

Chapter 2

Numbers and Arithmetic

2.1 Structure of Clifford Numbers

Clifford numbers are multi-dimensional entities which follow certain simple rules for the arithmetic operations of addition and multiplication. Each Clifford number is composed of several different *components*. Table 2.1 gives examples of Clifford numbers for dimensions from zero to three.

The number of components in any Clifford number depends on its dimension. In general an m -dimensional Clifford number has 2^m components. Of course, some of these components may be of zero value. For problems in electromagnetism four-dimensional Clifford numbers are required, providing $2^4 = 16$ components. However the electromagnetic field has only six non-zero components, three for the electric part of the field and three for the magnetic part of the field. The other ten components are identically zero.

Each component in a Clifford number is made up from two parts: a *coefficient* and a *unit*.

The coefficient is a numeric entity such as a real or complex number. In algebraic calculations, when the value of the numeric coefficient is unknown, a symbolic variable which represents the value of the coefficient is used instead. It is also possible to use matrices and operators as coefficients. For problems in electromagnetism two types of coefficients are used: complex numbers and partial differential operators. When coefficients combine in operations such as addition and multiplication they behave according to the normal rules for entities of their own particular types.

The unit is a symbolic entity. By convention, the symbol is constructed from the letter e with a subscript attached, either singly or as a product. Products of units follow Clifford's rules (constraint equations):

$$e_r e_s = \begin{cases} -1 & \text{when } r = s \\ -e_s e_r & \text{when } r \neq s \end{cases} \quad (2.1)$$

Table 2.1 Examples of Clifford numbers

dimension	grade				no. of components				
	$\wedge^0$	$\wedge^1$	$\wedge^2$	$\wedge^3$	$\wedge^0$	$\wedge^1$	$\wedge^2$	$\wedge^3$	all
0	$3+4i$				1				1
1	$3+4i$	$+7e_0$			1	1			2
2	$3+4i$	$+7e_0+\pi e_1$	$+ie_0e_1$		1	2	1		4
3	$3+4i$	$+7e_0+\pi e_1+xe_2$	$+ie_0e_1+be_0e_2+ce_1e_2$	$+21.4e_0e_1e_2$	1	3	3	1	8
coefficients	$(3+4i)$	7	π	x	i	b	c	21.4	
units		e_0	e_1	e_2	e_0e_1	e_0e_2	e_1e_2	$e_0e_1e_2$	
components	$(3+4i)$	$7e_0$	πe_1	xe_2	ie_0e_1	be_0e_2	ce_1e_2	$21.4e_0e_1e_2$	

Arbitrarily complicated products of units can always be reduced to the form $\prod_p e_p$ where no two subscripts p are the same.

Note that the product of units is not commutative. From this property it follows that the product of Clifford numbers themselves is not commutative, irrespective of the type of coefficients. Coefficients and units freely commute with one another, and coefficients may or may not commute amongst themselves depending on their nature, e.g. numbers do but matrices don't.

Components are classified into *grades* according to their units. Components with no visible units, such as $(3+4i)$ in Table 2.1, fall into grade zero, denoted symbolically by a large wedge with a superscript zero, i.e. $\wedge^0$. Components in grade zero play the geometric role of *scalars*. Components containing a single visible e factor (when reduced according to the rules of Eq. 2.1) e_p , fall into grade one: $\wedge^1$. These components play the geometric role of *vectors* (oriented line segments). Components with a pair of visible e factors, e_re_s where $r \neq s$, fall into grade two, $\wedge^2$, and play the role of *bivectors* (oriented areas). Similarly, in $\wedge^3$ are components with units $e_qe_re_s$ which play the role of *trivectors* (oriented volumes). In general components containing ℓ distinct units, such as $\prod_{p=1}^{\ell} e_p$, fall into grade ℓ and play the role of ℓ -vectors.

A linear combination of all ℓ -vectors from all grades is called a *multivector*. Clifford numbers are multivectors.

2.2 Notation

2.2.1 Units

The units in Clifford numbers are written in a variety of forms, in order to simply the notation where possible.

The unit for grade zero $\wedge^0$, the grade which accommodates scalars, is variously written as 1, $e_\emptyset$ where $\emptyset$ is the empty set or, as in Table 2.1, is left unwritten. Grassmann called this the *absolute* unit, and classified it as a unit of zeroth order. Note that the absolute unit is strictly not a Clifford unit, and that when written as $e_\emptyset$ does not follow the rules 2.1 for products of Clifford units.

The units of first order, those in grade one $\bigwedge^1$, the grade which accommodates vectors, are written either as the symbol e with integer subscripts, for example e_0 , e_1 , e_2 , or with lower case alphabetic subscripts, for example e_q , e_r , e_s . The alphabetic subscripts are usually taken to represent integers although that is by no means necessary. These units are referred to as *primal* units, being the basis from which all units of higher order are generated. Note that the (first order) primal (vector) unit $e_0 \in \bigwedge^1$ is not the same as the (zeroth order) absolute (scalar) unit $e_\emptyset \in \bigwedge^0$.

Higher order units, for grades beyond $\bigwedge^1$, are either written directly as products of primal units, for example e_0e_1 , e_re_s , $e_qe_3e_4$, or written as the symbol e with uppercase alphabetic subscripts, for example e_A , e_B , e_S . The uppercase alphabetic subscripts are taken to represent sets. These sets contain subscripts p for all of the primal units e_p which must be formed into a product to generate the units of higher order. For example:

$$\begin{aligned} e_A & \text{ where } A = \{0, 1, 2\} \Rightarrow e_A = e_0e_1e_2 \\ e_B & \text{ where } B = \{q, r, s\} \Rightarrow e_B = e_qe_re_s \\ e_S & \text{ where } S = \{A, B\} \Rightarrow e_S = e_Ae_B = e_0e_1e_2e_qe_re_s \end{aligned} \quad (2.2)$$

These higher order units are called *evolved* units, since the Clifford product which generates them reduces to the outer product (as in Sect. 1.2.2) and thereby to Grassmann's process of evolution. The unit e_S is an evolved unit composed of two factors e_A and e_B . Units e_A and e_B are themselves both evolved units, each composed of three factors. Those six factors are all in grade one (i.e. they are primal units), so they contain no further factors apart from themselves and the absolute unit. Strictly speaking that final (trivial) factorisation means that primal units can also be considered as evolved from the absolute unit as generating element using themselves as the measure of generation.

When units are expanded fully in primal form it is easy to identify the grade to which they belong. When units are written in evolved form, the grade to which the unit belongs is not always immediately apparent.

As sets, the subscripts of all evolved units contain also the empty (null) set $\emptyset$. The evolved unit e_S of Eq. 2.2 could also be expanded as:

$$e_S \text{ where } S = \{A, B\} \Rightarrow e_S = e_\emptyset e_A e_B = e_\emptyset e_0 e_1 e_2 e_q e_r e_s \quad (2.3)$$

Although it is usually convenient to leave the absolute unit $e_\emptyset = 1$ unwritten, it is important to take the empty set explicitly into account when performing set operations on the subscripts of evolved units, as in Sect. 2.3.3.

2.2.2 Vectors

The notation used here to write Clifford vectors takes the form of a summation over a lowercase (integer) index of components:

$$x = \sum_p x_p e_p \quad (2.4)$$

The x_p are the coefficients, and the e_p are the units, all being primal units (from grade one). The index p varies through some range of integers. For an m -dimensional vector the index plays the role of m integers, one for each of the m units therein. Integer values may well be chosen in the range from 0 to $m - 1$ or from 1 to m , according to preference.

For example, a four-dimensional Clifford vector can be written:

$$x = \sum_{p=0}^3 x_p e_p = x_0 e_0 + x_1 e_1 + x_2 e_2 + x_3 e_3 = \sum_{p=1}^4 x'_p e'_p = x'_1 e'_1 + x'_2 e'_2 + x'_3 e'_3 + x'_4 e'_4 \quad (2.5)$$

where $x'_p = x_{p-1}$ and $e'_p = e_{p-1}$.

For problems in electromagnetism the unit e_0 is taken to represent the dimension of time or frequency, and the units e_1, e_2, e_3 are taken to represent the three orthogonal dimensions of space.

2.2.3 Numbers

The notation used here to write Clifford numbers more general than vectors takes the form of a summation over an uppercase (set) index of components:

$$x = \sum_S x_S e_S \quad (2.6)$$

The x_S are the coefficients, and the e_S are units from all grades, being either absolute, primal, or evolved. The index S varies through some range of sets. For an m -dimensional number the index plays the role of 2^m sets, one for each of the 2^m units in an m -dimensional Clifford number.

For example, a four-dimensional Clifford number can be written:

$$\begin{aligned} x = \sum_{S \subseteq \{0,1,2,3\}} x_S e_S &= x_\emptyset e_\emptyset \\ &+ x_0 e_0 + x_1 e_1 + x_2 e_2 + x_3 e_3 \\ &+ x_{0,1} e_0 e_1 + x_{0,2} e_0 e_2 + x_{0,3} e_0 e_3 + x_{1,2} e_1 e_2 + x_{1,3} e_1 e_3 + x_{2,3} e_2 e_3 \\ &+ x_{0,1,2} e_0 e_1 e_2 + x_{0,1,3} e_0 e_1 e_3 + x_{0,2,3} e_0 e_2 e_3 + x_{1,2,3} e_1 e_2 e_3 \\ &+ x_{0,1,2,3} e_0 e_1 e_2 e_3 \end{aligned} \quad (2.7)$$

where each of the five grades is written on a separate line on the right hand side. The empty (null) set has been included explicitly in the expansion for grade zero, but left unwritten for the other grades.

2.3 Arithmetic Operators

2.3.1 Inversion of Signs

For complex numbers it is possible to invert the sign of either the real or imaginary components separately using the operations of negation and complex conjugation. For Clifford numbers there are many more components and many more possibilities. Table 2.2 shows in addition to the operation of negation, three other operations on Clifford numbers: involution, reversion and Clifford conjugation. All three of these involve inverting the signs of some of the components according to a particular rule or pattern based on the grade of the units in those components, as shown in Table 2.3.

The operation of *involution* (denoted by the symbol ‘ $\bar{}$ ’) involves changing the sign of every primal unit e_p in every evolved unit e_S . The effect is to multiply every component containing a unit in grade ℓ by the factor of $\xi = (-1)^\ell$. The overall effect is to invert the sign of all odd numbered grades, as shown in Table 2.3.

Table 2.2 Operations involving inversion of sign

Operation	$f(x)$	Definition		Effect on primal units		Effect on evolved unit
Identity	x	$x = \sum_S x_S e_S$	$e_S = \prod_{p=1}^\ell e_p = \xi e_S$			$\xi = 1$
Negation	$-x$	$-x = \sum_S x_S (-e_S)$	$-e_S = -\prod_{p=1}^\ell e_p = \xi e_S$			$\xi = -1$
Involution	$\bar{x}$	$\bar{x} = \sum_S x_S \bar{e}_S$	$\bar{e}_S = \prod_{p=1}^\ell (-e_p) = \xi e_S$		Change sign	$\xi = (-1)^\ell$
Reversion	$\tilde{x}$	$\tilde{x} = \sum_S x_S \tilde{e}_S$	$\tilde{e}_S = \prod_{p=1}^\ell e_{\ell+1-p} = \xi e_S$	Reverse order		$\xi = (-1)^{\ell(\ell-1)/2}$
Clifford conjugation	$\bar{x}$	$\bar{x} = \sum_S x_S \bar{e}_S$	$\bar{e}_S = \prod_{p=1}^\ell (-e_{\ell+1-p}) = \xi e_S$	Reverse order	Change sign	$\xi = (-1)^{\ell(\ell+1)/2}$

Table 2.3 Periodic behaviour of reversion, involution and Clifford conjugation

grade	0	1	2	3	4	5	6	7	8	9	10	11
involution	+	-	+	-	+	-	+	-	+	-	+	-
reversion	+	+	-	-	+	+	-	-	+	+	-	-
Clifford conjugation	+	-	-	+	+	-	-	+	+	-	-	+

The operation of *reversion* (denoted by the tilde symbol ‘ $\sim$ ’) involves reversing the entire sequence of primal units e_p in every evolved unit e_s . Returning all of the primal units to their original order then requires $\ell(\ell - 1)/2$ exchanges of adjacent pairs. The effect is to multiply every component containing a unit in grade ℓ by the factor of $\xi = (-1)^{\ell(\ell-1)/2}$.

The operation of *Clifford conjugation* (denoted by the overbar symbol ‘ $\bar{}$ ’) involves both involution and reversion, i.e. both changing the sign of every primal unit e_p and also reversing the order of the entire sequence of primal units. The effect is to multiply every component containing a unit in grade ℓ by the factor of $\xi = (-1)^{\ell(\ell+1)/2}$.

2.3.2 Addition and Subtraction

Addition and subtraction of Clifford numbers is straightforward. The coefficients of components with matching units are simply added or subtracted according to the normal rules for entities of their own particular types. For two Clifford numbers $a = \sum_S a_S e_S$ and $b = \sum_S b_S e_S$, the sum $a + b = c$ is:

$$a + b = \left(\sum_S a_S e_S \right) + \left(\sum_S b_S e_S \right) = \sum_S (a_S + b_S) e_S = \sum_S c_S e_S = c \quad (2.8)$$

and the difference $a - b = c$ is:

$$a - b = \left(\sum_S a_S e_S \right) - \left(\sum_S b_S e_S \right) = \sum_S (a_S - b_S) e_S = \sum_S c_S e_S = c \quad (2.9)$$

The index set S is taken to include all of the units in both a and b .

2.3.3 Multiplication

Real, complex and Clifford multiplications have zero, one and two constraint equations respectively. Each time more constraints are applied a new kind of multiplication is created.

For Clifford numbers, in addition to the constraints on the primal units e_r, e_s as in Eq. 2.1, a whole new layer of constraints is applied at the level of the evolved units e_A, e_B . Each different constraint (or set of constraints) leads to a different multiplication.

All of these kinds of multiplication can be defined in terms of a general unconstrained product operation (see [7]), as denoted here by the symbol ‘ $\diamond$ ’. For two Clifford numbers $a = \sum_A a_A e_A$ and $b = \sum_B b_B e_B$, the general product $a \diamond b = c$ is:

$$a \diamond b = \left(\sum_A a_A e_A \right) \diamond \left(\sum_B b_B e_B \right) = \sum_A \sum_B a_A b_B (e_A \diamond e_B) = \sum_C c_C e_C = c \quad (2.10)$$

The value of the product is determined by the conventional product of its coefficients $c_C = a_A b_B$ according to the normal rules for entities of their own particular types, and the general product of the units $e_C = e_A \diamond e_B$ adhering to the constraint equations for the particular kind of multiplication the general product operation $\diamond$ is intended to deliver.

Table 2.4 shows seven different kinds of product, all distinguished by the way they adhere to the general constraint equation:

$$e_A \diamond e_B = \begin{cases} e_A e_B & \text{when } \checkmark \\ 0 & \text{otherwise} \end{cases} \quad (2.11)$$

The tick $\checkmark$ is determined for each of the kinds of multiplication according to such relationships as whether the unit e_A contains all or none of the primal units e_p found in unit e_B , and/or *vice versa* or not. The empty set $\emptyset$ corresponding to the absolute unit $e_\emptyset = 1$ is taken into account as part of the determination.

Table 2.4 shows the constraint equations for the different kinds of multiplication in set notation, and Table 2.5 shows the result of applying the constraint equations to a product of two-dimensional Clifford numbers.

Table 2.4 Constraint equations for seven different kinds of Clifford multiplication

Product	$a \diamond b$	Constraint equations				
		Excludes $B \cap A = \emptyset$	Either includes $B \supseteq A$		Vector $ B A = 1$	Non-vector $ B A \neq 1$
			Includes $B \supseteq A$	Includes $B \subseteq A$		
Central	$a b$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
Outer	$a \wedge b$	$\checkmark$			$\checkmark$	$\checkmark$
Inner	$\bar{a} \vee b$		$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
Left inner	$\bar{a} \lrcorner b$		$\checkmark$		$\checkmark$	$\checkmark$
Right inner	$a \lrcorner \bar{b}$			$\checkmark$	$\checkmark$	$\checkmark$
Scalar	$(\bar{a}, b)$		$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
Dot	$\bar{a} \cdot b$		$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$

The *central* product ab (Clifford multiplication, denoted by no visible symbol) is the most general, with no constraint equations at the level of the evolved units. All of the other products derived from the general product are contained within the central product.

The *outer* product $a \wedge b$ (wedge product, denoted by the symbol ' $\wedge$ ') is constrained to produce non-zero values only when both factors e_A, e_B are composed of units, none of which match any primal unit in the other.

The *inner* product $\bar{a} \vee b$ (denoted by the symbol ' $\vee$ ') is constrained to produce non-zero values only when either of the factors e_A, e_B is composed of units, all of which match any unit in the other.

Grassmann called the outer and inner products by those names because they give non-zero results only if the two factors e_A, e_B either lie completely *outside* the linear span of the primal units contained in each other, or one lies completely *inside* the linear span of the primal units contained in the other.

The *left inner* product $\bar{a} \lrcorner b$ (denoted by the symbol ' $\lrcorner$ ') is constrained to produce non-zero values only when the leftmost of the factors e_A, e_B is composed of units, all of which match any unit in the other.

The *right inner* product $a \llcorner \bar{b}$ (denoted by the symbol ' $\llcorner$ ') is constrained to produce non-zero values only when the rightmost of the factors e_A, e_B is composed of units, all of which match any unit in the other.

The *scalar* product $(\bar{a}, b)$ (denoted by the parenthesis-comma-parenthesis notation ' $(,)$ ') is constrained to produce non-zero values only when both of the factors e_A, e_B are composed of units, all of which match any unit in the other.

For the three inner products, and for the scalar product, the conjugation can be moved by substituting $a' = \bar{a}$ or $b' = \bar{b}$, which then leads to a product of factors $e_A e_B = \bar{e}_A \bar{e}_B$ or $e_A e_B = e_A \bar{e}_{B'}$. For the scalar product the factor to which the conjugation is applied is of little consequence since the result, always a scalar, is the same in either case.

The *dot* product $\bar{a} \cdot b$ (denoted by the symbol ' $\cdot$ ') is constrained to produce non-zero values only when both of the factors e_A, e_B are composed of the one and same primal unit. For vectors $\mathbf{a}, \mathbf{b}$ the value is given in terms of Clifford multiplication by Eq. 1.14, and for primal units e_r, e_s the value is given in Table 1.2. The dot product is a special case of the inner and scalar products, just for vectors.

The seven products listed in Table 2.5 are not all independent. The left and right inner products are defined in terms of the outer and scalar products by the two duality relationships:

$$\begin{cases} (u \lrcorner x, y) = (x, u \wedge y) \\ (x \llcorner u, y) = (x, y \wedge u) \end{cases} \quad (2.12)$$

These relationships hold for both the Hamilton, Maxwell, Grassmann, Clifford convention $e_r^2 = -1$, and for the alternative convention $e_r^2 = +1$. For the alternative convention the products in Table 2.5 employing Clifford conjugation must be modified so as to use reversion instead. It is important when reading published literature relating to Clifford algebra to determine early which convention has been used so as to avoid confusion.

For problems in electromagnetism as treated here it is the unconstrained Clifford or central multiplication which is of primary importance. For problems in differential geometry it is the outer and left inner products which play the major roles.

2.3.4 Application

Figure 2.1 shows two Clifford vectors:

$$\begin{cases} a = a_{\parallel} + a_{\perp} = e_1 \\ b = b_{\parallel} + b_{\perp} = \frac{1}{2}e_1 + \frac{\sqrt{3}}{2}e_2 \end{cases} \quad (2.13)$$

which are each broken into two components, parallel ($\parallel$) and perpendicular ($\perp$) to the other:

$$\begin{cases} a_{\parallel} = \frac{1}{4}e_1 + \frac{\sqrt{3}}{4}e_2 \\ a_{\perp} = \frac{3}{4}e_1 - \frac{\sqrt{3}}{4}e_2 \end{cases} \quad \text{and} \quad \begin{cases} b_{\parallel} = \frac{1}{2}e_1 \\ b_{\perp} = \frac{\sqrt{3}}{2}e_2 \end{cases} \quad (2.14)$$

The vector $a_{\parallel}$ is the component of a parallel to the direction of b , and the vector $a_{\perp}$ is the component of a perpendicular to the direction of b . The vectors $b_{\parallel}$ and $b_{\perp}$ behave in a similar way with respect to the direction of a .

By direct calculation the outer product:

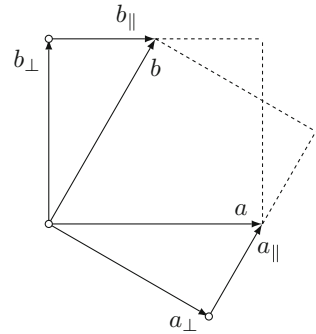
$$a_{\perp} \wedge b = a \wedge b = a \wedge b_{\perp} \quad (2.15)$$

depends only on the perpendicular components of the two vectors. By contrast, the inner product:

$$a_{\parallel} \vee b = a \vee b = a \vee b_{\parallel} \quad (2.16)$$

depends only on the parallel components of the two vectors.

Fig. 2.1 Vectors a and b broken into components perpendicular and parallel to one another



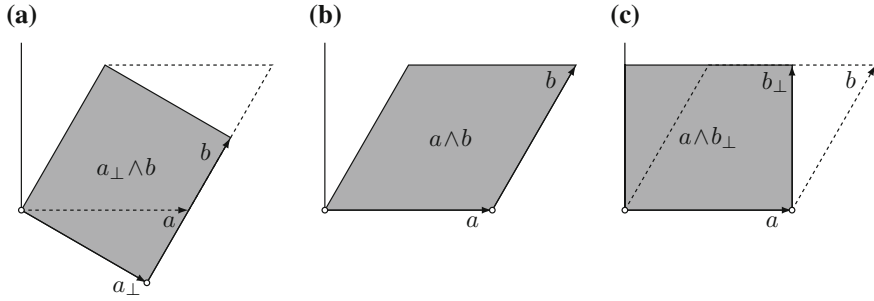


Fig. 2.2 Torque (*shaded*) calculated from outer product of (a) perpendicular component $a_{\perp}$ of force $a = F$ with whole lever-arm $b = d$, of (b) whole force $a = F$ with whole lever-arm $b = d$, and of (c) whole force $a = F$ with perpendicular component $b_{\perp}$ of lever-arm $b = d$

In the case that the vector a represents a force F applied to the tip of a lever-arm, and the vector b represents the displacement d from the tip of the level-arm to its pivot, then the geometrical interpretation shown in Fig. 2.2b for the outer product $a \wedge b = F \wedge d$, is the resulting torque as given in value by the area of the shaded portion of the figure. Figure 2.2a shows the case taking the perpendicular component of force F with the whole of the displacement d , and Fig. 2.2c shows the case taking the whole force F with the perpendicular component of the displacement d . In all cases the size of the shaded portion of the figures (i.e. the amount of torque) is the same.

In the case that the vector a represents a force F applied to an object, and the vector b represents the displacement d of that object under the influence of the force, then the geometrical interpretation shown in Figs. 2.3b and 2.4b for the inner product

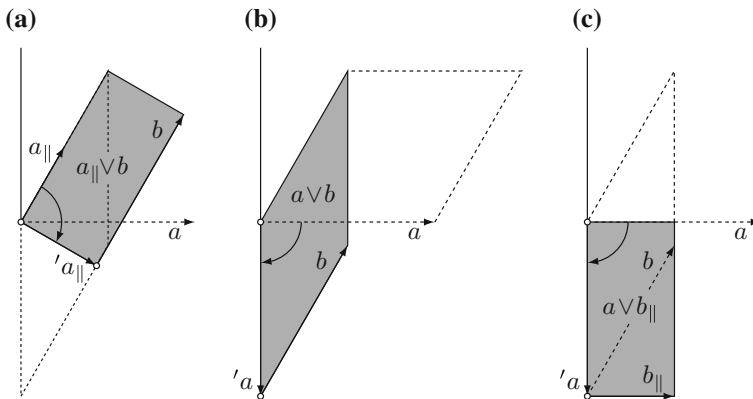


Fig. 2.3 Work (*shaded*) calculated from inner product of (a) parallel component $a_{\parallel}$ of force $a = F$ with whole displacement $b = d$, of (b) whole force $a = F$ with whole displacement $b = d$, and of (c) whole force $a = F$ with parallel component $b_{\parallel}$ of displacement $b = d$. The inner product with vector a is converted to an outer product with rotated vector $'a$

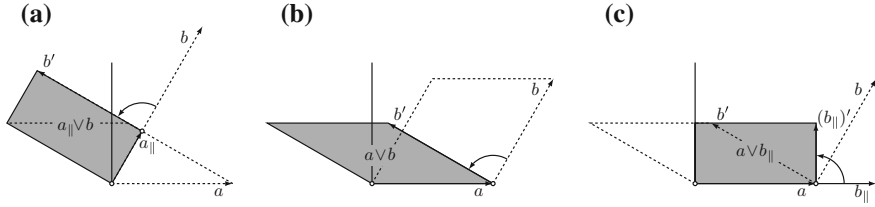


Fig. 2.4 Work (*shaded*) calculated from inner product of (a) parallel component $a_{\parallel}$ of force $a = F$ with whole displacement $b = d$, of (b) whole force $a = F$ with whole displacement $b = d$, and of (c) whole force $a = F$ with parallel component $b_{\parallel}$ of displacement $b = d$. The inner product with vector b is converted to an outer product with rotated vector b'

$a \vee b = F \vee d$, is the resulting work as given in value by the area of the shaded portion of the figures. Figures 2.3a and 2.4a show the case taking the parallel component of force F with the whole of the displacement d , and Figs. 2.3c and 2.4c shows the case taking the whole force F with the parallel component of the displacement d . In all cases the size of the shaded portion of the figures (i.e. the amount of work) is the same.

In Fig. 2.3b the vector $'a$ is constructed by rotating the vector a by -90° when angles are measured in a positive sense through the smallest arc from a to b . This converts the inner product of a and b into an evolution (outer product) of $'a$ and b . Similar conversions have been made in Fig. 2.3a, c.

In Fig. 2.4b the vector b' is constructed by rotating the vector b by $+90^\circ$. This converts the inner product of a and b into an evolution (outer product) of a and b' . Similar conversions have been made in Fig. 2.4a, c.

The torque is a bivector and the work is a scalar. Calculating the central product ab produces the torque in grade two of the result and produces the work in grade zero of the result. Depending on the physical situation it may be one or other of these components which is of most interest.

2.4 Numerical Implementation

2.4.1 Dimensional Recursion

A Clifford number $\mathcal{Z}$ of dimension $m > 0$ can be constructed by combining two Clifford numbers $\mathcal{X}$ and $\mathcal{Y}$ of dimension $m - 1$ with the unit e_m for dimension m :

$$\mathcal{Z}_{(m)} = \mathcal{X}_{(m-1)} + \mathcal{Y}_{(m-1)}e_m \quad (2.17)$$

Table 2.6 Recursive construction of Clifford numbers

Dimension	Number		Involution of number	
m	$\mathcal{Z}_{(m)}$	$\mathcal{X}_{(m-1)} + \mathcal{Y}_{(m-1)}e_m$	$\overline{\mathcal{Z}_{(m)}}$	$\overline{\mathcal{X}_{(m-1)}} - \overline{\mathcal{Y}_{(m-1)}} e_m$
$m - 1$	$\mathcal{Z}_{(m-1)}$	$\mathcal{X}_{(m-2)} + \mathcal{Y}_{(m-2)}e_{m-1}$	$\overline{\mathcal{Z}_{(m-1)}}$	$\overline{\mathcal{X}_{(m-2)}} - \overline{\mathcal{Y}_{(m-2)}} e_{m-1}$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
2	$\mathcal{Z}_{(2)}$	$\mathcal{X}_{(1)} + \mathcal{Y}_{(1)}e_2$	$\overline{\mathcal{Z}_{(2)}}$	$\overline{\mathcal{X}_{(1)}} - \overline{\mathcal{Y}_{(1)}} e_2$
1	$\mathcal{Z}_{(1)}$	$\mathcal{X}_{(0)} + \mathcal{Y}_{(0)}e_1$	$\overline{\mathcal{Z}_{(1)}}$	$\overline{\mathcal{X}_{(0)}} - \overline{\mathcal{Y}_{(0)}} e_1$
0	$\mathcal{Z}_{(0)}$	$x + iy$	$\overline{\mathcal{Z}_{(0)}}$	$x + iy$

The process can be repeated for $\mathcal{X}$ and $\mathcal{Y}$ with unit e_{m-1} , and for all lower dimensions down to Clifford numbers of dimension zero, as shown in Table 2.6. At that level, a Clifford number is simply constructed from a single complex number $z = x + iy$:

$$\mathcal{Z}_{(0)} = x + iy \quad (2.18)$$

Note that the involution operation (in the two rightmost columns) plays a role for the primal units e_p of Clifford numbers similar to the role the complex conjugate operation plays for the imaginary unit i of complex numbers. However, the involution does not operate on the imaginary unit i , so the involution of a zero-dimensional Clifford number is the same as the original number. (This is similar to complex numbers, where the complex conjugate of a real number is the same as the original number).

2.4.2 Recursive Product

The product of Clifford numbers of any dimension can be recursively defined in terms of Clifford numbers of lower dimension, in a form similar to the (more familiar) product of complex numbers. For complex numbers $u = v + iw$ and $z = x + iy$ the product is:

$$uz = (v + iw)(x + iy) = [vx - wy] + i[vy + wx] \quad (2.19)$$

For m -dimensional Clifford numbers $\mathcal{U}_{(m)} = \mathcal{V}_{(m-1)} + \mathcal{W}_{(m-1)}e_m$ and $\mathcal{Z}_{(m)} = \mathcal{X}_{(m-1)} + \mathcal{Y}_{(m-1)}e_m$ the product is:

$$\begin{aligned}
\mathcal{U}_{(m)} \mathcal{Z}_{(m)} &= (\mathcal{V}_{(m-1)} + \mathcal{W}_{(m-1)} e_m) (\mathcal{X}_{(m-1)} + \mathcal{Y}_{(m-1)} e_m) \\
&= [\mathcal{V}_{(m-1)} \mathcal{X}_{(m-1)} + \mathcal{W}_{(m-1)} e_m \mathcal{Y}_{(m-1)} e_m] \\
&\quad + [\mathcal{V}_{(m-1)} \mathcal{Y}_{(m-1)} e_m + \mathcal{W}_{(m-1)} e_m \mathcal{X}_{(m-1)}]
\end{aligned} \tag{2.20}$$

Changing the order of $e_m \mathcal{X}_{(m-1)}$ to $\mathcal{X}_{(m-1)} e_m$ introduces minus signs in all odd grades of $\mathcal{X}$. The same effect can be achieved by applying the involution operator (as described in Sect. 2.3.1) so that:

$$e_m \overline{\mathcal{X}_{(m-1)}} = \mathcal{X}_{(m-1)} e_m \tag{2.21}$$

Multiplying both sides from left and right by e_m now gives:

$$\begin{aligned}
e_m e_m \overline{\mathcal{X}_{(m-1)}} e_m &= e_m \mathcal{X}_{(m-1)} e_m e_m \\
\overline{\mathcal{X}_{(m-1)}} e_m &= e_m \mathcal{X}_{(m-1)}
\end{aligned} \tag{2.22}$$

Using this result for both $e_m \mathcal{X}_{(m-1)}$ and $e_m \mathcal{Y}_{(m-1)}$ in Eq. 2.20 gives:

$$\begin{aligned}
\mathcal{U}_{(m)} \mathcal{Z}_{(m)} &= [\mathcal{V}_{(m-1)} \mathcal{X}_{(m-1)} + \mathcal{W}_{(m-1)} \overline{\mathcal{Y}_{(m-1)}} e_m e_m] \\
&\quad + [\mathcal{V}_{(m-1)} \mathcal{Y}_{(m-1)} e_m + \mathcal{W}_{(m-1)} \overline{\mathcal{X}_{(m-1)}} e_m] \\
&= [\mathcal{V}_{(m-1)} \mathcal{X}_{(m-1)} - \mathcal{W}_{(m-1)} \overline{\mathcal{Y}_{(m-1)}}] \\
&\quad + [\mathcal{V}_{(m-1)} \mathcal{Y}_{(m-1)} + \mathcal{W}_{(m-1)} \overline{\mathcal{X}_{(m-1)}}] e_m
\end{aligned} \tag{2.23}$$

Comparison of Eq. 2.23 for the multiplication of Clifford numbers to the case for multiplication of complex numbers (Eq. 2.19) shows strong similarities. Structurally the equations are the same, with the only added complication in the case of Clifford multiplication being the need to apply the involution operation to some of the terms. The simple form of Eq. 2.23 can be exploited to develop simple (although recursive) computer implementations of Clifford multiplication, as shown in Sect. 2.4.3.

2.4.3 Machine Computation

Currently no computer languages exist which support generic data types for Clifford numbers. In order to perform machine-based arithmetic operations with Clifford numbers it is necessary both to design a suitable data construct which serves to represent them, and to develop accompanying computer routines which serve to operate on that construct.

The recursive representation of Clifford numbers described in Sect. 2.4.1 exhibits the form of a binary tree in which a number $\mathcal{Z}$ of dimension $m > 0$ consists of an ordered pair of elements (numbers) $\mathcal{X}, \mathcal{Y}$ each of dimension one lower, and each

branch of the tree terminates at the lowest dimension $m = 0$ with a single element (number) $\mathcal{Z} = a_0$, where a_0 is a complex number:

$$\mathcal{Z} := (\mathcal{X} \mathcal{Y}) \quad m > 0 \quad (2.24)$$

$$\mathcal{Z} := a_0 \quad m = 0 \quad (2.25)$$

Substituting two elements in the form of Eq. 2.25 into Eq. 2.24 gives the representation of a Clifford number of dimension $m = 1$:

$$\mathcal{Z} := (a_0 \ a_1) \quad m = 1 \quad (2.26)$$

$$\mathcal{X} \ \mathcal{Y}$$

where now both a_0 and a_1 are complex numbers. Similarly, substituting two elements in the form of Eq. 2.26 into Eq. 2.24 gives the representation of a Clifford number of dimension $m = 2$. Repeating the substitution recursively gives the representations of Clifford numbers for successively higher dimensions. The representation in tree structure of Clifford numbers $\mathcal{Z}$ of dimension $m \leq 4$ is shown in Table 2.7.

The tree structure contains only the coefficients a_s of the components of a Clifford number. The units e_s need never be recorded explicitly. They can always be determined from the position of the coefficients in the tree.

Multiplication as described in Sect. 2.4.2, of Clifford numbers represented in the tree structure, is only a little more complicated than multiplying two complex num-

Table 2.7 Machine representation in tree structure of Clifford numbers of dimension m from zero to four

$\mathcal{Z} := a_0$	$m = 0$
$\mathcal{Z} := (a_0 \ a_1)$ $\mathcal{X} \ \mathcal{Y}$	$m = 1$
$\mathcal{Z} := ((a_0 \ a_1) (a_2 \ a_3))$ $\underbrace{\hspace{1.5cm}}_{\mathcal{X}} \quad \underbrace{\hspace{1.5cm}}_{\mathcal{Y}}$	$m = 2$
$\mathcal{Z} := (((a_0 \ a_1)(a_2 \ a_3)) ((a_4 \ a_5)(a_6 \ a_7)))$ $\underbrace{\hspace{2.5cm}}_{\mathcal{X}} \quad \underbrace{\hspace{2.5cm}}_{\mathcal{Y}}$	$m = 3$
$\mathcal{Z} := (((((a_0 \ a_1)(a_2 \ a_3))((a_4 \ a_5)(a_6 \ a_7))) ((a_8 \ a_9)(a_{10} \ a_{11}))) ((a_{12} \ a_{13})(a_{14} \ a_{15}))))$ $\underbrace{\hspace{4.5cm}}_{\mathcal{X}} \quad \underbrace{\hspace{4.5cm}}_{\mathcal{Y}}$	$m = 4$

Table 2.8 Recursive multiplication of Clifford numbers in REDUCE code

```

procedure cmulx(U,Z);
if tlist(U)=1 and tlist(Z)=1
then list(subx(cmulx(first(U),first(Z)),cmulx(second(U),invc(second(Z)))),
      addx(cmulx(first(U),second(Z)),cmulx(second(U),invc(first(Z)))))
else if tlist(U)=0 and tlist(Z)=0
then U*Z else write "bad arguments to cmulx";

```

bers. By way of example Table 2.8 shows a computer implementation “`cmulx`” of Eq. 2.23 in the REDUCE programming language, for multiplying two Clifford numbers “`U`” and “`Z`” of equal but arbitrary dimension. This implementation invokes a recursive call on itself, along with calls to companion routines “`subx`”, “`addx`” and “`invx`”, as described in appendix A, for subtraction, addition and involution. The actual multiplication is performed by the single REDUCE expression “`U*Z`”.

Multiplying Clifford numbers of different dimensions is only slightly more complicated. Routines to do so are listed in appendix A, along with brief descriptions of the purpose of each routine.

Examples of machine computations on Clifford numbers with predominantly integer coefficients (for simplicity of reading) are shown in Table 2.9. Note that examples 2 and 5, of multiplication on one-dimensional Clifford numbers, behave according to the same rules as complex multiplication. Note also that the routines

Table 2.9 Examples of machine computations on Clifford numbers

```

1: printcliff(cmulx(2,5));
   10
2: printcliff(cmulx({1,2},{3,4}));
   10*e(0)-5
3: u:={v,w};
   u:={v,w}
4: z:={x,y};
   z:={x,y}
5: printcliff(cmulx(u,z));
   e(0)*(v*y+w*x)+v*x-w*y
6: printcliff(cmulx({{1,2},{3,4}},{5,6},{7,8}}));
   30*e(1)+24*e(0)*e(1)+12*e(0)-60
7: printcliff(cmulx({{1,2},{3,4}},{5,6},{7,8}},
   {{9,10},{11,12}},{13,14},{15,16}}));
   -38*e(2)-66*e(1)*e(2)+246*e(1)+204*e(0)*e(2)+192*e(0)*e(1)*e(2)
   -152*e(0)*e(1)-220*e(0)-218
8: z:=cmulx({{1,2},{3,4}},{5,6},{7,8}},
   {{9,10},{11,12}},{13,14},{15,16}});
   z:={{-218,-220},{246,-152}},{-38,204},{-66,192}}
9: printcliff(z);
   -38*e(2)-66*e(1)*e(2)+246*e(1)+204*e(0)*e(2)+192*e(0)*e(1)*e(2)
   -152*e(0)*e(1)-220*e(0)-218
10: dimc(z);
    3
11: printcliff(invz(z));
    38*e(2)-66*e(1)*e(2)-246*e(1)+204*e(0)*e(2)-192*e(0)*e(1)*e(2)
    -152*e(0)*e(1)+220*e(0)-218
12: printcliff(gradec(z,0));
    -218
13: printcliff(gradec(z,1));
    -38*e(2)+246*e(1)-220*e(0)
14: printcliff(gradec(z,2));
    -66*e(1)*e(2)+204*e(0)*e(2)-152*e(0)*e(1)
15: printcliff(gradec(z,3));
    192*e(0)*e(1)*e(2)

```

Table 2.10 Symbolic verification of Clifford product duality relationship from first line of Eq. 2.12

```

1: m:=4;
   m:=4

2: x:=makecliff(m,x);
   x:={{{{x0,x1},{x2,x3}},{x4,x5},{x6,x7}}},
       {{{x8,x9},{x10,x11}},{x12,x13},{x14,x15}}}}

3: y:=makecliff(m,y);
   y:={{{{y0,y1},{y2,y3}},{y4,y5},{y6,y7}}},
       {{{y8,y9},{y10,y11}},{y12,y13},{y14,y15}}}}

4: u:=makecliff(m,u);
   u:={{{{u0,u1},{u2,u3}},{u4,u5},{u6,u7}}},
       {{{u8,u9},{u10,u11}},{u12,u13},{u14,u15}}}}

5: lhs:=printcliff(sklar(left(u,x),y));
   lhs:=u0*x0*y0+u0*x1*y1+u0*x10*y10+u0*x11*y11+u0*x12*y12+u0*x13*y13+
u0*x14*y14+u0*x15*y15+u0*x2*y2+u0*x3*y3+u0*x4*y4+u0*x5*y5+u0*x6*y6
+u0*x7*y7+u0*x8*y8+u0*x9*y9+u1*x1*y0+u1*x11*y10+u1*x13*y12+u1*x15*
y14+u1*x3*y2+u1*x5*y4+u1*x7*y6+u1*x9*y8+u10*x10*y0+u10*x11*y1-u10*
x14*y4-u10*x15*y5+u11*x11*y0-u11*x15*y4+u12*x12*y0+u12*x13*y1+u12*
x14*y2+u12*x15*y3+u13*x13*y0+u13*x15*y2+u14*x14*y0-u14*x15*y1+u15*
x15*y0+u2*x10*y8-u2*x11*y9+u2*x14*y12-u2*x15*y13+u2*x2*y0-u2*x3*y1+
u2*x6*y4-u2*x7*y5+u3*x11*y8+u3*x15*y12+u3*x3*y0+u3*x7*y4+u4*x12*y8
-u4*x13*y9-u4*x14*y10+u4*x15*y11+u4*x4*y0-u4*x5*y1-u4*x6*y2+u4*x7*
y3+u5*x13*y8-u5*x15*y10+u5*x5*y0-u5*x7*y2+u6*x14*y8+u6*x15*y9+u6*
x6*y0+u6*x7*y1+u7*x15*y8+u7*x7*y0-u8*x10*y2+u8*x11*y3-u8*x12*y4+u8
*x13*y5+u8*x14*y6-u8*x15*y7+u8*x8*y0-u8*x9*y1-u9*x11*y2-u9*x13*y4+
u9*x15*y6+u9*x9*y0

6: rhs:=printcliff(sklar(x,outer(u,y)));
   rhs:=u0*x0*y0+u0*x1*y1+u0*x10*y10+u0*x11*y11+u0*x12*y12+u0*x13*y13+
u0*x14*y14+u0*x15*y15+u0*x2*y2+u0*x3*y3+u0*x4*y4+u0*x5*y5+u0*x6*y6
+u0*x7*y7+u0*x8*y8+u0*x9*y9+u1*x1*y0+u1*x11*y10+u1*x13*y12+u1*x15*
y14+u1*x3*y2+u1*x5*y4+u1*x7*y6+u1*x9*y8+u10*x10*y0+u10*x11*y1-u10*
x14*y4-u10*x15*y5+u11*x11*y0-u11*x15*y4+u12*x12*y0+u12*x13*y1+u12*
x14*y2+u12*x15*y3+u13*x13*y0+u13*x15*y2+u14*x14*y0-u14*x15*y1+u15*
x15*y0+u2*x10*y8-u2*x11*y9+u2*x14*y12-u2*x15*y13+u2*x2*y0-u2*x3*y1+
u2*x6*y4-u2*x7*y5+u3*x11*y8+u3*x15*y12+u3*x3*y0+u3*x7*y4+u4*x12*y8
-u4*x13*y9-u4*x14*y10+u4*x15*y11+u4*x4*y0-u4*x5*y1-u4*x6*y2+u4*x7*
y3+u5*x13*y8-u5*x15*y10+u5*x5*y0-u5*x7*y2+u6*x14*y8+u6*x15*y9+u6*
x6*y0+u6*x7*y1+u7*x15*y8+u7*x7*y0-u8*x10*y2+u8*x11*y3-u8*x12*y4+u8
*x13*y5+u8*x14*y6-u8*x15*y7+u8*x8*y0-u8*x9*y1-u9*x11*y2-u9*x13*y4+
u9*x15*y6+u9*x9*y0

7: lhs-rhs;
   0

```

operate equally well with numeric entities (as in example 2) and symbolic entities (as in example 5).

An example of machine computations to verify the first of the duality relationships from Eq. 2.12 is shown in Table 2.10. This example serves as quite a good check on the implementation of some of the different kinds of products.

2.5 Exercises

- Q1.** Show that the operations of involution, reversion and Clifford conjugation are all their own inverses.
- Q2.** Show that the application of any two of the operations of involution, reversion and Clifford conjugation has the same effect as the one not used.
- Q3.** Calculate the involution, reversion and Clifford conjugation of the number $e_1 e_3 e_3$ by applying the effects described in Table 2.2 on the primal units directly and then rearranging them into their original order.
- Q4.** Show that reversing the order of the entire sequence of primal units e_p in every evolved unit e_S in grade ℓ has the effect of multiplying by the factor of $\xi = (-1)^{\ell(\ell-1)/2}$.
- Q5.** Calculate the sum $a + b$ of the two Clifford numbers $a = 2 - 1e_1$ and $b = 1 + 3e_1 + 2e_2 + 4e_1e_2$.
- A5. $3 + 2e_1 + 2e_2 + 4e_1e_2$.
- Q6.** Calculate the difference $a - b$ of the two Clifford numbers $a = 2 - 1e_1$ and $b = 1 + 3e_1 + 2e_2 + 4e_1e_2$.
- A6. $1 - 4e_1 - 2e_2 - 4e_1e_2$.
- Q7.** Calculate the central product ab of the two Clifford numbers $a = 2 - 1e_1$ and $b = 1 + 3e_1 + 2e_2 + 4e_1e_2$.
- A7. $5 + 5e_1 + 8e_2 + 6e_1e_2$.
- Q8.** Calculate the outer product $a \wedge b$ of the two Clifford numbers $a = 2 - 1e_1$ and $b = 1 + 3e_1 + 2e_2 + 4e_1e_2$.
- A8. $2 + 5e_1 + 4e_2 + 6e_1e_2$.
- Q9.** Calculate the inner product $a \vee b$ of the two Clifford numbers $a = 2 - 1e_1$ and $b = 1 + 3e_1 + 2e_2 + 4e_1e_2$.
- A9. $-1 + 7e_1 + 8e_1e_2$.
- Q10.** Calculate the left inner product $a \lrcorner b$ of the two Clifford numbers $a = 2 - 1e_1$ and $b = 1 + 3e_1 + 2e_2 + 4e_1e_2$.
- A10. $-1 + 6e_1 + 8e_1e_2$.
- Q11.** Calculate the right inner product $a \llcorner b$ of the two Clifford numbers $a = 2 - 1e_1$ and $b = 1 + 3e_1 + 2e_2 + 4e_1e_2$.
- A11. $-1 - 1e_1$.
- Q12.** Calculate the scalar product (a, b) of the two Clifford numbers $a = 2 - 1e_1$ and $b = 1 + 3e_1 + 2e_2 + 4e_1e_2$.
- A12. -1 .
- Q13.** Calculate the dot product $a \cdot b$ of the two Clifford numbers $a = 2 - 1e_1$ and $b = 1 + 3e_1 + 2e_2 + 4e_1e_2$.
- A13. -3 .
- Q14.** Implement using a computer language of your choice, software to multiply two arbitrary Clifford numbers using the central product.
- A14. Check your software against question 7.

- Q15.** Implement using a computer language of your choice, software to multiply two arbitrary Clifford numbers using the outer product.
- A15. Check your software against question 8.
- Q16.** Implement using a computer language of your choice, software to multiply two arbitrary Clifford numbers using the inner product.
- A16. Check your software against question 9.
- Q17.** Implement using a computer language of your choice, software to multiply two arbitrary Clifford numbers using the left inner product.
- A17. Check your software against question 10.
- Q18.** Implement using a computer language of your choice, software to multiply two arbitrary Clifford numbers using the right inner product.
- A18. Check your software against question 11.
- Q19.** Implement using a computer language of your choice, software to multiply two arbitrary Clifford numbers using the scalar product.
- A19. Check your software against question 12.
- Q20.** Implement using a computer language of your choice, software to multiply two arbitrary Clifford numbers using the dot product.
- A20. Check your software against question 13.
- Q21.** Using your implementations from questions 16 to 19 to verify numerically or symbolically the first duality relationship from Eq. 2.12 for dimension $m = 4$ as in Table 2.10.
- Q22.** Using your implementations from questions 16 to 19 to verify numerically or symbolically the second duality relationship from Eq. 2.12 for dimensions m from 3 to 5.

Application of Geometric Algebra to Electromagnetic
Scattering

The Clifford-Cauchy-Dirac Technique

Seagar, A.

2016, XXII, 179 p. 53 illus. in color., Hardcover

ISBN: 978-981-10-0088-1