# Deriving Wedge Products and Rotors from the Matrix Determinant.

By Jarvis Carroll

Written November, 2022

## Contents

## Overview

Geometric Algebra provides a rich suite of tools and concepts that bridge the gap from simple vector arithmetic, to quaternion manipulation, which simultaneously makes quaternions easier to understand, and easier to solve problems with. However, many explanations of quaternions and Geometric Algebra are quite abstract, and unmotivated in their explanations. In this document we seek to derive rotors, including the quaternions, using nothing but rational vectors and matrices, as well as determinants and dot products as prior theory, and with as little contrivance as possible. The hope is that each step of this process introduces a new tool that is

- a logical solution to a problem,
- easy to visualise, using previous techniques and solutions as building blocks,
- easy to understand where the solution comes from, with small tricks to help certain parts of the process move along.

There are a total of seven significant concepts covered in this document, each introducing one or more techniques for reasoning about vectors, planes, and operations in those planes. These are

1. wedge product, maps lists of vectors to something that simultaneously captures the $ad - bc$ style area/volume/etc. they enclose, AND the plane/etc. that they lie in.
2. orthogonal component of a vector, in the plane represented by a given wedge product.
3. projection and rejection, using dot products and wedge products respectively,
4. reflection through a vector, a slightly weird operation that becomes dead simple with closed formulas for projection and rejection,
5. geometric product, as a combined dot/wedge product specifically intended to make reflections easier to compose,
6. conjugates and inverses of geometric products,
7. basic theory of rotors and quaternions, and how two reflections can make a rotation, which will now be easy to compose as well.

As an alternative to quaternions, rotors are easier to understand, precisely because they can be broken down into a composition of multiple simple techniques, each which do something logical.

## Wedge Products

The wedge product gives a way of representing oriented planes, hyperplanes, etc. along with a quantity that represents area, volume, etc. Representing both of these at the same time makes the algebra simpler, similar to how vectors are easier to work with than angles. The $i$, $j$, and $k$ objects, which are often represented as rotation around the $x$, $y$, and $z$ axes respectively, will turn out to correspond exactly to wedge products that represent the $yz$, $xz$, and $xy$ planes respectively.

There are many ways that the wedge product gets defined; some are "post-hoc" definitions, that just take the algebraic rules that ended up working, and presenting them bare to the reader, while others are unnecessarily abstract, involving tensor spaces between vectors, which make definitions simple, but completely disconnect us from the planes and hyperplanes we want to be working with. The approach that we will use here is to observe how matrix determinants can be used to test whether vectors lie in planes, thus giving a computational representation *of* a plane, which we can analyse in order to extract the minimum array of numbers needed to perform that test, which will happen to end up encoding area information as well.

### Determinant Review

For our purposes the important facts about determinants are

1. determinants somehow calculate the oriented volume of a matrix or list of vectors,
2. determinants are zero when the vectors are not linearly independent, and
3. determinants can be calculated recursively using cofactor expansion, i.e. smaller determinants.

These properties are closely related to the properties of wedge products that we end up wanting, which makes sense.

The relationship between determinants, volume, and linear independence, comes from the origins of the determinant as the lowest common denominator of the entries in a matrix inverse, e.g.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

Interpreting this further, in order to "undo" this matrix, we have to undo any rotation/skew operation using the adjugate

$$\begin{bmatrix} d & -b \\ -c & a \end{bmatrix},$$

and then undo the scaling effects of each of these using the determinant. The bigger the determinant, the more the matrix scaled its inputs, to the point where a simple scaling matrix

$$\begin{bmatrix} a & 0 \\ 0 & d \end{bmatrix}$$

has determinant $ad$.

If the determinant is zero, then there is no way to invert the matrix, which means there are two vectors that have the same image under this matrix. Due to the distributive property of matrices, this means that the image of their difference will be $A(u-v) = Au - Av = 0$, the zero vector. Put differently, the coordinates of this vector $(u-v)$ represent a linear combination of columns in $A$ that add up to the zero vector, meaning that those columns are not linearly independent, exactly what we need for representing planes and hyperplanes in higher dimensions.

Cofactor expansion of matrices is still mysterious to me, which is unfortunate since it is basically the whole reason that wedge products exist, but the idea is that to calculate large determinants by hand, one can recursively calculate smaller determinants, and weigh and add these using coefficients from the vector that didn't make it into the smaller determinants.

E.g.

$$\det \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} = c \det \begin{pmatrix} d & e \\ g & h \end{pmatrix} - f \det \begin{pmatrix} a & b \\ g & h \end{pmatrix} + i \det \begin{pmatrix} a & b \\ d & e \end{pmatrix}$$

Each of these smaller matrices is based on removing the last column and a different row from the original $3 \times 3$ matrix. The coefficient we end up multiplying this smaller determinant by turns out not to be from any of the rows are in that determinant, but precisely the row that was removed; this is related to the fact that vectors that point in the same direction don't encompass any area, so any area calculation has to be based on a complete set of orthogonal directions.

This cofactor expansion lets us calculate an $n$ dimensional determinant in terms of $n$ determinants each of dimension $n-1$. For $n \geq 4$ these $n-1$ determinants will expand into $n(n-1)$ smaller determinants, each of dimension $n-2$, and so on, but from this point on many of the smaller determinants will be the same, e.g. from removing row 3 and then row 4, vs removing row 4 and then row 3. This means we need some kind of memoisation in order to make this recursive calculation more efficient, which happens to be exactly what wedge products will end up implementing, in addition to their uses as geometric objects.

## Planes in 3D

Suppose that we have 3 vectors in 3D:

$$u = \begin{bmatrix} a \\ d \\ g \end{bmatrix}, \qquad v = \begin{bmatrix} b \\ e \\ h \end{bmatrix}, \qquad w = \begin{bmatrix} c \\ f \\ i \end{bmatrix}.$$

Reading each of these vectors it is easy to see that their coordinates were chosen from the columns of the general matrix we were using before,

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}.$$

The determinant we were calculating will therefore tell us exactly whether these vectors are linearly independent or not. If $u$ and $v$ are linearly independent, then they will represent some plane, and $w$ will be in that plane if and only if the determinant of the three vectors is zero. This means that the equation

$$c \det \begin{pmatrix} d & e \\ g & h \end{pmatrix} - f \det \begin{pmatrix} a & b \\ g & h \end{pmatrix} + i \det \begin{pmatrix} a & b \\ d & e \end{pmatrix} = 0$$

is satisfied exactly when $w$ is in the plane formed by $u$ and $v$. (Again, assuming that $u$ and $v$ are themselves linearly independent.)

This equation is the sort of thing that mathematicians like, of course, but it is also very straight-forwardly computed, making it a great complement for the other way of computing points in a plane formed by two vectors –taking linear combinations of them.

What is most interesting of all, though, is that the smaller determinants can be pre-computed *without* knowing what the test vector $w$ is, meaning the smaller determinants themselves contain all of the information needed to represent this plane. We could collect these determinants, and the negative sign on f, into a column vector, giving

$$\begin{bmatrix} dh - eg \\ bg - ah \\ ae - bd \end{bmatrix}.$$

This is the cross product of $u$ and $v$, but where the cross product interprets each of these coefficients as the coordinates of a vector,

$$u \times v = (dh - eg)e_1 + (bg - ah)e_2 + (ae - bd)e_3,$$

we want to instead interpret them as a sum of 'unit planes',

$$u \wedge v = (dh - eg)e_2 \wedge e_3 + (bg - ah)e_3 \wedge e_1 + (ae - bd)e_1 \wedge e_2,$$

since these unit planes represent the actual pair of axes that each of these coordinates rotate through. The fact that there are three such pairs is a coincidence which won't help us in higher dimensions, though, so the difference between axes and planes becomes important. Clearly it is circular to define $u \wedge v$ in terms of $e_i \wedge e_j$, so to be more careful we would define a separate 3D vector space notated $e_{2,3}$, $e_{3,1}$, $e_{1,2}$, and use that to define the wedge product, and then simply take the above formula as a theorem rather than a definition; either way, the point is that the unit 'vectors' of this wedge product formula are also wedges, that we can intuit as planes, and still do arithmetic with.

### Arbitrary Dimensions

The technique above generalises to sets of $n-1$ vectors in an $n$ dimensional space, but what about a smaller set of say, $k$ vectors? e.g. what if we have three vectors in 4D,

$$u = \begin{bmatrix} a \\ d \\ g \\ j \end{bmatrix}, \qquad v = \begin{bmatrix} b \\ e \\ h \\ k \end{bmatrix}, \qquad w = \begin{bmatrix} c \\ f \\ i \\ l \end{bmatrix},$$

and we again want to test whether $w$ is in the plane formed by $u$ and $v$? We can't simply use a determinant, since they won't form a square matrix, but we can use multiple determinants! In this case we would use 4 determinants, each $3 \times 3$, but in general we would have $\binom{n}{k+1}$ determinants, each of size $(k+1) \times (k+1)$, to cover all possible combinations of coordinates that we want to check for linear independence. [1]. This represents a linear space as a system of linear equations, which is nice. In this case it would be the equations

$$\det\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} = c\det\begin{pmatrix} d & e \\ g & h \end{pmatrix} - f\det\begin{pmatrix} a & b \\ g & h \end{pmatrix} + i\det\begin{pmatrix} a & b \\ d & e \end{pmatrix} = 0,$$

$$\det\begin{pmatrix} a & b & c \\ d & e & f \\ j & k & l \end{pmatrix} = c\det\begin{pmatrix} d & e \\ j & k \end{pmatrix} - f\det\begin{pmatrix} a & b \\ j & k \end{pmatrix} + l\det\begin{pmatrix} a & b \\ d & e \end{pmatrix} = 0,$$

$$\det\begin{pmatrix} a & b & c \\ g & h & i \\ j & k & l \end{pmatrix} = c\det\begin{pmatrix} g & h \\ j & k \end{pmatrix} - i\det\begin{pmatrix} a & b \\ j & k \end{pmatrix} + l\det\begin{pmatrix} a & b \\ g & h \end{pmatrix} = 0,$$

$$\det\begin{pmatrix} d & e & f \\ g & h & i \\ j & k & l \end{pmatrix} = f\det\begin{pmatrix} g & h \\ j & k \end{pmatrix} - i\det\begin{pmatrix} d & e \\ j & k \end{pmatrix} + l\det\begin{pmatrix} d & e \\ g & h \end{pmatrix} = 0.$$

Here we have four equations, which depend on 6 distinct determinants, which we could collect into a column vector as before,

$$\begin{bmatrix} ae - bd \\ ah - bg \\ ak - bj \\ dh - eg \\ dk - ej \\ gk - hj \end{bmatrix},$$

[1] For a proof that this works, see jarviscarroll.net/math/span-solving.html

which as a sum of unit planes would be written

$$
\begin{aligned}
u \wedge v = &(ae - bd)e_1 \times e_2 \\
&+(ah - bg)e_1 \times e_3 \\
&+(ak - bj)e_1 \times e_4 \\
&+(dh - eg)e_2 \times e_3 \\
&+(dk - ej)e_2 \times e_4 \\
&+(gk - hj)e_3 \times e_4.
\end{aligned}
$$

Note that to derive the cross product we wrote $(bg-ah)e_3 \wedge e_1$, whereas here we have represented this term with $(ah-bg)e_1 \wedge e_3$. If you calculate $e_1 \times e_3$ and $e_3 \wedge e_1$ explicitly *using* the above formula, then you'll see that they are equal and opposite, so we can use either to represent a more general bivector, (sum of wedges of two vectors,) and so here we write them in ascending order instead of cyclic order, since there are more terms to deal with now, which breaks the pattern.

## Wedges of Wedges

You may have realised this already, but the original system of equations, that checked if four $3 \times 3$ determinants were all zero, was really dealing with the coordinates of a 3-way wedge product $u \wedge v \wedge w$ all along, which we could have used to test if a fourth vector laid in the hyperplane formed by these vectors. Taking advantage of this, the plane equation can be written in one line as

$$u \wedge v \wedge w = 0.$$

This means that the cofactor expansions we performed not only tested whether $w$ was in the span of $u$ and $v$, but also calculated the coordinates of $u \wedge v \wedge w$ from the coordinates of $u \wedge v$ and $w$, which is where the real algebraic power of the wedge product starts to emerge. In general, we can use cofactor expansion to rewrite any $k$-way wedge as a $k-1$-way wedge, wedged with one additional vector. This is where the distinction in 3D becomes important, between the three unit vectors and the three unit planes; unit planes wedge with vectors to form volumes, whereas unit vectors wedge with vectors to form planes.

Also, if we come back to the property of cofactor expansion, we mentioned that calculating an $n \times n$ determinant required $n$ determinants of smaller matrices $(n-1) \times (n-1)$ each, and that these smaller determinants required even smaller determinants, with overlap between which size $n-1$ determinants required which size $n-2$ determinants, suggesting a need for memoisation; well, this formulation of the wedge product as a binary operation gives us exactly the incremental data structure we need to store all of these intermediate calculations, which might be obvious if that were the problem we were trying to solve, but what is amazing is that it happens to be such an independently useful geometric object on its own! In fact, even the $n \times n$ determinant we started with can now be interpreted as a single $n$ way determinant, with the very first cofactor expansion we did as one last binary wedge operation. Interestingly, though, in the same way that vectors are isomorphic to bivectors in 3D, but not the same, this calculation of determinants using wedges actually gives an $n$-vector, which is isomorphic to a scalar, but not necessarily the same; this makes sense, given the intuition that determinants represent an oriented volume/etc.

The wedge product as a binary operation is distributive, associative, and anticommutative,[2] allowing us to define it as the linear extension of the basic rules acting on the unit vectors,

$$\begin{aligned} &e_i \wedge e_i = 0, \\ &e_i \wedge e_j = -e_j \wedge e_i, \text{ whenever } i \neq j. \end{aligned}$$

which is often taken as the 'definition' of the wedge product, in courses that don't care about working out *why* things work, as long as they work. Computation of wedge products can be done by hand using these rules, e.g.

$$\begin{aligned} &(ae_1 + ce_2) \wedge (be_1 + de_2) \\ =&abe_1 \wedge e_1 + ade_1 \wedge e_2 + bce_2 \wedge e_1 + cde_2 \wedge e_2 \\ =&0 + ade_1 \wedge e_2 - bce_1 \wedge e_2 + 0 \\ =&(ad - bc)e_1 \wedge e_2, \end{aligned}$$

exactly as we would expect, calculating the two by two determinant using algebraic rules. If you take these rules as a given, you can even use them to 'define' the determinant with what appears to be a fairly small amount of theory, but this derivation wouldn't even begin to explain *why* such a determinant would have been useful to calculate. [3]

## Extracting Information from Wedges

We know how to construct wedges, and how to test if vectors lie in the subspace represented by those wedges, but can we use wedges to reason more directly about the subspaces they represent *as a whole*? The answer is yes, these objects not only represent area/volume information and direction information, but those areas, volumes, and directions can be extracted back out. This is the wedge analogue to defining a length or quadrance of vectors, and to defining the dot product of a pair of vectors.

[2] Proving these is fairly straight-forward to do if you move from a cofactor interpretation of determinants, to the deeper permutation interpretation, but for now we just leave these properties as magic, along with the whole concept of wedging a multivector with another multivector.

[3] Although we don't really need it, this algebraic approach can also be used to see what the wedge of a multivector with a multivector looks like, which is of course perfectly congruent with the permutation determinant approach to defining and reasoning about wedges.

## Wedge Products and Volume

Let's work out roughly how area and volume information is stored in a wedge product.

Start with the simple case of three axis-aligned vectors,

$$u = \begin{bmatrix} a \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad v = \begin{bmatrix} 0 \\ b \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad w = \begin{bmatrix} 0 \\ 0 \\ c \\ \vdots \\ 0 \end{bmatrix}.$$

We see that $u \wedge v = ab(e_1 \wedge e_2)$, and $u \wedge v \wedge w = abc(e_1 \wedge e_2 \wedge e_3)$, so in the axis aligned case the coordinates of a wedge product literally calculate area and volume, while tracking which axes that area or volume falls in. This pattern continues in a fairly straight-forward way for higher dimensions as well.

What happens if we rotate or skew these vectors? We should expect rotation to give something with the same size, for this operation to mean anything geometric. To test this, apply the rotation $A = \begin{bmatrix} x & -y \\ y & x \end{bmatrix}$ to the first two coordinates of all three vectors, and check their wedge afterwards[4],

$$\begin{aligned} Au \wedge Av \wedge Aw &= \begin{bmatrix} ax \\ ay \\ 0 \\ \vdots \\ 0 \end{bmatrix} \wedge \begin{bmatrix} -by \\ bx \\ 0 \\ \vdots \\ 0 \end{bmatrix} \wedge \begin{bmatrix} 0 \\ 0 \\ c \\ \vdots \\ 0 \end{bmatrix} \\ &= (abx^2 + aby^2)(e_1 \wedge e_2) \wedge \begin{bmatrix} 0 \\ 0 \\ c \\ \vdots \\ 0 \end{bmatrix} \\ &= (abx^2 + aby^2)c(e_1 \wedge e_2 \wedge e_3) \\ &= abc(x^2 + y^2)(e_1 \wedge e_2 \wedge e_3), \end{aligned}$$

so if $x^2 + y^2 = 1$, the volume is unchanged.

What is really happening here is closely related to the multiplicative property of determinants, so it should be little surprise that applying a skew matrix to these vectors would not change their volume, similar to how skewing a rectangle gives a parallelogram with the same area,

[4] Using the distributive properties of wedges and matrices, this can be defined as a more general action of an arbitrary matrix action on wedges *that have already been expanded out into coordinates.*

$$
\begin{aligned}
Bu \wedge Bv \wedge Bw &= \begin{bmatrix} a \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \wedge \begin{bmatrix} bx \\ b \\ 0 \\ \vdots \\ 0 \end{bmatrix} \wedge \begin{bmatrix} 0 \\ 0 \\ c \\ \vdots \\ 0 \end{bmatrix} \\
&= (ab - 0)(e_1 \wedge e_2) \wedge \begin{bmatrix} 0 \\ 0 \\ c \\ \vdots \\ 0 \end{bmatrix} \\
&= abc(e_1 \wedge e_2 \wedge e_3).
\end{aligned}
$$

These examples make sense, but what happens if we apply an operation that changes what plane some vectors occupy? e.g. what happens if we calculate $Av \wedge Aw$, without $u$ involved – do we stay in the $e_2 \wedge e_3$ plane?

$$
\begin{aligned}
Av \wedge Aw &= \begin{bmatrix} -by \\ bx \\ 0 \\ \vdots \\ 0 \end{bmatrix} \wedge \begin{bmatrix} 0 \\ 0 \\ c \\ \vdots \\ 0 \end{bmatrix} \\
&= -bcy(e_1 \wedge e_3) + bcx(e_2 \wedge e_3);
\end{aligned}
$$

We are in a linear combination of the $e_1 \wedge e_3$ plane, and the $e_2 \wedge e_3$ plane, meaning $A$ somehow acts exactly the same way in the $e_1 \wedge e_2 \wedge e_3$ hyperplane as it acts in the $e_1 \wedge e_2$ plane. It appears that the preserved quantity here, is the sum of squares of coordinates, in this case a squared area of $a^2c^2$ versus $a^2c^2(x^2 + y^2)$.

What about skewing these vectors out of the $e_2 \wedge e_3$ plane?

$$
\begin{aligned}
Bv \wedge Bw &= \begin{bmatrix} bx \\ b \\ 0 \\ \vdots \\ 0 \end{bmatrix} \wedge \begin{bmatrix} 0 \\ 0 \\ c \\ \vdots \\ 0 \end{bmatrix} \\
&= bcx(e_1 \wedge e_3) + bc(e_2 \wedge e_3);
\end{aligned}
$$

Here we have increased the squared area by a factor of $1 + x^2$, while also skewing which plane this area is in. This makes intuitive sense, if you can visualise taking a mesh rectangle, and skewing it into a parallelogram, versus trying to stretch it by pulling one edge out in the direction of the face of the rectangle.

### Extracting Vectors from Planes

In the examples above, we see hints of how matrices can act on bivectors in ways that capture the scaling effects of that matrix, when the matrix acts within a given plane, while also acting intuitively on the plane itself, when the matrix would have mapped elements of the plane to somewhere different. This is cool, and suggests that the coordinates of a bivector behave straight-forwardly like a normal vector, and that it might be quite easy to extract directional information from bivectors, as simple vectors.

To start with a simple example, if we have a vector like $u = xe_1 + ye_2$, is it possible to extract this vector $u$ from a bivector like $u \wedge e_3 = x(e_1 \wedge e_3) + y(e_1 \wedge e_3)$? The answer is obviously yes, we can just extract out the coordinates $x$ and $y$, (and 0 in the $e_3$ direction,) but what does this mean geometrically? And what do we do with bivectors that aren't aligned with any axis?

If we visualise the different planes that $u \wedge e_3$ could represent, then we can see how the coordinates of $u$ in $e_1 \wedge e_2$ directly correspond to the coordinates of different planes in $e_1 \wedge e_2 \wedge e_3$, and so by extracting $u$ we are essentially removing the $e_3$ direction from these planes.

- Given a wedge $u \wedge v$, and the vector $v$, find the direction of $u$.
- Given a wedge $u \wedge v$, and a vector $w$ in the plane formed by $u \wedge v$, find the orthogonal complement of $w$ in this plane.
- Given a wedge $u \wedge v$, and a vector $w$ in the plane formed by $u \wedge v$, rotate $w$ by a quarter turn in this plane.

If we assume that $u$ and $v$ are orthogonal unit vectors, then we want to implement a map $A$ with the properties that

$$\begin{aligned} Av &= u, \\ Au &= -v. \end{aligned}$$

Since this operation will be linear, we can map linear combinations of $u$ and $v$ as well, according to

$$A(au + bv) = bu - av.$$

If we let $w = au + bv$, then we can solve

$$\begin{aligned} a &= w \cdot u, \\ b &= w \cdot v, \\ \therefore Aw &= (w \cdot v)u - (w \cdot u)v. \end{aligned}$$

Now, we want to know what the columns of $A$ are, so let $w$ be a unit vector $e_i$, then

$$\begin{aligned} Ae_i &= (e_i \cdot v)u - (e_i \cdot u)v \\ &= v_i u - u_i v \\ \implies A_{i,j} &= v_i u_j - u_i v_j, \end{aligned}$$

but, this is the $e_j \wedge e_i$ coordinate of $u \wedge v$, so we can in fact use wedges to find orthogonal complements.

Orthogonal complements and rotation by a quarter turn are both what one would typically associate with cross products, and in a sense they are related to cross products, in that we need to use a wedge product in order for this to be possible, but in line with the way that $w$ gets projected onto $u \wedge v$, and the way that the projected part gets removed, we will actually notate this operation as a dot product, $(u \wedge v) \cdot w$, as the linear extension of the basic rules

$$\begin{aligned}(e_j \wedge e_i) \cdot e_i &= e_j, \text{ and} \\ (e_i \wedge e_j) \cdot e_k &= 0, \text{ when } i \neq k \text{ and } j \neq k.\end{aligned}$$

Note that by the anticommutativity of the wedge product, it follows that

$$\begin{aligned}(e_j \wedge e_i) \cdot e_j &= -(e_i \wedge e_j) \cdot e_j \\ &= -e_i.\end{aligned}$$

If $u$ and $v$ aren't orthogonal, or aren't unit vectors, then this definition will still work fine. If you have a multiple of orthogonal unit vectors, then it becomes a simple rotation-dilation operation, similar to multiplication by imaginary numbers in the complex plane,

$$(au \wedge bv) \cdot w = ab((u \wedge v) \cdot w),$$

but even if there is no such multiple we can compute, we still just use the linear extension above, as a continuous extension of these cases that are orthogonal, again, much like rational-coefficient complex numbers.

The notation and definition $(e_j \wedge e_i) \cdot e_i$ has been carefully chosen so that the dot operation cancels out the wedge operation, which makes expressions easy to simplify, and makes very powerful tricks possible later. Extending this idea, we also introduce a left dot product,

$$\begin{aligned}e_j \cdot (e_j \wedge e_i) &= e_i \\ &= -(e_j \wedge e_i) \cdot e_j,\end{aligned}$$

making this wedge dot product anti-commutative, just like the wedge product itself.

## Rejected Components

Dot products make it possible to calculate projections using the formula

$$u_{\parallel v} = \frac{(u \cdot v)v}{v \cdot v},$$

and now with wedge dot products, we can easily calculate rejected components too, with the almost identical formula

$$u_{\perp v} = \frac{(u \wedge v) \cdot v}{v \cdot v}.$$

This formula can be understood as finding the parallelogram between $u$ and $v$, and then finding the orthogonal component of $v$ in this parallelogram. Dividing by $v \cdot v$ corrects the size of this orthogonal component, to compensate for the two times that $v$ was multiplied into the formula.

It can be shown that these components have the following properties:

$$\begin{aligned} u_{\parallel v} \cdot v &= u \cdot v, \\ u_{\parallel v} \wedge v &= 0, \\ u_{\perp v} \cdot v &= 0, \\ u_{\perp v} \wedge v &= u \wedge v, \\ u_{\perp v} + u_{\parallel v} &= u, \\ u_{\perp v} \cdot u_{\parallel v} &= 0, \\ u_{\perp v} \wedge u_{\parallel v} &= \frac{(u \cdot v)(u \wedge v)}{v \cdot v}. \end{aligned}$$

## Geometric Products, Rotors, and Spinors

The basic trick that builds up to rotors and quaternions, is to work out a binary operation for conveniently composing two or more reflections, and then treating rotations as the special case of composing an even number of rotations with another even number of rotations.

While it is slightly daunting to imagine every rotation as some convoluted pair of reflections, we will find that once we have built up the algebra of rotations from that base case, it becomes possible to look for simpler and more sane ways of constructing the rotations we might want.

There are two ways of reflecting a vector $u$ inside another vector $v$; one is to treat $v$ as the normal of a mirror, and reflect $u$ in that mirror, flipping it in the direction of $v$, and keeping it the same in every other direction, and the other is to do the opposite, spinning it by a half turn around $v$, negating it in every direction except the $v$ direction. These two operations are simply the negative of each other, and so both of these are equally easy to implement using the operations we have implemented so far,

$$\begin{aligned} u_m &= u_{\perp v} - u_{\parallel v} \\ u_a &= u_{\parallel v} - u_{\perp v}. \end{aligned}$$

### Simplifying the Reflection Formula

The formula for reflection in/rotation around an axis can be written in a novel, symmetric format,

$$\begin{aligned} u_a &= u_{\parallel v} - u_{\perp v} \\ &= \frac{(u \cdot v)v}{v \cdot v} - \frac{(u \wedge v) \cdot v}{v \cdot v} \\ &= \frac{(u \cdot v)v - (u \wedge v) \cdot v}{v \cdot v} \\ &= \frac{v(u \cdot v) + v \cdot (u \wedge v)}{v \cdot v}, \end{aligned}$$

as a sum of terms with the rough form $v * u * v$. If we then apply another reflection, in the direction $w$, we would get a nesting pattern of four terms, each of the form $w*v*u*v*w$, with operations that might commute with each other, but the increase to four terms suggests an exponential growth of terms that is not going to be easy to analyse algebraically. When we look at the reflection formula, though, all of the operations involved are based on linear extensions of very simple rules. Dot products are a linear extension of the rules

$$\begin{aligned} e_i \cdot e_i &= 1, \\ e_i \cdot e_j &= 0, \text{ when } i \neq j, \end{aligned}$$

wedge products are a linear extension of the rules

$$\begin{aligned} e_i \wedge e_i &= 0, \\ e_i \wedge e_j &= -e_j \wedge e_i, \end{aligned}$$

and dot products on wedge products are a linear extension of the rules

$$\begin{aligned} (e_i \wedge e_j) \cdot e_j &= e_i, \\ (e_i \wedge e_j) \cdot e_k &= 0, \text{ when } i \neq k \text{ and } j \neq k. \end{aligned}$$

So we have these three operations for covering different mutually exclusive cases, and then to reflect a vector, we add these three cases up. We could simply combine them into a new operation, that applies all of the productive rules at once, and thus get rid of the explicit sum altogether. First, let's consider when we are multiplying a vector by another vector, which we will denote simply as the concatenation $uv$. The result is an ordered pair consisting of a scalar and then a bivector, i.e. a total of $\binom{n}{2} + 1$ coefficients, denoted simply by a sum of scalar terms and bivector terms, in general

$$uv = u \cdot v + u \wedge v.$$

This is called the geometric product. As a linear extension of basis operations, though, this operation becomes dramatically simpler to describe, with just the two rules

$$e_i e_i = 1,$$
$$e_i e_j = -e_j e_i, \text{ when } i \neq j.$$

For example, we can compute

$$\begin{aligned} &(e_1 + e_2)(e_1 + 2e_2) \\ = &\, e_1e_1 + 2e_1e_2 + e_2e_1 + 2e_2e_2 \\ = &\, 1 + 2e_1e_2 - e_1e_2 + 2 \\ = &\, 3 + e_1e_2. \end{aligned}$$

Then we could combine the wedge product and the dot product in our reflection formula into this geometric product operation; we then need to multiply the scalar part by a vector, and dot the wedge part with that same vector, with a rule kind of like

$$wuv = w(u \cdot v + u \wedge v) = w(u \cdot v) + w \cdot (u \wedge v).$$

Now, the $e_ie_j = 1$ rule of the geometric product almost makes this work as is! But, since the geometric product also includes a wedge operation, a three way geometric product would really give

$$wuv = w(u \cdot v) + w \cdot (u \wedge v) + w \wedge u \wedge v.$$

Fortunately, in the reflection formula, the $w$ here is also just $v$, and

$$v \wedge u \wedge v = -u \wedge v \wedge v = 0,$$

so we absolutely can use the geometric product, simplifying the reflection formula to

$$u_a = \frac{vuv}{v \cdot v}.$$

Further, since $v$ is parallel to itself, we can even write $v \cdot v$ as $vv$, or $v^2$, without ambiguity, giving

$$u_m = \frac{vuv}{v^2}.$$

Now this is a formula we can work with, multiplying on either side by an operation that was built to be distributive, and associative, and divided by a scalar. For the remainder of this document we'll be working with this operation, and so from now on we can even write unit planes/volumes/etc. as geometric products of basis vectors, like $e_1e_2$, instead of $e_1 \wedge e_2$.

## Dimensions and Brackets

It would seem that to reflect a vector $u$ in a vector $v$, and then in a vector $w$, we would simply write

$$u'_a = \frac{wvuvw}{v^2w^2}.$$

This is exactly the formula for composing two reflections, but now there are two significantly different ways of bracketing this.

$$(w(vuv)w)$$

will produce a vector $vuv$, and then turn that into the vector $wvuvw$, whereas

$$(wv)u(vw)$$

will produce some more arbitrary $v \cdot w + v \wedge w$, multiply it by $u$ to get an even more arbitrary $u(v \cdot w) + u \cdot (v \wedge w) + u \wedge v \wedge w$, and then finally cancel this back down to a vector by multiplying by $wv$. In this latter approach, we need an almost arbitrary linear combination of *all* $2^n$ possible multivectors, giving an abstract sum of scalars, vectors, bivectors, trivectors, and so on. Each specific product will consist only of either even terms, (scalars, bivectors, etc.) or odd terms, (vectors, trivectors, etc.) so that limits the dimension by a half, and the fewer terms you compose, the fewer dimensions will actually get used, but the potential still remains for up to $2^{n-1} + \frac{n}{2}$ degrees of freedom in these intermediate objects.[5] For the purposes of algebra, though, we can ignore this, and assume that all geometric products are closed in a $2^n$ dimensional space, and save the optimisations for later, when dealing with specific problems.

Apart from this subtlety that the minimum datatype you could be working with might change every time you rebracket these expressions, they otherwise behave as an associative datatype, ready for us to start building algebraic and geometric tools out of.

## Inverses

Given a chain of reflections,

$$\frac{v_k \dots v_1 u v_1 \dots v_k}{v_1^2 \dots v_k^2}$$

we can define two operations to extract out the concept of a reflection/rotation,[6] allowing this expression to be simplified,[7] and reasoned about further. First, we simply create a notation for reversing long products of vectors,

$$(v_1 \dots v_k)^* = v_k \dots v_1,$$

[5] As $n$ gets large you probably want to represent this stuff with sparse coefficient lists, but for $n < 5$ it wouldn't be that bad to represent densely.

[6] As we will discuss soon, this covers the 'orthogonal group' $O(n)$.

[7] This simplification saves on redundant compositions, but with significant overhead, analogous to the difference between $(AB)v$ and $A(Bv)$ in matrix computations, where the first is cheaper to do if you have a hundred different $v$s, but the second is cheaper to do if you have a hundred different $A$s and $B$s.

more specifically, as the linear extension of the rule

$$(e_{i_1} \dots e_{i_k})^* = e_{i_k} \dots e_{i_1}.$$

Further, if we have some arbitrary sum of multivectors $\xi$, and $\xi\xi^*$ equals $\xi^*\xi$, and is a non-zero scalar, then we can define

$$\xi^{-1} = \frac{\xi^*}{\xi\xi^*}$$

so that $\xi\xi^{-1} = \xi^{-1}\xi = 1$. In particular, a vector $v$ will have inverse

$$v^{-1} = \frac{v}{v^2},$$

and a product of vectors $v_1 \dots v_k$ will have inverse

$$(v_1 \dots v_k)^{-1} = \frac{v_k \dots v_1}{v_1^2 \dots v_k^2}.$$

Now we can rewrite a repeated reflection as

$$\frac{v_k \dots v_1 u v_1 \dots v_k}{v_1^2 \dots v_k^2} = (v_k \dots v_1) u (v_k \dots v_1)^{-1}.$$

## Rotations, Rotors, and Spinors

Now we can start thinking of products such as $v_k \dots v_1$ as objects unto themselves, with a simple conjugation $\xi u \xi^{-1}$ applying them to any vector $u$, resulting in some linear operation $R(\xi)$. $R$ is a homomorphism, meaning it has the property that

$$R(\xi_2\xi_1) = R(\xi_2)R(\xi_1),$$

and therefore that computations with sums of multivectors perfectly track the resulting operation they would effect when applied to vectors. The linear operations $R(\xi)$ are from the class of matrices denoted $\mathrm{O}(n)$, and they are said to 'double-cover' this class, in that all operations in the class have a corresponding $\xi$, and yet $R(-\xi) = R(\xi)$, so there is a redundancy in multivectors, in that there are 'twice as many' of them.

For any vector $v$,

$$R(v) = -1,$$

meaning that an odd number of reflections will remain a reflection, inverting all volumes, and thus changing the coordinate system in a fundamentally discontinuous way. An even number of reflections, on the other hand, will not have this effect, but will still result in rotations of the space, if reflections were applied in a way that did not negate each other, and were not orthogonal to each other. For example, in 2D,

$$(e_1 + e_2)e_1 = 1 - e_1e_2,$$

and

$$
\begin{aligned}
R((e_1+e_2)e_1)\begin{bmatrix}a\\b\end{bmatrix} &= R((e_1+e_2)e_1)(ae_1+be_2)\\
&= \frac{(e_1+e_2)e_1(ae_1+be_2)e_1(e_1+e_2)}{2}\\
&= \frac{a(e_1+e_2)e_1e_1e_1(e_1+e_2)}{2} + \frac{b(e_1+e_2)e_1e_2e_1(e_1+e_2)}{2}\\
&= \frac{a(e_1+e_2)e_1(e_1+e_2)}{2} - \frac{b(e_1+e_2)e_2(e_1+e_2)}{2}\\
&= \frac{a(1-e_1e_2)(e_1+e_2)}{2} - \frac{b(1+e_1e_2)(e_1+e_2)}{2}\\
&= \frac{a(e_1+e_2+e_2-e_1)}{2} - \frac{b(e_1-e_2+e_2+e_1)}{2}\\
&= ae_2-be_1\\
&= \begin{bmatrix}0&1\\-1&0\end{bmatrix}\begin{bmatrix}a\\b\end{bmatrix}
\end{aligned}
$$

Here we have applied the operation corresponding to $1-e_1e_2$, and gotten a clockwise quarter-turn of the plane. Crucially, the wedge $e_1e_2$ is the very plane that $R(1-e_1e_2)$ ended up rotating, suggesting that we should stop thinking about reflections altogether at this point, and just construct objects like this, which rotate planes in well defined ways.

Any product of an even number of vectors[8] is called a rotor, due to its ability to rotate $n$-dimensional space in one or more planes. In the same way that the products we had before covered $\mathrm{O}(n)$, the special case where there are an even number of reflections, covers $\mathrm{SO}(n)$, which should not be surprising for those familiar.

As we started to see in the example of $1-e_1e_2$, rotors that are made out of a scalar and a bivector rotate in precisely the same plane that the bivector represents. Specifically, if a rotation $A$ can rotate a vector $v$ to a vector $u$, then the rotor $uv$ will represent the rotation $A^2$. [9] This means that unlike complex numbers, we represent a half turn with a bivector $u\wedge v$, and, just like the quaternion examples discussed previously, we represent a quarter turn with expressions like $1+u\wedge v$, assuming that $u\wedge v$ has unit area, such as if $u$ and $v$ are perpendicular, and unit length.

Here are a list of useful rotor formulas, along with their effect in the $u\wedge v$ plane.

[8] I.e. any sum of wedges of even numbers of vectors, with a well defined inverse under the geometric product. Proof not included.

[9] Although in general, thinking about reflections is not necessary for dealing with rotors, this relationship is easily derived from properties of reflections, $R(uv)=R(u)R(v)=R(u)A^{-1}R(u)A=AA=A^2$.

$$
\begin{array}{rl}
u \wedge v & \text{half turn,} \\
1 + u \wedge v & \text{quarter turn, if } u \wedge v \text{ has unit area,} \\
1 - u \wedge v & \text{reverse quarter turn, if } u \wedge v \text{ has unit area,} \\
x + y(u \wedge v) & \dfrac{1}{x^2 + y^2} \begin{bmatrix} x & -y \\ y & x \end{bmatrix}^2, \text{ if } u \wedge v \text{ has unit area,} \\
uv & \text{rotate by twice the angle between } v \text{ and } u, \\
(u + v)v & \text{rotate } v \text{ to } u, \text{ if } u^2 = v^2. \\
(u + v)u & \text{rotate } u \text{ to } v, \text{ if } u^2 = v^2.
\end{array}
$$

If the conditions for these formulas aren't met, then the vectors or wedges can be normalised by approximate square root calculations, to get approximately the same effect.

Since rotors that are rescalings of each other, or even negatives of each other, all end up having the same effect, we consider them equivalent. There is another action that we can make using the geometric product, though, which is to apply them without normalisation,

$$S(\xi)u = \xi u \xi^*.$$

This will rotate the vector as before, but might also dilate it, acting like a higher-dimensional analogue to the complex numbers. When this action is used, we call $\xi$ a spinor, and in this case we do not consider spinors equivalent when they are rescalings of each other, making them the same class of things, but different datatypes.[10]

## Quaternions

The quaternions are often described as something like, the unique 4D vector space with associative multiplication satisfying $i^2 = j^2 = k^2 = ijk = -1$, and then all the useful properties of them are derived by magic. We now have the tools to build operations that we know are rotations, and so when applied to the 3D case they should derive these same algebraic properties, but this time we already know they are rotations!

Taking the spinor and rotor theory we have described, and simply setting $n = 3$, gives us a rather right theory of operations, including

- A 3D vector space, with basis vectors $e_1$, $e_2$, $e_3$,
- a 3D vector space of 2D wedges/bivectors, including the basis planes $e_1e_2$, $e_2e_3$, $e_1e_3$, acquired using the wedge product, analogous to the cross product,

[10]Normally rotors are defined as the special case of spinors where $\xi\xi^* = 1$, but this approach requires square root operations whenever we want to interpret a spinor as a rotors, and with that comes unnecessary computational cost and inaccuracy.

- Oriented volumes, with a single dimension, represented by a single basis unit $e_1e_2e_3$,
- 3D projections, rejections, and reflections,
- 3D rotors, i.e. rotations, represented by an even number of reflections, i.e. an arbitrary bivector, and a scalar,
- 3D rotation-reflection operations, represented by an odd number of reflections, i.e. a vector and a volume,
- The ability to compose rotations and reflection-rotations with the geometric product, and to apply them with conjugation $\xi u \xi^{-1}$,
- The ability to apply rotations and reflection-rotations without compensating for scaling, $\xi u \xi^*$, and thus representing spinors/rotation-dilations, and reflection-rotation-dilations.

The main difference between 3D spinors, and higher dimensions, is that 3D spinors never have quadvector parts, or higher, since there are only three dimensions to choose from, and so all planes through the origin intersect. This means the geometric product is closed, among just the scalars and bivectors. E.g.

$$\begin{aligned}(e_1e_2)(e_2e_3) &= e_1e_3\\ (e_1e_2)(e_2e_1) &= 1\\ (e_1e_2)(e_1e_2) &= -1.\end{aligned}$$

These examples also give hints as to how Hamilton's defining equation for quaternions will hold for 3D spinors as well. If we set

$$i = e_2e_3 \qquad j = e_1e_3 \qquad k = ij = e_1e_2,$$

then we indeed find that

$$(e_2e_3)^2 = (e_1e_3)^2 = (e_1e_2)^2 = (e_2e_3)(e_1e_3)(e_1e_2) = -1.$$

$$(e_3e_2)^2 = (e_1e_3)^2 = (e_2e_1)^2 = (e_3e_2)(e_1e_3)(e_2e_1) = -1.$$

($e_3e_2e_1e_3e_2e_1$ requires three swaps to eliminate all the vectors entirely. Since this is an odd number, this leaves -1.)

Further, the multivector reversal operation is exactly the quaternion conjugate operation,

$$\begin{aligned}(w + xe_3e_2 + ye_1e_3 + ze_2e_1)^* &= w + xe_2e_3 + ye_3e_1 + ze_1e_2\\ &= w - xe_3e_2 - ye_1e_3 - ze_2e_1,\end{aligned}$$

and so the spinor inverse $\xi^*/\xi\xi^*$ is the same as the quaternion inverse as well. Since all of these operations come out the same, we can see that 3D spinors are isomorphic to the quaternions.

Just like spinors, quaternions are often more useful when treated as rotors, but this is usually done by mapping the vector to a pure-imaginary quaternion, and applying the conjugation $\xi u \xi^{-1}$. This step of mapping vectors to quaternions is unnecessary when dealing with rotors, though, since rotors and spinors can be applied to vectors, as-is. For example, to rotate $e_2$ by a quarter turn in the $e_2e_3$ plane, we could map it to the quaternion $j$, and then apply the quaternion rotation $1 + i$, giving

$$\begin{aligned}
&(1+i)j(1+i)^{-1} \\
=\ &(1+i)j\frac{(1-i)}{2} \\
=\ &\frac{j - ji + ij - iji}{2} \\
=\ &\frac{j + k + k - j}{2} \\
=\ &\frac{2k}{2} \\
=\ &k.
\end{aligned}$$

On the other hand, we can leave the vector $e_2$ as is, and apply the rotor $1 + e_3e_2$ to it, giving

$$\begin{aligned}
&(1+e_3e_2)e_2(1+e_3e_2)^{-1} \\
=\ &(1+e_3e_2)e_2\frac{1+e_2e_3}{2} \\
=\ &\frac{e_2 + e_3e_2e_2 + e_2e_2e_3 + e_3e_2e_2e_2e_3}{2} \\
=\ &\frac{e_2 + e_3 + e_3 - e_2}{2} \\
=\ &\frac{2e_3}{2} \\
=\ &e_3.
\end{aligned}$$

Although there were more terms to deal with at any given time, the rules of rotor simplification are less contrived, and at the end we get the literal vector we want, instead of some abstract $xi + yj + zk$ that we need to map to a vector.

There is more that can be done with quaternions, and rotors in general. Since we can easily create rotors based on the plane we want to rotate in, and since we can easily compose rotors, a very powerful application of rotors is to smoothly interpolation between two rotations. This operation requires a theory of angles, and requires approximate operations, compared to the exact rational arithmetic we have gotten away with using until now. Angles and rotation are something that we take for granted, but really they are a big topic all on their own, that could be given a separate twenty-page document like this one, which could then include a discussion of spherical interpolation using rotors.