The Regressive Product Per Hestenes

Kurt Nalty

August 1, 2017

Abstract

Hermann Grassmann's wedge product has been widely adopted in physics and mathematics, but the related regressive product is rarely presented. Here I work through David Hestenes' presentation of the regressive product [1] in two, three and four dimensional spaces, providing multiplication tables and component formulas.

Nomenclature

Scalar values are shown as standard math type, such as q. Vector basis are shown as e with a subscript, such as e_x and e_y. Bivector basis are shown as e with two subscripts, order sensitive, such as e_{xy} and e_{zx}. Trivector basis are shown as e with three suffixes, such as e_{xyz} or e_{yzt}, and the quadvector basis will be shown as e_{xyzt}.

Generic multivectors are the linear combination of basis elements, and will be usually represented by a capital Roman letter.

The generic one dimensional multivector is

$$L = a + be_x$$

which has two elements, being a scalar and a vector.

The generic two dimensional multivector is

$$P = a + (be_x + ce_y) + de_{xy}$$

which has one scalar component, a two component vector, and a planar bivector element.

The generic three dimensional multivector is

$$V = a + (be_x + ce_y + de_z) + (ee_{xy} + fe_{zx} + ge_{yz}) + he_{xyz}$$

which has one scalar component, a three component vector portion, a three component bivector portion, and a single trivector component. The order of terms in e_{zx} is a matter of convention which differs among authors. The use of e as a factor for the basis e_{xy} should not be a cause for confusion.

The generic four dimensional (spacetime) multivector is

$$\begin{aligned} B = \; & + \; a \\ & + \; (be_x + ce_y + de_z + ee_t) \\ & + \; (fe_{xy} + ge_{zx} + he_{yz} + ie_{xt} + je_{yt} + ke_{zt}) \\ & + \; (le_{xyz} + me_{xyt} + ne_{xzt} + oe_{yzt}) \\ & + \; pe_{xyzt} \end{aligned}$$

which has a single component scalar, a four component vector, a six component bivector, a four component trivector, and a single component quadvector.

Each of the basis terms is a simple set of factors of the vector basis. A *blade* is a simple set of vector factors. The number of factors defines the *step* of the blade. The scalar term is associated with a blade of step 0. The vector basis a_y is a blade of step 1. The bivector e_{zx} is a blade of step 2. The trivector e_{xyz} is a blade of step 3, and so on. Hestenes typically uses a lower case k to represent the step of a blade.

The *pseudo-scalar* is the highest order blade in a dimension. In three dimensions, the pseudo-scalar is e_{xyz}, and it mimics the complex number $i = \sqrt{-1}$. Hestenes commonly uses I to represent a pseudo-scalar, regardless of dimension. It is important to be aware of the context when seeing this symbol.

When evaluating the regressive product, we are typically using a subset of the available dimensionality. As an example, $e_y \vee e_z$ only uses two dimensions of the implied three dimensional space. Hestenes typically uses r for the step of the forefactor, and s for the step of the postfactor. The unique set of basis vectors actually used in the two terms is called the *support* of the subspace. The ordered cascaded wedge product of the support members creates the working pseudoscalar for the calculation. The step of this pseudoscalar is the working dimensionality n for the specific calculation. As an example,

$e_{xt} \vee e_y$ has step $r = 2$, step $s = 1$, support $= x, y$, and t, pseudoscalar $I = x \wedge y \wedge t$, and dimensionality $n = 3$. This is a three dimensional subspace of an implied four dimensional space.

The wedge product, also called the progressive product, is denoted with a $\wedge$ operator, as in $e_x \wedge e_y$. The wedge product using a scalar term is simple scalar multiplication. However, the wedge product between vector elements is antisymmetric. As an example, $e_x \wedge e_y = -e_y \wedge e_x$. This antisymmetry make the square wedge of a basis vector identically zero, as in $e_x \wedge e_x = -e_x \wedge e_x = 0$. Unique products of basis vector create multivector elements (blades), which are often written with multiple subscripts rather than explicitly showing the wedge products. As an example, $e_x \wedge e_y = e_{xy}$, and $e_{zx} \wedge e_y = e_{xyz}$. The default basis for a multivector has an arbitrary choice of sign preference among the multivector components. Some authors prefer e_{zx}, for example, while other prefer e_{xz}. This can result in sign difference in specific formulas and calculations, and is a detail to be aware of.

The geometric product, also called the Clifford product, is denoted by simple juxtoposition of terms, such as $e_x e_{xy}$. The geometric product with a scalar is simple scalar multiplication. However, the geometric product using vector basis is given by $\vec{a}\vec{b} = \vec{a} \cdot \vec{b} + \vec{a} \wedge \vec{b}$, where I have included overarrows to emphasis that this formula uses vectors, not general multivectors. For orthonormal basis, the dot products are zero for different basis, leading to the wedge product, while the square of a basis vector now becomes one, due to the dot product. As an example, $e_x e_y = e_x \wedge e_y = e_{xy}$ while $e_x e_x = e_x \cdot e_x = 1$.

Hestenes defines reversion as an operator which reverses the order of multiplication in the basis blades of the multivector. The dagger symbol is used to indicate reversion by Hestenes. An example of reversion is

$$\begin{aligned}(a + be_x + ce_{yz} + de_{xyz})^\dagger &= a + be_x + ce_{zy} + de_{zyx} \\ &= a + be_x - ce_{yz} - de_{xyz}\end{aligned}$$

If k indicates the step of a multivector component, the reverse of that component has a sign change given by $(-1)^k(k-1)/2$, or $+ + - - + + - -$ organized by ascending step for Euclidean space.

The motivation behind the reverse is to have an easy definition for the inverse of a blade. For Euclidean space, a blade M times it reverse is equal to one. $MM^\dagger = 1$. For more generic spacetime, where the dot product square of the time basis is negative, we need to pay attention to sign based upon the actual basis being used.

∧	1	e_x
1	1	e_x
e_x	e_x	0

Table 1: Wedge Product in One Dimensional Euclidean Space

If I represents the pseudoscalar for a space, and s is number of timelike basis vectors, then $I^{-1} = (-1)^s I^\dagger$.

As the reverse and inverse are unary operations, there is no distinction between prefactor or postfactor formulas. However, duality will occur in two forms. Hestenes only uses the postfactor format for his duality. Using a tilde to indicate the dual of a blade, Hestenes defines

$$\tilde{A} = AI^{-1}$$

where I wil be restricted to the support of the terms involved.

Hestenes then defines the regressive product implicitly through a DeMorgan style formula.

$$(A \vee B)\tilde{} = \tilde{A} \wedge \tilde{B}$$

or

$$A \vee B = (\tilde{A} \wedge \tilde{B})\tilde{}$$

Our goal now is to take these formulas and create specific example implementations.

One Dimensional Euclidean Space

In one dimensional Euclidean space, we have the basis blades of 1 and e_x. Our generic multivector is

$$A = a + be_x$$

Our pseudoscalar is $I = e_x$, and the inverse of this pseudoscalar is also $I^{-1} = e_x$. Our table for wedge multiplication and geometric multiplication are shown in Table 1 and 2, with forefactors on the left column, and postfactors on the top row.

	1	e_x
1	1	e_x
e_x	e_x	1

Table 2: Geometric Product in One Dimensional Euclidean Space

Our first entry is purely scalar multiplication. This is a zero dimensional subspace for the one dimensional space. For this space, the pseudoscalar is $I = 1$, and the inverse of the pseudoscalar is also $I^{-1} = 1$.

Walking throught this first calculation for our regressive product, we have

$$\begin{aligned} A \vee B &= (\tilde{A} \wedge \tilde{B})\tilde{} \\ 1 \vee 1 &= (\tilde{1} \wedge \tilde{1})\tilde{} \\ &= (1 \wedge 1)\tilde{} \\ &= (1)\tilde{} = 1 \end{aligned}$$

For the other entries in this space, the pseudoscalar is $I = e_x$. The inverse of the pseudoscalar is also $I^{-1} = e_x$. The dual of 1 is $\tilde{1} = e_x$, and the dual of $\tilde{e}_x = 1$.

$$\begin{aligned} 1 \vee e_x &= (\tilde{1} \wedge \tilde{e}_x)\tilde{} \\ &= (e_x \wedge 1)\tilde{} \\ &= (e_x)\tilde{} = 1 \\ e_x \vee 1 &= (\tilde{e}_x \wedge \tilde{1})\tilde{} \\ &= (1 \wedge e_x)\tilde{} \\ &= (e_x)\tilde{} = 1 \\ e_x \vee e_x &= (\tilde{e}_x \wedge \tilde{e}_x)\tilde{} \\ &= (1 \wedge 1)\tilde{} \\ &= (1)\tilde{} = e_x \end{aligned}$$

Table 3 provides the One Dimensional Regressive Product in multiplication table format.

$\vee$	1	e_x
1	1	1
e_x	1	e_x

Table 3: Regressive Product in One Dimensional Euclidean Space

$\wedge$	1	e_x	e_y	e_{xy}
1	1	e_x	e_y	e_{xy}
e_x	e_x	0	e_{xy}	0
e_y	e_y	-e_{xy}	0	0
e_{xy}	e_{xy}	0	0	0

Table 4: Wedge Product in Two Dimensional Euclidean Space

In C programming format, the one dimensional component equations are

```
c.q = +a.q*b.q + a.q*b.x + a.x*b.q
c.x = +a.x*b.x
```

Two Dimensional Euclidean Space

In two dimensional Euclidean space, we have a scalar basis of 1, two vector basis being e_x and e_y, and a bivector e_{xy}. The pseudoscalar for this space is $I = e_{xy}$, although for many calculations here we will be using a one dimensional subspace with different pseudoscalar. For this two dimensional space, the inverse of the pseudoscalar is $I^{-1} = -I = -e_{xy}$.

In multiplication table format, the wedge product is shown in Table 4. Table 5 shows the geometric product for two dimensional Euclidean space.

We have sixteen entries in our worksheet for the regressive product. However, many of these are known from the one dimensional case. Table 6 is a partial table with the knowns from the one dimensional case filled in.

	1	e_x	e_y	e_{xy}
1	1	e_x	e_y	e_{xy}
e_x	e_x	1	e_{xy}	e_y
e_y	e_y	-e_{xy}	1	-e_x
e_{xy}	e_{xy}	-e_y	e_x	-1

Table 5: Geometric Product in Two Dimensional Euclidean Space

$\vee$	1	e_x	e_y	e_{xy}
1	1	1	1	?
e_x	1	e_x	?	?
e_y	1	?	e_y	?
e_{xy}	?	?	?	?

Table 6: Partial Regressive Product in Two Dimensional Euclidean Space

We now walk throught the remaining calculations for our regressive product. For $1 \vee e_{xy}$, the pseudoscalar is $I = e_{xy}$ and $I^{-1} = -e_{xy}$.

$$\begin{aligned} A \vee B &= (\tilde{A} \wedge \tilde{B})\tilde{} \\ 1 \vee e_{xy} &= (\tilde{1} \wedge \tilde{e}_{xy})\tilde{} \\ &= (-e_{xy} \wedge 1)\tilde{} = -e_{xy}\tilde{} \\ &= -1 \end{aligned}$$

For $e_x \vee e_y$, the pseudoscalar is $I = e_{xy}$ and $I^{-1} = -e_{xy}$.

$$\begin{aligned} A \vee B &= (\tilde{A} \wedge \tilde{B})\tilde{} \\ e_x \vee e_y &= (\tilde{e}_x \wedge \tilde{e}_y)\tilde{} \\ &= (-e_y \wedge e_x)\tilde{} = e_{xy}\tilde{} \\ &= 1 \end{aligned}$$

For $e_x \vee e_{xy}$, the pseudoscalar is $I = e_{xy}$ and $I^{-1} = -e_{xy}$.

$$\begin{aligned} A \vee B &= (\tilde{A} \wedge \tilde{B})\tilde{} \\ e_x \vee e_{xy} &= (\tilde{e}_x \wedge \tilde{e}_{xy})\tilde{} \\ &= (-e_y \wedge 1)\tilde{} = -e_y\tilde{} \\ &= -e_x \end{aligned}$$

For $e_y \vee e_x$, the pseudoscalar is $I = e_{xy}$ and $I^{-1} = -e_{xy}$.

$$\begin{aligned} A \vee B &= (\tilde{A} \wedge \tilde{B})\tilde{} \\ e_y \vee e_x &= (\tilde{e}_y \wedge \tilde{e}_x)\tilde{} \\ &= (e_x \wedge -e_y)\tilde{} = -e_{xy}\tilde{} \\ &= -1 \end{aligned}$$

For $e_y \vee e_{xy}$, the pseudoscalar is $I = e_{xy}$ and $I^{-1} = -e_{xy}$.

$$\begin{aligned} A \vee B &= (\tilde{A} \wedge \tilde{B})\tilde{} \\ e_y \vee e_{xy} &= (\tilde{e}_y \wedge \tilde{e}_{xy})\tilde{} \\ &= (e_x \wedge 1)\tilde{} = e_x\tilde{} \\ &= -e_y \end{aligned}$$

For $e_{xy} \vee 1$, the pseudoscalar is $I = e_{xy}$ and $I^{-1} = -e_{xy}$.

$$\begin{aligned} A \vee B &= (\tilde{A} \wedge \tilde{B})\tilde{} \\ e_{xy} \vee 1 &= (\tilde{e}_{xy} \wedge \tilde{1})\tilde{} \\ &= (1 \wedge -e_{xy})\tilde{} = -e_{xy}\tilde{} \\ &= -1 \end{aligned}$$

For $e_{xy} \vee e_x$, the pseudoscalar is $I = e_{xy}$ and $I^{-1} = -e_{xy}$.

$$\begin{aligned} A \vee B &= (\tilde{A} \wedge \tilde{B})\tilde{} \\ e_{xy} \vee e_x &= (\tilde{e}_{xy} \wedge \tilde{e}_x)\tilde{} \\ &= (1 \wedge -e_y)\tilde{} = -e_y\tilde{} \\ &= -e_x \end{aligned}$$

$\vee$	1	e_x	e_y	e_{xy}
1	1	1	1	-1
e_x	1	e_x	1	-e_x
e_y	1	-1	e_y	-e_y
e_{xy}	-1	-e_x	-e_y	-e_{xy}

Table 7: Regressive Product in Two Dimensional Euclidean Space

For $e_{xy} \vee e_y$, the pseudoscalar is $I = e_{xy}$ and $I^{-1} = -e_{xy}$.

$$\begin{aligned} A \vee B &= (\tilde{A} \wedge \tilde{B})\tilde{} \\ e_{xy} \vee e_y &= (\tilde{e}_{xy} \wedge \tilde{e}_y)\tilde{} \\ &= (1 \wedge e_x)\tilde{} = e_x\tilde{} \\ &= -e_y \end{aligned}$$

For $e_{xy} \vee e_{xy}$, the pseudoscalar is $I = e_{xy}$ and $I^{-1} = -e_{xy}$.

$$\begin{aligned} A \vee B &= (\tilde{A} \wedge \tilde{B})\tilde{} \\ e_{xy} \vee e_{xy} &= (\tilde{e}_{xy} \wedge \tilde{e}_{xy})\tilde{} \\ &= (1 \wedge 1)\tilde{} = 1\tilde{} \\ &= -e_{xy} \end{aligned}$$

We now fill in the missing entries in our Table 6 to get Table 7.

Unlike Eric Lengyel, this definition has a negative sign on the pseudoscalar product terms.

In C programming format, the two dimensional component equations are

```
c.q  = +a.q*b.q + a.q*b.x  + a.q*b.y - a.q*b.xy + a.x*b.q
       + a.x*b.y + a.y*b.q - a.y*b.x - a.xy*b.q
c.x  = +a.x*b.x - a.x*b.xy - a.xy*b.x
c.y  = +a.y*b.y - a.y*b.xy - a.xy*b.y
c.xy = -a.xy*b.xy
```

$\wedge$	1	e_x	e_y	e_z	e_{xy}	e_{xz}	e_{yz}	e_{xyz}
1	1	e_x	e_y	e_z	e_{xy}	e_{xz}	e_{yz}	e_{xyz}
e_x	e_x	0	e_{xy}	e_{xz}	0	0	e_{xyz}	0
e_y	e_y	-e_{xy}	0	e_{yz}	0	-e_{xyz}	0	0
e_z	e_z	-e_{xz}	-e_{yz}	0	e_{xyz}	0	0	0
e_{xy}	e_{xy}	0	0	e_{xyz}	0	0	0	0
e_{xz}	e_{xz}	0	$-e_{xyz}$	0	0	0	0	0
e_{yz}	e_{yz}	e_{xyz}	0	0	0	0	0	0
e_{xyz}	e_{xyz}	0	0	0	0	0	0	0

Table 8: Wedge Product in Three Dimensional Euclidean Space

	1	e_x	e_y	e_z	e_{xy}	e_{xz}	e_{yz}	e_{xyz}
1	1	e_x	e_y	e_z	e_{xy}	e_{xz}	e_{yz}	e_{xyz}
e_x	e_x	1	e_{xy}	e_{xz}	e_y	e_z	e_{xyz}	e_{yz}
e_y	e_y	-e_{xy}	1	e_{yz}	- e_x	-e_{xyz}	e_z	-e_{xz}
e_z	e_z	-e_{xz}	-e_{yz}	1	e_{xyz}	-e_x	-e_y	e_{xy}
e_{xy}	e_{xy}	-e_y	e_x	e_{xyz}	-1	-e_{yz}	e_{xz}	-e_z
e_{xz}	e_{xz}	-e_z	-e_{xyz}	e_x	e_{yz}	-1	-e_{xy}	e_y
e_{yz}	e_{yz}	e_{xyz}	-e_z	e_y	-e_xz	e_{xy}	-1	-e_x
e_{xyz}	e_{xyz}	e_yz	-e_{xz}	e_{xy}	-e_z	e_y	-e_x	-1

Table 9: Geometric Product in Three Dimensional Euclidean Space

Three Dimensional Euclidean Space

In three dimensional Euclidean space, we have a scalar basis of 1, three vector basis beinge_x, e_y and e_z, three bivector basis being e_{xy}, e_{xz} and e_{yz}, and a trivector e_{xyz}. I am using e_{xz} for the bivector, rather than e_{zx}, to reduce careless mistakes in building my tables. It is easy to change the basis definition once the work is done. The pseudoscalar for this space is $I = e_{xyz}$. For this three dimensional space, the inverse of the pseudoscalar is $I^{-1} = -I = -e_{xyz}$.

In multiplication table format, the wedge product is shown in Table 8. Table 9 shows the geometric product for three dimensional Euclidean space.

$\vee$	1	e_x	e_y	e_z	e_{xy}	e_{xz}	e_{yz}	e_{xyz}
1	1	1	1	1	-1	-1	-1	?
e_x	1	e_x	1	1	?	?	?	?
e_y	1	-1	e_y	1	?	?	?	?
e_z	1	-1	-1	e_z	?	?	?	?
e_{xy}	-1	?	?	?	$-e_{xy}$	?	?	?
e_{xz}	-1	?	?	?	?	-e_{xz}	?	?
e_{yz}	-1	?	?	?	?	?	-e_{yz}	?
e_{xyz}	?	?	?	?	?	?	?	?

Table 10: Partial Regressive Product in Three Dimensional Euclidean Space

$\vee$	1	e_x	e_y	e_z	e_{xy}	e_{xz}	e_{yz}	e_{xyz}
1	1	1	1	1	-1	-1	-1	-1
e_x	1	e_x	1	1	-e_x	-e_x	-1	-e_x
e_y	1	-1	e_y	1	-e_y	1	-e_y	-e_y
e_z	1	-1	-1	e_z	-1	-e_z	-e_z	-e_z
e_{xy}	-1	-e_x	-e_y	-1	$-e_{xy}$	e_x	e_y	$-e_{xy}$
e_{xz}	-1	-e_x	1	-e_z	-e_x	-e_{xz}	e_z	-e_{xz}
e_{yz}	-1	-1	-e_y	-e_z	-e_y	-e_z	-e_{yz}	-e_{yz}
e_{xyz}	-1	-e_x	-e_y	-e_z	-e_{xy}	-e_{xz}	-e_{yz}	-e_{xyz}

Table 11: Regressive Product in Three Dimensional Euclidean Space

In Table 10, we fill in the known regressive product terms from one and two dimensions. In Table 11, courtesy of the software program Regressive-ProductPerHestenes.c, we have the full table.

In C programming format, the component equations are

```
c.q   = +a.q*b.q + a.q*b.x  + a.q*b.y  + a.q*b.z   - a.q*b.xy
        - a.q*b.xz  - a.q*b.yz - a.q*b.xyz + a.x*b.q
        + a.x*b.y + a.x*b.z - a.x*b.yz + a.y*b.q
        - a.y*b.x + a.y*b.z + a.y*b.xz + a.z*b.q
        - a.z*b.x - a.z*b.y - a.z*b.xy - a.xy*b.q
        - a.xy*b.z - a.xz*b.q + a.xz*b.y - a.yz*b.q
```

```
        - a.yz*b.x - a.xyz *b.q
c.x   = +a.x*b.x - a.x*b.xy - a.x*b.xz - a.x*b.xyz - a.xy*b.x
        + a.xy*b.xz - a.xz*b.x - a.xz*b.xy - a.xyz*b.x
c.y   = +a.y*b.y - a.y*b.xy - a.y*b.yz - a.y*b.xyz - a.xy*b.y
        + a.xy*b.yz - a.yz*b.y - a.yz*b.xy - a.xyz*b.y
c.z   = +a.z*b.z - a.z*b.xz - a.z*b.yz - a.z*b.xyz - a.xz*b.z
        + a.xz*b.yz - a.yz*b.z - a.yz*b.xz - a.xyz*b.z
c.xy  = -a.xy*b.xy - a.xy*b.xyz - a.xyz*b.xy
c.xz  = -a.xz*b.xz - a.xz*b.xyz - a.xyz*b.xz
c.yz  = -a.yz*b.yz - a.yz*b.xyz - a.xyz*b.yz
c.xyz = -a.xyz*b.xyz
```

Minkowski Spacetime

The novelty with Minkowski spacetime is that the metric for the time basis is negative. This means that $e_t e_t = -1$, rather than $+1$ for Euclidean space. I wrote a program to handle the calculations. The results are presented in Tables 12-14.

http://www.kurtnalty.com/RegressiveProductPerHestenes.c

The component equations in C programming format are

```
c.q    =  + a.q   *b.q    + a.q   *b.x    + a.q   *b.y    + a.q   *b.z
 - a.q   *b.t    - a.q   *b.xy   - a.q   *b.xz   - a.q   *b.yz   + a.q   *b.xt
  + a.q   *b.yt   + a.q   *b.zt   - a.q   *b.xyz  + a.q   *b.xyt  + a.q   *b.xzt
  + a.q   *b.yzt  - a.q   *b.xyzt + a.x   *b.q    + a.x   *b.y    + a.x   *b.z
   - a.x   *b.t    - a.x   *b.yz   + a.x   *b.yt   + a.x   *b.zt   + a.x   *b.yzt
 + a.y   *b.q    - a.y   *b.x    + a.y   *b.z    - a.y   *b.t    + a.y   *b.xz
  - a.y   *b.xt   + a.y   *b.zt   - a.y   *b.xzt  + a.z   *b.q    - a.z   *b.x
   - a.z   *b.y    - a.z   *b.t    - a.z   *b.xy   - a.z   *b.xt   - a.z   *b.yt
  + a.z   *b.xyt  - a.t   *b.q    + a.t   *b.x    + a.t   *b.y    + a.t   *b.z
   + a.t   *b.xy   + a.t   *b.xz   + a.t   *b.yz   - a.t   *b.xyz  - a.xy  *b.q
   - a.xy  *b.z    + a.xy  *b.t    - a.xy  *b.zt   - a.xz  *b.q    + a.xz  *b.y
    + a.xz  *b.t    + a.xz  *b.yt   - a.yz  *b.q    - a.yz  *b.x    + a.yz  *b.t
   - a.yz  *b.xt   + a.xt  *b.q    - a.xt  *b.y    - a.xt  *b.z    - a.xt  *b.yz
  + a.yt  *b.q    + a.yt  *b.x    - a.yt  *b.z    + a.yt  *b.xz   + a.zt  *b.q
  + a.zt  *b.x    + a.zt  *b.y    - a.zt  *b.xy   - a.xyz *b.q    + a.xyz *b.t
  + a.xyt *b.q    - a.xyt *b.z    + a.xzt *b.q    + a.xzt *b.y    + a.yzt *b.q
  - a.yzt *b.x    - a.xyzt*b.q
c.x    =  + a.x   *b.x    - a.x   *b.xy   - a.x   *b.xz   + a.x   *b.xt
  - a.x   *b.xyz  + a.x   *b.xyt  + a.x   *b.xzt  - a.x   *b.xyzt - a.xy  *b.x
   + a.xy  *b.xz   - a.xy  *b.xt   - a.xy  *b.xzt  - a.xz  *b.x    - a.xz  *b.xy
 - a.xz  *b.xt   + a.xz  *b.xyt  + a.xt  *b.x    + a.xt  *b.xy   + a.xt  *b.xz
```

```
 - a.xt  *b.xyz  - a.xyz *b.x    - a.xyz *b.xt   + a.xyt *b.x    + a.xyt *b.xz
  + a.xzt *b.x    - a.xzt *b.xy   - a.xyzt*b.x
c.y    =  + a.y   *b.y    - a.y   *b.xy   - a.y   *b.yz   + a.y   *b.yt
   - a.y   *b.xyz  + a.y   *b.xyt  + a.y   *b.yzt  - a.y   *b.xyzt - a.xy  *b.y
   + a.xy  *b.yz   - a.xy  *b.yt   - a.xy  *b.yzt  - a.yz  *b.y    - a.yz  *b.xy
  - a.yz  *b.yt   + a.yz  *b.xyt  + a.yt  *b.y    + a.yt  *b.xy   + a.yt  *b.yz
  - a.yt  *b.xyz  - a.xyz *b.y    - a.xyz *b.yt   + a.xyt *b.y    + a.xyt *b.yz
  + a.yzt *b.y    - a.yzt *b.xy   - a.xyzt*b.y
c.z    =  + a.z   *b.z    - a.z   *b.xz   - a.z   *b.yz   + a.z   *b.zt
 - a.z   *b.xyz  + a.z   *b.xzt  + a.z   *b.yzt  - a.z   *b.xyzt - a.xz  *b.z
   + a.xz  *b.yz   - a.xz  *b.zt   - a.xz  *b.yzt  - a.yz  *b.z    - a.yz  *b.xz
  - a.yz  *b.zt   + a.yz  *b.xzt  + a.zt  *b.z    + a.zt  *b.xz   + a.zt  *b.yz
 - a.zt  *b.xyz  - a.xyz *b.z    - a.xyz *b.zt   + a.xzt *b.z    + a.xzt *b.yz
 + a.yzt *b.z    - a.yzt *b.xz   - a.xyzt*b.z
c.t    =  - a.t   *b.t    + a.t   *b.xt   + a.t   *b.yt   + a.t   *b.zt
  + a.t   *b.xyt  + a.t   *b.xzt  + a.t   *b.yzt  - a.t   *b.xyzt + a.xt  *b.t
   - a.xt  *b.yt   - a.xt  *b.zt   - a.xt  *b.yzt  + a.yt  *b.t    + a.yt  *b.xt
 - a.yt  *b.zt   + a.yt  *b.xzt  + a.zt  *b.t    + a.zt  *b.xt   + a.zt  *b.yt
 - a.zt  *b.xyt  + a.xyt *b.t    - a.xyt *b.zt   + a.xzt *b.t    + a.xzt *b.yt
 + a.yzt *b.t    - a.yzt *b.xt   - a.xyzt*b.t
c.xy   =  - a.xy  *b.xy   - a.xy  *b.xyz  + a.xy  *b.xyt  - a.xy  *b.xyzt
 - a.xyz *b.xy   + a.xyz *b.xyt  + a.xyt *b.xy   - a.xyt *b.xyz  - a.xyzt*b.xy
c.xz   =  - a.xz  *b.xz   - a.xz  *b.xyz  + a.xz  *b.xzt  - a.xz  *b.xyzt
 - a.xyz *b.xz   + a.xyz *b.xzt  + a.xzt *b.xz   - a.xzt *b.xyz  - a.xyzt*b.xz
c.yz   =  - a.yz  *b.yz   - a.yz  *b.xyz  + a.yz  *b.yzt  - a.yz  *b.xyzt
 - a.xyz *b.yz   + a.xyz *b.yzt  + a.yzt *b.yz   - a.yzt *b.xyz  - a.xyzt*b.yz
c.xt   =  + a.xt  *b.xt   + a.xt  *b.xyt  + a.xt  *b.xzt  - a.xt  *b.xyzt
 + a.xyt *b.xt   + a.xyt *b.xzt  + a.xzt *b.xt   - a.xzt *b.xyt  - a.xyzt*b.xt
c.yt   =  + a.yt  *b.yt   + a.yt  *b.xyt  + a.yt  *b.yzt  - a.yt  *b.xyzt
 + a.xyt *b.yt   + a.xyt *b.yzt  + a.yzt *b.yt   - a.yzt *b.xyt  - a.xyzt*b.yt
c.zt   =  + a.zt  *b.zt   + a.zt  *b.xzt  + a.zt  *b.yzt  - a.zt  *b.xyzt
 + a.xzt *b.zt   + a.xzt *b.yzt  + a.yzt *b.zt   - a.yzt *b.xzt  - a.xyzt*b.zt
c.xyz  =  - a.xyz *b.xyz  - a.xyz *b.xyzt - a.xyzt*b.xyz
c.xyt  =  + a.xyt *b.xyt  - a.xyt *b.xyzt - a.xyzt*b.xyt
c.xzt  =  + a.xzt *b.xzt  - a.xzt *b.xyzt - a.xyzt*b.xzt
c.yzt  =  + a.yzt *b.yzt  - a.yzt *b.xyzt - a.xyzt*b.yzt
c.xyzt =  - a.xyzt*b.xyzt
```

Comments

The Hestenes regressive product is non-associative, as demonstrated by the program linked below.

q	+x	+y	+z	+t	+xy	+xz	+yz	+xt	+yt	+zt	+xyz	+xyt	+xzt	+yzt	+xyzt
x	0	+xy	+xz	+xt	0	0	+xyz	0	+xyt	+xzt	0	0	0	+xyzt	0
y	-xy	0	+yz	+yt	0	-xyz	0	-xyt	0	+yzt	0	0	-xyzt	0	0
z	-xz	-yz	0	+zt	+xyz	0	0	-xzt	-yzt	0	0	+xyzt	0	0	0
t	-xt	-yt	-zt	0	+xyt	+xzt	+yzt	0	0	0	-xyzt	0	0	0	0
xy	0	0	+xyz	+xyt	0	0	0	0	0	+xyzt	0	0	0	0	0
xz	0	-xyz	0	+xzt	0	0	0	0	-xyzt	0	0	0	0	0	0
yz	+xyz	0	0	+yzt	0	0	0	+xyzt	0	0	0	0	0	0	0
xt	0	-xyt	-xzt	0	0	0	+xyzt	0	0	0	0	0	0	0	0
yt	+xyt	0	-yzt	0	0	-xyzt	0	0	0	0	0	0	0	0	0
zt	+xzt	+yzt	0	0	+xyzt	0	0	0	0	0	0	0	0	0	0
xyz	0	0	0	+xyzt	0	0	0	0	0	0	0	0	0	0	0
xyt	0	0	-xyzt	0	0	0	0	0	0	0	0	0	0	0	0
xzt	0	+xyzt	0	0	0	0	0	0	0	0	0	0	0	0	0
yzt	-xyzt	0	0	0	0	0	0	0	0	0	0	0	0	0	0
xyzt	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12: Minkowski Wedge Product Multiplication Table

q	x	y	z	t	xy	xz	yz	xt	yt	zt	xyz	xyt	xzt	yzt	xyzt
x	q	xy	xz	xt	y	z	xyz	t	xyt	xzt	yz	yt	zt	xyzt	yzt
y	-xy	q	yz	yt	-x	-xyz	z	-xyt	t	yzt	-xz	-xt	-xyzt	zt	-xzt
z	-xz	-yz	q	zt	xyz	-x	-y	-xzt	-yzt	t	xy	xyzt	-xt	-yt	xyt
t	-xt	-yt	-zt	-q	xyt	xzt	yzt	x	y	z	-xyzt	-xy	-xz	-yz	xyz
xy	-y	x	xyz	xyt	-q	-yz	xz	-yt	xt	xyzt	-z	-t	-yzt	xzt	-zt
xz	-z	-xyz	x	xzt	yz	-q	-xy	-zt	-xyzt	xt	y	yzt	-t	-xyt	yt
yz	xyz	-z	y	yzt	-xz	xy	-q	xyzt	-zt	yt	-x	-xzt	xyt	-t	-xt
xt	-t	-xyt	-xzt	-x	yt	zt	xyzt	q	xy	xz	-yzt	-y	-z	-xyz	yz
yt	xyt	-t	-yzt	-y	-xt	-xyzt	zt	-xy	q	yz	xzt	x	xyz	-z	-xz
zt	xzt	yzt	-t	-z	xyzt	-xt	-yt	-xz	-yz	q	-xyt	-xyz	x	y	xy
xyz	yz	-xz	xy	xyzt	-z	y	-x	yzt	-xzt	xyt	-q	-zt	yt	-xt	-t
xyt	yt	-xt	-xyzt	-xy	-t	-yzt	xzt	-y	x	xyz	zt	q	yz	-xz	-z
xzt	zt	xyzt	-xt	-xz	yzt	-t	-xyt	-z	-xyz	x	-yt	-yz	q	xy	y
yzt	-xyzt	zt	-yt	-yz	-xzt	xyt	-t	xyz	-z	y	xt	xz	-xy	q	-x
xyzt	-yzt	xzt	-xyt	-xyz	-zt	yt	-xt	yz	-xz	xy	t	z	-y	x	-q

Table 13: Minkowski Geometric Algebra Multiplication Table

V	+q	+x	+y	+z	+t	+xy	+xz	+yz	+xt	+yt	+zt	+xyz	+xyt	+xzt	+yzt	+xyzt
q	+q	+q	+q	+q	-q	-q	-q	-q	+q	+q	+q	-q	+q	+q	+q	-q
x	+q	+x	+q	+q	-q	-x	-x	-q	+x	+q	+q	-x	+x	+x	+q	-x
y	+q	-q	+y	+q	-q	-y	+q	-y	-q	+y	+q	-y	+y	-q	+y	-y
z	+q	-q	-q	+z	-q	-q	-z	-z	-q	-q	+z	-z	+q	+z	+z	-z
t	-q	+q	+q	+q	-t	+q	+q	+q	+t	+t	+t	-q	+t	+t	+t	-t
xy	-q	-x	-y	-q	+q	-xy	+x	+y	-x	-y	-q	-xy	+xy	-x	-y	-xy
xz	-q	-x	+q	-z	+q	-x	-xz	+z	-x	+q	-z	-xz	+x	+xz	-z	-xz
yz	-q	-q	-y	-z	+q	-y	-z	-yz	-q	-y	-z	-yz	+y	+z	+yz	-yz
xt	+q	+x	-q	-q	+t	+x	+x	-q	+xt	-t	-t	-x	+xt	+xt	-t	-xt
yt	+q	+q	+y	-q	+t	+y	+q	+y	+t	+yt	-t	-y	+yt	+t	+yt	-yt
zt	+q	+q	+q	+z	+t	-q	+z	+z	+t	+t	+zt	-z	-t	+zt	+zt	-zt
xyz	-q	-x	-y	-z	+q	-xy	-xz	-yz	-x	-y	-z	-xyz	+xy	+xz	+yz	-xyz
xyt	+q	+x	+y	-q	+t	+xy	+x	+y	+xt	+yt	-t	-xy	+xyt	+xt	+yt	-xyt
xzt	+q	+x	+q	+z	+t	-x	+xz	+z	+xt	+t	+zt	-xz	-xt	+xzt	+zt	-xzt
yzt	+q	-q	+y	+z	+t	-y	-z	+yz	-t	+yt	+zt	-yz	-yt	-zt	+yzt	-yzt
xyzt	-q	-x	-y	-z	-t	-xy	-xz	-yz	-xt	-yt	-zt	-xyz	-xyt	-xzt	-yzt	-xyzt

Table 14: Minkowski Regressive Product Multiplication Table

http://www.kurtnalty.com/Hestenes_Regressive_Is_NonAssociative.cp

The negative signs on the pseudoscalar product terms in three and four dimensions are bothersome.

My next effort will be to revisit Eric Lengyel's antiwedge product, with the left and right dual forms taking a parameter for the span of the calculation.

References

[1] David Hestenes, *Universal Geometric Algebra* in SIMON STEVIN, A Quarterly Journal of Pure and Applied Mathematics, Volume 62 (1988), Number 3-4 (September-December 1988)

[2] John Browne, *Grassmann Algebra*, Barnard Publishing, 9-781479-197637

[3] Hermann Grassmann, *A New Branch of Mathematics*, Translated by Lloyd C. Kannenberg, Open Court Publishing, ISBN 0-8126-9276-4

[4] Chris Doran and Anthony Lasenby, *Geometric Algebra for Physicists*, Cambridge University Press, ISBN 978-0-521-71595-9

[5] Leo Dorst, Daniel Fontune and Stephen Mann, *Geometric Algebra for Computer Science*, Morgan Kaufmann Publishers, ISBN 978-0-12-374942-0

[6] David Hestenes and Garret Sobczyk, *Clifford Algebra to Geometric Calculus*, D. Reidal Publishing Company, ISBN 978-90-277-2581-5

[7] Anthony Lasenby and Chris Doran, *Lectures and Handouts 1999*, www.mrao.cam.ac.uk/clifford/ptIIIcourse/

[8] Eric Lengyel, *Foundations of Game Engine Development*, Terathon Software LLC, ISBN 9-780985-811747