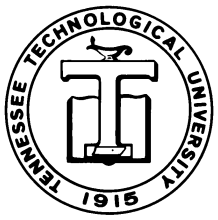
DEPA E A E A
E A EP

MATHEMATICS OF CLIFFORD - A MAPLE PACKAGE FOR CLIFFORD AND GRASSMANN ALGEBRAS

(REVISED S

A A A A
A D
E ED A E

DE E E



E E EE E A E

CLIFFORD

Rafal Ablamowicz [a] and Bertfried Fauser [b]

[a] *Department of Mathematics, Box 5054, Tennessee Technological University, Cookeville, TN 38505, USA*

[b] *Max Planck Institute for Mathematics in the Sciences, Inselstrasse 22-26, D-04103 Leipzig, Germany*

CLIFFORD performs various computations in Graßmann and Cli ord algebras. It can compute with quaternions, octonions, and matrices with entries in $C\ell(B)$ - the Cli ord algebra of a vector space V endowed with an arbitrary bilinear form B. Two user-selectable algorithms for the Cli ord product are implemented: cmulNUM - based on Chevalley's recursive formula, and cmulRS - based on a non-recursive Rota-Stein sausage. Graßmann and Cli ord bases can be used. Properties of reversion in undotted and dotted wedge bases are discussed.

Key words: quantum Cli ord algebra, contraction, dotted wedge product, grade involution, Graßmann algebra, Hopf algebra, multivector, octonions, quaternions, reversion, wedge product

1 Introduction

As many programs CLIFFORD emerged from a practical problem. Relatively complicated algebraic manipulations with octonions, which can be performed in spin(7), started a thread which has now developed into a multi purpose algebra tool. It is the basic structure of a vector space V endowed with a quadratic form Q

categorial sense ‘for free’– an algebra structure, the Cli ord algebra $C\ell(V, Q)$. While in a conventional vector calculus one makes a good use of the vector space structure, one does not have yet a vector algebra since vector multiplication is missing. Having established a Cli ord algebra structure provides one with an entirely new formalism that now can be applied to solving completely di erent problems.

In this sense, CLIFFORD is a basic tool for all such investigations and applications which can be carried in finite dimensional vector spaces equipped with a

2 Notations and basic computations

CLIFFORD uses as default a standard Graßmann basis (Graßmann multivectors) in $\bigwedge V$ where V = span $\{e_i \mid 1 \le i \le n\}$ for $1 \le n \le 9$. Then $\bigwedge V$ = span $\{1, e_i, e_i \wedge e_j, \ldots, e_i \wedge e_j \wedge \ldots \wedge e_k \mid 0 \le i < j < \ldots < k \le n\}$. In CLIFFORD these basis monomials are written as strings $\{I, 1, \ldots, 9, 1w2, 1w3, \ldots, 1w2w3, \ldots\}$ although they can be aliased to shorten input. Here $1w2$ is a string that denotes $e_1 \wedge e_2$ and Id denotes the identity 1 in $\bigwedge V$. However, CLIFFORD can also use one-character long symbolic indices as in $eiwej$. Thus, in principle, it can compute with Clifford algebras in dimensions higher than 9. For example, when $n = 3$, Graßmann basis monomials are:

```
> W=cbasis(3);
```

$$W = [I,\ 1,\ 2,\ 3,\ 1w2,\ 1w3,\ 2w3,\ 1w2w3]$$

but aliases can also be used to shorten input/output:

```
> eval(makealiases(3));
```

$$I,\ 12,\ 21,\ 13,\ 31,\ 23,\ 32,\ 123,\ 132,\ 213,\ 231,\ 312,\ 321$$

In the above, $jk = w2w3$ is the wedge product of three 1-vectors: e_1, e_2, e_3. Thus, the most general element in the Graßmann algebra $\bigwedge V$ is a Graßmann polynomial which is just a linear combination of Graßmann basis monomials with real coefficients. Notice that symbolic indices are allowed:

```
> p1:=Id+4.5*ei-alpha*e1we2we3;
```

$$p1 := I + 4.5\ ei - \alpha\ 123$$

Reordering of Graßmann monomials can be explicitly accomplished with a procedure reorder. CLIFFORD procedures ordinarily return their results in the standard (reordered) basis.

```
> p2:=-e3we2we1-x0*Id+x12*e2we1+a*ejwei;reorder(p2);
```

$$p2 := -321 - x0\ I + x12\ 21 + a\ jw$$

$$123 - x0\ I - x12\ 12 + a\ w\ j$$

The wedge product $\wedge$ is computed with a procedure wedge or its ampersand counterpart &w:

```
> wedge(e1,e2),e1 &w e2;wedge(ea,eb,ec),ea &w eb &w ec;p1 &w p2;
```

$$12,\ 12$$

$$w\ w,\ w\ w$$

$$123 - x0\ I + 4.500000000\ x0\ 1 + \alpha\ x0\ 123 - x12\ 12$$

Of course, irrespective of the bilinear form chosen, the Graßmann multiplication table will always remain as:

```
wedgetable:=matrix(4,4,(i,j)->wedge(cbas[i],cbas[j]));
```

$$w := \begin{bmatrix} I & 1 & 2 & 12 \\ 1 & 0 & 12 & 0 \\ 2 & 12 & 0 & 0 \\ 12 & 0 & 0 & 0 \end{bmatrix}$$

Let $B = g + F$ where g and F are the symmetric and the antisymmetric parts of B :

```
g,F:=matrix(2,2,[g11,g12,g12,g22]),matrix(2,2,[0,F12,-F12,0]);
B:=evalm(g+F);
```

$$g, F := \begin{bmatrix} 11 & 12 \\ 12 & 22 \end{bmatrix}, \begin{bmatrix} 0 & F12 \\ F12 & 0 \end{bmatrix}$$

$$B := \begin{bmatrix} 11 & 12 + F12 \\ 12 & F12 \quad 22 \end{bmatrix}$$

Then, the Cli ord multiplication table of the basis monomials in $C(B)$ will be as follows:

```
MultTable:=matrix(4,4,(i,j)->cmul(cbas[i],cbas[j]));
```

$M_{\mathcal{U}} T :=$

$[I, 1, 2, 12]$

$[1, 11\, I, 12 + (12 + F12)\, I, 11\ 2 \quad (12 + F12)\ 1]$

$[2, (12 \quad F12)\, I \quad 12, 22\, I, (12 \quad F12)\ 2 \quad 22\ 1]$

$[12, (12 \quad F12)\ 1 \quad 11\ 2, 22\ 1 \quad (12 + F12)\ 2,$

$(12$

```
> clisort(simplify(%));
```

$$2\ \ 12\ I$$

It is well known [16,19] that real Cli ord algebras $C(V,Q) = C$ are classified in terms of the signature (p,q) of Q and the dimension $\dim V = n = p+q$. Information about all Cli ord algebras C, $1 \quad n \quad 9$, for any signature (p,q) has been pre-computed and stored in `CLIFFORD`, and it can be retrieved with a procedure `clidata`. For example, for the Cli ord algebra C_{2} (also denoted as C_{2}) of the Euclidean plane $\mathbb{R}^{2}$ we find:

```
> clidata([2,0]); #Clifford algebra of the Euclidean plane
```

$$[\quad, 2, \quad \omega\ , \frac{1}{2}I + \frac{1}{2}\ 1, [I\ ,\ 2], [I\ ], [I\ ,\ 2]]$$

The meaning of the first three entries in the above output list is that C_{2} is a simple algebra isomorphic to Mat$(2,\mathbb{R})$. The 4th entry in the list gives a primitive idempotent f that has been used to generate a minimal left spinor ideal $S = C_{2}f$ and, subsequently, the left spinor (lowest dimensional and faithful) representation of C_{2} in S. In general it is known that, depending on (p,q) and $n = \dim V$, the spinor ideal $S = C\ f$ is a right K-module where K is either $\mathbb{R}, \mathbb{C}$, or $\mathbb{H}$ for simple Cli ord algebras when $(p\ q) \equiv 1 \bmod 4$, or $\mathbb{R}\ \mathbb{R}$ and $\mathbb{H}\ \mathbb{H}$ for semisimple algebras when $(p\ q) = 1 \bmod 4$ [14,17]. Elements in the 5th entry (here $[I\ ,\ 2]$) generate a real basis in S with respect to f, that is, $S = \text{span}\ Id\&cf, e2\&cf^{1} = \text{span}\ f, e2\&cf^{1}$. Elements in the 6th entry span a subalgebra F of $C(Q)$ that is isomorphic to K. In the case of C_{2} we find that $F = \text{span}\ Id^{1} = \mathbb{R}$. The last entry in the output gives 2 generators of S (with respect to f) viewed as a right module over K where $k = q\ \ r$ and r is the Radon-Hurwitz number.[1] Number k is the number of factors $\frac{1}{2}(1+T)$, where T^{1}, $i = 1,\ldots,k$, is a set of commuting basis Graßmann monomials squaring in $C(Q)$ to 1, whose product gives a primitive idempotent f in $C(Q)$. Spinor representation for all Cli ord algebras $C(Q)$, $1 \quad n = p+q \quad 9$, and for any signature (p,q) has been pre-computed [1] and can be retrieved from `CLIFFORD` with a procedure `matKrepr`. For example, 1-vectors $\mathbf{e}_{1}$ and $\mathbf{e}_{2}$ in C_{2} have the following spinor representation in the basis $f, e2\&cf^{1}$ of $S = C_{2}f$:[2]

```
> matKrepr([2,0]);
```

$$[\ 1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ \ 2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}]$$

In another example, Cli ord algebra C of $\mathbb{R}$ is isomorphic with Mat$(2,\mathbb{C})$:

```
> B:=linalg[diag](1,1,1):clidata([3,0]);
```

---

[1] Type `?RHnumber` in a Maple session when `CLIFFORD` is installed for more help.

[2] We use the sloppy notation $1 \equiv$ in Cli ord algebra valued matrices which produces a simpler display.

$$[\ \ \omega\ x,\ 2,\ \ \omega\ ,\ \frac{1}{2}\ I\ +\frac{1}{2}\ \ 1,\ [I\ ,\ \ 2,\ \ 3,\ \ 23],\ [I\ ,\ \ 23],\ [I\ ,\ \ 2]]$$

and its spinor representation is given in terms of Pauli matrices:

```
matKrepr([3,0]);
```

$$[\ 1\ =\ \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ \ 2\ =\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ \ 3\ =\ \begin{bmatrix} 0 & 23 \\ 23 & 0 \end{bmatrix}\ ]$$

Notice that F = span $\{Id, e23\}$ ($e23 = 2w\ 3$) is a subalgebra of C isomorphic to $\mathbb{C}$. Since Pauli matrices belong to $\mathrm{Mat}(2, F)$, it is necessary for `CLIFFORD` to compute with Clifford matrices, that is, matrices of a type `climatrix` with entries in a Clifford algebra.

```
M1,M2,M3:=rhs(%[1]),rhs(%[2]),rhs(%[3]);
```

$$M1,\ M2,\ M3\ :=\ \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ \begin{bmatrix} 0 & 23 \\ 23 & 0 \end{bmatrix}.$$

Of course Pauli matrices satisfy the same defining relations as the basis vectors e_1, e_2, and e_3: For example:

```
'M1 &cm M2 + M2 &cm M1' = evalm(M1 &cm M2 + M2 &cm M1);
'e1 &c e2 + e2 &c e1'=e1 &c e2 + e2 &c e1;
```

in $C\ (Q)$ in terms of a single matrix over a double field $\mathbb{R}\quad\mathbb{R}$ or $\mathbb{H}\quad\mathbb{H}$ rather than as pair of matrices. [4]

One can easily list signatures of the quadratic form Q for which $C\ (Q)$ is simple or semisimple. For more information, type `?all_sigs`. For example, $C\quad$ has a spinor representation given in terms of 2 by 2 quaternionic matrices, whose entries belong to a subalgebra F of $C\quad$ spanned by $Id, e2, e3, e2we3$ [1] :

```
> B:=linalg[diag](1,-1,-1,-1):clidata([1,3]);
```

$$[q\ \ ,\ 2,\ \ ,\ \frac{1}{2}\,I\ +\frac{1}{2}\ \ 1w\ 4, [I\ ,\ 1,\ 2,\ 3,\ 12,\ 13,\ 23,\ 123],$$
$$[I\ ,\ 2,\ 3,\ 23], [I\ ,\ 1]]$$

```
> matKrepr([1,3]); #quaternionic matrices
```

$$[\ 1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ \ 2 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix},\ \ 3 = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix},\ \ 4 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}]$$

`CLIFFORD` includes several special-purpose procedures to deal with quaternions and octonions (type `?quaternion` and `?octonion` for help). In particular, following [18], octonions are treated as para-vectors in C

3 Cli ord product

cursive evaluation many superfluous terms appear that later cancel out at the next recursive call. When the bilinear form is sparse numeric, many branches of the recursion are cut out by Maple quite early due to automatic evaluation that takes precedence over the recursion. In this case, the superfluous terms disappear and are not passed on to the next recursive step. However, in the symbolic case, in general, all these terms might be non-zero which prevents fast completion of the recursion. Fortunately, Hopf combinatorial methods free of the drawbacks of the recursion can also be applied and have been encoded in `cmulRS`. Thus, the two ways to evaluate the Cli ord product in `CLIFFORD` have emerged.

We introduce the Chevalley deformation and the Cli ord map to explain the algorithm used in `cmulNUM`. The Cli ord map $\gamma_{\mathbf{x}}$ is defined on $u \in \bigwedge V$ as

(i) $\gamma_{\mathbf{x}}(u) = \mathtt{LC}(x, u, B) + \mathtt{wedge}(x, u) = \mathrm{x} \lrcorner u + \mathrm{x} \wedge u$
(ii) $\gamma_{\mathbf{x}}\gamma_{\mathbf{y}} = \gamma_{\mathbf{x} \wedge \mathbf{y}} + B(\mathrm{x}, \mathrm{y})\gamma_{\mathbf{1}}$
(iii) $\gamma_{\mathbf{x}+\mathbf{y}} = a\gamma_{\mathbf{x}} + b\gamma_{\mathbf{y}}$

where $\mathrm{x}, \mathrm{y} \in V$ (see, for example, [19]). One knows how to compute with the wedge $\mathrm{x} \wedge u$ and the left contraction $\mathrm{x} \lrcorner u$ with respect to the bilinear form B (in `CLIFFORD`, the left contraction $\lrcorner$ is given by the procedure $\mathtt{LC}(\mathrm{x}, u, B)$). Following Chevalley, the left contraction has the following properties:

(i) $\mathrm{x} \lrcorner \mathrm{y} = B(\mathrm{x}, \mathrm{y})$
(ii) $\mathrm{x} \lrcorner (u \wedge v) = (\mathrm{x} \lrcorner u) \wedge v + \hat{u} \wedge (\mathrm{x} \lrcorner v)$
(iii) $(u \wedge v) \lrcorner w = u \lrcorner (v \lrcorner w)$

where $\mathrm{x} \in V$, $u, v \in \bigwedge V$ and $\hat{u}$ is the Graßmann grade involution. Hence we can use the Cli ord map $\gamma_{\mathbf{x}}$ (Chevalley deformation of the Graßmann algebra) to define a Cli ord product of a one-vector x and a multivector u as

$$\mathrm{x}u = \mathrm{x} \lrcorner u + \mathrm{x} \wedge u.$$

Analogous formula can also be given for a right Cli ord map using the right contraction $\llcorner$ implemented as the procedure

$$
\begin{aligned}
(e_- \; e_2) \;\&c\; (e_- \; e_4) &= (e \;\&c\; e_2) \;\&c\; (e_- \; e_4) \quad B(e \, , e_2)1 \;\&c\; (e_- \; e_4) \\
&= e \;\&c\; (B(e_2, e \,)e_4 \quad B(e_2, e_4)e \; + e_2 \; e_- \; e_4) \\
&\qquad B(e \, , e_2)1 \;\&c\; (e_- \; e_4)
\end{aligned}
$$

and a second recursion of the process gives now

$$
\begin{aligned}
= B(e_2, e \,)B(e \, , e_4) \quad B(e_2, e_4)B(e \, , e \,) + B(e_2, e \,)(e_- \; e_4) \\
B(e_2, e_4)(e_- \; e \,) + \mathrm{B}(e_1, e_2)(e_3 \; e_4) \quad B(e \, , e \,)(e_2 \; e_4) \\
+ B(e \, , e_4)(e_2 \; e \,) + e_- \; e_2 \; e_- \; e_4 \quad \mathrm{B}(e_1, e_2)(e_3 \; e_4)
\end{aligned}
$$

with the bolded terms cancelling out. Note that the last term in the r.h.s. was superfluously generated in the first step of the recursion.

The Cli ord product can be derived from the above recursion by linearity and associativity. The induction starts with a left factor of grade one or grade zero which is trivial, i.e., $1 \;\&c\; e_- \; \ldots \; e = e_- \; \ldots \; e$. In the case when the left factor is of grade one, we use the Cli ord product expressed by the Cli ord map of Chevalley, i.e., $e \;\&c\; e_- \; \ldots \; e = e \lrcorner (e_- \; \ldots \; e) + e_- \; e_- \; \ldots \; e$. We make a complete induction in the following way: If the left factor is of higher grade, say n, one application of the recursion yields Cli ord products where the new left factor is of grade either $n \quad 1$ or $n \quad 2$, hence the recursion stops after at most $n \quad 1$ steps.

A disadvantage of the recursive approach is that additional terms are produced by shifting Graßmann wedge products into Cli ord products in order to swap one factor to the right. While these terms eventually cancel out, their computation increases unnecessarily the total computing time. More importantly, they may easily exhaust any computer memory available and prevent Maple from completing the computation of the product.

An advantage of the recursive approach is realized when the bilinear form B is numeric and sparse, that is, with many zeros. In this case, after each recursive step many terms drtr28putercomputer memo32(tom)-3mo3autom(ofccancel)tertastantlys.

computation in Maple will be performed as follows:

```
> cmul(e1we2,e3we4);
```

$$(B_{2}\ B_{4} - B_{24}B\ )I + B_{2}\ 14 - B_{24}\ 13 - B\ 24 + B_{4}\ 23 + 1234$$

Notice also that `cmul` accepts an arbitrary bilinear form K as its argument:

```
> cmul[K](e1we2,e3we4);
```

$$(K_{2}\ K_{4} - K_{24}K\ )I + K_{2}\ 14 - K_{24}\ 13 - K\ 24 + K_{4}\ 23 + 1234$$

and likewise its ampersand form [8]

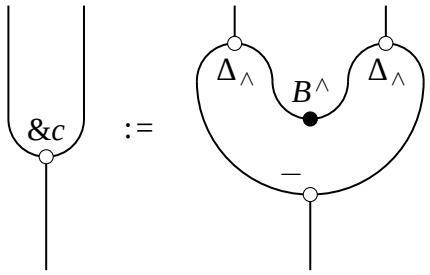
```
> &c[K](ei,ejwekwel);
```

$$w\ jw\ kw + K\ kw - K\ jw + K\ jw\ k$$

where we have also shown the ability of `CLIFFORD` to use symbolic indices. For clarity and to show our approach we display the algorithm of `cmulNUM` in Appendix A.

3.2 P ‘ cmulRS ’ h - ’ ’ ’

The procedure `cmulRS` is computed using the non-recursive Rota-Stein cliffordization. See [4,5,12,20] and `BIGEBRA` help pages for additional references. The cli ordization process is based on the Hopf algebra theory. The Cli ord product is obtained from the Graßmann wedge product and its Graßmann co-product as shown by the following tangle:



(2)

Here — is the Graßmann exterior wedge product and $\Delta_{\wedge}$ is the Graßmann exterior co-product which is obtained from the wedge product by a categorial duality: To every algebra over a linear space A with a product we find a co-algebra with a co-product over the same space by reversing all arrows in all axiomatic commutative diagrams. Note that the co-product splits each input ‘factor’ x into a sum of tensor products of ordered pairs $x_{(\ )}, x_{(2)}$. The main

[8] Procedures `cmulNUM` and `cmulRS` do not have their special ampersand forms. Procedure &*c* uses internally `cmulNUM` or `cmulRS` depending on the current value of an environmental variable `_default_Clifford_product`. Current value of this and other environmental variables can be displayed by a procedure `CLIFFORD_ENV`.

requirement is that every such pair multiplies back to the input x when the dual operation of multiplication is applied, i.e., $x_{(1)} \cdot x_{(2)} = x$ for each i-th pair. The 'cup' like part of the tangle decorated with $B^\wedge$ is the bilinear form B on the generating space V extended to the whole Graßmann algebra: It is a map $B^\wedge : \bigwedge V \otimes \bigwedge V \to k$ with $B : V \otimes V \to k$ evaluating to $B(x, y)$ on vectors in V. Hence, `cmulRS` computes the Cli ord product on Graßmann basis monomials x and y for the given B, which is later extended to Cli ord polynomials by bilinearity, as follows:

$$\texttt{cmulRS}(x, y, B) = \sum_{=} \sum_{=} (\quad) x_{(1)} \cdot y_{(2)} \, B(x_{(2)}, y_{(1)}) \tag{3}$$

where n and m give the cardinalities of the required splits and the sign is due to the parity of a permutation needed to arrange the factors.

A simplified algorithm of `cmulRS` looks as follows:

```
cmulRS(x,y,B) [x, y two Grassmann monomials, B - bilinear form]
begin
 lstx <- list of indices from x
 lsty <- list of indices from y
 NX <- length of lstx
 NY <- length of lsty
 funx <- function maps integers 1..NX onto elements of lstx keeping their order
 funy <- function maps integers 1..NY onto elements of lsty keeping their order
  (this is to calculate with arbitrary indices and to compute necessary signs)
 psetx <- power set of 1..NX (actually a list in a certain order)
  (the i-th and (2^NX+1-i)-th element are disjoint adding up to the set 1..NX)
 psety <- power set of 1..NY (actually a list in a certain order)
  (the i-th and (2^NY+1-i)-th element are disjoint adding up to the set 1..NY)
  (for faster computation we sort this power sets by grade)
  (we compute the sign for any term in the power set)
 psetx <- sort psetx by grade
 psety <- sort psety by grade
 pSgnx <- sum_(i in psetx) (-1)^sum_(j in psetx[i]) (psetx[i][j]-j)
 pSgny <- sum_(i in psety) (-1)^sum_(j in psety[i]) (psety[i][j]-j)
 (we need a subroutine for cup tangle computing the bilinear form cup(x,y,B))
  begin cup
   if |x| <> |y| then return 0 end if
   if |x| = 0 then return 1 end if
   if |x| = 1 then return B[x[1],y[1]] end if
   return sum_(j in 1..|x|)(-1)^(j-1)*B(x[1],y[j])*cup(x[2..-1],y/y[j],B)
  end cup
 (now we compute the double sum, to gain e ciency we do this grade wise)
 (note that there are r over NX r-vectors in psetx, analogously for psety)
```

```
max_grade - |lstx <- convert_to_set union lsty <- convert_to_set|
res <- 0, pos1 <- 0
for j from 0 to NX (iterate over all j-vectors of psetx)
 begin
   F1 <- N1!/((N1-j)!*j!) (number of terms (N1 over j))
   pos2 <- 0
  for i from 0 to min(N2,max_grade-j)
  0
```

in various situations when one needs pairs of such algebras. This leads to a relative isomorphism, which is then mathematically and physically relevant. We just mention two places where the dotted wedge appears.

In quantum field theory one needs to study various orderings of field operator products and/or correlation functions. In fermionic quantum field theory, a normal ordered product is expressed in terms of graded-commutative wedge products. A transition to time ordered products resp. correlation functions is equivalent to a transition to the dotted wedge products. The antisymmetric bilinear form in this case is called a W h , see [9–11].
In the theory of group representations one wants to deduce characters of subgroups of a given group by branching laws. If one derives the branching $U(n)$ $U(n$ 1) one encounters a pair of products which are related to the transition from the undotted to the dotted wedge, see [13].

In general, one can use Hopf algebra cohomology to classify maps which connect the various products. From this analysis it is known that algebra isomorphisms are related to 1-cocycles. The 1-cocycle condition guarantees that the transition is an algebra homomorphism. Below, we investigate in which way the wedge product –related to the creation operators– and the contraction –related to the annihilation operators– is a ected by the algebra isomorphism induced by the antisymmetric part F of a bilinear form B. This analysis can be extended to symmetric algebras [13] and to superspaces [9].

It was shown above that `CLIFFORD` uses the Graßmann algebra $\bigwedge V$ as the underlying vector space of the Cli ord algebra $C\ (V, B)$. Thus, the Graßmann wedge basis of monomials is the standard basis used in `CLIFFORD`. A general element u in $C\ (V, B)$ can be therefore viewed as a Graßmann polynomial.

When the bilinear form B has an antisymmetric part F = F , it is convenient to split it as $B = g + F$, where g is the symmetric part of B, and to introduce the so called "dotted Graßmann basis" [6] and the dotted wedge product $\dot{}$. The original Graßmann basis will be referred to here as the "undotted Graßmann basis". In `CLIFFORD`, the wedge product is given by the procedure `wedge` and &*w* while the dotted wedge product is given by `dwedge` and &*dw*.

According to the Chevalley definition of the Cli ord product &*c*, we have

$$\mathtt{x}\ \&\mathtt{c}\ u = \mathtt{x} \lrcorner u + \mathtt{x}\ \&\mathtt{w}\ u = \mathtt{LC}(\mathtt{x}, u, B) + \mathtt{wedge}(\mathtt{x}, u) \qquad (4)$$

for a 1-vector x and an arbitrary element u of $C\ (B)$. As before, $\mathtt{LC}(\mathtt{x}, u, B)$ denotes the left contraction of u by x with respect to the bilinear form B.

However, when $B = g + F$ then

$$\mathrm{x} \lrcorner u = \mathtt{LC}(\mathrm{x}, u, B) = \mathrm{x} \lrcorner u + \mathrm{x} \lrcorner_F u = \mathtt{LC}(\mathrm{x}, u, g) + \mathtt{LC}(\mathrm{x}, u, F) \quad (5)$$

and

$$\begin{aligned} \mathrm{x} \,\&\mathtt{c}\, u &= \mathtt{LC}(\mathrm{x}, u, B) + \mathrm{x} \,\&\mathtt{w}\, u && (6)\\ &= \mathtt{LC}(\mathrm{x}, u, g) + \mathtt{LC}(\mathrm{x}, u, F) + \mathrm{x} \,\&\mathtt{w}\, u && (7)\\ &= \mathtt{LC}(\mathrm{x}, u, g) + \mathtt{dwedge}[F](\mathrm{x}, u) = \mathtt{LC}(\mathrm{x}, u, g) + \mathrm{x} \,\&\mathtt{dw}\, u && (8) \end{aligned}$$

where $\mathrm{x} \,\&\mathtt{dw}\, u = \mathrm{x} \,\&\mathtt{w}\, u + \mathtt{LC}(\mathrm{x}, u, F)$. That is, the wedge and the dotted wedge "di er" by the contraction term(s) with respect to the antisymmetric part F of B. This dotted wedge $\&dw$ can be extended to elements of higher grades. Its properties are discussed next.

4.2 dwedge &dw

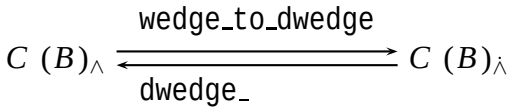
Procedure dwedge (and its infix form $\&dw$) requires an index which can be a symbol or an antisymmetric matrix. That is, dwedge computes the dotted wedge product of two Graßmann polynomials and expresses its answer in the undotted basis. Special procedures exist which convert polynomials between the undotted and dotted bases. When no index is used, the default is F :

```
dwedge[K](e1+2*e2we3,e4+3*e1we2);&dw(ei+2*ejwek,ei+2*ejwek);
```

$$\begin{aligned} &(K_4 + 6K_2 K_2) I - 6K_2\, 2w\, 3 - 6K_2\, 1w\, 2 - 2K_2\, 4\, 3 + \\ &\quad 2K_4\, 2 - 3K_2\, 1 + 1w\, 4 + 2\, 2w\, 3w\, 4 \\ &\qquad 4\, w\, jw\, k - 4F\, j + 4F\, k - 8F\, jw\, k - 4F^2 I \end{aligned}$$

Observe that conversion from the undotted wedge basis to the dotted wedge basis using antisymmetric form F and dwedge[F] are related through the following convert function:

$$\mathtt{dwedge}[F](e1, e2, \ldots, en) = \mathtt{convert}(e1we2w\ldots wen, \mathtt{wedge\_to}$$


wedge_to_dwedge
C (B)∧
C (B)∧̇
dwedge_

Here u

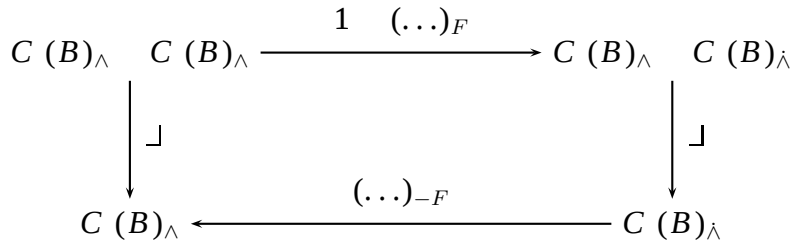
Now we map the convert function onto this basis to get the dotted wedge basis:

```
> d_bas:=map(convert,w_bas,wedge_to_dwedge,F);
> test_wbas:=map(convert,d_bas,dwedge_to_wedge,-F);
```

$$\_ := [I,\ 1,\ 2,\ 3,\ 1w\ 2 + F_2 I,\ 1w\ 3 + F\ I,\ 2w\ 3 + F_2\ I,\ 1w\ 2w\ 3 + F_2\ 1\ F\ 2 + F_2\ 3]$$

$$\_w := [I,\ 1,\ 2,\ 3,\ 1w\ 2,\ 1w\ 3,\ 2w\ 3,\ 1w\ 2w\ 3]$$

Notice that only the unity 1 and the one vector basis elements e remain


$C\ (B)_\wedge$
$C\ (B)_\wedge$
$1 \quad (\ldots)_F$
$C\ (B)_\wedge$
$C\ (B)_{\dot\wedge}$
$(\ldots)_{-F}$
$C\ (B)_\wedge$
$C\ (B)_{\dot\wedge}$

Diagram

procedure `cmul` that takes a bilinear form as its index. As an example, we will use two most general elements u and v in $\bigwedge V$ when $\dim V = 3$. Most output will be eliminated.

```
u:=add(x.k*w_bas[k+1],k=0..7):v:=add(y.k*w_bas[k+1],k=0..7):
```

We can then define in $\bigwedge V$ a Cli ord product `cmul[g]` with respect to the symmetric part g and another Cli ord product `cmul[B]` with respect to the entire form B :

```
cmulg:=proc() return cmul[g](args) end proc:
cmulB:=proc() return cmul[B](args) end proc:
```

Thus, we are ready to perform computations around our next commutative diagram, however most output will be eliminated to save space.

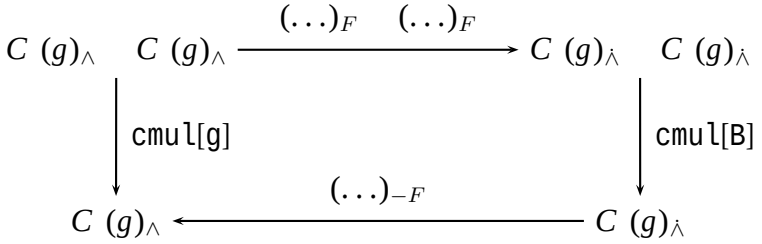


Diagram 5. Cli ord multiplications `cmul[g]` and `cmul[B]` w.r.t. dotted and undotted basis.

First, we compute the Cli ord product `cmul[g]`(u, v) in $C(g)$ in undotted Graßmann basis.

```
uv:=cmulg(u,v): #Clifford product w.r.t. g in Cl(g) in wedge basis
```

Now, we convert u and v to u_F and v_F, respectively, expressed in the dotted wedge basis:

```
uF:=convert(u,wedge_to_dwedge,F):vF:=convert(v,wedge_to_dwedge,F):
```

We now compute the Cli ord product of u_F and v_F in $C(B)$ in the dotted wedge basis,

```
uFvF:=cmulB(uF,vF): #Clifford product in Cl(B) in dwedge basis
```

convert back the above result back to the undotted wedge basis:

```
uv2:=convert(uFvF,dwedge_to_wedge,-F): #convert result dwedge->wedge
```

and verify that the results are the same:

```
simplify(uv-uv2); #show equality!
```

$$0$$

Thus, we have shown that the following identity involving `cmul[g]` and `cmul[B]` is true (at least when $\dim V = 3$). The result is folklore, and may be found e.g. in [7,15].

$$(u\,v) \;=\; u \,\&\mathtt{c}\; v = (u_F \,\&\mathtt{c}\; v_F)_{-F} = ((u_F\, v_F)\;)_{-F} \tag{11}$$

This shows that the Cli ord algebra $C(g)$ of the symmetric part g of B using the undotted exterior basis is isomorphic, as an associative algebra, to the Cli ord algebra $C(B)$ of the entire bilinear form $B = g + F$ spanned by the dotted wedge

---

Here, $(u\,v)$ is the Cli ord product with respect to g while $u_F \,\&\mathtt{c}\; v_F$ and $(u_F v_F)$ are the Cli ord products with respect to B, that is, in $C(g)$ and $C(B)$, respectively.

basis if the antisymmetric part F of B is exactly the same as F used to connect the two bases.

$$(\ldots)_F \in \mathrm{Hom}_{\mathrm{Alg}}(C\ (g), C\ (B)), \qquad B = g + F$$

4.7 *v* [illegible]

We proceed to show that the expansion of the Cli ord basis elements into the dotted or undotted exterior products has also implications for other well known operations such as the Cli ord reversion anti-automorphism $\tilde{}\ : C\ (B) \to C\ (B)$, $uv \mapsto \tilde{v}\tilde{u}$, which preserves the grades in $\dot{\bigwedge} V$ [but not in $\bigwedge V$ unless B is symmetric.] Only when the bilinear form is symmetric, we find that the reversion is grade preserving, otherwise it reflects only the filtration: That is, reversed elements are in general sums of terms of the same and lower degrees.

```
map(reversion,cbas,B);
```

$$[I,\ 1,\ 2,\ 3,\ 1w\ 2\ \ 2F_{\cdot 2}I,\ 1w\ 3\ \ 2F\ \ I,\ 2w\ 3\ \ 2F_{\cdot 2}\ I,\ 2F_{\cdot 2}\ 1 + 2F\ \ 2\ \ 2F_{\cdot 2}\ 3\ \ 1w\ 2w\ 3]$$

If instead of B we use a symmetric matrix $g = g$ (or the symmetric part of B), then

```
map(reversion,cbas,g);
```

$$[I,\ 1,\ 2,\ 3,\ 1w\ 2,\ 1w\ 3,\ 2w\ 3,\ 1w\ 2w\ 3]$$

Convert now $\wedge_{\cdot 2}$ to the dotted basis to get $\dot{\wedge}_{\cdot 2} = e1We2$:

```
convert(e1we2,wedge_to_dwedge,F);
```

$$1W\ 2$$

Apply reversion to $e1We2$ with respect to F to get the reversed element in the dotted basis:

```
reversed_e1We2:=reversion(e1We2,F);
```

$$v\ \ \_\ 1W\ 2 := \ \ 1w\ 2\ \ F_{\cdot 2}I$$

Observe, that the above element is equal to the negative of $e1We2$ just like reversing $e1we2$ with respect to the symmetric part g of B :

```
reversed_e1We2+e1We2;
```

$$0$$

Finally, convert reversed $e1We2$ to the undotted standard Graßmann basis to get

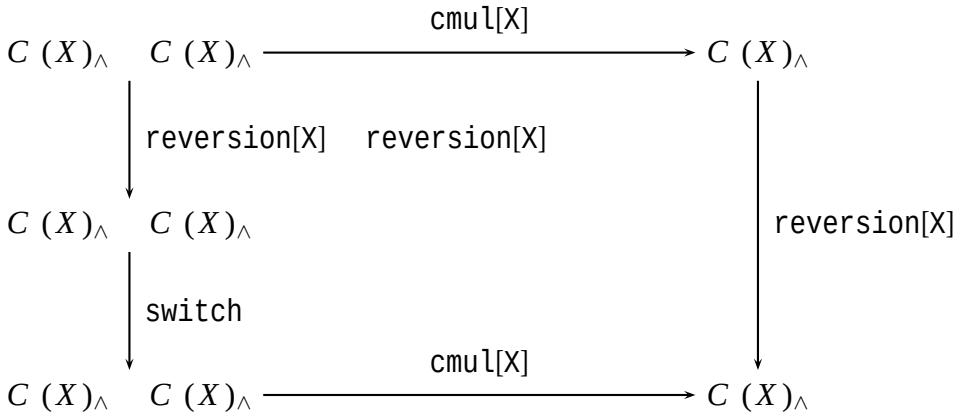


Diagram 7. Relation between the `reversion[X]` of type `X` $\in \{$`g,F,B`$\}$ with the corresponding Cli ord multiplication `cmul[X]`. The map called `switch` is the ungraded switch of tensor factors, that is, `switch`$(A \otimes B) = B \otimes A$.

5 Conclusions

This paper continues with the second part [5] about `BIGEBRA` where further aims and outlooks for the future applications of `CLIFFORD` and `BIGEBRA` are given.

Appendix A: Code of `cmulNUM`

Here is a shortened code of the recursive procedure `cmulNUM`.

```
(a1,a2,B) [a1, a2 - two Grassmann monomials, B - name of bilinear form]
  nargs <>3 h ”exactly three arguments are needed”
  has(0,map(simplify,[a1,a2])) h 0
  a2=‘Id‘ h a1
  a1=‘Id‘ h a2
```

```
p1 <- substring(a1,1..(3*N-4))
p2 <- p      (x,a2,B)
S <- clibilinear(p1,p2,p      ,B)
   -add((-1)^(i)*coB*nameB[L[-i],L[-1]]*
   p      (makeclibasmon(subs(L[-i]=NULL,L[1..-2])),a2,B),i=2..N)
   reorder(simplify(S))
```

The first author, R.A., gratefully acknowledges financial support from the College of Arts and Sciences, Tennessee Technological University, to present a preliminary version of this paper at the ACA 2002 in Volos, Greece.

References

[1] Ablamowicz, R.: Spinor representations of Cli ord algebras: A symbolic approach,

[10] Fauser, B.: On an easy transition from operator dynamics to generating functionals by Cli ord algebras, *J. Math. Phys.* **39** (1998) 4928–47 (Preprint hep-th/9710186)

[11] Fauser, B.: Cli ord geometric parameterization of Wick normal-ordering, *J. Phys. A: Math. Gen.* **34** (2001) 105–15

[12] Fauser, B.: *A Treatise on Quantum Cli ord Algebras* (Habilitationsschrift, Universität Konstanz, Konstanz, 2002)

[13] Fauser, B., and Jarvis, P.D.: A Hopf laboratory for symmetric functions, *J. Phys. A: Math. Gen.* **37** (2004) 1633–1663

[14] Helmstetter, J.: Algèbres de Cli ord et algèbres de Weyl, *Cahiers Math* **25** (1982)

[15] Helmstetter, J.: Monoïdes de Cli ord et déformations d'algèbres de Cli ord, *J. of Alg.* ( ) (1987) 14–48

[16] Lam, T.Y.: *The Algebraic Theory of Quadratic Forms* (The Benjamin Cummings Publishing Company, Reading, 1973)

[17] Lounesto, P.: Scalar products of spinors and an extension of Brauer-Wall groups, *Foundations of Physics* (1981) 721–740

[18] Lounesto, P., Mikkola, R., and Vierros, V.: