seq(CS(cmul(cmul(b.i,cat(b,i+1)),b.i)-
> cmul(cmul(cat(b,i+1),b.i),cat(b,i+1)))=0,i=1..n-1)]);
```

$$[true,\ true,\ true]$$

Let's define the remaining basis elements of the Hecke image $\rho(H_{\mathsf{F}}(4,q))$ in $C\ell(B)$:

```
> b12:=CS(cmul(b1,b2)):b21:=CS(cmul(b2,b1)):b23:=CS(cmul(b2,b3)):
> b32:=CS(cmul(b3,b2)):b34:=CS(cmul(b3,b4)):b43:=CS(cmul(b4,b3)):
> b121:=CS(cmul(b1,b21)):b123:=CS(cmul(b12,b3)):b234:=CS(cmul(b23,b4)):
> b321:=CS(cmul(b32,b1)):b432:=CS(cmul(b43,b2)):b4321:=CS(cmul(b432,b1)):
> b1234:=CS(cmul(b123,b4)):
```

Thus, ρ defines a homomorphism from the Hecke algebra $H_{\mathsf{F}}(4,q)$ into the Clifford algebra $C\ell(B)$. It was shown in [17] that ρ is not injective for $n \geq 4$, and that its kernel contains all Young diagrams which are not L-shaped.

3.1 Hecke Algebra $H_{\mathsf{F}}(2,q)$

We begin with the Hecke algebra $H_{\mathsf{F}}(2,q)$ generated by $\{Id, b_1\}$ which reduces to S_2 in the limit $q \to 1$. We have thus only one q-transposition b_1 from which we can calculate a q-symmetrizer

$R(12)$ and a q-antisymmetrizer $C(12)$. One of the features of the construction presented in [7] is that $R(12)$ and $C(12)$ are related by the reversion in $C\ell(B)$:

```
> R.12:=CS(q*Id+b1);   #S_2 symmetrizer
> C.12:=CS(Id-b1);     #S_2 antisymmetrizer
> evalb(reversion(C.12)=R.12);
```

$$R12 := e15 + q\,Id, \quad c12 := Id - e15$$

$$true$$

Observe, that $R(12)$ is the reversion of $C(12)$, that is, $R(12) = C(12)^{\tilde{}}$ where $\tilde{}$ is the reversion in the Clifford algebra $C\ell_{1,1}$. Operators $C(12)$ and $R(12)$ are almost idempotent and they annihilate each other:

```
> factor(CS(R.12 &c R.12));
```

$$(q+1)\,(q\,Id + e15)$$

```
> factor(CS(C.12 &c C.12));
```

$$(q+1)\,(Id - e15)$$

```
> R.12 &c C.12, C.12 &c R.12;
```

$$0,\ 0$$

Upon normalization, the symmetrizer $R(12)$ becomes the q-Young operator $Y_{1,2}^{(2)}$ while the antisymmetrizer $C(12)$ becomes $Y_{1,2}^{(11)}$. The following computation verifies that $Y_{1,2}^{(2)}$ and $Y_{1,2}^{(11)}$ are mutually annihilating idempotents adding up to the identity element.

```
> Y2.12 := R.12/(1+q);Y11.12:= C.12/(1+q);
```

$$Y212 := \frac{q\,Id + e15}{q+1}, \quad Y1112 := \frac{Id - e15}{q+1}$$

```
> ybas:=[Y2.12,Y11.12]:
> f:=(i,j)->CS(cmul(ybas[i],ybas[j]));
> YM:=evalm(linalg[matrix](2,2,(i,j)->f(i,j)));
> CS(Y11.12+Y2.12);
```

$$f := (i,\ j) \to \mathrm{CS}(\mathrm{climul}(ybas_i,\ ybas_j))$$

$$YM := \begin{bmatrix} \dfrac{q\,Id}{q+1} + \dfrac{e15}{q+1} & 0 \\ 0 & \dfrac{Id}{q+1} - \dfrac{e15}{q+1} \end{bmatrix}$$

$$Id$$

In our construction, the Hecke algebra $H_{\mathsf{F}}(2,q)$ is a subalgebra of the even part $C\ell_{1,1}^{+}$ of $C\ell_{1,1}$ and it is generated by $\{\mathbf{e}_1, \mathbf{e}_5\}$. That is, the bilinear form G on the vector space spanned by $\{\mathbf{e}_1, \mathbf{e}_5\}$ is

```
> G:=matrix(2,2,[B[1,1],B[1,5],B[5,1],B[5,5]]);
```

$$G := \begin{bmatrix} 0 & q \\ 1 & 0 \end{bmatrix}.$$

After symmetrization, diagonalization, and the limit $q \to 1$, G becomes $\mathrm{diag}(1,-1)$. Therefore, due to the isomorphism $C\ell(B) \simeq C\ell(g)$ (as associative algebras) we can view $H_{\mathsf{F}}(2,q)$ as the subalgebra of $C\ell_{1,1}^{+}$. This embedding implies that any idempotent of the Hecke algebra $H_{\mathsf{F}}(2,q)$

must be an even Clifford element. The following computation shows that the Young operators found above are the only two nontrivial mutually annihilating idempotents in $H_{\mathsf{F}}(2,q)$.

```
> xx:=CS(a*Id+b*b1);
```

$$xx := a\,Id + b\,e15$$

```
> sol:=clisolve2(CS(cmul(xx,xx)-xx),[a,b]);
```

$$sol := [b=0,\ a=0,\ a=1,\ b=0,\ b=-\frac{1}{q+1},\ a=\frac{1}{q+1},\ b=\frac{1}{q+1},\ a=\frac{q}{q+1}]$$

```
> f.1:=normal(subs(sol[4],xx));f.2:=normal(subs(sol[3],xx));
```

$$f1 := \frac{q\,Id + e15}{q+1}, \quad f2 := \frac{Id - e15}{q+1}.$$

In this case, the Young operators happen to be the two even primitive idempotents f_1, f_2 in $C\ell^+_{1,1}$. Notice, that in the case when $q=1$, the idempotents f_1 and f_2 reduce to the well-known primitive idempotents

```
> f11:=subs(q=1,f1);
```

$$f11 := \frac{1}{2}\,Id + \frac{1}{2}\,e15$$

```
> f22:=subs(q=1,f2);
```

$$f22 := \frac{1}{2}\,Id - \frac{1}{2}\,e15$$

in the Clifford algebra $C\ell_{1,1}$.

3.2 Hecke algebra $H_{\mathsf{F}}(3,q)$

In [7] Young operators related to various symmetries were constructed as idempotent elements in the Hecke algebra $H_{\mathsf{F}}(3,q)$ embedded into the even subalgebra $C\ell^+_{2,2}$ of $C\ell_{2,2}$. As one of the main features of this construction, the Young operators corresponding to conjugate Young tableaux in the sense of McDonald [24] were related through the reversion in the Clifford algebra. The goal was to find four Young operators known to exist from the general theory of the Hecke algebras for $n=3$. The four q-Young operators found had one parameter and generalized the four Young operators of S_3 described in Hamermesh [18] on p. 245. One of them was a full symmetrizer, one was a full antisymmetrizer and two were of mixed symmetry.

The construction began with finding the most general element X in $H_{\mathsf{F}}(3,q)$ such that[5]:

$$X + \tilde{X} = Id \tag{15}$$

$$X^2 = X \tag{16}$$

$$X\tilde{X} = 0 \tag{17}$$

In the first step, the most general element in the Hecke algebra $H_{\mathsf{F}}(3,q)$ was found that satisfied (15)[6]:

```
> bset:=[Id,b1,b2,b12,b21,b121]:X:=bexpand(add(K[i]*bset[i],i=1..6)):
> sollist[1]:=clisolve2(X+reversion(X)-Id,[seq(K[i],i=1..6)]):
> X:=bexpand(subs(sollist[1],X));
```

[5] See formulas (24), (25) and (26) in [7].

[6] Procedure `clisolve2` is capable of solving the equation $X + \tilde{X} - 1 = 0$ for the free parameters $K_1, \ldots, K_6$ appearing in X. However, when used repeatedly, it randomly selects free parameters in the solution.

$$X := (-\frac{1}{2}\,K_6\,q + \frac{1}{2}\,K_3\,q + \frac{1}{2}\,K_6\,q^2 + \frac{1}{2}\,K_2\,q + \frac{1}{2} - \frac{1}{2}\,K_3 - \frac{1}{2}\,K_2)\,Id + K_2\,b1 + K_3\,b2 + K_4\,b12$$
$$+ (-K_6 + K_6\,q - K_4)\,b21 + K_6\,b121$$

In the above, procedure `bexpand` expands elements in the Hecke algebra, which are normally expressed in the Grassmann basis of $C\ell_{2,2}$, in terms of the Hecke basis $\{\mathbf{1}, b_1, b_2, b_{12}, b_{21}, b_{121}\}$.

Thus, the solution to (15) gives an element that belongs to a family parameterized by four real or complex parameters. In order to simplify Maple output, we define two aliases and then we substitute X found above into the second equation (16) .

```
> alias(alpha=RootOf((1+q)*_Z^2+(-q^2*K[4]+K[4]+q*K[2]-1+K[2])*_Z+K[4]*K[2]+
> K[2]^2-K[4]-K[2]+q*K[4]-q^2*K[4]^2-q*K[4]^2-q^2*K[2]*K[4]+q*K[2]^2));
> alias(kappa=RootOf((1+q)*_Z^2+(-q^2*K[4]+K[4]+q*K[2]+1+K[2])*_Z+K[4]*
> K[2]+K[2]^2+K[4]+K[2]-q*K[4]-q^2*K[4]^2-q*K[4]^2-q^2*K[2]*K[4]+q*K[2]^2));
```

$$I,\ \alpha,\ \kappa$$

```
> sollistxx1:=clisolve2(cmul(X,X)-X,[seq(K[i],i=1..6)]):
> xxlist:=[seq(bexpand(subs(sollistxx1[i],X)),i=1..nops(sollistxx1))]:
> for i from 1 to nops(xxlist) do r.i:=bexpand(xxlist[i]) od:
```

The six representatives $r_1, \ldots, r_6$ defined in this last Maple command are the same as the elements displayed in [7] after formula (30). Using the command `findbasis` we can determine that the set $\{r_i\}$, $i = 1, \ldots, 6$, is of rank 4 :

```
> nops(findbasis([r.(1..6)]));
```

$$4$$

Since the representatives are related by the reversion, we seek four linearly independent elements. For example, r_1, r_2, r_3 and r_5 are linearly independent. From now on we assign them to f_1, f_2, f_3, f_4.

```
> nops(findbasis([xxlist[1],xxlist[2],xxlist[3],xxlist[5]]));
```

$$4$$

```
> f1:=r.1:f2:=r.2:f3:=r.3:f4:=r.5:
```

We can easily verify that the four elements $\{f_1, f_2, f_3, f_4\}$ satisfy equation (15):

```
> CS(f.1+reversion(f.1)-Id),CS(f.2+reversion(f.2)-Id),
> CS(f.3+reversion(f.3)-Id),CS(f.4+reversion(f.4)-Id);
```

$$0,\ 0,\ 0,\ 0$$

and equation (16):

```
> simplify(CS(f.1 &c f.1 - f.1)),simplify(CS(f.2 &c f.2 - f.2)),
> simplify(CS(f.3 &c f.3 - f.3)),simplify(CS(f.4 &c f.4 - f.4));
```

$$0,\ 0,\ 0,\ 0$$

Each of the four non-primitive idempotents f_i generates a three-dimensional one-sided ideal in $H_{\mathsf{F}}(3, q)$. In order to split such ideal into a one-dimensional space and a two-dimensional space, one had to find a way of splitting at least one of these idempotents into a sum of two mutually annihilating idempotents. It was observed in [7] that f_1 contained a full symmetrizer. Since the full symmetrizer $Y^{(3)}_{1,2,3}$ was defined there as

$$Y^{(3)}_{1,2,3} \;:=\; \frac{q^3\mathbf{1} + q^2 b_1 + q^2 b_2 + q b_{12} + q b_{21} + b_{121}}{(1+q+q^2)(1+q)}, \tag{18}$$

the full antisymmetrizer $Y^{(111)}_{1,2,3}$ was, by construction, the reversion of the symmetrizer, that is, $Y^{(111)}_{1,2,3} := Y^{(3)}_{1,2,3}{}^{\sim}$,

$$Y^{(111)}_{1,2,3} := \frac{\mathbf{1} - b_1 - b_2 + b_{12} + b_{21} - b_{121}}{(1+q+q^2)(1+q)}. \tag{19}$$

Therefore, by subtracting the full antisymmetrizer $Y^{(111)}_{1,2,3}$ from f_1, the first Young operator $Y^{(21)}_{1,3,2}$ of a mixed type was found. Then, the second Young operator $Y^{(21)}_{1,2,3}$ of a mixed type was computed by applying the reversion (conjugate) to $Y^{(21)}_{1,3,2}$, that is, $Y^{(21)}_{1,2,3} := Y^{(21)}_{1,3,2}{}^{\sim}$. Namely[7],

```
> Y21.132:=bexpand(f.1-Y111.123):Y21.123:=bexpand(reversion(Y21.132)):
```

Furthermore, $Y^{(111)}_{1,2,3}$ annihilates $Y^{(21)}_{1,3,2}$ when multiplied from both sides. This is a reflection of the fact that the left ideal (the representation space) in the Hecke algebra $H_{\mathsf{F}}(3,q)$ generated by f_1 decomposes into a direct sum of one-dimensional left ideal and a two-dimensional left ideal.

```
> simplify(Y.111.123+Y.21.132-f1);
```

$$0$$

```
> simplify(Y.21.123+Y.3.123-reversion(f1));
```

$$0$$

```
> CS(Y.111.123 &c Y.21.132),CS(Y.21.132 &c Y.111.123);
```

$$0,\ 0$$

```
> CS(Y.21.132 &c Y.3.123),CS(Y.3.123 &c Y.21.123);
```

$$0,\ 0$$

```
> CS(Y.111.123 &c Y.111.123 - Y.111.123),CS(Y.21.132 &c Y.21.132 - Y.21.132);
```

$$0,\ 0$$

In addition to the symmetrizer $Y^{(3)}_{1,2,3}$ and the antisymmetrizer $Y^{(111)}_{1,2,3}$ displayed in (18) and (19), we have also two Young operators $Y^{(21)}_{1,2,3}$ and $Y^{(21)}_{1,3,2}$ of mixed symmetry:

```
> 'Y21.123'=bexpand(Y21.123);
```

$$\begin{aligned} Y21.123 = {} & \frac{q\,Id}{q+1+q^2} + \frac{(K_4\,q^3 + 2\,K_4\,q^2 - q^2 + 2\,K_4\,q + K_4)\,b1}{(q+1+q^2)\,(1+q)} \\ & - \frac{(K_4\,q^3 + 2\,K_4\,q^2 + q + 2\,K_4\,q + K_4)\,q\,b2}{(q+1+q^2)\,(1+q)} \\ & - \frac{(K_4\,q^3 + 2\,K_4\,q^2 + q + 2\,K_4\,q + K_4)\,b12}{q^3 + 2\,q^2 + 2\,q + 1} \\ & + \frac{(q^5\,K_4 + q^4\,K_4 + q^3 + K_4\,q^3 - q^2 + K_4\,q^2 + K_4\,q + K_4 - 1)\,b21}{q\,(q^3 + 2\,q^2 + 2\,q + 1)} \\ & + \frac{(q^4\,K_4 + K_4\,q^3 + q^2 - K_4\,q - K_4 + 1)\,b121}{(q+1+q^2)\,q\,(1+q)} \end{aligned}$$

```
> 'Y21.132'=bexpand(Y21.132);
```

[7] Due to lengthy displays, we refer Reader to formulas (33) and (34) in [7].

$$Y21.132 = \frac{q\,Id}{q+1+q^2} - \frac{(K_4\,q^3 + 2\,K_4\,q^2 + 2\,K_4\,q - 1 + K_4)\,b1}{(q+1+q^2)\,(1+q)}$$
$$+ \frac{(q^4\,K_4 + 2\,K_4\,q^3 + 2\,K_4\,q^2 + K_4\,q + 1)\,b2}{(q+1+q^2)\,(1+q)}$$
$$+ \frac{(K_4\,q^3 + 2\,K_4\,q^2 + 2\,K_4\,q - 1 + K_4)\,b12}{q^3 + 2\,q^2 + 2\,q + 1}$$
$$- \frac{(q^5\,K_4 + q^4\,K_4 + K_4\,q^3 + q^3 + K_4\,q^2 + K_4\,q + q + K_4 - 1)\,b21}{q\,(q^3 + 2\,q^2 + 2\,q + 1)}$$
$$- \frac{(q^4\,K_4 + K_4\,q^3 + q^2 - K_4\,q - K_4 + 1)\,b121}{(q+1+q^2)\,q\,(1+q)}$$

In order to represent our Young operator $Y^{(21)}_{1,3,2}$ as a product of a row-symmetrizer $R(13)$ and a column-antisymmetrizer $C(12) = f_1$, we use f_1 defined above and compute $R(13)$ from the equation

$$Y^{(21)}_{132} = R(13) f_1. \tag{20}$$

In order to solve the above equation in Maple for $R(13)$, we need to find an element Y in the Hecke algebra which would not only satisfy equation (20) but also such that $Y + Y^{\sim} = \mathbf{1}$. This is because we want the column antisymmetrizer $C(12)$ to remain related to $R(12)$ through the reversion. Notice also that we are justified in defining $C(12)$ as equal to f_1 (modulo a normalizing factor) because f_1 generalizes the antisymmetrizer $C(12)$ from S_2 to S_3 :

```
> bexpand(c12/(1+q));
```

$$\frac{Id}{1+q} - \frac{b1}{1+q}$$

```
> bexpand(f1);
```

$$\frac{Id}{1+q} - K_4\,b1 + K_4\,q\,b2 + K_4\,b12 - \frac{(K_4\,q^3 + q + K_4 - 1)\,b21}{q\,(1+q)} - \frac{(-K_4 + K_4\,q^2 + 1)\,b121}{q\,(1+q)}$$

That is, f_1 is seen to contain $C(12)/(1+q)$ if we replace K_4 with $1/(1+q)$. Thus, we first express Y in the Hecke basis contained in the list `bset` below and then we make sure that $Y + Y^{\sim} = \mathbf{1}$:

```
> bset:=[Id,b1,b2,b12,b21,b121]:
> Y:=bexpand(add(P[i]*bset[i],i=1..6)):
> Y:=bexpand(op(clisolve2(Y+reversion(Y)-Id,Y))):
```

$$Y := (\frac{1}{2}\,P_3\,q - \frac{1}{2}\,P_6\,q + \frac{1}{2}\,P_6\,q^2 + \frac{1}{2}\,P_2\,q - \frac{1}{2}\,P_2 + \frac{1}{2} - \frac{1}{2}\,P_3)\,Id + P_2\,b1 + P_3\,b2$$
$$+ (-P_5 - P_6 + P_6\,q)\,b12 + P_5\,b21 + P_6\,b121$$

Next we require that the above found element Y satisfies equation (20):

```
> R.13:=bexpand(op(clisolve2(Y21.132-cmul(Y,f_1),Y)));
```

$$R13 := \frac{q\,Id}{1+q} - \frac{(-q^2+q^2\,P_3+P_3\,q-1+P_3)\,b1}{(q+1+q^2)\,q} + P_3\,b2 + \frac{(q^2\,P_3+P_3\,q+P_3-1)\,b12}{q\,(q+1+q^2)}$$
$$- \frac{(q^5\,P_3+q^4\,P_3+q^3\,P_3-q^2+q^2\,P_3+P_3\,q-1+P_3)\,b21}{(1+q)\,q^2\,(q+1+q^2)}$$
$$- \frac{(q^4\,P_3+q^3\,P_3-P_3\,q+1-P_3)\,b121}{q^2\,(q^3+2\,q^2+2\,q+1)}$$

We verify that $R(13)$ is an idempotent:

```
> CS(R13 &c R13-R13);
```

$$0$$

It was pointed out in [7] that when the Clifford product $R(13)f_1$ is computed, the free parameter P_3 disappears:

```
> bexpand(cmul(R13,f1));
```

$$\frac{q\,Id}{q+1+q^2} - \frac{(K_4\,q^3+2\,K_4\,q^2+2\,K_4\,q-1+K_4)\,b1}{\%1}$$
$$+ \frac{(q^4\,K_4+2\,K_4\,q^3+2\,K_4\,q^2+K_4\,q+1)\,b2}{\%1}$$
$$+ \frac{(K_4\,q^3+2\,K_4\,q^2+2\,K_4\,q-1+K_4)\,b12}{\%1}$$
$$- \frac{(q^5\,K_4+q^4\,K_4+K_4\,q^3+q^3+K_4\,q^2+K_4\,q+q+K_4-1)\,b21}{q\,\%1}$$
$$- \frac{(q^4\,K_4+K_4\,q^3+q^2-K_4\,q-K_4+1)\,b121}{(q+1+q^2)\,q\,(1+q)}$$
$$\%1 := q^3+2\,q^2+2\,q+1$$

which gives $Y^{(21)}_{1,3,2}$ computed above:

```
> CS(%-Y21.132);
```

$$0$$

In order to construct the representation spaces from the four Young operators, one needs to find at least one *Garnir* $G^{(\lambda_i)}_{i,j}$ element in the Hecke algebra [7, 20]. All Garnir elements can be seen to act as row or column cycles in Young tableaux thereby generating non-standard tableaux which correspond to the basis vectors of the representation space. A Garnir element has the following defining properties:

$$Y^{(21)}_{1,2,3}\,G^{(21)}_{1,1} = 0, \tag{21}$$

$$G^{(21)}_{1,1}\,Y^{(21)}_{1,2,3} \neq 0. \tag{22}$$

The reason for requiring (22) is that we want $G^{(21)}_{1,1}Y^{(21)}_{1,2,3}$ to be a second basis element in the left Hecke ideal generated by $Y^{(21)}_{1,2,3}$. In order to solve (21) and (22), we begin by assigning to X a general element of the Hecke algebra, that is, X is a linear combination of the Hecke basis elements with some undefined coefficients $K_i, i = 1, \ldots, 6$:

```
> X:=bexpand(K[1]*Id+K[2]*b1+K[3]*b2+K[4]*b12+K[5]*b21+K[6]*b121);
```

$$X = K_1\,Id + K_2\,b1 + K_3\,b2 + K_4\,b12 + K_5\,b21 + K_6\,b121$$

We use `clisolve2` to solve equation (21). All three solutions returned by Maple are assigned to a list `sol`.

```
> sol:=clisolve2(cmul(Y21.123,X),[seq(K[i],i=1..6)]):nops(sol);
```

$$3$$

```
> for i from 1 to nops(sol) do X.i:=bexpand(subs(sol[i],X)) od;
```

$$X1 := (K_2\,q - K_6\,q + K_4)\,Id + K_2\,b1 + K_3\,b2 + K_4\,b12 + \frac{(K_6\,q^2 + K_4\,q^2 - K_3\,q - K_6\,q - K_4\,q + K_2 + K_4)\,b21}{q} + K_6\,b121$$

$$X2 := K_1\,Id + \frac{(K_6\,q^3 + q^2\,K_1 + K_6\,q^2 + q\,K_1 + 1)\,b1}{q^2\,(1+q)} + K_3\,b2 - \frac{b12}{q\,(1+q)} + \frac{(q^5\,K_6 - q^4\,K_3 - q^3\,K_3 - q^3 + K_6\,q^2 + q^2\,K_1 + q^2 + q\,K_1 - q + 1)\,b21}{q^3\,(1+q)} + K_6\,b121$$

$$X3 := K_1\,Id + \frac{(q - 1 + K_6\,q + 2\,K_6\,q^2 + 2\,K_6\,q^3 + K_1 + 2\,q\,K_1 + q^4\,K_6 + 2\,q^2\,K_1 + q^3\,K_1)\,b1}{q\,(q^3 + 2\,q^2 + 2\,q + 1)} + K_3\,b2 - \frac{(q-1)\,b12}{q^3 + 2\,q^2 + 2\,q + 1} + ((q^6\,K_6 + q^5\,K_6 - q^5\,K_3 - 2\,q^4\,K_3 - q^4 + q^4\,K_6 + 2\,q^3 + q^3\,K_1 + K_6\,q^3 - 2\,q^3\,K_3 + K_6\,q^2 + 2\,q^2\,K_1 - q^2\,K_3 - 2\,q^2 + K_6\,q + 2\,q\,K_1 + 2\,q + K_1 - 1)b21)/(q^2\,(q^3 + 2\,q^2 + 2\,q + 1)) + K_6\,b121$$

One way to find out if the three solutions returned by Maple are really linearly independent is to use a procedure `findbasis` which from the given list of Clifford polynomials extracts all polynomials which are linearly independent.

```
> nops(findbasis([X.1,X.2,X.3]));
```

$$3$$

The three solutions X_1, X_2, X_3 are therefore linearly independent. Another way to verify that fact would be to try to find three coefficients c_1, c_2, c_3, not all equal to zero, that would satisfy the following linear combination:

$$c_1\,X_1 + c_2\,X_2 + c_3\,X_3 = 0. \tag{23}$$

Thus, we can use again the procedure `clisolve2` and try to solve equation (23) as follows:

```
> clisolve2(c[1]*X.1+c[2]*X.2+c[3]*X.3,[c[1],c[2],c[3]]);
```

$$[\{c_1 = 0,\ c_2 = 0,\ c_3 = 0\}]$$

As expected, all coefficients are zero. Thus, X_1, X_2, X_3 are three linearly independent solutions of (21), that is, $Y^{(21)}_{1,2,3} X_i = 0$ for $i = 1, 2, 3$.

We proceed now to verify whether the solutions we have just obtained satisfy also equation (22). In particular, we will try to see if there any non-zero values of the parameters K_2, K_4, K_5, K_6 in X_1 so that $X_1 Y^{(21)}_{1,2,3}$ would be 0. We will again use the procedure `clisolve2`.

```
> var1:=select(type,indets(X1),indexed);
```

$$var1 := \{K_2,\ K_4,\ K_5,\ K_6\}$$

```
> clisolve2(X.1 &c Y.21.123,vars1);
```

$$[\,]$$

As expected, Maple returns an empty solution set.

3.3 Automorphism α_q and the Garnir elements in the Hecke algebra

In [7], an automorphism α_q was introduced in the Hecke algebra via the formula (48). α_q replaces the reversion $\tilde{}$ and gives the inverse of the basis element b_K for any multi-index K. The automorphism α_q is then extended to the whole Hecke algebra by the following definition:

$$\begin{aligned} \alpha_q(b_{i_1}\dots b_{i_s}) &= (\tfrac{-1}{q})^s (b_{i_1}\cdots b_{i_s})\tilde{} \\ &= (\tfrac{-1}{q})^s (\tilde{b_{i_s}}\cdots\tilde{b_{i_1}}) \\ &= \alpha_q(b_{i_s})\cdots\alpha_q(b_{i_1}). \end{aligned} \tag{24}$$

In CLIFFORD, the α_q automorphism has been programmed as a Maple procedure `alpha2`. For example, when $s=2$, verification of (24) can be done as follows:[8]

```
> evalb(bexpand(alpha2(b1 &c b2))=bexpand(alpha2(b2) &c alpha2(b1)));
> evalb(bexpand(alpha2(b2 &c b1))=bexpand(alpha2(b1) &c alpha2(b2)));
```

$$true,\ true,$$

The following is a verification that indeed $\alpha_q(b_i) = b_i^{-1} = \frac{(q-1)}{q}\mathbf{1} + \frac{b_i}{q}$, $i = 1, 2$.[9]

```
> bexpand(alpha2(b1)),evalb(bexpand(cinv(b1)) = bexpand(alpha2(b1)));
> bexpand(alpha2(b2)),evalb(bexpand(cinv(b2)) = bexpand(alpha2(b2)));
```

$$\frac{(q-1)\,Id}{q} + \frac{b1}{q},\ true$$

$$\frac{(q-1)\,Id}{q} + \frac{b2}{q},\ true$$

On the other hand, (24) implies that $\alpha_q(b_{12}) = (\frac{-1}{q})^2\,\tilde{b_{12}} = b_{12}^{-1}$:

```
> bexpand(alpha2(b12));
> map(normal,bexpand((-1/q)^2*(reversion(b12))));
> evalb(bexpand(cinv(b12))=bexpand(alpha2(b12)));
```

$$\frac{(1-2\,q+q^2)\,Id}{q^2} + \frac{(q-1)\,b1}{q^2} + \frac{(q-1)\,b2}{q^2} + \frac{b21}{q^2}$$

$$\frac{(1-2\,q+q^2)\,Id}{q^2} + \frac{(q-1)\,b1}{q^2} + \frac{(q-1)\,b2}{q^2} + \frac{b21}{q^2}$$

$$true$$

and similarly for b_{21}. For completeness we only show that $\alpha_q(b_{121}) = (\frac{-1}{q})^3\,\tilde{b_{121}} = b_{121}^{-1}$:

[8] Procedure `alpha2` is part of a package `suppl`. Rather than displaying and comparing long expressions, it is often convenient to use Maple's built-in Boolean procedure `evalb` which returns `true` or `false` when the two expressions are equal or not.

[9] In CLIFFORD, the symbolic inverse of any element can be found using a procedure `cinv`.

```
> bexpand(alpha2(b121));
> map(normal,bexpand((-1/q)^3*(reversion(b121))));
> evalb(bexpand(cinv(b121))=bexpand(alpha2(b121)));
```

$$\frac{(-2\,q^2+q^3+2\,q-1)\,Id}{q^3}+\frac{(1-2\,q+q^2)\,b1}{q^3}+\frac{(1-2\,q+q^2)\,b2}{q^3}+\frac{(q-1)\,b12}{q^3}+\frac{(q-1)\,b21}{q^3}+\frac{b121}{q^3}$$

$$\frac{(-2\,q^2+q^3+2\,q-1)\,Id}{q^3}+\frac{(1-2\,q+q^2)\,b1}{q^3}+\frac{(1-2\,q+q^2)\,b2}{q^3}+\frac{(q-1)\,b12}{q^3}+\frac{(q-1)\,b21}{q^3}+\frac{b121}{q^3}$$

$$true$$

Thus, we have verified that $\alpha_q(b_K)=(\frac{-1}{q})^{|K|}\,\tilde{b_K}=b_K^{-1}$ for a multi-index K of length $|K|$. However, this property does not extend to non-homogeneous (non-versor like) elements of the Hecke algebra. Let `hecke` be a Maple variable representing an arbitrary element in $H_{\mathsf{F}}(3,q)$ expanded in the Hecke basis $\{\mathbf{1},b_1,b_2,b_{12},b_{21},b_{121}\}$:

```
> hecke:=h[1]*Id + h[2]*'b1' + h[3]*'b2' + h[4]*'b12' + h[5]*'b21' + h[6]*'b121';
```

$$hecke := h_1\,Id + h_2\,b1 + h_3\,b2 + h_4\,b12 + h_5\,b21 + h_6\,b121$$

Then the action of α_q on `hecke` can be written as follows:

```
> alpha2hecke:=bexpand(alpha2(hecke));
```

$$alpha2hecke := \frac{H1\,Id}{q^3}+\frac{H2\,b1}{q^3}+\frac{H3\,b2}{q^3}+\frac{H4\,b12}{q^3}+\frac{H5\,b21}{q^3}+\frac{H6\,b121}{q^3}$$

where the expressions $H_i, i=1,\ldots,6,$ are q-polynomials with coefficients expressed in terms of $h_i, i=1,\ldots,6.$[10] With a little experimentation it can be easily verified that, for example, α_q does not give the inverse of the element $\mathbf{1}+b_1$:

```
> h[1],h[2],h[3],h[4],h[5],h[6]:=1,1,0,0,0,0:
> bexpand(hecke);
```

$$Id + b1$$

```
> bexpand(CS(cinv(hecke))),bexpand(alpha2(hecke));
```

$$\frac{1}{2}\,\frac{(q-2)\,Id}{q-1}+\frac{1}{2}\,\frac{b1}{q-1},\quad \frac{(2\,q-1)\,Id}{q}+\frac{b1}{q}.$$

We return now to the problem of finding the Garnir element $G_{1,1}^{(21)}$ that would satisfy equations (21) and (22). Recall that equation (21) had three linearly independent solutions X_1, X_2, X_3. Notice, that X_1 does not annihilate the Young operator $Y_{1,2,3}^{(21)}$ and $\alpha_q(X_1)$ does not annihilate the Young operator $Y_{1,3,2}^{(21)}$ no matter what values are assigned to the parameters K_2, K_4, K_5, K_6 :

```
> clisolve2(X.1 &c Y.21.123,vars1),clisolve2(alpha2(X.1) &c Y.21.132,vars1);
```

$$[\,],\ [\,]$$

[10] In order to get the display for $\alpha_q(hecke)$ shown above expressions $H_i, i=1,\ldots,6,$ have been defined in Maple as `aliases`. See Appendix 2 for their definitions.

Furthermore, the following shows that the elements $X_1 Y^{(21)}_{1,2,3}$ and $\alpha_q(X_1) Y^{(21)}_{1,3,2}$ are linearly independent:

```
> nops(findbasis([X.1 &c Y.21.123,alpha2(X.1) &c Y.21.132]));
```

$$2$$

Since the six elements

$$\{Y^{(3)}_{1,2,3}, Y^{(21)}_{1,2,3}, X_1 Y^{(21)}_{1,2,3}, \alpha_q(X_1) Y^{(21)}_{1,3,2}, Y^{(21)}_{1,3,2}, Y^{(111)}_{1,2,3}\}$$

are linearly independent,

```
> nops(findbasis([Y.3.123,Y.21.123,X.1 &c Y.21.123,alpha2(X.1) &c Y.21.132,
> Y.21.132,Y.111.123]));
```

$$6$$

they may form a basis S for the Hecke algebra $H_{\mathsf{F}}(3,q)$. We define the Garnir element $G^{(21)}_{1,1}$ to be equal to X_1.[11]

```
> alias(t1=K[6]*q^2+K[4]*q^2-K[5]*q-K[6]*q-K[4]*q+K[2]+K[4]):
> G21.1.1:=bexpand(X1);
```

$$G2111 := (K_2\, q - K_6\, q + K_4)\, Id + K_2\, b1 + \frac{t1\ b2}{q} + K_4\, b12 + K_5\, b21 + K_6\, b121$$

```
> S:=[Y.3.123,Y.21.123,G21.1.1 &c Y.21.123,
> alpha2(G21.1.1) &c Y.21.132,Y.21.132,Y.111.123]:
```

Procedure `yexpand` (See Appendix 2) is used to expand elements in the Hecke algebra $H_{\mathsf{F}}(3,q)$ in terms of the Young basis S. For example,

```
> yexpand(Y3.123);
```

$$Y3.123$$

```
> yexpand(&c(G21.1.1,Y21.123));
```

$$G21.1.1\ \&c\ Y21.123$$

```
> yexpand(&c(alpha2(G21.1.1),Y21.132));
```

$$\alpha 2(G21.1.1)\ \&c\ Y21.132$$

Recall that the original basis in the Hecke algebra was: $\{\mathbf{1}, b_1, b_2, b_{12}, b_{21}, b_{121}\}$. Each original basis element should be representable in terms of the Young basis S :

```
> 'Id'=yexpand(Id);
```

$$Id = Y3.123 + Y21.123 + Y21.132 + Y111.123$$

```
> 'b1'=yexpand(b1);
```

$$b1 = Y3.123 + \frac{w2\ Y21.123}{1+q} + \frac{q\, w1\ (G21.1.1\ \&c\ Y21.123)}{w3\ (1+q)} - \frac{q^2\ (\alpha 2(G21.1.1)\ \&c\ Y21.132)}{w4} + \frac{q\, w5\ Y21.132}{w6} - q\ Y111.123$$

```
> 'b2'=yexpand(b2);
```

[11] The above computations could have been performed with X_2 or X_3 instead of X_1, and the results would have been similar.

$$b2 = Y3.123 - \frac{w7\, q\, Y21.123}{1+q} - \frac{w8\, q\, (G21.1.1 \,\&c\, Y21.123)}{w3\,(1+q)}$$
$$+ \frac{q^3\,(\alpha 2(G21.1.1)\,\&c\, Y21.132)}{w4} - \frac{w9\ Y21.132}{w6} - q\ Y111.123$$

where $w_i, i = 1, \ldots, 9,$ are polynomials in q parameterized in terms of K_2, K_4, K_5, K_6. They have been defined as Maple aliases and are shown in Appendix 2.

4 Singular Value Decomposition

Our next application of Clifford algebras will be to the Singular Value Decomposition (SVD) of a matrix [28]. There are many uses of SVD such as in image processing, description of the so called *principal gains* in a multivariable system [25], or in an automated data indexing known as *Latent Semantic Indexing* (or LSI). LSI presents a very interesting and useful technique in information retrieval models and it is based on the SVD [10]. While in these practical cases computations are done numerically, it may be of interest to ask whether such decomposition of a matrix can be performed in the framework of Clifford algebras. That is, if any new insights, theoretical or otherwise, into such decomposition could be gained when stated in the Clifford algebra language. In this section we will present examples of such computations.

We will explore a well-known fact that when $p - q \neq 1 \bmod 4$, Clifford algebra $C\ell_{p,q}$ is a simple algebra of dimension 2^n, $n = p + q$, isomorphic to a full matrix algebra $\mathrm{Mat}(2^k, \mathsf{K})$ of $2^k \times 2^k$ matrices[12] with entries in K which is R, C, or H (see [5]). Thus, any operation performed on a matrix A can be expressed as an operation on a corresponding to it element p in $C\ell_{p,q}$. The choice of the signature (p, q) depends on the size of A and the division ring K. Of course, for computational reasons one should find the smallest Clifford algebra $C\ell_{p,q}$ such that the given matrix A can be embedded into $\mathrm{Mat}(2^k, \mathsf{K}) \overset{\varphi}{\simeq} C\ell_{p,q}$. In the following we will use the same approach as in [4] where a technique for matrix exponentiation based on the isomorphism φ was presented. In particular, we will use a faithful spinor representation of $C\ell_{p,q}$ in a minimal left ideal $S = C\ell_{p,q} f$ generated by a primitive idempotent f. Symbolic computations of such representations with CLIFFORD were shown in [5].

Following [28], let A be an $m \times n$ real matrix of rank r. Then the SVD of A is defined a factorization of A into a product of three matrices U, Σ, V^{-1} where U and V are orthogonal matrices $m \times m$ and $n \times n$ respectively, and Σ is a $m \times n$ matrix containing *singular values* of A on its "diagonal".

$$A = U\Sigma V^{-1}, \quad U^T U = I, \quad V^T V = I. \tag{25}$$

The matrices $V = [v_1|v_2|\ldots|v_n]$ and $U = [u_1|u_2|\ldots|u_m]$ contain orthonormal bases for all four fundamental spaces of A. Namely, the first r columns $v_1, v_2, \ldots, v_r$ of V provide a basis for the row space $\mathcal{R}(A^T)$ while the remaining $n - r$ columns of V provide a basis for the null space $\mathcal{N}(A)$. Likewise, the first r columns $u_1, u_2, \ldots, u_r$ of U provide a basis for the column space $\mathcal{C}(A)$ while the remaining $m - r$ columns of U provide a basis for the left-null space $\mathcal{N}(A^T)$. Vectors v_i are the normalized eigenvectors of $A^T A$ while vectors u_i are the normalized eigenvectors of AA^T. For

[12] The value of $k = q - r_{q-p}$, where r_i is the Radon-Hurwitz number. The Radon-Hurwitz number is defined by a recursion as $r_{i+8} = r_i + 4$ and these initial values: $r_0 = 0$, $r_1 = 1$, $r_2 = r_3 = 2$, $r_4 = r_5 = r_6 = r_7 = 3$.

$i = 1, \ldots, r$, these vectors can be chosen to be related via the positive singular values σ_i of A which are just the square roots of the eigenvalues of $A^T A$ (or of AA^T.) Namely,

$$Av_i = \sigma_i u_i, \quad i = 1, \ldots, r. \tag{26}$$

It is a little tricky to make sure that the above relation is satisfied: this is because the choice of vectors u_i is independent of the choice of vectors v_i. However, it is always possible to do so as we will see below (see also [28]). In order to complete the picture, the orthonormal set $\{v_1, \ldots, v_r\}$ needs to be completed to a full orthonormal basis for R^n while $\{u_1, \ldots, u_r\}$ needs to be completed to a full orthonormal basis for R^m. Since the additional vectors are being annihilated by A and A^T respectively, that is, they are eigenvectors of A and A^T (or of $A^T A$ and AA^T) that correspond to the eigenvalue 0, care has to be exercised when finding them. For example, while the eigenvectors of the symmetric matrix AA^T are automatically orthogonal provided they correspond to different eigenvalues, eigenvectors of AA^T that correspond to the 0 eigenvalue don't need to be orthogonal: in this case the Gram-Schmidt orthogonalization process is used to complete the two sets.

4.1 Singular Value Decomposition of a 2×2 matrix of rank 2

In this section we present our first example of SVD applied to a 2×2 real matrix of rank 2. The purpose of this example is just to show step by step how finding the SVD of a matrix can be done in the Clifford algebra language. Reader is encouraged to perform these computations with CLIFFORD and an additional package asvd which is described in Appendix 3.

```
> A:=matrix(2,2,[2,3,1,2]);#defining A
> m:=rowdim(A): #number of rows of A is m
> n:=coldim(A): #number of columns of A is n
```

$$A := \begin{bmatrix} 2 & 3 \\ 1 & 2 \end{bmatrix}$$

Since $A \in \mathrm{Mat}(2, \mathsf{R})$, we need to find (p, q) such that $C\ell_{p,q} \overset{\varphi}{\simeq} \mathrm{Mat}(2, \mathsf{R})$. As shown next, we have two choices for the signature:

```
> all_sigs(2..2,real,simple);
```

$$[[1,\ 1],\ [2,\ 0]]$$

Thus, we can pick either $C\ell_{1,1}$ or $C\ell_{2,0}$. Our choice is $C\ell_{2,0}$. We define a bilinear form $B = \mathrm{diag}(1, 1)$ and display information about $C\ell_{2,0}$.

```
> dim:=2:B:=diag(1,1):eval(makealiases(dim)):
> data:=clidata();
```

$$data := [real,\ 2,\ simple,\ \frac{1}{2}\,Id + \frac{1}{2}\,e1,\ [Id,\ e2],\ [Id],\ [Id,\ e2]]$$

The above output means that $C\ell_{2,0}$ is a simple algebra isomorphic with $\mathrm{Mat}(2, \mathsf{R})$; that the element $\frac{1}{2} + \frac{1}{2}\mathbf{e}_1$ is a primitive idempotent which we will call f; that the list $[Id, e2]$ shown as the fourth entry displays generators of a minimal left-ideal $C\ell_{2,0}f$ considered as vector space over R; that the division ring $\mathsf{K} = fC\ell_{2,0}f =< Id >_{\mathsf{R}} \simeq \mathsf{R}$; and that the last list $[Id, e2]$ gives generators of $C\ell_{2,0}f$ over K, and since $\mathsf{K} \simeq \mathsf{R}$, it is the same as the fourth list.[13]. In the following, we define a Grassmann basis in $C\ell_{2,0}$, assign the primitive idempotent to f, and generate a spinor basis in $C\ell_{2,0}f$.

[13] For more information see [5] and CLIFFORD's help pages.

```
> clibas:=cbasis(dim); #ordered basis in Cl(2,0)
```

$$clibas := [Id,\ e1,\ e2,\ e12]$$

```
> f:=data[4]:#a primitive idempotent in Cl(2,0)
> SBgens:=data[5]:#generators for a real basis in S
> FBgens:=data[6]:#generators for K
```

Here `SBgens` is a K-basis for $S = C\ell_{2,0} f$. Since for the signature $(2,0)$ we have $\mathsf{K} \simeq \mathsf{R}$, $S \simeq \mathsf{R}^2$, and $C\ell_{2,0} \simeq \mathrm{Mat}(2,\mathsf{R})$, the output from the procedure `spinorKbasis` shown below has two basis elements and their generators modulo f :

```
> Kbasis:=spinorKbasis(SBgens,f,FBgens,'left');
```

$$Kbasis := [[\frac{1}{2}\, Id + \frac{1}{2}\, e1,\ \frac{1}{2}\, e2 - \frac{1}{2}\, e12],\ [Id,\ e2],\ left]$$

Thus, the real spinor basis in S consists of the following two polynomials:

```
> for i from 1 to nops(Kbasis[1]) do f.i:=Kbasis[1][i] od;
```

$$f1 := \frac{1}{2}\, Id + \frac{1}{2}\, e1, \quad f2 := \frac{1}{2}\, e2 - \frac{1}{2}\, e12.$$

Now, we compute matrices M_1, M_2, M_3, M_4 representing each of the four basis elements $\{\mathbf{1}, \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_{12}\}$ in $C\ell_{2,0}$.[14]

```
> for i from 1 to nops(clibas) do M[i]:=subs(Id=1,matKrepr(clibas[i])) od:
```

We will use a procedure `phi` which gives the isomorphism φ from $\mathrm{Mat}(2,\mathsf{R})$ to $C\ell_{2,0}$. This way we can find the image p in $C\ell_{2,0}$ of any real 2×2 real matrix A. Knowing the image $\varphi(M_i)$ of each matrix M_i in terms of some Clifford polynomial in $C\ell_{2,0}$, we can easily find the image $p = \varphi(A)$ of A as follows:[15]

```
> p:=phi(A,M); #finding image of A in Cl(2,0)
```

$$p := 2\, Id + 2\, e2 + e12$$

```
> pT:=phi(t(A),M); #finding image of t(A) in Cl(2,0)
```

$$pT := 2\, Id + 2\, e2 - e12$$

Next, we compute a symmetric matrix $A^T A$ (denoted in Maple as `ATA`), its characteristic polynomial, eigenvalues, and its orthonormal eigenvectors v_1, v_2. Vectors v_1 and v_2 will become columns of an orthogonal matrix V needed for SVD of A[16]:

```
> ATA:=evalm(t(A) &* A); #finding matrix ATA
```

$$ATA := \begin{bmatrix} 5 & 8 \\ 8 & 13 \end{bmatrix}$$

```
> pTp:=phi(ATA,M); #finding image of ATA in Cl(2,0)
```

$$pTp := 9\, Id - 4\, e1 + 8\, e2$$

which should be the same as

```
> 'pTp'=cmul(pT,p);
```

$$pTp = 9\, Id - 4\, e1 + 8\, e2$$

The minimum polynomial of $A^T A$ (or $pTp = \varphi(A^T A)$) is:

```
> climinpoly(pTp);
```

[14] Since a similar computation was done in [4, 5], we won't display the matrices.

[15] From now on, as use Maple `alias(t=transpose)`, that is, `t(A)` denotes the matrix transposition in Maple.

[16] We use procedure `radsimplify` to simplify radicals in matrices and vectors.

$$x^2 - 18\,x + 1$$

and, in this case, it is the same as the characteristic polynomial of $A^T A$:

```
> pol:=charpoly(ATA,x);#characteristic polynomial of ATA
```

$$pol := x^2 - 18\,x + 1$$

In order to find eigenvalues and eigenvectors of $A^T A$, we will use Maple's procedure `eigenvects` modified by our own sorting via a new procedure `assignL`. The latter displays a list containing two lists: one has the eigenvalues while the second has the eigenvectors.[17] In the following, we will assign the eigenvalues to λ_1, λ_2 and the (un-normalized, but orthogonal) eigenvectors we assign to v_1, v_2.

```
> P:=assignL(sort([eigenvects(ATA)],byeigenvals));N:=P[1]:
```

$$P := [2,\ [9 + 4\sqrt{5},\ 9 - 4\sqrt{5}],\ [\left[1,\ \frac{1}{2} + \frac{1}{2}\sqrt{5}\right],\ \left[1,\ \frac{1}{2} - \frac{1}{2}\sqrt{5}\right]]]$$

```
> for i from 1 to N do lambda.i:=P[2][i];
> v.i:=map(simplify,normalize(P[3][i]))
> od:
```

We can now verify that vectors v_1, v_2 are eigenvectors of $A^T A$ with the eigenvalues λ_1, λ_2.

```
> for i from 1 to N do map(simplify,evalm(ATA &* v.i - lambda.i*v.i)) od;
```

$$[0,\ 0],\ [0,\ 0]$$

Similar verification can be done in $C\ell_{2,0}$ since one can view the 1-column eigenvectors v_1, v_2 as one-column spinors in S. We simply convert the two vectors to spinors sv_1, sv_2 which we express in the previously computed spinor basis f_1, f_2.

```
> spinorbasis:=[''f1'',''f2'']:
> for i from 1 to N do sv.i:=convert(v.i,spinor,spinorbasis) od;
```

$$sv1 := 2\,\frac{f1}{\sqrt{10 + 2\sqrt{5}}} + \frac{(1 + \sqrt{5})\,f2}{\sqrt{10 + 2\sqrt{5}}},\quad sv2 := 2\,\frac{f1}{\sqrt{10 - 2\sqrt{5}}} - \frac{(-1 + \sqrt{5})\,f2}{\sqrt{10 - 2\sqrt{5}}}.$$

Since v_1, v_2 are eigenvectors of ATA, spinors sv_1, sv_2 must be eigenspinors of $pTp = \varphi(A^T A)$.

```
> for i from 1 to N do simplify((pTp - lambda.i) &c sv.i) od;
```

$$0, 0$$

We are now in position to define the orthogonal matrix $V = [v_1|v_2]$.

```
> V:=radsimplify(augment(v.(1..N))); #defining matrix V
```

$$V := \begin{bmatrix} 2\,\dfrac{1}{\sqrt{10 + 2\sqrt{5}}} & 2\,\dfrac{1}{\sqrt{10 - 2\sqrt{5}}} \\ \dfrac{1 + \sqrt{5}}{\sqrt{10 + 2\sqrt{5}}} & \dfrac{-\sqrt{5} + 1}{\sqrt{10 - 2\sqrt{5}}} \end{bmatrix}$$

Since later we will need images of V and V^T under φ, we compute them now and store under the variables pV and pVt. The fact that V is orthogonal can be easily verified in the matrix language; in $C\ell_{2,0}$ it can be done as follows:

```
> simplify(cmul(pVt,pV));
```

$$Id$$

[17] The first entry 2 in the output is just the number of eigenvectors.

Now we repeat the above steps and apply them to AA^T. In the process, we will find its eigenvectors u_1, u_2. We must make sure that $Av_i = \sigma_i u_i$ where $\sigma_i = \sqrt{\lambda_i}$, $i = 1, 2$. This will require extra checking and possibly redefining of the u's.

```
> AAT:=evalm(A &* transpose(A)); #computing AAT
```

$$AAT := \begin{bmatrix} 13 & 8 \\ 8 & 5 \end{bmatrix}$$

The image of AA^T under φ in $C\ell_{2,0}$ we denote as ppT.

```
> ppT:=phi(AAT,M); #finding image of AAT in Cl(2,0)
```

$$ppT := 9\,Id + 4\,e1 + 8\,e2$$

In this case, the minimal polynomial of ppT and the characteristic polynomial of AA^T are the same.

```
> pol2:=charpoly(AAT,lambda); #finding the characteristic polynomial of AAT
```

$$pol2 := \lambda^2 - 18\,\lambda + 1$$

```
> 'ppT'=climinpoly(ppT);
```

$$ppT = x^2 - 18\,x + 1$$

Since matrices A^TA and AA^T have the same characteristic polynomials, their eigenvalues will be the same. We define therefore the *singular values* σ_1 and σ_2 of A:

```
> for i from 1 to N do sigma.i:=sqrt(lambda.i) od;
```

$$\sigma 1 := \sqrt{5} + 2, \quad \sigma 2 := \sqrt{5} - 2$$

When we compute the eigenvectors u_1, u_2 of AA^T, we will not necessarily have $Av_i = \sigma_i u_i, i = 1, 2$. This is because the choice of u_1, u_2 is not consistent with the choice of v_1, v_2.

```
> P:=assignL(sort([eigenvects(AAT)],byeigenvals)):
> for i from 1 to N do lambda.i:=P[2][i];
> u.i:=map(simplify,normalize(P[3][i]))
> od:
```

However, $Av_1 = \sigma_1 u_1$ while $Av_2 = -\sigma_2 u_2$:

```
> radsimplify(evalm(A &* v1-sigma1*u1)); #this one checks out but
```

$$[0,\ 0]$$

```
> radsimplify(evalm(A &* v2-sigma2*u2)); #this one does not check out
> radsimplify(evalm(A &* v2+sigma2*u2)); #this one does check out
```

$$\left[\frac{14 - 6\sqrt{5}}{\sqrt{10 - 2\sqrt{5}}},\ \frac{8 - 4\sqrt{5}}{\sqrt{10 - 2\sqrt{5}}}\right]$$

$$[0,\ 0]$$

Notice that the set $\{u_1, u_2\}$ is orthonormal, but so is $\{u1, -u2\}$. Let's re-define u_2 as $-u_2$ and call it u_{22}. For completeness we rename u_1 as u_{11}:

```
> u11:=evalm(u1):u22:=evalm(-u2):
```

In the Clifford algebra $C\ell_{2,0}$, we need to perform similar computations with v_1, v_2. The images $\varphi(u_{11}), \varphi(u_{22})$ contained in the spinor ideal need to be found first. We call them su_1 and su_2.

```
> for i from 1 to N do su.i:=convert(u.i.i,spinor,spinorbasis) od;
```

$$su1 := \frac{(1+\sqrt{5})\,f1}{\sqrt{10+2\sqrt{5}}} + 2\,\frac{f2}{\sqrt{10+2\sqrt{5}}}, \quad su2 := \frac{(-1+\sqrt{5})\,f1}{\sqrt{10-2\sqrt{5}}} - 2\,\frac{f2}{\sqrt{10-2\sqrt{5}}}.$$

The verification of the condition (26) in $C\ell_{2,0}$ looks as follows:

```
> for i from 1 to N do simplify(p &c sv.i-sigma.i*su.i) od;
```

$$0,\ 0$$

Now we may define the orthogonal matrix $U = [u_{11}|u_{22}]$ and its image $\varphi(U)$ in $C\ell_{2,0}$ which we call pU :[18]

```
> U:=radsimplify(augment(u11,u22)); #defining matrix U
```

$$U := \begin{bmatrix} \dfrac{1+\sqrt{5}}{\sqrt{10+2\sqrt{5}}} & \dfrac{-1+\sqrt{5}}{\sqrt{10-2\sqrt{5}}} \\ 2\,\dfrac{1}{\sqrt{10+2\sqrt{5}}} & -2\,\dfrac{1}{\sqrt{10-2\sqrt{5}}} \end{bmatrix}$$

```
> pU:=phi(U,M);pUt:=phi(t(U),M):
```

$$pU := Id\,(\frac{1}{20}\,\%1\,\sqrt{5} - \frac{1}{8}\,\%2 - \frac{1}{40}\,\%2\,\sqrt{5}) + (\frac{1}{20}\,\%1\,\sqrt{5} + \frac{1}{8}\,\%2 + \frac{1}{40}\,\%2\,\sqrt{5})\,e1$$
$$+ (\frac{1}{20}\,\%2\,\sqrt{5} + \frac{1}{8}\,\%1 - \frac{1}{40}\,\%1\,\sqrt{5})\,e2 + (\frac{1}{20}\,\%2\,\sqrt{5} - \frac{1}{8}\,\%1 + \frac{1}{40}\,\%1\,\sqrt{5})\,e12$$
$$\%1 := \sqrt{10+2\sqrt{5}}$$
$$\%2 := \sqrt{10-2\sqrt{5}}$$

The fact that U is an orthogonal matrix can be easily now checked both in the matrix language and in the Clifford language:

```
> radsimplify(evalm(t(U) &* U));#U is an orthogonal matrix
```

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

```
> simplify(pUt &c pU);
```

$$Id$$

Finally, we define matrix Σ using a procedure `makediag`. Recall [28] that Σ has the same dimensions as the original matrix A and that $\Sigma^T\Sigma, \Sigma\Sigma^T$ are the diagonal forms of A^TA and AA^T respectively. In this example matrices $\Sigma^T\Sigma$ and $\Sigma\Sigma^T$ are the same since Σ is a square diagonal matrix. Normally these matrices are different although their nonzero "diagonal" entries are the same.

$$A^TA = V\Sigma^T\Sigma V^T, \quad AA^T = U\Sigma\Sigma^TU^T, \quad \Sigma = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}, \quad \Sigma^T\Sigma = \Sigma\Sigma^T = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}. \qquad (27)$$

```
> Sigma:=makediag(m,n,[seq(sigma.i,i=1..N)]);
> STS,SST:=evalm(t(Sigma) &* Sigma),evalm(Sigma &* t(Sigma));
```

$$\Sigma := \begin{bmatrix} \sqrt{5}+2 & 0 \\ 0 & \sqrt{5}-2 \end{bmatrix}$$

[18] For a later verification we will also need $pUt = \varphi(U^T)$. Expressions %1 and %2 showing up in the Maple output for pU are just place holders for $\sqrt{10+2\sqrt{5}}$ and $\sqrt{10-2\sqrt{5}}$ respectively as shown at the end of the display.

$$STS,\ SST := \begin{bmatrix} (\sqrt{5}+2)^2 & 0 \\ 0 & (\sqrt{5}-2)^2 \end{bmatrix}, \begin{bmatrix} (\sqrt{5}+2)^2 & 0 \\ 0 & (\sqrt{5}-2)^2 \end{bmatrix}$$

```
> pSigma,pSTS,pSST:=phi(Sigma,M),phi(STS,M,FBgens),phi(SST,M);
```

$$pSigma,\ pSTS,\ pSST := \sqrt{5}\,Id + 2\,e1,\ 9\,Id + 4\sqrt{5}\,e1,\ 9\,Id + 4\sqrt{5}\,e1$$

We should be able to verify in $C\ell_{2,0}$ the following two factorizations of AA^T and A^TA:

$$A^TA = V\Sigma^T\Sigma V^T \tag{28}$$
$$AA^T = U\Sigma\Sigma^T U^T \tag{29}$$

like this:

```
> evalb(pTp=simplify(pV &c pSTS &c pVt)), evalb(ppT=simplify(pU &c pSST &c pUt));
```

$$true,\ true$$

We check the SVD of A, which is $A = U\Sigma V^T$,[19] in the Clifford algebra language:

```
> evalb(p=simplify(pU &c pSigma &c pVt)),
```

$$true,$$

where

```
> 'pU'=pU;
```

$$pU = Id\,(\frac{1}{20}\,\%1\,\sqrt{5} - \frac{1}{8}\,\%2 - \frac{1}{40}\,\%2\,\sqrt{5}) + (\frac{1}{20}\,\%1\,\sqrt{5} + \frac{1}{8}\,\%2 + \frac{1}{40}\,\%2\,\sqrt{5})\,e1$$
$$+ (\frac{1}{20}\,\%2\,\sqrt{5} + \frac{1}{8}\,\%1 - \frac{1}{40}\,\%1\,\sqrt{5})\,e2 + (\frac{1}{20}\,\%2\,\sqrt{5} - \frac{1}{8}\,\%1 + \frac{1}{40}\,\%1\,\sqrt{5})\,e12$$
$$\%1 := \sqrt{10 + 2\sqrt{5}}$$
$$\%2 := \sqrt{10 - 2\sqrt{5}}$$

```
> 'pSigma'=pSigma;
```

$$pSigma = \sqrt{5}\,Id + 2\,e1$$

```
> 'pVt'=pVt;
```

$$pVt = Id\,(-\frac{1}{20}\,\%1\,\sqrt{5} + \frac{1}{8}\,\%2 - \frac{1}{40}\,\%2\,\sqrt{5}) + (\frac{1}{20}\,\%1\,\sqrt{5} + \frac{1}{8}\,\%2 - \frac{1}{40}\,\%2\,\sqrt{5})\,e1$$
$$+ (\frac{1}{20}\,\%2\,\sqrt{5} + \frac{1}{8}\,\%1 + \frac{1}{40}\,\%1\,\sqrt{5})\,e2 + (\frac{1}{20}\,\%2\,\sqrt{5} - \frac{1}{8}\,\%1 - \frac{1}{40}\,\%1\,\sqrt{5})\,e12$$
$$\%1 := \sqrt{10 - 2\sqrt{5}}$$
$$\%2 := \sqrt{10 + 2\sqrt{5}}$$

4.2 Singular Value Decomposition of a 3×2 matrix of rank 2

In this section we will show our second example of SVD applied to a non-square matrix. The matrix will need to be embedded first into an appropriate matrix algebra before its image can be found in a suitable Clifford algebra.

[19] The SVD of A is not unique: For example, $A = (-U)\Sigma(-V^T)$ is another such factorization.

```
> C:=matrix(3,2,[3,0,0,-1,0,1]);
> m:=rowdim(C): #number of rows of C is m
> n:=coldim(C): #number of columns of C is n
```

$$C := \begin{bmatrix} 3 & 0 \\ 0 & -1 \\ 0 & 1 \end{bmatrix}$$

Since our matrix C is 3×2, we will embed it into $\mathrm{Mat}(4, \mathsf{R}) \simeq C\ell_{p,q}$ where the signature (p, q) could be either $(2, 2)$ or $(3, 1)$. Since the symbolic spinor representation of $C\ell_{3,1}$ was already computed in [5], we will work with the signature $(3, 1)$.

```
> dim:=4:B:=diag(1$3,-1$1):eval(makealiases(dim)):
```

Let's recall information about $C\ell_{3,1}$ stored in CLIFFORD[20]:

```
> data:=clidata();
```

$$data := [real,\ 4,\ simple,\ 'cmulQ'(\frac{1}{2}\ Id + \frac{1}{2}\ e1,\ \frac{1}{2}\ Id + \frac{1}{2}\ e34),\ [Id,\ e2,\ e3,\ e23],\ [Id],$$
$$[Id,\ e2,\ e3,\ e23]]$$

We begin by defining a Grassmann basis in $C\ell_{3,1}$, by assigning the fourth entry in the list to a primitive idempotent f, and by generating a spinor basis in $S = C\ell_{3,1}f$.

```
> clibas:=cbasis(dim):  #ordered basis in Cl(3,1)
> f:=data[4]; #a primitive idempotent in Cl(3,1)
> SBgens:=data[5]; #generators for a real basis in S
> N:=data[2]:  #dimension of the spinor representation
```

$$f := \mathrm{cmulQ}(\frac{1}{2}\ Id + \frac{1}{2}\ e1,\ \frac{1}{2}\ Id + \frac{1}{2}\ e34)$$

$$SBgens := [Id,\ e2,\ e3,\ e23]$$

Thus, the real spinor basis in $S = C\ell_{3,1}f$ consists of the following four polynomials:

```
> for i from 1 to N do f.i:=SBgens[i] &c f od;
```

$$f1 := \frac{1}{4}\ Id + \frac{1}{4}\ e34 + \frac{1}{4}\ e1 + \frac{1}{4}\ e134, \quad f2 := \frac{1}{4}\ e2 + \frac{1}{4}\ e234 - \frac{1}{4}\ e12 - \frac{1}{4}\ e1234$$

$$f3 := \frac{1}{4}\ e3 + \frac{1}{4}\ e4 - \frac{1}{4}\ e13 - \frac{1}{4}\ e14, \quad f4 := \frac{1}{4}\ e23 + \frac{1}{4}\ e24 + \frac{1}{4}\ e123 + \frac{1}{4}\ e124$$

We compute matrices $M_1, \ldots, M_{16}$ representing each basis element in $C\ell_{3,1}$. These matrices can be computed also with the help of a procedure `matKrepr` which, being less general than `spinorKrepr` used earlier, is also simpler to use. We won't display these matrices since they can be found in [5].

```
> for i from 1 to nops(clibas) do M[i]:=subs(Id=1,matKrepr(clibas[i]))od:
```

Before we can use the procedure `phi` that realizes the isomorphism $\mathrm{Mat}(4, \mathsf{R}) \overset{\varphi}{\simeq} C\ell_{3,1}$, we need to embed matrix C into $\mathrm{Mat}(4, \mathsf{R})$. This is accomplished with a procedure `embed`.

```
> A:=embed(C);m1:=rowdim(A):n1:=coldim(A):
```

[20] In the following display, $'cmulQ'(\frac{1}{2} * Id + \frac{1}{2} * e1, \frac{1}{2} * Id + \frac{1}{2} * e34)$ denotes an unevaluated product of two non-primitive idempotents $\frac{1}{2} * Id + \frac{1}{2} * e1$ and $\frac{1}{2} * Id + \frac{1}{2} * e34$ in $C\ell_{3,1}$, and `cmulQ` is a name of a simplified version of the procedure `cmul`. While `cmul` gives the Clifford product in $C\ell(B)$, `cmulQ` gives the Clifford product in $C\ell(Q)$.

$$A := \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

```
> p:=phi(A,M); #finding image of A (and C) in Cl(3,1)
> pT:=phi(t(A),M); #finding image of AT (and CT) in Cl(3,1)
```

$$p := \frac{1}{2}\,Id + e1 - \frac{1}{4}\,e23 - \frac{1}{4}\,e24 + \frac{1}{2}\,e34 + \frac{1}{4}\,e123 + \frac{1}{4}\,e124 + e134$$

$$pT := \frac{1}{2}\,Id + e1 + \frac{1}{4}\,e23 - \frac{1}{4}\,e24 + \frac{1}{2}\,e34 - \frac{1}{4}\,e123 + \frac{1}{4}\,e124 + e134$$

Next we compute A^TA, AA^T, their images in the Clifford algebra $C\ell_{3,1}$ under φ, their eigenvalues, and orthonormal eigenvectors v_i and u_i, $i = 1, \ldots, 4$, which will become columns of two orthogonal matrices V and U.

```
> ATA,AAT:=evalm(t(A) &* A),evalm(A &* t(A));
```

$$ATA,\ AAT := \begin{bmatrix} 9 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 9 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

```
> pTp,ppT:=phi(ATA,M),phi(AAT,M); #finding images of ATA and AAT in Cl(3,1)
```

$$pTp,\ ppT :=$$
$$\frac{11}{4}\,Id + \frac{7}{4}\,e1 + \frac{11}{4}\,e34 + \frac{7}{4}\,e134,\ \frac{11}{4}\,Id + \frac{7}{4}\,e1 + \frac{1}{2}\,e24 + \frac{9}{4}\,e34 - \frac{1}{2}\,e124 + \frac{9}{4}\,e134$$

The characteristic polynomial of A^TA and AA^T is $(x-9)(x-2)x^2$ while the minimal polynomial of pTp and ppT is $x(x-2)(x-9)$.

```
> P1:=assignL(sort([eigenvects(ATA)],byeigenvals));
> P2:=assignL(sort([eigenvects(AAT)],byeigenvals));
> for i from 1 to N do lambda.i:=P1[2][i];
> v.i:=map(simplify,normalize(P1[3][i]));
> u.i:=map(simplify,normalize(P2[3][i]));
> od:
```

$$P1 := [4,\ [9,\ 2,\ 0,\ 0],\ [[1,\ 0,\ 0,\ 0],\ [0,\ 1,\ 0,\ 0],\ [0,\ 0,\ 1,\ 0],\ [0,\ 0,\ 0,\ 1]]]$$

$$P2 := [4,\ [9,\ 2,\ 0,\ 0],\ [[1,\ 0,\ 0,\ 0],\ [0,\ -1,\ 1,\ 0],\ [0,\ 1,\ 1,\ 0],\ [0,\ 0,\ 0,\ 1]]]$$

The third list in $P1$ and the third list in $P2$ contain vectors v_i and u_i that will make up matrices V and U respectively. Clearly, since V is the 4×4 identity matrix, $\varphi(V) = \varphi(V^T) = \mathbf{1}$. In order to translate this matrix picture $C\ell_{3,1}$, we need to find spinors $sv_i = \varphi(v_i)$ and $su_i = \varphi(u_i)$ in $S = C\ell_{3,1}f$.

```
> spinorbasis:=[''f1'',''f2'',''f3'',''f4'']:
> for i from 1 to N do sv.i:=convert(v.i,spinor,spinorbasis);
> su.i:=convert(u.i,spinor,spinorbasis) od;
```

$$sv1 := f1,\ sv2 := f2,\ sv3 := f3,\ sv4 := f4$$

$$su1 := f1,\ su2 := -\frac{1}{2}\sqrt{2}\,f2 + \frac{1}{2}\sqrt{2}\,f3,\ su3 := \frac{1}{2}\sqrt{2}\,f2 + \frac{1}{2}\sqrt{2}\,f3,\ sv4 := f4$$

Spinors sv_i are eigenspinors of pTp, while su_i are eigenspinors of ppT with the eigenvalues $\lambda_1 = 9,\ \lambda_2 = 2, \lambda_3 = 0, \lambda_4 = 0,$ namely:

```
> for i from 1 to N do simplify((pTp - lambda.i) &c sv.i);
> simplify((ppT - lambda.i) &c su.i) od;
```

$$0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0$$

```
> V:=radsimplify(augment(v.(1..N))): #defining matrix V (identity matrix)
> pV,pVt:=phi(V,M),phi(t(V),M): #finding images of V and t(V) in Cl(3,1)
```

We check that $Av_i = \sigma_i u_i$ where $\sigma_i = \sqrt{\lambda_i}$ by verifying this fact in $C\ell_{3,1}$.[21]

```
> for i from 1 to N do sigma.i:=sqrt(lambda.i) od;
```

$$\sigma 1 := 3,\ \sigma 2 := \sqrt{2},\ \sigma 3 := 0,\ \sigma 4 := 0$$

```
> for i from 1 to N do simplify(p &c sv.i-sigma.i*su.i) od;
```

$$0,\ 0,\ 0,\ 0$$

Thus, we may now define an orthogonal matrix $U = [u_1|u_2|u_3|u_4]$ and find its image $\varphi(U)$ and the image of its transpose in $C\ell_{3,1}$:

```
> U:=radsimplify(augment(seq(u.i,i=1..N))); #defining matrix U
```

$$U := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -\frac{1}{2}\sqrt{2} & \frac{1}{2}\sqrt{2} & 0 \\ 0 & \frac{1}{2}\sqrt{2} & \frac{1}{2}\sqrt{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
> pU:=phi(U,M);pUt:=phi(t(U),M);
```

$$pU := \frac{1}{2}\,Id + \frac{1}{2}\,e1 - \frac{1}{4}\sqrt{2}\,e24 - \frac{1}{4}\sqrt{2}\,e34 + \frac{1}{4}\sqrt{2}\,e124 + \frac{1}{4}\sqrt{2}\,e134$$

$$pUt := \frac{1}{2}\,Id + \frac{1}{2}\,e1 - \frac{1}{4}\sqrt{2}\,e24 - \frac{1}{4}\sqrt{2}\,e34 + \frac{1}{4}\sqrt{2}\,e124 + \frac{1}{4}\sqrt{2}\,e134$$

The fact that U is orthogonal is reflected in the following:

```
> pUt &c pU;
```

$$Id$$

Finally, it is just enough to find matrix Σ and verify SVD for $\varphi(A)$ in the Clifford algebra $C\ell_{3,1}$.

```
> Sigma:=makediag(m1,n1,[seq(sigma.i,i=1..N)]);
> pSigma:=phi(Sigma,M);
```

$$\Sigma := \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & \sqrt{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$pSigma := Id\,(\frac{3}{4} + \frac{1}{4}\sqrt{2}) + (-\frac{1}{4}\sqrt{2} + \frac{3}{4})\,e1 + (\frac{3}{4} + \frac{1}{4}\sqrt{2})\,e34 + (-\frac{1}{4}\sqrt{2} + \frac{3}{4})\,e134$$

```
> evalb(p=pU &c pSigma &c pVt); #SVD of phi(A)
```

$$true$$

[21] As before, we are following engineering practice [25] of considering square roots of zero eigenvalues of AA^T and A^TA also as the singular values of a matrix. While it is convenient to do so for computational reasons, it is not conceptually correct since by the definition singular values of a matrix are always positive.

Since the original matrix C was 3 by 2 and not 4 by 4, in order to find SVD of C we need to project out certain columns and rows out of the matrices $U, \Sigma,$ and V. This can also be done internally in the Clifford algebra. The original matrix C has therefore this factorization:

```
> U2,Sigma2,V2t:=submatrix(U,1..m,1..m),submatrix(Sigma,1..m,1..n),
> submatrix(t(V),1..n,1..n);
```

$$U2,\ \Sigma 2,\ V2t := \begin{bmatrix} 1 & 0 & 0 \\ 0 & -\frac{1}{2}\sqrt{2} & \frac{1}{2}\sqrt{2} \\ 0 & \frac{1}{2}\sqrt{2} & \frac{1}{2}\sqrt{2} \end{bmatrix}, \begin{bmatrix} 3 & 0 \\ 0 & \sqrt{2} \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

```
> evalm(C)=radsimplify(evalm(U2 &* Sigma2 &* V2t));
```

$$\begin{bmatrix} 3 & 0 \\ 0 & -1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 3 & 0 \\ 0 & -1 \\ 0 & 1 \end{bmatrix}$$

4.3 Additional comments

In this section we have shown that it is possible to translate the matrix algebra picture of the Singular Value Decomposition of a matrix A into the Clifford algebra language. Although we have not abandoned entirely the linear algebra formalism in our examples, e.g., we have computed the eigenvalues and the eigenvectors of $A^T A$ and AA^T, and only then we have found images of the eigenvectors in the spinor space $C\ell_{p,q} f$, these computations including solving the eigenvalue problems can be done entirely in $C\ell_{p,q}$ without using matrices. For example, if we consider element $pTp = \varphi(A^T A)$ in $C\ell_{3,1}$ from the last example, we can find its eigenvalues from its minimal polynomial while its eigenvectors can be found by solving the eigenvalue equation. For example, we know that the minimal polynomial of pTp is

```
> pol:=factor(climinpoly(pTp));
```

$$pol := x\,(x-2)\,(x-9)$$

hence the eigenvalues of pTp are $9, 2, 0$. Of course, the minimal polynomial doesn't give us their geometric multiplicities. However, we can find them by solving the eigenvalue equation directly in $C\ell_{3,1}$ for each of the eigenvalues. Let's assign these three known eigenvalues to $\lambda_1, \lambda_2, \lambda_3$:

```
> lambda1,lambda2,lambda3:=9,2,0;
```

$$\lambda 1,\ \lambda 2,\ \lambda 3 := 9,\ 2,\ 0$$

Let $\psi \in C\ell_{3,1} f$ be an arbitrary spinor expressed in the spinor basis $\{f_1, f_2, f_3, f_4\}$ computed earlier, and let $c_1, \ldots, c_4$ be its undefined (real) coefficients.

```
> psi:=c[1]*'f1'+c[2]*'f2'+c[3]*'f3'+c[4]*'f4';
```

$$\psi := c_1\, f1 + c_2\, f2 + c_3\, f3 + c_4\, f4$$

We will now use the procedure `clisolve2` from the `suppl` package (see Section 2) to solve the eigenvalue equation

$$pTp\,\psi = \lambda\psi. \tag{30}$$

First we solve it when $\lambda = \lambda_1 = 9$.

```
> sol1:=clisolve2(pTp &c psi - lambda1*psi,[c[1],c[2],c[3],c[4]]);
```

$$sol1 := [\{c_2 = 0,\ c_3 = 0,\ c_4 = 0,\ c_1 = c_1\}]$$

```
> psi1:=c[1]*'f1'; #eigenspinor of pTp with eigenvalue lambda1=9
```

$$\psi 1 := c_1\, f1$$

Thus, the first solution to (30) for $\lambda_1 = 9$ is a one-parameter solution that belongs to a one-dimensional subspace spanned by f_1. That is, the geometric multiplicity of the eigenvalue $\lambda_1 = 9$ is 1. Similarly for $\lambda = \lambda_2 = 2$:

```
> sol2:=clisolve2(pTp &c psi - lambda2*psi,[c[1],c[2],c[3],c[4]]);
```

$$sol2 := [\{c_3 = 0,\, c_4 = 0,\, c_1 = 0,\, c_2 = c_2\}]$$

```
> psi2:=c[2]*'f2'; #eigenspinor of pTp with eigenvalue lambda2=2
```

$$\psi 2 := c_2\, f2$$

The second solution to (30) for $\lambda_2 = 2$ is also a one-parameter solution that belongs to a one-dimensional subspace spanned by f_2. The geometric multiplicity of the eigenvalue $\lambda_2 = 2$ is also 1.

```
> sol34:=clisolve2(pTp &c psi - lambda3*psi,[c[1],c[2],c[3],c[4]]);
```

$$sol34 := [\{c_3 = c_3,\, c_4 = c_4,\, c_2 = 0,\, c_1 = 0\}]$$

The third solution to (30) for $\lambda = \lambda_3 = 0$ is parameterized by two parameters c_3 and c_4 and belongs therefore to a two-dimensional subspace of S spanned by $\{f_3, f_4\}$. This implies that the geometric multiplicity of $\lambda = \lambda 3 = 0$ is two.

```
> psi3:=c[3]*'f3'; #eigenspinor of pTp with eigenvalue lambda3=0
```

$$\psi 3 := c_3\, f3$$

```
> psi4:=c[4]*'f4'; #eigenspinor of pTp with eigenvalue lambda3=0
```

$$\psi 4 := c_4\, f4$$

So, up to a normalizing scalar, the eigenspinors $\psi_1, \psi_2, \psi_3, \psi_4$ give previously computed normalized spinors sv_1, sv_2, sv_3, sv_4. Likewise for the image ppT of AA^T :

```
> sol1:=clisolve2(ppT &c psi - lambda1*psi,[c[1],c[2],c[3],c[4]]);
```

$$sol1 := [\{c_2 = 0,\, c_3 = 0,\, c_4 = 0,\, c_1 = c_1\}]$$

```
> phi1:=c[1]*'f1'; #eigenspinor of ppT with eigenvalue lambda1=9
```

$$\phi 1 := c_1\, f1$$

The first one-parameter eigenspace of ppT corresponding to the eigenvalue $\lambda_1 = 9$ is one-dimensional and it is spanned by f_1.

```
> sol2:=clisolve2(ppT &c psi - lambda2*psi,[c[1],c[2],c[3],c[4]]);
```

$$sol2 := [\{c_3 = c_3,\, c_4 = 0,\, c_1 = 0,\, c_2 = -c_3\}]$$

```
> phi2:=-c[3]*'f2'+c[3]*'f3'; #eigenspinor of ppT with eigenvalue lambda2=2
```

$$\phi 2 := -c_3\, f2 + c_3\, f3$$

The second one-parameter eigenspace of ppT corresponding to the eigenvalue $\lambda_2 = 0$ is also one-dimensional and it is spanned by $\{f_2, f_3\}$.

```
> sol34:=clisolve2(ppT &c psi - lambda3*psi,[c[1],c[2],c[3],c[4]]);
```

$$sol34 := [\{c_3 = c_3,\, c_4 = c_4,\, c_1 = 0,\, c_2 = c_3\}]$$

The third solution to (30) is parameterized by two parameters c_3 and c_4. Thus, we get a two dimensional vectors space spanned by $\{f_3, f_4\}$.

```
> phi3:=c[3]*'f2'+c[3]*'f3'; #eigenspinor of ppT with eigenvalue lambda3=0
```

$$\phi 3 := c_3\, f2 + c_3\, f3$$

```
> phi4:=c[4]*'f4'; #eigenspinor of ppT with eigenvalue lambda3=0
```

$$\phi 4 := c_4\, f4$$

So, again up to a normalizing scalar, $\phi_1, \phi_2, \phi_3, \phi_4$ give the eigenspinors su_1, su_2, su_3, su_4 computed earlier.

5 Clifford algebras in robotics

Clifford algebras $C\ell(V,Q)$ on a quadratic space (V,Q) endowed with a *degenerate* quadratic form Q and associated groups **Spin**, **Pin**, Clifford, etc., were studied in [3, 8] and [2, 12, 13]. In contrast to the Clifford algebras of a non-degenerate quadratic form, these algebras possess a non-trivial two-sided nilpotent ideal called *Jacobson radical*. The Jacobson radical J is generated by the null-vectors in V which are orthogonal to the entire space V (that is, J is generated by the orthogonal complement $V^\perp$ of V). It is known [15, 16] that J contains every nilpotent left and right ideal in $C\ell(V,Q)$. From the point of view of the spinorial representation theory of Clifford algebras used in Section 4, an important difference is that $C\ell(V,Q)$ does not possess faithful matrix representation when Q is degenerate.

Let $V = V' \perp V^\perp$ where V' is endowed with a non-degenerate part Q' of Q of signature (p,q). Let $\dim(V^\perp) = d$, hence $p+q+d = \dim(V)$. Let's denote $C\ell(Q)$ as $C\ell_{d,p,q}$. Then we have a direct sum decomposition $C\ell_{d,p,q} = C\ell_{p,q} \oplus J$ into $C\ell_{p,q}$-modules. It was shown in [2] that when $d = 1$ this decomposition is responsible for a semi-direct product structure of the group of units $C\ell^*_{1,p,q}$ of $C\ell_{1,p,q}$ and of all of its subgroups such as the *Clifford group* $\Gamma(1,p,k)$ and the *special Clifford groups* $\Gamma^\pm(1,p,k) = \Gamma(1,p,k) \cap C\ell^\pm_{1,p,q}$, where $C\ell^+_{1,p,q}$ (resp. $C\ell^-_{1,p,q}$) denotes the even (resp. odd) part of $C\ell_{1,p,q}$. The Clifford group was defined as $\Gamma(1,p,k) = \{g \in C\ell^*_{1,p,q} \,|\, gvg^{-1} \in V, v \in V\}$, that is, without a twist.[22] Let $N : \Gamma(1,p,k) \to C\ell^*_{1,p,k}$ be defined as $N(g) = \tilde{g}g$. Then we define the *reduced Clifford groups* as $\Gamma^\pm_0(1,p,k) = \ker N \cap \Gamma^\pm(1,p,k)$. The $\mathbf{Pin}(1,p,q)$ and $\mathbf{Spin}(1,p,k)$ groups are then:

$$\mathbf{Pin}(1,p,q) = \{g \in \Gamma(1,p,q) \,|\, N(g) = \pm 1\}, \quad \mathbf{Spin}(1,p,q) = \{g \in \Gamma^+(1,p,q) \,|\, N(g) = \pm 1\}. \tag{31}$$

In preparation for our computations below, from now on we assume that $p+q$ is an odd positive integer. Let $G = \{1 + v\mathbf{e}_1 | v \in V', \mathbf{e}_1^2 = 0\}$ be a subgroup of $C\ell^*_{1,p,q}$. Then it was proven in [2] that

$$\mathbf{Pin}(1,p,q) = G \,\mathsf{o}\, \Gamma^\pm_0(p,k), \quad \mathbf{Spin}(1,p,q) = G \,\mathsf{o}\, \mathbf{Spin}(p,k). \tag{32}$$

In the above, symbol o denotes a semi-direct product with the group on the right acting on the group on the left. For example, it will be of interest to us to note that the homogeneous Galilei group of rigid motions $G_6 = \mathsf{R}^3 \,\mathsf{o}\, \mathrm{SO}(3)$ in R^3 is isomorphic to $\mathrm{SO}^+(1,0,3)$ and it is doubly covered by $\mathbf{Spin}^+(1,0,3)$, the identity component of $\mathbf{Spin}(1,0,3)$. For a similar result to (32) when one considers the twisted Clifford group $\Gamma_\alpha(1,p,k)$ and a twisted map $N_\alpha : \Gamma_\alpha(1,p,k) \to C\ell^*_{1,p,k}$ defined as $N_\alpha(g) = \bar{g}g$ where $^-$ denotes the conjugation in $C\ell_{1,p,q}$, see [12, 26].

[22] See Crumeyrolle [14] for a definition of the Clifford group with the twist given by α : in Crumeyrolle's notation α denotes the principal automorphism or the grade involution in $C\ell(Q)$. Then the *twisted Clifford group* is defined as $\Gamma_\alpha(1,p,k) = \{g \in C\ell^*_{1,p,q} \,|\, \alpha(g)vg^{-1} \in V, v \in V\}$.

In the following two sections we will use approach and notation from Selig [27] where the author denotes the degenerate Clifford algebra $C\ell_{d,p,q}$ as $C(p,q,d)$. Furthermore Selig uses twisted groups and defines the **Pin** and **Spin** groups as follows:

$$\mathbf{Pin}(n) = \{\mathbf{g} \in C(0,n,0) : \mathbf{g}\mathbf{g}^* = \mathbf{1} \text{ and } \alpha(\mathbf{g})\mathbf{x}\mathbf{g}^* \in V \text{ for all } \mathbf{x} \in V\}, \quad (33)$$

$$\mathbf{Spin}(n) = \{\mathbf{g} \in C^+(0,n,0) : \mathbf{g}\mathbf{g}^* = \mathbf{1} \text{ and } \mathbf{g}\mathbf{x}\mathbf{g}^* \in V \text{ for all } \mathbf{x} \in V\}, \quad (34)$$

with * denoting the conjugation $^-$ in the Clifford algebra $C(p,q,d)$. It is implicit in the definitions above that the actions of **Pin** and **Spin** on V are $\mathbf{x} \mapsto \alpha(\mathbf{g})\mathbf{x}\mathbf{g}^*$ and $\mathbf{x} \mapsto \mathbf{g}\mathbf{x}\mathbf{g}^*$ respectively.

5.1 Group Pin(3)

In this section we will perform some computations with **Pin**(3). In particular, we will find all possible forms of the elements in **Pin**(3) and verify some facts about that group. We begin by assigning a diagonal matrix to the bilinear form B. Grassmann basis for $C\ell_{0,3}$ will be stored in the variable `clibas`. Following Selig we re-name Clifford conjugation as a procedure `star` and define a Euclidean norm on $V = \mathsf{R}^3$ as a procedure `Enorm`. We will also define some additional Maple procedures that will be useful below.

```
> B:=diag(-1$3);eval(makealiases(3)):clibas:=cbasis(3);
```

$$B := \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

$$clibas := [Id,\ e1,\ e2,\ e3,\ e12,\ e13,\ e23,\ e123]$$

```
> star:=proc(x) conjugation(x) end:  #star (conjugation) operation in Cl(0,3)
> Enorm:=v->simplify(scalarpart(v &c star(v))):  #(pseudo)Euclidean norm in V
> alpha:=proc(x) gradeinv(x) end:  #alpha (grade involution) operation in Cl(0,3)
> scalarprod:=(x,y)->scalarpart(1/2*(x &c star(y) + star(y) &c x)):  #scalar product
> Pin_action:=(x,g)->clicollect(simplify(alpha(g) &c x &c star(g)));#action of
Pin(3)
```

$$Pin\_action := (x,\ g) \to \mathrm{clicollect}(\mathrm{simplify}((\alpha(g)\ \text{`\&c`}\ x)\ \text{`\&c`}\ \mathrm{star}(g)))$$

```
> Spin_action:=(x,g)->clicollect(simplify(g &c x &c star(g)));#action of Spin(3)
```

$$Spin\_action := (x,\ g) \to \mathrm{simplify}((g\ \text{`\&c`}\ x)\ \text{`\&c`}\ \mathrm{star}(g))$$

Let v, v_1, v_2 be three arbitrary vectors in R^3 with some undetermined coefficients expressed in a pseudo-orthonormal basis $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$:

```
> v:=c1*e1+c2*e2+c3*e3:v1:=c11*e1+c12*e2+c13*e3:v2:=c21*e1+c22*e2+c23*e3:
```

Then the Euclidean norm in R^3 is:

```
> Enorm(v);
```

$$c1^2 + c2^2 + c3^2$$

The action of **Pin** on $C\ell_{0,3}$ is realized as the procedure `Pin_action` defined above. Let's verify Selig's claim ([27], page 153) that when $\mathbf{x}, \mathbf{g}$ are both in V, then $\mathbf{g}\mathbf{x}\mathbf{g}^*$ automatically belongs to V :

```
> Pin_action(v,v1);
```

$$(c11^2\ c3 - c13^2\ c3 - 2\ c13\ c12\ c2 + c12^2\ c3 - 2\ c13\ c11\ c1)\ e3$$
$$+ (-c11^2\ c1 - 2\ c11\ c13\ c3 + c12^2\ c1 + c13^2\ c1 - 2\ c11\ c12\ c2)\ e1$$
$$+ (c11^2\ c2 - 2\ c12\ c11\ c1 - 2\ c12\ c13\ c3 + c13^2\ c2 - c12^2\ c2)\ e2$$

As we can see from the above, the output of `Pin_action(v,v1)` belongs to V. In order to check that indeed the action of **Pin** in V preserves the scalar product, we will first find all possible forms of $\mathbf{g} \in \mathbf{Pin}(3)$. Recall that according to (33) any element $\mathbf{g} \in \mathbf{Pin}(n)$ must satisfy two conditions: (1) $\mathbf{g}\mathbf{g}^* = \mathbf{1}$ and (2) $\alpha(\mathbf{g})\mathbf{v}\mathbf{g}^* \in V$ for any $\mathbf{v} \in V$. Suppose that $\mathbf{g}$ is an arbitrary element in $C\ell_{0,3,0}$ expressed in CLIFFORD in terms of the Grassmann basis $\{\mathbf{1}, \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_{12}\}$ [23]

```
> g:=add(x.i * clibas[i],i=1..nops(clibas)); #a general element in Cl(0,3,0)
```

$$g := x1\ Id + x2\ e1 + x3\ e2 + x4\ e3 + x5\ e12 + x6\ e13 + x7\ e23 + x8\ e123$$

We will now attempt to find conditions that the coefficients $x_i, i = 1, \ldots, 8,$ must satisfy so that $\mathbf{g}\mathbf{g}^* = \mathbf{1}$. We will again use the command `clisolve2`. In order to shorten its outputs, additional aliases $\kappa_j, j = 1, \ldots, 5,$ need to be defined (see Appendix 2).

The first condition (1) gives:

```
> sol:=clisolve2(cmul(g,star(g))-Id,[x.(1..8)]);
```

$$sol := [\{x1 = -\frac{-x7\ \kappa5 - x5\ x4 + x6\ x3}{x8},\ x8 = x8,\ x4 = x4,\ x7 = x7,\ x6 = x6,\ x5 = x5,\ x2 = \kappa5,$$
$$x3 = x3\}, \{x8 = 0,\ x1 = \frac{\kappa4}{x7},\ x4 = x4,\ x7 = x7,\ x6 = x6,\ x5 = x5,\ x3 = x3,\ x2 = \frac{-x5\ x4 + x6\ x3}{x7}\},$$
$$\{x7 = 0,\ x8 = 0,\ x1 = \frac{\kappa3}{x6},\ x3 = \frac{x4\ x5}{x6},\ x2 = x2,\ x4 = x4,\ x6 = x6,\ x5 = x5\},$$
$$\{x4 = 0,\ x7 = 0,\ x6 = 0,\ x8 = 0,\ x2 = x2,\ x5 = x5,\ x3 = x3,\ x1 = \kappa2\},$$
$$\{x7 = 0,\ x6 = 0,\ x5 = 0,\ x8 = 0,\ x2 = x2,\ x4 = x4,\ x3 = x3,\ x1 = \kappa1\}]$$

Thus, there are five different possible solutions, three of which requiring respectively that x_6, x_7 and x_8 be non-zero. Let's substitute these solutions into $\mathbf{g}$.

```
> for i from 1 to nops(sol) do g.i:=subs(sol[i],g) od;
```

$$g1 := -\frac{(-x7\ \kappa5 - x5\ x4 + x6\ x3)\ Id}{x8} + \kappa5\ e1 + x3\ e2 + x4\ e3 + x5\ e12 + x6\ e13 + x7\ e23$$
$$+ x8\ e123$$

$$g2 := \frac{\kappa4\ Id}{x7} + \frac{(-x5\ x4 + x6\ x3)\ e1}{x7} + x3\ e2 + x4\ e3 + x5\ e12 + x6\ e13 + x7\ e23$$

$$g3 := \frac{\kappa3\ Id}{x6} + x2\ e1 + \frac{x4\ x5\ e2}{x6} + x4\ e3 + x5\ e12 + x6\ e13$$

$$g4 := \kappa2\ Id + x2\ e1 + x3\ e2 + x5\ e12$$

$$g5 := \kappa1\ Id + x2\ e1 + x3\ e2 + x4\ e3$$

The above are five different types of $\mathbf{g}$ in $C\ell_{0,3,0}$ satisfying $\mathbf{g}\mathbf{g}^* = \mathbf{1}$.

```
> for i from 1 to nops(sol) do simplify(cmul(g.i,star(g.i))) od;
```

$$Id,\ Id,\ Id,\ Id,\ Id$$

[23] Recall that $\mathtt{e}_{12}$ was defined above as an alias of $\mathtt{e}_1 \wedge \mathtt{e}_2$ with the command `makealiases`.

We need to make sure now that each g_i displayed above satisfies also the second condition (2), namely, $\alpha(\mathbf{g})\mathbf{v}\mathbf{g}^*$ is in V for any $\mathbf{v} \in V$. We begin with the simplest element g_5. By computing the **Pin** group action on $\mathbf{v}$ and requiring that the result be a 1-vector, we get for g_5 :

```
> Pin_action(v,g5);
```

$$(2\,\kappa 1\,c3\,x4 + 2\,\kappa 1\,c1\,x2 + 2\,\kappa 1\,c2\,x3)\,Id + (-2\,x3\,x4\,c2 - 2\,x4\,x2\,c1 - 2\,x4^2\,c3 + c3)\,e3$$
$$+ (-2\,x4\,x2\,c3 - 2\,x3\,x2\,c2 - 2\,x2^2\,c1 + c1)\,e1$$
$$+ (-2\,x3^2\,c2 - 2\,x4\,x3\,c3 - 2\,x2\,x3\,c1 + c2)\,e2$$

It should be clear from the above that since the coefficient of Id must be zero for any c_1, c_2, c_3, either $x_2 = x_3 = x_4 = 0$ or $\kappa_1 = 0$. Let $\varepsilon = \pm 1$.[24] Thus, the former gives $\mathbf{g} = \pm\mathbf{1}$,

```
> g.5.1:=subs({x2=0,x3=0,x4=0},g5);
```

$$g51 := eps\,Id$$

while the latter gives

```
> g.5.2:=subs({kappa1=0,x4=lambda1},g5);
```

$$g52 := \lambda 1\,e3 + x2\,e1 + x3\,e2$$

where $\lambda_1 = \pm\sqrt{1 - x_2^2 - x_3^2}$.[25] We will collect all **Pin** group elements in a set `Pin_group`.

```
> Pin_group:={g.5.1,g.5.2};
```

$$Pin\_group := \{eps\,Id,\ \lambda 1\,e3 + x2\,e1 + x3\,e2\}$$

Similarly, we consider g_4. We assign the identity coefficient of the **Pin** action $\alpha(\mathbf{g})\mathbf{v}\mathbf{g}^*$ to a variable eq and find a solution to the resulting two equations that will be parameterized by c_1, c_2 :

```
> a:=Pin_action(v,g4);
```

$$a := (2\,x3\,\kappa 2\,c2 - 2\,x5\,x2\,c2 + 2\,x5\,x3\,c1 + 2\,x2\,\kappa 2\,c1)\,Id + c3\,e3$$
$$+ (-2\,x2^2\,c1 - 2\,x5\,\kappa 2\,c2 - 2\,x3\,x2\,c2 + c1 - 2\,x5^2\,c1)\,e1$$
$$+ (-2\,x5^2\,c2 - 2\,x2\,x3\,c1 + 2\,\kappa 2\,x5\,c1 - 2\,x3^2\,c2 + c2)\,e2$$

```
> eq:=collect(coeff(a,Id),{c1,c2}):eq1:=coeff(eq,c1):eq2:=coeff(eq,c2):
> sol:=[solve({eq1,eq2},{x2,x5,x3})];
```

$$sol := [\{x5 = 0,\ x2 = \lambda 2,\ x3 = x3\},\ \{x3 = 0,\ x2 = 0,\ x5 = x5\}]$$

In the above, $\lambda_2 = \pm\sqrt{1 - x_3^2}$. Likewise, we set $\lambda_3 = \pm\sqrt{1 - x_5^2}$.[26] The two new elements we assign to g_{41}, g_{42} and add to `Pin_group`.

```
> for i from 1 to nops(sol) do g.4.i:=simplify(subs(sol[i],g4)) od:
> Pin_group:=Pin_group union {g41,g42};
```

$$Pin\_group := \{eps\,Id,\ \lambda 1\,e3 + x2\,e1 + x3\,e2,\ \lambda 2\,e1 + x3\,e2,\ \lambda 3\,Id + x5\,e12\}$$

In order to continue with g_3 displayed above, we must make the assumption $x_6 \neq 0$ known to Maple. Then, the action of g_3 on a vector can be computed.[27]

```
> assume(x6>0,x6<0);
```

[24] In Maple one way to make $\varepsilon = \pm 1$ is to define `alias(eps=RootOf(_Z^2-1)):`. In the following, Maple outputs will contain the alias *eps*.

[25] In Maple we define `alias(lambda1=RootOf(_Z^2+x2^2+x3^2-1)):`.

[26] Both are defined as aliases: `alias(lambda2=RootOf(_Z^2-1+x3^2)): alias(lambda3=RootOf(_Z^2-1+x5^2)):`.

[27] We won't display it due to its length.

```
> a:=Pin_action(v,g3):
```

As in the previous two case, the quantity a is spanned by $\{Id, e_1, e_2, e_3\}$. We will isolate the coefficient of the identity element in a, assign it to a variable eq, and then determine for which values of x_2, x_4, x_5, x_6 it will be automatically zero for every choice of c_1, c_2, c_3. This will require solving a set of three equations $\{eq_1, eq_2, eq_3\}$ for x_2, x_4, x_5, x_6. Maple reminds us that $x_6 \neq 0$ by displaying it as $x6\tilde{}$:

```
> cliterms(a);
```

$$\{Id,\ e3,\ e1,\ e2\}$$

```
> eq:=collect(coeff(a,Id),{c.(1..3)});
```

$$eq := -2\,\frac{(x6\tilde{}^3\ x2 - \lambda 4\ x4\ x6\tilde{})\ c3}{x6\tilde{}^2} - 2\,\frac{(-\lambda 4\ x2\ x6\tilde{} - x5^2\ x4\ x6\tilde{} - x4\ x6\tilde{}^3)\ c1}{x6\tilde{}^2}$$
$$- 2\,\frac{(-\lambda 4\ x5\ x4 + x2\ x5\ x6\tilde{}^2)\ c2}{x6\tilde{}^2}$$

```
> for i from 1 to 3 do eq.i:=coeff(eq,c.i) od:
> sol:=solve({eq.(1..3)},{x6,x4,x5,x2});
```

$$sol := \{x2 = 0,\ x6\tilde{} = x6\tilde{},\ x5 = x5,\ x4 = 0\}$$

This time we only have one solution which we then substitute into g_3.

```
> g31:=subs(sol,g3);
```

$$g31 := \frac{\lambda 5\ Id}{x6\tilde{}} + x5\ e12 + x6\tilde{}\ e13$$

where λ_5 is another alias (see Appendix 2).

```
> Pin_group:=Pin_group union {g31};
```

$$Pin\_group := \{\lambda 3\ Id + x5\ e12,\ \frac{\lambda 5\ Id}{x6\tilde{}} + x5\ e12 + x6\tilde{}\ e13,\ \lambda 1\ e3 + x2\ e1 + x3\ e2,$$
$$\lambda 2\ e1 + x3\ e2,\ eps\ Id\}$$

By continuing in the similar fashion with the elements g_2 and g_1, one can find all general types of $\mathbf{Pin}(3)$. Finally, all general elements of $\mathbf{Pin}(3)$ can be displayed:

```
> 'Pin_group'=Pin_group;
```

$$Pin\_group := \{\lambda 3\ Id + x5\ e12,\ \frac{\lambda 5\ Id}{x6\tilde{}} + x5\ e12 + x6\tilde{}\ e13,$$
$$\frac{\lambda 7\ Id}{x7\tilde{}} + x5\ e12 + x6\ e13 + x7\tilde{}\ e23,\ \lambda 9\ e1 + x3\ e2 + x4\ e3 + x8\tilde{}\ e123,$$
$$\lambda 1\ e3 + x2\ e1 + x3\ e2,\ \lambda 2\ e1 + x3\ e2,\ eps\ Id\}$$

where λ_7 and λ_9 are displayed in the Appendix 2. It is a simple matter now to verify that all elements of $\mathbf{Pin}$ displayed in Pin_group satisfy both conditions (1) and (2) from the definition (33).

```
> for g in Pin_group do evalb(simplify(cmul(g,star(g))=Id)) od;#Condition (1)
```

$$true,\ true,\ true,\ true,\ true,\ true$$

```
> for g in Pin_group do
> evalb(Pin_action(v,g)=vectorpart(Pin_action(v,g),1)) od;#Condition (2)
```

true, *true*, *true*, *true*, *true*, *true*

We are now in position to verify Selig's claim [27], page 153, that the scalar product on $V = \mathsf{R}^3$ defined in $C\ell_{d,p,q}$ as

$$\mathbf{v}_1 \cdot \mathbf{v}_2 = \frac{1}{2}(\mathbf{v}_1\mathbf{v}_2^* + \mathbf{v}_2\mathbf{v}_1^*), \quad \text{for any } \mathbf{v}_1, \mathbf{v}_2 \in \mathsf{R}^3,$$

is preserved under the action of the **Pin** group. Let v_1, v_2 be two arbitrary 1-vectors defined earlier. Procedure `scalarprod` that gives the scalar product may be defined as follows:

```
> scalarprod:=(x,y)->scalarpart(1/2*(x &c star(y) + star(y) &c x)):#scalar product
> for g in Pin_group do
> simplify(scalarprod(Pin_action(v1,g),Pin_action(v2,g))-
> scalarprod(v1,v2)) od;
```

0, 0, 0, 0, 0, 0

Thus, **Pin**(3) preserves the scalar product in R^3 and, therefore, we have a homomorphism from **Pin**(3) to O(3) which is known to be a double-covering map. In the process, we have found all types of elements in **Pin**(3).

5.2 Group Spin(3)

In this section we will perform a few computations with **Spin**(3). Once we have found general elements in **Pin**(3), it is much easier to find elements in **Spin**(3). Recall from (34) that

$$\mathbf{Spin}(3) = \{\mathbf{g} \in C\ell^+_{0,3,0} : \mathbf{g}\mathbf{g}^* = \mathbf{1} \text{ and } \mathbf{g}\mathbf{x}\mathbf{g}^* \in \mathsf{R}^3 \text{ for all } \mathbf{x} \in \mathsf{R}^3\}.$$

Let's find *gSpin*, a general element in **Spin**(3). Since $\mathbf{Spin}(3) \subset C\ell^+_{0,3,0}$, we will begin with decomposing *gSpin* over the even basis elements.

```
> clibaseven:=cbasis(3,'even');
```

clibaseven := [*Id*, *e12*, *e13*, *e23*]

```
> gSpin:=c0*Id+c3*e12+c2*e13+c1*e23;
```

gSpin := *c0 Id* + *c3 e12* + *c2 e13* + *c1 e23*

Notice that under the **Spin** group action defined as a procedure `Spin_action`

```
> Spin_action:=(x,g)-> simplify(g &c x &c star(g));
```

Spin_action := $(x, g) \to \text{simplify}((g \text{ `\&c` } x) \text{ `\&c` } \text{star}(g))$

vectors are automatically mapped into vectors:

```
> Spin_action(v,gSpin)-vectorpart(Spin_action(v,gSpin),1);
```

0

We just need to make sure that $\mathbf{g}\mathbf{g}^* = \mathbf{1}$ for each $\mathbf{g} \in \mathbf{Spin}(3)$. To simplify Maple output, we define $\kappa = \sqrt{1 - c_1^2 - c_2^2 - c_3^2}$ as a Maple alias.

```
> alias(kappa=sqrt(-c1^2-c2^2-c3^2+1)):
> sol:=clisolve2(cmul(gSpin,star(gSpin))-Id,[c.(0..3)]);
```

sol := [{*c1* = *c1*, *c2* = *c2*, *c3* = *c3*, *c0* = κ}, {*c1* = *c1*, *c2* = *c2*, *c3* = *c3*, *c0* = $-\kappa$}]

```
> gSpin:=eps*kappa*Id+c3*e12+c2*e13+c1*e23;
```

gSpin := *eps* κ *Id* + *c3 e12* + *c2 e13* + *c1 e23*

Thus, the most general element in $\mathbf{Spin}(3)$ is just $\mathbf{g} = \varepsilon\kappa\mathbf{1} + c_3\mathbf{e}_{12} + c_2\mathbf{e}_{13} + c_1\mathbf{e}_{23}$ where $\varepsilon = \pm 1$. Notice, that the defining properties of $\mathbf{g}$ are easily checked:

```
> simplify(cmul(gSpin,star(gSpin)));
```

$$Id$$

```
> evalb(Spin_action(v,gSpin)=vectorpart(Spin_action(v,gSpin),1));
```

$$true$$

In fact, element $\mathbf{g} \in \mathbf{Spin}(3)$ could be identified with a unit quaternion spanned over the basis $\{\mathbf{1}, \mathbf{e}_{12}, \mathbf{e}_{13}, \mathbf{e}_{23}\}$. Then, the * conjugation becomes the quaternionic conjugation. It can be easily checked by hand or with CLIFFORD that the basis (bi)vectors anticommute and square to $-\mathbf{1}$.

```
> quatbasis:=[e12,e13,e23];
```

$$quatbasis := [e12,\ e13,\ e23]$$

```
> M:=matrix(3,3,(i,j)->cmul(quatbasis[i],quatbasis[j]));
```

$$M := \begin{bmatrix} -Id & e23 & -e13 \\ -e23 & -Id & e12 \\ e13 & -e12 & -Id \end{bmatrix}$$

We have unit quaternions on a unit sphere in R^4 isomorphic to $\mathbf{Spin}(3)$ while the even part of $C\ell_{0,3}$ is isomorphic with the quaternionic division ring H. $\mathbf{Spin}(3)$ acts on R^3 through the rotations. In Appendix 3 one can find a procedure `rot` which takes as its first argument a vector $\mathbf{v}$ in R^3 while as its second argument it takes a quaternion. For example, a counter-clockwise rotation in the $\{\mathbf{e}_1, \mathbf{e}_2\}$ is accomplished with a help of a unit quaternion $\cos(\frac{\theta}{2}) + \sin(\frac{\theta}{2})\,\mathbf{e}_{12}$:

```
> rot(e1,cos(theta/2)+sin(theta/2)*e12);
> rot(e2,cos(theta/2)+sin(theta/2)*e12);
> rot(e3,cos(theta/2)+sin(theta/2)*e12);
```

$$\cos(\theta)\,e1 + e2\sin(\theta),\ -e1\sin(\theta) + \cos(\theta)\,e2,\ e3$$

Let's now take a general element from $\mathbf{Spin}(3)$ and act on all three unit basis vectors $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$. We can easily verify that the new elements $\mathbf{e}_{11}, \mathbf{e}_{22}, \mathbf{e}_{33}$ provide another orthonormal basis with the same orientation:

```
> e11:=rot(e1,gSpin);e22:=rot(e2,gSpin);e33:=rot(e3,gSpin);
```

$$e11 := 2\,(eps\,\kappa\,c2 + c3\,c1)\,e3 - (2\,c2^2 + 2\,c3^2 - 1)\,e1 + 2\,(eps\,\kappa\,c3 - c2\,c1)\,e2$$

$$e22 := 2\,(eps\,\kappa\,c1 - c3\,c2)\,e3 - 2\,(eps\,\kappa\,c3 + c2\,c1)\,e1 - (2\,c1^2 - 1 + 2\,c3^2)\,e2$$

$$e33 := -(2\,c2^2 + 2\,c1^2 - 1)\,e3 - 2\,(-c3\,c1 + eps\,\kappa\,c2)\,e1 - 2\,(eps\,\kappa\,c1 + c3\,c2)\,e2$$

```
> e1 &c e2 + e2 &c e1, e1 &c e3 + e3 &c e1, e2 &c e3 + e3 &c e2;
```

$$0,\ 0,\ 0$$

```
> e11 &c e22 + e22 &c e11, e11 &c e33 + e33 &c e11, e22 &c e33 + e33 &c e22;
```

$$0,\ 0,\ 0$$

```
> e1 &w e2 &w e3,e11 &w e22 &w e33;
```

$$e123,\ e123$$

Length of a vector $\mathbf{v}$ under the action $\mathbf{Spin}(3)$ is of course preserved:

```
> Enorm(v),Enorm(Spin_action(v,gSpin));
```

$$c3^2 + c1^2 + c2^2,\ c3^2 + c1^2 + c2^2$$

Example 1: Rotations in coordinate planes

Let's define unit quaternions responsible for the rotations in the coordinate planes. These are counter-clockwise rotations when looking down the rotation axis. We will define a pure-quaternion basis consisting of $\{\mathbf{q}_i, \mathbf{q}_j, \mathbf{q}_k\}$ in place of traditionally used $\{\mathbf{i}, \mathbf{j}, \mathbf{k}\}$.

```
> qi:=e23:qj:=e13:qk:=e12:
> q12:=cos(alpha/2)*Id+sin(alpha/2)*'qk';#rotation in the xy-plane
```

$$q12 := \cos(\frac{1}{2}\,\alpha)\,Id + \sin(\frac{1}{2}\,\alpha)\,qk$$

```
> q13:=cos(beta/2)*Id+sin(beta/2)*'qj';#rotation in the xz-plane
```

$$q13 := \cos(\frac{1}{2}\,\beta)\,Id + \sin(\frac{1}{2}\,\beta)\,qj$$

```
> q23:=cos(gamma/2)*Id+sin(gamma/2)*'qi';#rotation in the yz-plane
```

$$q23 := \cos(\frac{1}{2}\,\gamma)\,Id + \sin(\frac{1}{2}\,\gamma)\,qi$$

Notice that to rotate by an angle $n\alpha$ it is enough to find the n-th Clifford power of the appropriate quaternion and then apply it to a vector.

```
> q12 &c q12; #rotation by the angle 2*alpha
```

$$\cos(\alpha)\,Id + e12\,\sin(\alpha)$$

```
> q12 &c q12 &c q12; #rotation by the angle 3*alpha
```

$$\cos(\frac{3}{2}\,\alpha)\,Id + \sin(\frac{3}{2}\,\alpha)\,e12$$

```
> q12 &c q12 &c q12 &c q12; #rotation by the angle 4*alpha
```

$$\cos(2\,\alpha)\,Id + \sin(2\,\alpha)\,e12$$

Let's see now how these basis rotations in the coordinate planes act on an arbitrary vector $\mathbf{v} = a\mathbf{e}_1 + b\mathbf{e}_2 + c\mathbf{e}_3$:

```
> v:=a*e1+b*e2+c*e3;
```

$$v := a\,e1 + b\,e2 + c\,e3$$

The norm of $\mathbf{v}$ is $\|\mathbf{v}\| = \mathbf{v}\mathbf{v}^*$ and it can be defined in CLIFFORD as follows:

```
> vlength:=sqrt(scalarpart(v &c star(v)));
```

$$vlength := \sqrt{c^2 + a^2 + b^2}$$

Certainly, rotations do not change length. For example, let's rotate $\mathbf{v}$ by $q_{13}q_{12}$:

```
> v123:=rot(v,q13 &c q12); #rotation q12 followed by q13
```

$$v123 := (-b\sin(\beta)\sin(\alpha) + a\sin(\beta)\cos(\alpha) + c\cos(\beta))\,e3$$
$$+ (-b\sin(\alpha)\cos(\beta) - c\sin(\beta) + a\cos(\beta)\cos(\alpha))\,e1 + (b\cos(\alpha) + a\sin(\alpha))\,e2$$

```
> vlength:=sqrt(scalarpart(v123 &c star(v123)));
```

$$vlength := \sqrt{c^2 + a^2 + b^2}$$

Thus, the length of $\mathbf{v}_{123}$ is the same as the length of $\mathbf{v}$. However, rotations do not commute. We will show that by applying quaternion $q_{12}q_{13}$ to $\mathbf{v}$ and by comparing it with q_{123} :

```
> v132:=rot(v,q12 &c q13); #rotation q13 followed by q12
```

$$v132 := (a\sin(\beta) + c\cos(\beta))\ e3 + (a\cos(\beta)\cos(\alpha) - c\sin(\beta)\cos(\alpha) - b\sin(\alpha))\ e1 + (b\cos(\alpha) - c\sin(\beta)\sin(\alpha) + a\sin(\alpha)\cos(\beta))\ e2$$

```
> clicollect(v123-v132);
```

$$(-b\sin(\beta)\sin(\alpha) + a\sin(\beta)\cos(\alpha) - a\sin(\beta))\ e3 + (-b\sin(\alpha)\cos(\beta) - c\sin(\beta) + c\sin(\beta)\cos(\alpha) + b\sin(\alpha))\ e1 + (a\sin(\alpha) + c\sin(\beta)\sin(\alpha) - a\sin(\alpha)\cos(\beta))\ e2$$

As it can be seen, $\mathbf{v}_{123} \neq \mathbf{v}_{132}$.

Example 2: Counter-clockwise rotation by an angle α around the given axis

In this example we will find a way to rotate a given vector $\mathbf{v}$ by an angle α in a plane orthogonal to the given axis vector $\mathbf{axis} = a_1\mathbf{e}_1 + a_2\mathbf{e}_2 + a_3\mathbf{e}_3$ vector. This rotation will be counter-clockwise when looking down the axis towards to origin $(0,0,0)$ of the coordinate system. In order to derive symbolic formulas, we will assume that the symbolic vector **axis** has been normalized by defining $\lambda = \pm\sqrt{1 - a_1^2 - a_2^2}$ and $\mathbf{axis} = a_1\mathbf{e}_1 + a_2\mathbf{e}_2 + \lambda\mathbf{e}_3$. However, it won't be necessary for **axis** to be of unit length when its components are numeric.

```
> alias(lambda=RootOf(-a1^2-a2^2-_Z^2+1)):
> axis:=a1*e1+a2*e2+lambda*e3;
```

$$axis := a1\ e1 + a2\ e2 + \lambda\ e3$$

```
> simplify(axis &c star(axis));
```

$$Id$$

Thus, in the symbolic case we will always have that $\|\mathbf{axis}\| = 1$. Notice that in order to represent a rotation around **axis** we need to find a dual unit quaternion which we will call **qaxis**. It will need to be defined in such a way as to give a desired orientation for the rotation. Since we have opted for counter-clockwise rotations, we define $\mathbf{qaxis} = -\mathbf{axis}\,\mathbf{e}_{123}$ where $\mathbf{e}_{123}$ is a unit pseudo-scalar in $C\ell_{0,3}$. In the following we will refer to the $\mathbf{axis} = a_1\mathbf{e}_1 + a_2\mathbf{e}_2 + a_3\mathbf{e}_3$ as the axis (a_1, a_2, a_3).

```
> qaxis:=axis &c (-e123);
```

$$qaxis := \lambda\ e12 + a1\ e23 - a2\ e13$$

In Appendix 3 Reader can find a procedure `qrot` which finds the dual quaternion **qaxis**. The first three arguments to `qrot` are the (numeric or symbolic) components of the **axis** vector in the basis $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ while the fourth argument is the angle of rotation. For example, we can define various rotation quaternions:

```
> q100:=qrot(1,0,0,theta);#rotation about the axis (1,0,0)
```

$$q100 := \cos(\frac{1}{2}\theta)\ Id + \sin(\frac{1}{2}\theta)\ e23$$

```
> q010:=qrot(0,1,0,theta);#rotation about the axis (0,1,0)
```

$$q010 := \cos(\frac{1}{2}\theta)\ Id - \sin(\frac{1}{2}\theta)\ e13$$

```
> q001:=qrot(0,0,1,theta);#rotation about the axis (0,0,1)
```

$$q001 := \cos(\frac{1}{2}\theta)\ Id + \sin(\frac{1}{2}\theta)\ e12$$

```
> q101:=qrot(1,0,1,theta);#rotation about the axis (1,0,1)
```

$$q101 := \cos(\frac{1}{2}\,\theta)\,Id + \sin(\frac{1}{2}\,\theta)\,(\frac{1}{2}\,e12\,\sqrt{2} + \frac{1}{2}\,\sqrt{2}\,e23)$$

```
> q011:=qrot(0,1,1,theta);#rotation about the axis (0,1,1)
```

$$q011 := \cos(\frac{1}{2}\,\theta)\,Id + \sin(\frac{1}{2}\,\theta)\,(\frac{1}{2}\,e12\,\sqrt{2} - \frac{1}{2}\,\sqrt{2}\,e13)$$

```
> q110:=qrot(1,1,0,theta);#rotation about the axis (1,1,0)
```

$$q110 := \cos(\frac{1}{2}\,\theta)\,Id + \sin(\frac{1}{2}\,\theta)\,(\frac{1}{2}\,\sqrt{2}\,e23 - \frac{1}{2}\,\sqrt{2}\,e13)$$

```
> q111:=qrot(1,1,1,theta);#rotation about the axis (1,1,1)
```

$$q111 := \cos(\frac{1}{2}\,\theta)\,Id + \sin(\frac{1}{2}\,\theta)\,(\frac{1}{3}\,e12\,\sqrt{3} - \frac{1}{3}\,\sqrt{3}\,e13 + \frac{1}{3}\,\sqrt{3}\,e23)$$

Let's try to rotate first a basis vector $\mathbf{e}_1$ around various axes listed above by some angle α. In the next example, after we find a general formula for the components of the rotated e1, we will verify it for various angles.

```
> v:=e1:
> vnew:=rot(v,q100); #rotation around the (100) axis
```

$$vnew := e1$$

```
> vnew:=rot(v,q010); #rotation around the (010) axis
```

$$vnew := -\sin(\theta)\,e3 + \cos(\theta)\,e1$$

```
> eval(subs(theta=Pi/2,vnew));
```

$$-e3$$

```
> vnew:=rot(v,q001); #rotation around the (001) axis
```

$$vnew := \cos(\theta)\,e1 + e2\,\sin(\theta)$$

```
> eval(subs(theta=Pi/2,vnew));
```

$$e2$$

```
> vnew:=rot(v,q101); #rotation around the (101) axis
```

$$vnew := -\frac{1}{2}\,(-1 + \cos(\theta))\,e3 + \frac{1}{2}\,(\cos(\theta) + 1)\,e1 + \frac{1}{2}\,\sqrt{2}\,e2\,\sin(\theta)$$

```
> eval(subs(theta=Pi/2,vnew));
```

$$\frac{1}{2}\,e3 + \frac{1}{2}\,e1 + \frac{1}{2}\,e2\,\sqrt{2}$$

```
> vnew:=rot(v,q011); #rotation around the (011) axis
```

$$vnew := -\frac{1}{2}\,\sqrt{2}\,e3\,\sin(\theta) + \cos(\theta)\,e1 + \frac{1}{2}\,\sqrt{2}\,e2\,\sin(\theta)$$

```
> eval(subs(theta=Pi/2,vnew));
```

$$-\frac{1}{2}\,e3\,\sqrt{2} + \frac{1}{2}\,e2\,\sqrt{2}$$

```
> vnew:=rot(v,q110); #rotation around the (110) axis
```

$$vnew := -\frac{1}{2}\,\sqrt{2}\,e3\,\sin(\theta) + \frac{1}{2}\,(\cos(\theta) + 1)\,e1 - \frac{1}{2}\,(-1 + \cos(\theta))\,e2$$

```
> eval(subs(theta=Pi/2,vnew));
```

$$-\frac{1}{2}\,e3\,\sqrt{2}+\frac{1}{2}\,e1+\frac{1}{2}\,e2$$

```
> vnew:=rot(v,q111); #rotation around the (111) axis
```

$$vnew := -\frac{1}{3}\,(\sqrt{3}\sin(\theta)-1+\cos(\theta))\,e3+\frac{1}{3}\,(2\cos(\theta)+1)\,e1 + \frac{1}{3}\,(\sqrt{3}\sin(\theta)+1-\cos(\theta))\,e2$$

```
> eval(subs(theta=Pi/2,vnew));
```

$$-\frac{1}{3}\,(\sqrt{3}-1)\,e3+\frac{1}{3}\,e1+\frac{1}{3}\,(\sqrt{3}+1)\,e2$$

```
> eval(subs(theta=Pi,vnew));
```

$$\frac{2}{3}\,e3-\frac{1}{3}\,e1+\frac{2}{3}\,e2$$

Example 3: A general formula

Finally, we derive a general formula for a rotation of an arbitrary vector $\mathbf{v}=v_1\mathbf{e}_1+v_2\mathbf{e}_2+v_3\mathbf{e}_3$ around an arbitrary $\mathbf{axis}=a_1\mathbf{e}_1+a_2\mathbf{e}_2+a_3\mathbf{e}_3$ and by an arbitrary angle α. In the purely symbolic case we assume that the axis is of unit length, that is, $a_3=\lambda$.

```
> v:=v1*e1+v2*e2+v3*e3; #an arbitrary vector
```

$$v := v1\,e1+v2\,e2+v3\,e3$$

```
> qnew:=clicollect(rot(v,qrot(a1,a2,lambda,theta)));# new vector after rotation
```

$$\begin{aligned} qnew :=\ & (-\lambda\,v1\,a1\cos(\theta)-\lambda\,v2\,a2\cos(\theta)+a1^2\,v3\cos(\theta)-v1\,a2\sin(\theta)+\lambda\,v1\,a1 \\ & +\lambda\,v2\,a2+a2^2\,v3\cos(\theta)+v2\,a1\sin(\theta)+v3-a2^2\,v3-a1^2\,v3)e3+(v1\cos(\theta) \\ & -a1\,v2\,a2\cos(\theta)-\lambda\,v3\,a1\cos(\theta)+\lambda\,v3\,a1+a1^2\,v1-a1^2\,v1\cos(\theta) \\ & -v2\,\lambda\sin(\theta)+v3\,a2\sin(\theta)+a1\,v2\,a2)e1+(a2^2\,v2+a1\,v1\,a2-v3\,a1\sin(\theta) \\ & -a2^2\,v2\cos(\theta)+\lambda\,v3\,a2+v1\,\lambda\sin(\theta)-\lambda\,v3\,a2\cos(\theta)+v2\cos(\theta) \\ & -a1\,v1\,a2\cos(\theta))e2 \end{aligned}$$

For example, let's rotate vector $\mathbf{v}=\mathbf{e}_1+2\mathbf{e}_2+3\mathbf{e}_3$ around the axis $(1,2,3)$ by any angle α. Certainly, since the vector $\mathbf{v}$ is on the axis of rotation, it should not change:

```
> clicollect(rot(e1+2*e2+3*e3,qrot(1,2,3,alpha)));
```

$$e1+2\,e2+3\,e3$$

Let's rotate $\mathbf{v}=\mathbf{e}_1-2\mathbf{e}_2+4\mathbf{e}_3$ around the axis $(2,-3,4)$ by an angle $\alpha=\pi/4$.

```
> clicollect(rot(e1-2*e2+4*e3,qrot(2,-3,4,Pi/4)));
```

$$\begin{aligned} & (-\frac{1}{58}\,\sqrt{29}\,\sqrt{2}+\frac{10}{29}\,\sqrt{2}+\frac{96}{29})\,e3+(-\frac{2}{29}\,\sqrt{29}\,\sqrt{2}+\frac{48}{29}-\frac{19}{58}\,\sqrt{2})\,e1 \\ & +(\frac{7}{29}\,\sqrt{2}-\frac{2}{29}\,\sqrt{29}\,\sqrt{2}-\frac{72}{29})\,e2 \end{aligned}$$

Thus, in this section we have shown how easy it is to derive vector rotation formulas from vector analysis using elements of $\mathbf{Spin}(3)$ considered as unit quaternions. It has been very helpful to be able to embed $\mathbf{Spin}(3)$ in $C\ell_{0,3}$.

5.3 Degenerate Clifford algebra and the proper rigid motions $SE(3)$

In this final section we will use the ability of CLIFFORD to perform computations in Clifford algebras of an arbitrary quadratic form including, of course, degenerate forms. We will consider the semi-direct product $\mathbf{Spin}(3)$ o R^3 that double covers the group of proper rigid motions $SE(3)$. We will follow the notation used in [27], page 156, except that our basis vector that squares to 0 will be $\mathbf{e}_4$ and not $\mathbf{e}$. We begin by defining B as a degenerate diagonal form $\mathrm{diag}(-1,-1,-1,0)$ of signature $(0,3,1)$. Recall from the previous section that procedure `star` gives conjugation in $C\ell_{0,3,1} = C(0,3,1)$.

```
> dim:=4:n:=dim-1:eval(makealiases(dim)):
> B:=diag(-1$n,0); #Selig's C(0,3,1)
```

$$B := \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Let the vector basis in R^3 be stored in `vbasis` and let $\mathbf{t}$ an arbitrary vector in R^3.

```
> vbasis:=cbasis(n,1):t:=add(t.i*vbasis[i],i=1..n);
```

$$t := t1\ e1 + t2\ e2 + t3\ e3$$

Elements of the form $\mathbf{1} + \mathbf{t}\,\mathbf{e}_4$ are invertible in $C\ell_{0,3,1}$ since $u\,\mathbf{e}_4$ and $\mathbf{e}_4\,u$ are nilpotent for any u in $C\ell_{0,3,1}$. That is, the Jacobson radical J in $C\ell_{0,3,1}$ is generated by $\mathbf{e}_4$.

```
> cinv(1+t &c e4;);#symbolic inverse of 1+t & c e4;
```

$$Id - t1\ e14 - t2\ e24 - t3\ e34$$

We will verify now statements made on page 156. Let $\kappa = \sqrt{-c_1^2 - c_2^2 - c_3^2 + 1}$ and $\varepsilon = \pm 1$ as before. Then the most general element $\mathbf{g}$ in $\mathbf{Spin}(3)$ has the form:

```
> alias(kappa=sqrt(-c1^2-c2^2-c3^2+1)):alias(eps=RootOf(_Z^2-1)):
> gSpin:=eps*kappa*Id+c3*e12+c2*e13+c1*e23;
```

$$gSpin := eps\,\kappa\,Id + c3\ e12 + c2\ e13 + c1\ e23$$

We consider a subgroup G of the group of units of $C\ell_{0,3,1}$ of the form $\mathbf{g} + \frac{1}{2}\mathbf{t}\mathbf{g}\mathbf{e}_4$ where $\mathbf{g}$ belongs to $\mathbf{Spin}(3)$ and $\mathbf{t}$ is a 1-vector. Elements in G will be given by a procedure `ge`.

```
> ge:=proc(g,t) RETURN(clicollect(simplify(g+1/2*t &c g &c e4))) end:
```

For the most general $\mathbf{g}$ in $\mathbf{Spin}(3)$ and $\mathbf{t} \in \mathsf{R}^3$, procedure `ge` gives:

```
> 'ge(gSpin,t)'=ge(gSpin,t);
```

$$\begin{aligned} ge(gSpin,t) &= (\tfrac{1}{2}\,t2\,\kappa\,eps - \tfrac{1}{2}\,t1\,c3 + \tfrac{1}{2}\,t3\,c1)\,e24 + (\tfrac{1}{2}\,t3\,\kappa\,eps - \tfrac{1}{2}\,t2\,c1 - \tfrac{1}{2}\,t1\,c2)\,e34 \\ &+ (\tfrac{1}{2}\,t2\,c3 + \tfrac{1}{2}\,t3\,c2 + \tfrac{1}{2}\,t1\,\kappa\,eps)\,e14 + (-\tfrac{1}{2}\,t2\,c2 + \tfrac{1}{2}\,t3\,c3 + \tfrac{1}{2}\,t1\,c1)\,e1234 \\ &+ eps\,\kappa\,Id + c3\ e12 + c2\ e13 + c1\ e23 \end{aligned}$$

First, let's verify Selig's statement that

$$(\mathbf{g} - \frac{1}{2}\mathbf{t}\,\mathbf{g}\,\mathbf{e}_4)^* = (\mathbf{g}^* + \frac{1}{2}\mathbf{g}^*\,\mathbf{t}\,\mathbf{e}_4). \tag{35}$$

Notice that the left-hand-side in (35) is just the conjugation of $ge(gSpin, -t)$ while the right-hand-side is equal to $ge(gSpin^*, -t)$ where $gSpin^*$ denotes the conjugate of $gSpin$.

```
> L:=clicollect(star(ge(gSpin,-t))):
> R:=simplify(star(gSpin)+1/2*star(gSpin) &c t &c e4):
> simplify(L-R);
```

$$0$$

Next, we define the action of the group G on the subspace of $C\ell_{0,3,1}$ consisting of the elements of the form $1 + \mathbf{x}\,\mathbf{e}_4$ as follows:

$$(\mathbf{g} + \frac{1}{2}\mathbf{t}\,\mathbf{g}\,\mathbf{e}_4)(1 + \mathbf{x}\,\mathbf{e}_4)(\mathbf{g} - \frac{1}{2}\mathbf{t}\,\mathbf{g}\mathbf{e}_4)^* = 1 + (\mathbf{g}\,\mathbf{x}\,\mathbf{g}^* + \mathbf{t})\,\mathbf{e}_4 \tag{36}$$

where $\mathbf{x}, \mathbf{t} \in \mathsf{R}^3$ and $\mathbf{g} \in \mathbf{Spin}(3)$. The above identity can be shown as follows. We define a procedure `rigid` which will give this action on R^3 : $\mathbf{x} \mapsto \mathbf{g}\,\mathbf{x}\,\mathbf{g}^* + \mathbf{t}$.

```
> rigid:=proc(x,g,t) local p;
> if not evalb(x=vectorpart(x,1)) or
> not evalb(t=vectorpart(t,1)) then ERROR('x and t must be vectors')
> fi:
> if not type(g,evenelement) then ERROR('g must be even') fi;
> RETURN(clicollect(simplify(cmul(g,x,star(g))+t)))
> end:
```

This action will give us the rigid motion on R^3. Thus, we can compute the right-hand-side of (36) by using `rigid` while the left-hand-side will be computed directly.

```
> x:=add(x.i*vbasis[i],i=1..n);
```

$$x := x1\ e1 + x2\ e2 + x3\ e3$$

```
> LHS:=simplify(ge(gSpin,t) &c (1 + x &c e4) &c star(ge(gSpin,-t))):
> RHS:=simplify(Id+rigid(x,gSpin,t) &c e4):
> simplify(LHS-RHS);
```

$$0$$

Finally, we will verify directly that the action $\mathbf{x} \mapsto \mathbf{g}\,\mathbf{x}\,\mathbf{g}^* + \mathbf{t}$ is a rigid motion. If we denote by $\mathbf{x}_p, \mathbf{y}_p$ the images of $\mathbf{x}, \mathbf{y}$ under this action, we will need to show that $\|\mathbf{x}_p - \mathbf{y}_p\| = \|\mathbf{x} - \mathbf{y}\|$ in the Euclidean norm. We can compute the norm $\|\mathbf{x} - \mathbf{y}\|$ by taking $\sqrt{(\mathbf{x} - \mathbf{y})(\mathbf{x} - \mathbf{y})^*}$ in $C\ell_{0,3,1}$, or by using a procedure `distance`.

```
> y:=add(y.i*vbasis[i],i=1..n):
> distance:=proc(x,y) sqrt(simplify(scalarpart(cmul(x-y,star(x-y))))) end:
> xp:=rigid(x,gSpin,t):yp:=rigid(y,gSpin,t):
> evalb(distance(x,y)=distance(xp,yp));
```

$$true$$

Thus the action of G defined as $\mathbf{x} \mapsto \mathbf{g}\,\mathbf{x}\,\mathbf{g}^* + \mathbf{t}$ is a rigid motion in R^3. In view of the presence of the radical in $C\ell_{0,3,1}$, this group G is in fact a semi-direct product of $\mathbf{Spin}(3)$ and R^3, that is of rotations and translations. It is well known of course that $\mathbf{Spin}(3)$ o R^3 doubly covers the group of proper rigid motions $SE(3)$. We leave it as an exercise for the Reader to check in CLIFFORD that the composition of two rigid motions is a rigid motions.

6 Summary

The main purpose of this paper has been to show a variety of computational problems that can be approached with the symbolic package CLIFFORD. It is through an extensive experimentation with the package that the results reported in [7] were found; we have seen some of the computations

that have led to them in Section 3. In view of their complexity it is unlikely that they could have been performed by hand. Relation between the Clifford products in $C\ell(g)$ and $C\ell(B)$ through the Helmstetter's formula appears more clear once it has been checked with CLIFFORD. The SVD of a matrix as performed in Section 4 is clearly feasible in the Clifford algebra language; however it is not clear if the approach presented there is the best. More study would need to be done here in order to possibly simplify the computations, perform them uniquely in the Clifford algebra language, and possibly better utilize the nilpotent-idempotent basis in the Clifford algebra rather than the Grassmann basis. In robotics applications presented in Section 5.3, CLIFFORD appears to be a very convenient tool to carry out practical computations in the low dimensional algebras such as $C\ell_{0,3}$ and $C\ell_{1,3,1}$.

7 Acknowledgments

The Author would like to acknowledge collaboration with and contribution from Bertfried Fauser, Universität Konstanz, Fakultät für Physik, Fach M678, 78457 Konstanz, Germany, to the development of the additional procedures used in Section (3).

8 Appendix 1

In addition to the main package CLIFFORD, in Section 2 we have used new procedures from a supplementary package `suppl`. These procedures are: `cinvg`, `cmulg`, `LCg`, `makeF`, `RCg`, `revg`, and `splitB`.

- Procedure `cinvg` finds the Clifford inverse with respect to the symmetric part g of the bilinear form B, that is, it finds the inverse (if it exists) of u in $C\ell(g)$.
- Procedure `cmulg` performs Clifford multiplication with respect to the symmetric part g of the bilinear form B. That is, it gives the Clifford product $\underset{g}{uv}$ for any two elements u and v in $C\ell(g)$.
- Procedures `LCg` and `RCg` give the left and right contraction in $C\ell(g)$.
- Procedure `makeF` computes element $F \in \bigwedge^2 V^*$ defined in (3). It uses additional procedures `pairs` and `mysign` from the package `suppl`.
- Procedure `revg` performs the reversion anti-automorphism $\tilde{}$ in $C\ell(g)$.
- Procedure `splitB` splits a bilinear form B in V into its symmetric part g and its antisymmetric part A which are returned as a sequence g, A. If B is purely symbolic and a second optional parameter (of any type) is used, in addition to g and A the output sequence contains two lists of symbolic substitutions that relate entries of g and A to the entries of B. If B is not assigned, in order for `splitB` to work it internally assigns a blank 9×9 matrix to B and then it calls itself.

In Section 3 we have used the following additional procedures from `suppl`:

- Procedure `cliexpand` expands the given Clifford number $u \in C\ell(B)$ from the default Grassmann basis to a Clifford basis consisting of un-evaluated Clifford products of the generators $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$. Procedure `clieval` converts from the Clifford basis back to the Grassmann basis.
- Procedure `reversion` gives the reversion in $C\ell(B)$.
- Procedure `clisolve2` solves equations of the type $u = 0$ for the unknown parameters in u. Its code is shown in Appendix 3.
- Procedure `defB` is needed to define a bilinear form B in an even-dimensional vector space V.
- Procedure `bexpand` expands any element in the Hecke algebra $H_{\mathsf{F}}(n, q)$ in terms of the Hecke b-basis.
- Procedure `alpha2` provides a Hecke algebra automorphism.

In Section 4 we have used the following additional procedures from a supplementary package `avsd`.

- Procedure `phi` provides an isomorphism between a matrix algebra and a Clifford algebra.
- Procedure `radsimplify` simplifies radical expressions in matrices and vectors.
- Procedure `assignL` is needed to write output from a Maple procedure `eigenvects` in a suitable form, it sorts eigenvectors according to the corresponding eigenvalues, and it uses the Gram-Schmidt orthogonalization process, if necessary, to return a complete list of orthogonal eigenvectors.
- Procedure `climinpoly` belongs to the main package CLIFFORD. It computes a minimal polynomial of any element of a Clifford algebra.
- Procedure `makediag` makes a "diagonal" Σ matrix consisting of singular values.
- Procedure `embed` embeds the given non-square matrix or a matrix of smaller dimensions into a $2^k \times 2^k$ matrix of smallest k such that it can be mapped into a Clifford algebra.

9 Appendix 2

We display a Maple code of some more important procedures used in Section 3.
Procedure `bexpand` expands any element in the Hecke algebra in terms of the Hecke b-basis:

```
> suppl[bexpand]:=proc(X)
> local a,setbs,setbe,i,eq,T,sys,sol;
> option remember;
> setbs:=[Id,'b1','b2','b12','b21','b121']:
> setbe:=[Id,b1,b2,cmul(b1,b2),b21,cmul(b1,b21)]:
> eq:=CS(X-add(a[i]*setbe[i],i=1..6));
> T:=cliterms(eq);
> sys:={coeffs(eq,T)};
> sol:=solve(sys,{seq(a[i],i=1..6)});
> subs(sol,add(a[i]*setbs[i],i=1..6));
> end:
```

Procedure `alpha2` gives an automorphism in the Hecke algebra:

```
> suppl[alpha2]:=proc(X)
> local a,setbb,setbe,setba,i,eq,T,sys,sol;
> option remember;
> setbe:=[Id,b1,b2,cmul(b1,b2),b21,cmul(b1,b21)]:
> setba:=[Id,-1/q*reversion(b1),-1/q*reversion(b2),1/q^2*reversion(setbe[4]),
> 1/q^2*reversion(setbe[5]),-1/q^3*reversion(setbe[6])]:
> eq:=CS(X-add(a[i]*setbe[i],i=1..6));
> T:=cliterms(eq);
> sys:={coeffs(eq,T)};
> sol:=solve(sys,{seq(a[i],i=1..6)});
> subs(sol,add(a[i]*setba[i],i=1..6));
> end:
```

Aliases needed in Section (3.3):

```
> alias(w1=q^4*K[4]^2+3*q^3*K[4]^2-K[4]*q^3+4*q^2*K[4]^2-K[4]*q^2+
> 3*q*K[4]^2-K[4]*q-q+K[4]^2-K[4]):
> alias(w2=K[4]*q^3+2*K[4]*q^2-q^2+2*K[4]*q+K[4]):
> alias(w3=-K[4]^2+K[4]+K[2]+K[4]*K[5]*q-q^3*K[4]*K[2]-q*K[2]*K[4]+
> K[6]*q^2*K[4]+K[4]*q^4*K[6]+K[6]*q^3*K[4]+K[4]*K[6]*q-q^2*K[2]*K[4]+
> q^2*K[4]*K[5]-K[2]*q-K[5]*q+K[4]*q^2-K[4]*q+q^4*K[4]^2-q^2*K[4]^2-
> q*K[4]^2-K[6]*q+K[6]*q^2-K[4]*K[2]);
> alias(w4=K[6]*q^3+K[4]*q^3-K[6]*q^2-K[5]*q^2-K[6]*q-K[5]*q+K[6]+K[4]):
> alias(w5=K[4]*q+K[2]*q-K[6]-K[5]):
> alias(w6=K[6]*q^2+K[4]*q^2-2*K[6]*q-K[5]*q-K[4]*q+K[6]+K[4]):
> alias(w7=K[4]*q^3+2*K[4]*q^2+q+2*K[4]*q+K[4]):
> alias(w8=q^5*K[4]^2+3*q^4*K[4]^2+4*q^3*K[4]^2+K[4]*q^3+3*q^2*K[4]^2+
> K[4]*q^2+q*K[4]^2+K[4]*q-1+K[4]):
> alias(w9=K[4]*q^3+K[2]*q^3-2*K[6]*q^2-K[5]*q^2-K[4]*q^2+K[5]*q+
> 2*K[6]*q+K[4]*q-K[4]-K[6]):
```

Aliases needed in Section (3.3.)

```
> alias(H1=h[1]*q^3+2*h[6]*q-h[6]+h[5]*q^3-h[3]*q^2-2*h[5]*q^2-2*h[6]*q^2
> +h[5]*q+h[6]*q^3+h[4]*q^3+h[2]*q^3-h[2]*q^2+h[4]*q+h[3]*q^3-2*h[4]*q^2):
> alias(H2=-2*h[6]*q+h[6]-h[4]*q+h[6]*q^2+h[2]*q^2+h[4]*q^2+h[5]*q^2-h[5]*q):
> alias(H3=h[3]*q^2-h[5]*q-h[4]*q-2*h[6]*q+h[4]*q^2+h[6]*q^2+h[5]*q^2+h[6]):
> alias(H4=-h[6]+h[5]*q+h[6]*q):
> alias(H5=h[6]*q+h[4]*q-h[6]):
> alias(H6=h[6]):
```

Aliases needed in Section (5.1):

```
> alias(kappa1=RootOf(_Z^2-1+x3^2+x2^2+x4^2)):
> alias(kappa2=RootOf(_Z^2-1+x5^2+x3^2+x2^2)):
> alias(kappa3=RootOf(-x6^2+x5^2*x6^2+x4^2*x5^2+x6^4+x2^2*x6^2+x4^2*x6^2+_Z^2)):
> alias(kappa4=RootOf(-x7^2+x5^2*x7^2+x3^2*x7^2+x6^2*x7^2+x7^4+x4^2*x5^2-
> 2*x4*x5*x6*x3+x6^2*x3^2+x4^2*x7^2+_Z^2)):
> alias(kappa5=RootOf((x7^2+x8^2)*_Z^2+(2*x5*x4*x7-2*x3*x6*x7)*_Z-x8^2+
> x6^2*x3^2+x8^4+x4^2*x5^2-2*x4*x5*x6*x3+x6^2*x8^2+x7^2*x8^2+x5^2*x8^2+x
> 3^2*x8^2+x4^2*x8^2)):
> alias(eps=RootOf(_Z^2-1)):
> alias(lambda1=RootOf(_Z^2+x2^2+x3^2-1)):
> alias(lambda2=RootOf(_Z^2-1+x3^2)):
> alias(lambda3=RootOf(_Z^2-1+x5^2)):
> alias(lambda4=RootOf(-x6^2+x5^2*x6^2+x4^2*x5^2+x6^4+x2^2*x6^2+x4^2*x6
> ^2+_Z^2)):
> alias(lambda5=RootOf(_Z^2-x6^2+x5^2*x6^2+x6^4)):
> alias(lambda6=RootOf(-x7^2+x5^2*x7^2+x3^2*x7^2+x6^2*x7^2+x7^4+x4^2*x5
> ^2-2*x4*x5*x6*x3+x6^2*x3^2+x4^2*x7^2+_Z^2)):
> alias(lambda7=RootOf(_Z^2-x7^2+x5^2*x7^2+x6^2*x7^2+x7^4)):
> alias(lambda8=RootOf((x7^2+x8^2)*_Z^2+(2*x5*x4*x7-2*x3*x6*x7)*_Z-x8^2
> +x6^2*x3^2+x8^4+x4^2*x5^2-2*x4*x5*x6*x3+x6^2*x8^2+x7^2*x8^2+x5^2*x8^2+
> x3^2*x8^2+x4^2*x8^2)):
> alias(lambda9=RootOf(_Z^2*x8^2-x8^2+x8^4+x3^2*x8^2+x4^2*x8^2)):
```

10 Appendix 3

We display code for two procedures `qrot` and `rot` that were needed in Section 5.2. Procedure `qrot` finds a unit quaternion that is dual to the rotation axis vector **axis**.

```
> qrot:=proc(p1,p2,p3,theta) local bas,c,e,k,l,i,q,n;global qaxis;
> if type(p1,name) or type(p2,name) then
> RETURN(cos(theta/2)*Id+sin(theta/2)*qaxis) fi;
> if evalb(simplify(p1^2+p2^2+p3^2)=1) then
> q:=simplify(subs({a1=p1,a2=p2},qaxis)) fi;
> n:=sqrt(p1^2+p2^2+p3^2):
> if n=0 then ERROR(‘axis vector must be a non-zero vector‘) elif
> has(n,RootOf) then n:=max(allvalues(n)) fi;
> q:=simplify(subs({a1=p1/n,a2=p2/n},qaxis));
> bas:=[Id,e12,e13,e23]:c:=[]:k:=0:
> for i from 1 to 4 do l:=coeff(q,bas[i]);
> if has(l,RootOf) then k:=i fi:
> c:=[op(c),l] od:
> if k<>0 then e:=allvalues(c[k]);
> if p3>0 then c:=subsop(k=max(e),c) elif
> p3<0 then c:=subsop(k=min(e),c) else ERROR(‘p3=0‘) fi;
> fi;
> q:=add(c[i]*bas[i],i=1..4);
> RETURN(cos(theta/2)*Id+sin(theta/2)*q)
> end:
```

Procedure `rot` performs a rotation of a vector by a quaternion through a certain angle.

```
> rot:=proc(v,q) local qs;
> qs:=star(q):
> RETURN(map(factor,clicollect(simplify(cmul(q,v,qs)))))
> end:
```

Procedure `clisolve2` was used extensively throughout this paper to solve linear equations in a Clifford algebra $C\ell(B)$.

```
> suppl[clisolve2]:=proc(eq,indet) local i,T,vars,sol,sys;
> if type(indet,list) then
> vars:=convert(indet,set)
> else
> vars:=select(type,indets(indet),indexed)
> fi;
> T:=cliterms(eq);
> sys:={coeffs(clicollect(simplify(eq)),T)};
> sol:=[solve(sys,vars)];
> if type(indet,list) then
> RETURN(sol)
> else
> RETURN([seq(subs(sol[i],indet),i=1..nops(sol))]);
> fi;
> end:
```

References

[1] R. Abłamowicz; *Clifford algebra computations with Maple*, Proc. Clifford (Geometric) Algebras, Banff, Alberta Canada, 1995. Ed. W. E. Baylis, Birkhäuser, Boston, 1996, pp. 463–501.

[2] R. Abłamowicz; *Structure of spin groups associated with degenerate Clifford algebras*, Journal of Mathematical Physics, Vol. **27**, No. **1**, January 1986, pp. 1–6.

[3] R. Abłamowicz; *Deformation and contraction in Clifford algebras*, Journal of Mathematical Physics, Vol. **27**, No. **2**, January 1986, pp. 1–6.

[4] R. Abłamowicz; *Matrix exponential via Clifford algebras*, Journal of Nonlinear Mathematical Physics, Vol. **27**, No. **2**, August 1998, pp. 423–427.

[5] R. Abłamowicz; *Spinor Representations of Clifford Algebras: A Symbolic Approach*, CPC Thematic Issue "Computer Algebra in Physics Research", Physics Communications **115** (1998), pp. 510–535.

[6] R. Abłamowicz; CLIFFORD - Maple V package for Clifford algebra computations, ver. 4 (Copyright 1995-1999) and two supplementary packages suppl and asvd are available from http://math.tntech.edu/rafal/cliff4/.

[7] R. Abłamowicz and B. Fauser, *Hecke algebra representations in ideals generated by q-Young Clifford idempotents*, Proceedings of the 5th International Conference on Clifford Algebras, Ixtapa, Mexico, 1999, Vol. **1** (submitted) and math.QA/9908062.

[8] R. Abłamowicz and P. Lounesto, *Primitive idempotents and indecomposable left ideals in degenerate Clifford algebras*, Proc. of "Clifford Algebras and Their Applications in Mathematical Physics (Canterbury, 1985), pp. 61–65, NATO Adv. Sci. Inst. Ser. C: Math. Phys. Sci., 183, Reidel, Dordrecht-Boston, Mass.

[9] R. Abłamowicz and P. Lounesto, *On Clifford algebras of a bilinear form with an antisymmetric part*, in: R. Abłamowicz, P. Lounesto, and J. M. Parra, eds., *On Clifford Algebras with Numeric and Symbolic Computations* (Birkhäuser, Boston, 1996), pp. 167–188.

[10] M. Berry and J. Dongarra; *Atlanta Organizers Put Mathematics to Work for the Math Sciences Community*, SIAM News, Vol. **32**, No. **6**, July/August 1999, and references therein.

[11] N. Bourbaki; *Algebra 1*, Chapters 1–3, Springer Verlag, Berlin, 1989.

[12] J.A.Brooke; *A Galileian formulation of spin. I. Clifford algebras and Spin groups*, J. Math. Phys. **19**, no. **5**, 952–959 (1978); *A Galileian formulation of spin. I. Explicit realizations*, J. Math. Phys. **21**, no. **4**, pp. 617–621 (1980).

[13] J.A.Brooke; *Spin groups associated with degenerate orthogonal spaces*, Proc. of "Clifford Algebras and Their Applications in Mathematical Physics (Canterbury, 1985), pp. 93–102, NATO Adv. Sci. Inst. Ser. C: Math. Phys. Sci., 183, Reidel, Dordrecht-Boston, Mass.

[14] A. Crumeyrolle, *Orthogonal and Symplectic Clifford Algebras: Spinor Structures* (Kluwer, Dordrecht, 1990).

[15] C.W.Curtis and I. Reiner; *Methods of Representation Theory with Applications to Finite Groups and Orders*, Vol. 1, Wiley-Interscience, Now York, 1981.

[16] I.N.Herstein; *Noncommutative Rings*, The Carus Mathematical Monographs, Number 15, Mathematical Association of America, Chicago, 1968.

[17] B. Fauser; *Hecke algebra representations within Clifford geometric algebras of multivectors*, J. Phys. A: Math. Gen. **32**, 1999, pp. 1919–1936.

[18] M. Hamermesh; *Group Theory and Its application to Physical Problems*, Addison-Wesley, London, 1962.

[19] J. Helmstetter, *Monoïdes de Clifford et déformations d'algèbres de Clifford*, Journal of Algebra, Vol. **111** (1987), pp. 14–48.

[20] R.C. King, B.G. Wybourne; *Representations and traces of Hecke algebras $H_n(q)$ of type $\mathbf{A}_{n-1}$*, J. Math. Phys. **33**, 1992, pp. 4–14.

[21] P. Lounesto, R. Mikkola, and V. Vierros, 'CLICAL User Manual', Helsinki University of Technology, Institute of Mathematics, Research Reports **A248**, Helsinki, 1987.

[22] P. Lounesto; *Clifford Algebras and Spinors*, Cambridge University Press, Cambridge, 1997.

[23] P. Lounesto; Private communication, 1997.

[24] I. G. Macdonald; *Symmetric Functions and Hall Polynomials*, Oxford University Press, Oxford, 1979.

[25] J.M. Maciejowski; *Multivariable Feedback Design*, Addison-Wesley, Wokingham, England, 1989.

[26] I. R. Porteous: *Clifford Algebras and the Classical Groups* (Cambridge University Press, Cambridge, 1995).

[27] J.M. Selig; *Geometrical Methods in Robotics*, Monographs in Computer Science, Springer-Verlag, New York, 1996.

[28] G. Strang; *Introduction to Linear Algebra*, Wellesley-Cambridge Press, Wellesley, 1998.