Quaternion Quantum Field Theory Demystified: Special Relativity with Intervals but Without Metrics

Douglas Sweetser, sweetser@alum.mit.edu

A quaternion variation on the metric is explored. The quaternion approach takes into account phase, which could be an improvement. Rotations and boosts work as expect for quaternions. The Einstein summation convention will not be used in this body of work, nor with covariant or contravariant vectors because there are no indices. A variety of conjugates will be used in their place. An open technical issue is how to deal with derivatives in curved quaternion spacetime.

QMN.1009.6242

1. The Metric and Its Quaternion Square Root

A spacetime metric tensor takes as inputs 2 4-vectors and returns a Lorentz invariant scalar. An inverse metric is well-defined, but curiously, not a product of two metrics. Write the definition of a metric in a non-standard way:

$$g_{\mu\nu} = \begin{pmatrix} g_0\, g_0 & g_0\, g_1 & g_0\, g_2 & g_0\, g_3 \\ g_1\, g_0 & g_1\, g_1 & g_1\, g_2 & g_1\, g_3 \\ g_2\, g_0 & g_2\, g_1 & g_2\, g_2 & g_2\, g_3 \\ g_3\, g_0 & g_3\, g_1 & g_3\, g_2 & g_3\, g_3 \end{pmatrix} \tag{1}$$

A metric has 2 tasks: account for signs and values of a possibly dynamic curvature. Only three of these flip signs, those down the diagonal. The metric is symmetric.

Herman Weyl began the work with tetrads (or frames), which can be viewed as the square root of a metric using this definition:

$$g^{\mu\nu} = e^{\mu}_{a}\, e^{\nu}_{b}\, \eta^{ab} \tag{2}$$

where η^{ab} is the flat Minkowski metric. Together, the two e's do the work of one of the ten g_{xx}'s. The detailed relationship between these two ways of representing metrics has been the subject of much study.

For the quaternion approach to differential geometry, treat the e's as basis vectors. The basis vectors sit right next to the real value amplitudes assigned to those basis vectors:

calc

```
qvp[{C t es, R Ev}, {t es, R Ev}]
```

calc

$$\{-\mathrm{Ev}^2\, \mathrm{R}^2 + \mathrm{C\,es}^2\, \mathrm{t}^2,\ \mathrm{es\ Ev\ R\ t} + \mathrm{C\ es\ Ev\ R\ t}\}$$

$$\left(c\, t\, e_0, R\, \vec{E}\right)^2 = \begin{pmatrix} c^2\, t^2 [\![e_0]\!] - R^2 [\![\vec{E}]\!] & -2\, c\, t\, e_0\, R\, \vec{E} \\ 2\, c\, t\, e_0\, R\, \vec{E} & c^2\, t^2 [\![e_0]\!] - R^2 [\![\vec{E}]\!] \end{pmatrix} \tag{3}$$

The signs of the product of two basis vectors are fixed to make multiplication consistent with the rules of quaternion products. The off-diagonal elements are anti-symmetric, but the non-zero elements along the diagonal make the product asymmetric overall. The scalar is the Lorentz invariant interval. The phase is not used by most physicists because it is discarded by using the metric. The magnitude of the basis vectors will be one in flat spacetime. In curved spacetime, the magnitude does not need to be unity. That is the impact of curving spacetime: unity is always unity in the local reference frame, but one person's measure of unity does not have to be equal to another's measure of unity. Unity varies over volumes of curved spacetime. This leads to a generalization, if one compares the product of two slightly different basis vectors:

calc

```
qvp[{c t es, R Ev}, {c t esp, R Evp}]
```

calc

$$\{-\text{Ev Evp R}^2 + \text{c}^2\ \text{es esp t}^2,\ \text{c esp Ev R t} + \text{c es Evp R t} + (\text{Ev R}) \times (\text{Evp R})\}$$

$$\left(c\,t\,e_0, R\,\vec{E}\right)\left(c\,t\,e'_0, R\,\vec{E'}\right) = \begin{pmatrix} c^2\,t^2\,e_0\,e'_0 - R^2\,\vec{E}\cdot\vec{E'} & -c\,t\,\vec{R}\left(e_0\,\vec{E'} + \vec{E}\,e'_0\right) - R\,\vec{E}\times R\,\vec{E'} \\ c\,t\,R\left(e_0\,\vec{E'} + \vec{E}\,e'_0\right) + R\,\vec{E}\times R\,\vec{E'} & c^2\,t^2\,e_0\,e'_0 - R^2\,\vec{E}\cdot\vec{E'} \end{pmatrix} \tag{4}$$

The cross product might provide a path to torsion.

There are notable differences in the quaternion approach when compared to using a metric. The quaternion approach is automorphic - the inputs are quaternions as are the outputs. Consider the term $g_0\,g_1$, which looks like the product of a scalar and a vector. In the metric approach, such a term gets mapped to the scalar. That does not sound consistent with standard rules of vectors, where a scalar times a vector can either increase, decrease, or do nothing to the length of a vector. For the quaternion approach, the curvature contained in such a term is still there, but it is placed in a different location - one for a vector instead of a scalar.

The metric approach used throughout mathematical physics presumes there is no rule for multiplication or division. That assumption is essential for all the work being done in higher dimensions. Without a product rule, the only place to put the any information about curvature is in the scalar. This work always stays confined to 3 vectorial dimensions and a scalar dimensions [note, I will try to avoid saying 4D since it implies incorrectly that all dimensions are the same when there is a critical difference between the scalar and 3-vectors]. For quaternions, there is a product rule. This means the information about curvature can go into vector slot if it belongs there.

2. Rotations and Boosts

I must confess I am not enthusiastic about replacing metrics. Far too many brilliant people have worked with them, two of the bigger old names being Riemann and Einstein, but also any professional working today. This is not a battle between people however. It is an accounting issue. Once I saw a scalar times a vector value going into the scalar position, that formed the basis of my objection to details of implementation.

The big picture was defined well by Einstein back in 1905:

1 - the laws of physics are the same for all inertial reference frames

2 - the speed of light is the same for all inertial observers

Quaternion as basis vectors must achieve these goals. Does a rotation in space alter the spacetime interval? Spacial rotations where the first practical use for quaternions, and remain their primary role today for rocket scientists and game designers alike. They are done by forming a quaternion triple product: sandwich a quaternion to be rotated between two unitary quaternions:

calc

```
zrot = {Cos[α], Sin[α], 0, 0};
v = {c t, x, y, z};
rotatedv = triple[zrot, v, conj[zrot]] // Simplify
```

calc

$$\{\text{c t, x, y Cos}[2\,\alpha] - \text{z Sin}[2\,\alpha],\ \text{z Cos}[2\,\alpha] + \text{y Sin}[2\,\alpha]\}$$

calc

```
{c t, x, y Cos[2 α] - z Sin[2 α], z Cos[2 α] + y Sin[2 α]}
```

calc

$$\{\text{c t, x, y Cos}[2\,\alpha] - \text{z Sin}[2\,\alpha],\ \text{z Cos}[2\,\alpha] + \text{y Sin}[2\,\alpha]\}$$

$$\begin{aligned} &\text{R}_\text{x} = (\cos(\alpha), \sin(\alpha), 0, 0) \\ &\text{v'} = \text{R}_\text{x}\ \text{v}\ \text{R}_\text{x}^* = (\text{c t, x, y cos}[2\,\alpha] - \text{z sin}[2\,\alpha],\ \text{z cos}[2\,\alpha] + \text{y sin}[2\,\alpha]) \end{aligned} \tag{5}$$

Time and position along the x axis are unaltered, while y and z mix it up with sines and cosines. Calculate the squares:

calc
```
square[v]
square[rotatedv] // Simplify
```

calc
```
{c^2 t^2 - x^2 - y^2 - z^2, 2 c t x, 2 c t y, 2 c t z}
```

calc
```
{c^2 t^2 - x^2 - y^2 - z^2, 2 c t x, 2 c t (y Cos[2 α] - z Sin[2 α]), 2 c t (z Cos[2 α] + y Sin[2 α])}
```

$$\begin{aligned} v^2 &= \left(c^2 t^2 - x^2 - y^2 - z^2,\ 2\,c\,t\,x,\ 2\,c\,t\,y,\ 2\,c\,t\,z\right) \\ v'^2 &= \left(c^2 t^2 - x^2 - y^2 - z^2,\ 2\,c\,t\,x,\ 2\,c\,t\,(y\cos[2\alpha] - z\sin[2\alpha]),\ 2\,c\,t\,(z\cos[2\alpha] + y\sin[2\alpha])\right) \end{aligned} \tag{6}$$

As expected, rotation does not change the scalar, but it does alter the phase. The metric approach ignores the phase, a bad practice since the phase holds valid information.

Boosts are handled in a similar way. This time hyperbolic sines and cosines take the place in the rotation matrix. There is a problem that the hyperbolic sign does not appear in the correct place, but that can be fixed (see http://visualphysics.org/preprints/qmn10091026):

calc
```
Bx = {Cosh[β], Sinh[β], 0, 0}
boostedv = triple[Bx, v, conj[Bx]] +
    (conj[triple[Bx, Bx, v]] - conj[triple[conj[Bx], conj[Bx], v]]) / 2 // Simplify
```

calc
```
{Cosh[β], Sinh[β], 0, 0}
```

calc
```
{c t Cosh[2 β] - x Sinh[2 β], x Cosh[2 β] - c t Sinh[2 β], y, z}
```

$$\begin{aligned} B_x &= (\cosh[\beta], \sinh[\beta], 0, 0) \\ v' &= B_x\, v\, B_x^* + \left((B_x\, B_x\, v)^* - \left(B_x^*\, B_x^*\, v\right)^*\right)/2 \\ &= (c\,t\cosh[2\,\beta] - x\sinh[2\,\beta], x\cosh[2\,\beta] - c\,t\sinh[2\,\beta], y, z) \\ &= (\gamma\,c\,t - \gamma\beta\,x, \gamma\,x - \gamma\beta\,c\,t, y, z) \end{aligned} \tag{7}$$

Square the boosted vector:

calc
```
square[boostedv] // Simplify
```

calc
```
{c^2 t^2 - x^2 - y^2 - z^2, 2 (x Cosh[2 β] - c t Sinh[2 β]) (c t Cosh[2 β] - x Sinh[2 β]),
 2 y (c t Cosh[2 β] - x Sinh[2 β]), 2 z (c t Cosh[2 β] - x Sinh[2 β])}
```

$$\begin{aligned} v'^2 &= \left(c^2 t^2 - x^2 - y^2 - z^2, 2\,(x\cosh[2\,\beta] - c\,t\sinh[2\,\beta])\,(c\,t\cosh[2\,\beta] - x\sinh[2\,\beta]),\right. \\ &\quad \left. 2\,y\,(c\,t\cosh[2\,\beta] - x\sinh[2\,\beta]), 2\,z\,(c\,t\cosh[2\,\beta] - x\sinh[2\,\beta])\right) \\ &= \left(c^2 t^2 - x^2 - y^2 - z^2, 2\,(\gamma\,x - \gamma\beta\,c\,t)\,(\gamma\,c\,t - \gamma\beta\,x),\ 2\,y\,(\gamma\,c\,t - \gamma\beta x),\ 2\,z\,(\gamma\,c\,t - \gamma\beta x)\right) \end{aligned} \tag{8}$$

The scalar is unchanged as it must be if this quaternion approach is to be valid. The stretched out factors of time and x are clear in the phase.

3. Conjugate Operators

One thing that will never be used in this work is the Einstein summation convention. That tool keeps the scalar and tosses the phase. There is no means or need to switch between covariant and contravariant indexes. What will be used extensively are conjugate operators. There is the familiar conjugate operator:

calc
```
conj[v]
```

calc
```
{c t, -x, -y, -z}
```

$$v^* = (c\,t, -x, -y, -z) \tag{9}$$

The scalar keeps its sign, while the others flip. A profound lesson of quantum field theory is to not treat time differently from space. Therefore we need involutive anti-automorphism for the spacial coordinates:

calc

```
conj0[v]
conj1[v]
conj2[v]
conj3[v]
```

calc

```
{c t, -x, -y, -z}
```

calc

```
{-c t, x, -y, -z}
```

calc

```
{-c t, -x, y, -z}
```

calc

```
{-c t, -x, -y, z}
```

$$
\begin{aligned}
v^{*00} &\equiv (1\ v\ 1)^* = (c\,t, -x, -y, -z) \\
v^{*11} &\equiv (i\ v\ i)^* = (-c\,t, x, -y, -z) \\
v^{*22} &\equiv (j\ v\ j)^* = (-c\,t, -x, y, -z) \\
v^{*33} &\equiv (k\ v\ k)^* = (-c\,t, -x, -y, z)
\end{aligned}
\tag{10}
$$

These conjugate operators are starting to look isomorphic to the gamma matrices introduce in an earlier preprint (http://visualphysics.org/preprints/qmn10096241). Now the gamma matrices can be viewed as a complete set of space-time involutive anti-automorphisms.

One important issue that has not been worked out at this time are derivatives in curved spacetime. Fortunately for work in quaternion quantum field theory, curvature does not play a role in practical calculations. It is important to acknowledge a significant technical omission at this time, even as this work proceeds.

Notebook tools skipped in output.

calc

Tools

calc

For treating a quaternion as a scalar and 3-vector:

calc

```
qvp[q1_, q2_] := Module[{s, V},
  s = q1[[1]] q2[[1]] - q1[[2]] q2[[2]];
  V = q1[[1]] q2[[2]] + q1[[2]] q2[[1]];
  If[(q1[[2]] == q2[[2]]) || (q1[[2]] == 0) || (q2[[2]] == 0), , ,
   V += Cross[q1[[2]], q2[[2]]]];
  {s, V}]
squarev[q1_] := qvp[q1, q1]
triplev[q1_, q2_, q3_] := qvp[qvp[q1, q2], q3]
conjv[q1_] := {q1[[1]], -q1[[2]]}
```

calc

For treating all 4 components:

calc

```
qp[q1_, q2_] := {q1[[1]] q2[[1]] - q1[[2]] q2[[2]] - q1[[3]] q2[[3]] - q1[[4]] q2[[4]],
  q1[[1]] q2[[2]] + q1[[2]] q2[[1]] + q1[[3]] q2[[4]] - q1[[4]] q2[[3]],
  q1[[1]] q2[[3]] + q1[[3]] q2[[1]] + q1[[4]] q2[[2]] - q1[[2]] q2[[4]],
  q1[[1]] q2[[4]] + q1[[4]] q2[[1]] + q1[[2]] q2[[3]] - q1[[3]] q2[[2]]}
square[q1_] := qp[q1, q1]
triple[q1_, q2_, q3_] := qp[qp[q1, q2], q3]
conj[q1_] := {q1[[1]], -q1[[2]], -q1[[3]], -q1[[4]]}
conj0[q1_] := conj[triple[{1, 0, 0, 0}, q1, {1, 0, 0, 0}]]
conj1[q1_] := conj[triple[{0, 1, 0, 0}, q1, {0, 1, 0, 0}]]
conj2[q1_] := conj[triple[{0, 0, 1, 0}, q1, {0, 0, 1, 0}]]
conj3[q1_] := conj[triple[{0, 0, 0, 1}, q1, {0, 0, 0, 1}]]
conj13[q1_] := conj[triple[{0, 1, 0, 0}, q1, {0, 0, 0, 1}]]
```