Rotations and boost without matrices

Keith Pedersen

NOTE: This paper is a reprint from the Appendix of my doctoral thesis "Expanding the HEP frontier with boosted b-tags and the QCD power spectrum". It uses the nomenclature that $\vec{w}$ is a 3-vector, $\hat{w}$ is a unit 3-vector, and $\boldsymbol{p}$ is a 4-vector.

Given my investigation into unsafe operations in floating point arithmetic, I decided to write my own C++ vector classes, so that I could be assured that they internally respect floating point limitations (e.g., the options available with ROOT use `acos` to find interior angle). My `Vector2`, `Vector3`, and `Vector4` classes store 2, 3, and 4-vectors (respectively), and are highly modified extensions of classes originally written by Z. Sullivan. In addition to defining basic functionality (add, scale, dot, cross, etc.), I needed tools to rotate 3-vectors and boost 4-vectors; this section presents the mathematical formalism I chose to use to accomplish those latter tasks.

NOTE: The Rodrigues formula (both for rotations and boost) is shown for reference, whereas Section I B presents my own work.

I. ROTATING VECTORS

The most natural method to rotate 3-vectors is via 3×3 square matrices. However, it can be cumbersome to implement these matrices for an arbitrary rotation, since they require hard-coding large trigonometric expressions. The size of these expressions lends itself to the possibility of floating point cancellation, and ensuring their numerical stability is a headache. However, one does not have to use matrices to define rotations; the Rodrigues formula rotates using only the dot and cross product.

A. The Rodrigues rotation formula

To implement an active, right-handed (RH) rotation of vector $\vec{w}$ by some angle ψ about some axis $\hat{x}$, the victim is projected into two pieces: its longitudinal length and transverse

vector

$$w_L = \vec{w} \cdot \hat{x}\,, \tag{1}$$

$$\vec{w}_T = \vec{w} - w_L \hat{x}\,. \tag{2}$$

We can then calculate another transverse $\vec{w}_T$, rotated by 90° relative to the first;

$$\vec{w}_{T'} = \hat{x} \times \vec{w}\,. \tag{3}$$

The two transverse $\vec{w}$ form a basis for the rotation, which $\hat{x}$ is unaltered by; this allows us to calculate the rotated vector quite simply

$$\vec{w}' = R(\vec{w}) = w_L \hat{x} + \cos\psi\, \vec{w}_T + \sin\psi\, \vec{w}_{T'}\,. \tag{4}$$

We can validate this scheme by showing it preserves the defining properties of rotations. First, the angle to the axis is unaltered;

$$\begin{aligned} \vec{w}' \cdot \hat{x} &= (\vec{w} \cdot \hat{x})\cancelto{1}{\hat{x} \cdot \hat{x}} + \cos\psi(\vec{w} \cdot \hat{x} - (\vec{w} \cdot \hat{x})\cancelto{1}{\hat{x} \cdot \hat{x}}) + \sin\psi\cancelto{0}{((\hat{x} \times \vec{w}) \cdot \hat{x})} \\ &= \vec{w} \cdot \hat{x}\,. \end{aligned} \tag{5}$$

(since $\vec{a} \cdot (\vec{b} \times \vec{c}) = \vec{b} \cdot (\vec{c} \times \vec{a}) = \vec{c} \cdot (\vec{a} \times \vec{b})$). Second, the vector's length is unaltered;

$$\begin{aligned} |\vec{w}'|^2 &= (\vec{w} \cdot \hat{x}) + (\sin^2\psi + \cos^2\psi)(|\vec{w}|^2 - (\vec{w} \cdot \hat{x})^2) \\ &= |\vec{w}|^2\,. \end{aligned} \tag{6}$$

This result uses the identity $|\vec{u} \times \vec{v}|^2 = |\vec{u}|^2 |\vec{v}|^2 - (\vec{u} \cdot \vec{v})^2$ and implicitly draws upon the mutual orthogonality of $\vec{w}_L$, $\vec{w}_T$ and $\vec{w}_{T'}$.

B. The hidden degree for freedom when rotating $\vec{u}$ to $\vec{v}$

Instead of an axis $\hat{x}$ and angle ψ as our input degrees of freedom, we may want the rotation that takes vector $\vec{u} \to \vec{v}$. We can reuse Equation 4, and define:

$$\cos\psi = \hat{u} \cdot \hat{v}\,, \tag{7}$$

$$\sin\psi = |\vec{u} \times \vec{v}|\,, \tag{8}$$

$$\hat{x}_1 = \frac{\hat{u} \times \hat{v}}{|\hat{u} \times \hat{v}|} = \frac{\hat{u} \times \hat{v}}{\sin\psi}\,. \tag{9}$$

However, $\hat{x}_1$ is only one *possible* rotation axis which takes $\vec{u} \to \vec{v}$. Consider a rotation of $\psi = \pi$ about the axis bisecting the two normalized vectors

$$\hat{x}_2 = \frac{\hat{u} + \hat{v}}{|\hat{u} + \hat{v}|} = \frac{\hat{u} + \hat{v}}{\sqrt{2(1 + \hat{u} \cdot \hat{v})}} \,. \tag{10}$$

This rotation also clearly takes $\vec{u} \to \vec{v}$. In fact, any axis which satisfies

$$\hat{u} \cdot \hat{x} = \hat{v} \cdot \hat{x} \tag{11}$$

defines a valid rotation (since $\vec{u}$ and $\vec{v}$ will have the same latitude relative to $\hat{x}$, and thus trace out the same circle during the rotation). The "rotation which takes $\vec{u} \to \vec{v}$" is ambiguous!

This ambiguity stems from a hidden degree of freedom. We are free to choose a coordinate system where

$$\vec{u} = (0,\, 0,\, 1) \,, \tag{12}$$

$$\vec{v} = (\cos\phi \sin\theta,\, \sin\phi \sin\theta,\, \cos\theta) \,. \tag{13}$$

We can get from $\vec{u}$ to $\vec{v}$ in two steps: (i) a RH rotation about $\hat{z}$ by angle ϕ and (ii) a RH rotation about $\hat{y}'$ (the new y-axis, after the first rotation) by angle θ. This procedure uses only two of the three Euler angles required to cover SO(3) (the group of 3-dimensional rotations). To complete the coverage, we need a final RH rotation about $\hat{z}''$ (the final z-axis, which in our case is the newly minted $\vec{v}$).

By construction, this post-rotation about $\vec{v}$ by angle ω cannot alter $\vec{v}$, so it does not spoil the original purpose of this rotation (take $\vec{u} \to \vec{v}$). Instead, ω determines what happens to *every other* vector, and does so by selecting *one* axis $\hat{x}$ from the set which map $\vec{u} \to \vec{v}$. If we can find this $\hat{x}$, we can describe the complete operation as a single rotation (instead of two sequential rotations). We have already found two valid $\hat{x}$ (Eqs. 9 and 10), and they are fortuitously orthogonal, so we can use them to construct a basis that parameterizes all possible axes of rotation

$$\hat{x} = a\,\hat{x}_1 + b\,\hat{x}_2 \,. \tag{14}$$

Our task is now clear; given $\vec{u}$, $\vec{v}$ and ω, determine a and b to find $\hat{x}$.

R_1 is the rotation about $\hat{x}_1$ by $\theta = \arccos(\hat{u} \cdot \hat{v})$ and R_2 is the rotation about $\hat{x}_2$ by ω. Applying them consecutively produces a composite rotation which cannot alter the axis of rotation;

$$\hat{x} = R_2(R_1(\hat{x})) \,. \tag{15}$$

Since this defining property applies to both of $\hat{x}$'s component individually (and linearly), we can solve for a and b by using unitarity as one equation (i.e., $a^2 + b^2 = 1$), then obtain the other equation by calculating

$$\begin{aligned} a &= R_2(R_1(a\,\hat{x}_1 + b\,\hat{x}_2)) \cdot \hat{x}_1 \\ &= a\,R_2(R_1(\hat{x}_1)) \cdot \hat{x}_1 + b\,R_2(R_1(\hat{x}_2)) \cdot \hat{x}_1\,. \end{aligned} \tag{16}$$

Beginning with $\hat{x}_1$, we are lucky that R_1 does not alter its own axis, while R_2 creates only one term parallel to $\hat{x}_1$;

$$R_1(\hat{x}_1) = \hat{x}_1\ ; \tag{17}$$

$$R_2(R_1(\hat{x}_1)) \cdot \hat{x}_1 = \cancelto{0}{(\hat{x}_1 \cdot \hat{v})\hat{v}} + \cos\omega(\hat{x}_1 - 0) + \sin\omega\underbrace{(\hat{v} \times \hat{x}_1)}_{\perp \text{ to } \hat{x}_1}) \cdot \hat{x}_1 = \cos\omega\,. \tag{18}$$

The effect on $\hat{x}_2$ is slightly more complicated;

$$\begin{aligned} R_1(\hat{x}_2) &= \cancelto{0}{(\hat{x}_2 \cdot \hat{x}_1)\hat{x}_1} + \cos\theta(\hat{x}_2 - 0) + \sin\theta(\hat{x}_1 \times \hat{x}_2) \\ &= \frac{1}{\sqrt{2(1+\cos(\theta))}}\left(\cos\theta\,(\hat{u}+\hat{v}) + \sin\theta\,\frac{(\hat{u}\times\hat{v})}{\sin\theta} \times (\hat{u}+\hat{v})\right) \\ &= \frac{1}{\sqrt{2(1+\cos(\theta))}}\left(\cos\theta\,(\hat{u}+\hat{v}) + \hat{v} - \hat{u}\cos\theta + \hat{v}\cos\theta - \hat{u}\right) \\ &= \frac{1}{\sqrt{2(1+\cos(\theta))}}\left((2\cos\theta+1)\hat{v} - \hat{u}\right)\,, \end{aligned} \tag{19}$$

using $(\vec{a}\times\vec{b})\times\vec{c} = \vec{b}\,(\vec{a}\cdot\vec{c}) - \vec{a}\,(\vec{b}\cdot\vec{c})$. Before we put all of Equation 19 through R_2, we should recall that the final equation uses only terms parallel to $\hat{x}_1$. R_2 returns only terms which are (i) parallel to the incoming vector (which in this case is in the uv-plane, and thus orthogonal to $\hat{x}_1$), (ii) parallel to $\hat{v}$ (also in the uv plane) and (iii) perpendicular to $\hat{v}$ (via $\hat{v}\times\vec{w}$). Hence, only the 3$^{\text{rd}}$ piece is meaningful, and since $\hat{v}\times\hat{v} = 0$, we only need to give it Equation 19's $\hat{u}$ term;

$$\begin{aligned} R_2(R_1(\hat{x}_2)) \cdot \hat{x}_1 &= \frac{1}{\sqrt{2(1+\cos(\theta))}}\,R_2(-\hat{u}) \cdot \hat{x}_1 = \frac{1}{\sqrt{2(1+\cos(\theta))}}(\hat{v}\times(-\hat{u})) \cdot \hat{x}_1 \\ &= \sin\omega\frac{\sin\theta}{\sqrt{2(1+\cos(\theta))}}\,, \end{aligned} \tag{20}$$

using $\hat{u}\times\hat{v} = \sin\theta\,\hat{x}_1$. Combining Equation 16, 17 and 20 we get our system of equations

$$a = a\cos\omega + b\sin\omega\frac{\sin\theta}{\sqrt{2(1+\cos(\theta))}}\,, \tag{21}$$

$$1 = a^2 + b^2\,. \tag{22}$$

Having done the hard work (see the Appendix), we can plug our system of equations into Mathematica to obtain

$$a = 2\cos(\omega/2)\frac{\sin(\theta/2)}{c(\theta)}\,, \tag{23}$$

$$b = 2\sin(\omega/2)\frac{1}{c(\theta)}\,, \tag{24}$$

$$c(\theta) = \sqrt{3 - \cos(\omega) - \cos(\theta)(1+\cos(\omega))}\,. \tag{25}$$

I then used Mathematica to validate this solution by checking that $\hat{x}$ matches the eigenvector of the composite rotation $R_2(R_1(\vec{w}))$ (up to the parity operation $\hat{x} \to -\hat{x}$, an irreducibly ambiguity of the eigenvector).

However, $c(\theta)$ is numerically unstable if used naïvely, due to the cosine cancellations. These terms should be rewritten in a form which *doubles* the precision of the floating point result

$$1 - \cos(t) \mapsto 2\sin^2(t/2)\,. \tag{26}$$

This gives us

$$c(\theta) \mapsto \sqrt{2\sin^2(\theta/2) + 2\sin^2(\omega/2) + (1 - \cos(\theta)\cos(\omega))}\,. \tag{27}$$

The final cancellation can be corrected using

$$\cos(\theta)\cos(\omega) = \frac{1}{2}(\cos(\theta+\omega) + \cos(\theta-\omega))\,; \tag{28}$$

$$(1 - \cos(\theta)\cos(\omega)) \mapsto \frac{1}{2}(1 - \cos(\theta+\omega)) + \frac{1}{2}(1 - \cos(\theta-\omega))\,. \tag{29}$$

Again using Equation 26, we obtain the final expression

$$c(\theta) \mapsto \sqrt{2\sin^2(\theta/2) + 2\sin^2(\omega/2) + \sin^2(\theta+\omega) + \sin^2(\theta-\omega)}\,. \tag{30}$$

Given $\vec{u}$, $\vec{v}$ and ω, we now have the tools to find the axis $\hat{x}$ about which the composite rotations occur.[1] But what is the angle ψ of rotation? We can determine ψ empirically by projecting $\hat{u}$ and $\hat{v}$ into the plane of rotation (e.g. $\hat{u}_\perp = \hat{u} - (\hat{u}\cdot\hat{x})\hat{x}$). The rotation angle is then defined via

$$\psi = \text{atan2}(\sin\psi, \cos\psi) = \text{atan2}\left(\text{sign}(a)\,|\hat{u}_\perp \times \hat{v}_\perp|\,,\ \hat{u}_\perp \cdot \hat{v}_\perp\right)\,. \tag{31}$$

[1] There is one class of system where $\hat{x}$ remains ambiguous; when $\vec{u}$ and $\vec{v}$ are antiparallel, $\hat{x}_1$ and $\hat{x}_2$ are both null. If $\vec{x}$ is supplied externally, ω rotates it around the shared uv axis, but $\vec{x}$ cannot be determined from ω alone.

It is best to use atan2 because it is more precise for angles near 0, $\pi/2$ and π. Note that we have to inject the *sign* of a into $\sin(\psi)$, because when $a < 0$, the RH rotation becomes larger than π, so we must instead uses a *negative* RH rotation.

I have tested an implementation of this algorithm and it works quite well (it is both length and angle preserving). I have additionally validated that rotating once about $\hat{x}$ gives the same result as the two-step, composite rotation.

II. BOOSTING BETWEEN FRAMES

Since SO(3) is a sub-group of the Lorentz group, it is not surprising that there is a boost analog of the Rodrigues formula. A four momentum $\boldsymbol{p}$ (with energy p^0 and 3-momentum $\vec{p}$) can be boosted by some speed $\vec{\beta}$ by splitting $\vec{p}$ into its longitudinal p_L and transverse momentum $\vec{p}_T$ relative to $\hat{\beta}$ (see Eqs. 1 and 2). Unlike the Rodrigues rotation formula, we do not need to find a second perpendicular axis, because the "rotation" is always between the longitudinal momentum and energy/time. This allows us to use the standard definition of the Lorentz boost matrix, but with a coordinate system defined by $\hat{\beta}$; writing p^0 and p_L as scalars, but keeping the transverse momentum $\vec{p}_T$ as a vector

$$\boldsymbol{p}' = \begin{pmatrix} \gamma & 0 & \beta\gamma \\ 0 & \mathbb{1} & 0 \\ \beta\gamma & 0 & \gamma \end{pmatrix} \begin{pmatrix} p^0 \\ \vec{p}_T \\ p_L \end{pmatrix} = \begin{pmatrix} \gamma p^0 + \beta\gamma\, p_L \\ \vec{p}_T \\ \beta\gamma\, p^0 + \gamma p_L \end{pmatrix} \tag{32}$$

$$= \begin{pmatrix} p^0 \\ \vec{p}_T \\ p_L \end{pmatrix} + \begin{pmatrix} (\gamma-1)p^0 + \beta\gamma\, p_L \\ \vec{0} \\ \beta\gamma\, p^0 + (\gamma-1)p_L \end{pmatrix} . \tag{33}$$

Hence, adding the correct four-momenta implements the desired boost;

$$\boldsymbol{p}' = \boldsymbol{p} + \boldsymbol{b}\,, \tag{34}$$

$$b^0 = (\gamma-1)p^0 + \beta\gamma\, p_L\,, \tag{35}$$

$$\vec{b} = \left(\beta\gamma\, p^0 + (\gamma-1)p_L\right)\hat{\beta}\,. \tag{36}$$

There are intrinsic floating point limitations to this boost scheme. When an ultra-relativistic particle $\boldsymbol{p}$ is boosted back to its CM frame, machine ϵ relative rounding errors when adding $\boldsymbol{b}$ to $\boldsymbol{p}$ can cause relative errors of $\mathcal{O}(\gamma\,\epsilon)$ in $p^0_{\rm cm}$ (and absolute errors at the same order in $\vec{p}_{\rm cm}$, which should always be a null vector in the CM frame).

Appendix: The apology

I see three steps in answering any question scientifically:

1. Write down a model which describes the system with sufficient accuracy.

2. Try and fail ad naseum until the Eureka moment allows you to *describe* the solution.

3. Find the solution.

The difference between step 2 and 3 is subtle, but important. A *description* of the solution is a complete statement of *where* the solution can be found, its GPS coordinates — it is a differential equation, an unevaluated integral, the polynomial whose roots we will find. Step 3 is the vehicle that takes us there — an algorithm, a clever change of variable, an analytic continuation. It is math, and frequently the heavy kind.

In my opinion, competence and creativity in step 1 and 2 are the mark of a good scientist. This requires a vast array of knowledge and experience (e.g. you have to know what an eigenvalue is, and have previously used them to solve simple problems, before you can use them to describe the solution for some novel problem). But once we reach step 3 (calculate the eigenvalue), we should beg, borrow and steal. Of course, someone has to blaze the original trail when a new technique is found, and for that they earn their reverence in the pages of a textbook. But since they bled to build that trail, we should have the common decency to use it. Conversely, each of us should forge our own trail for the step 3 in which *we* are the experts, and then tell the world.

But we cannot be experts in the huge library of step 3's. Furthermore, since step 3's are often quite general, they lend themselves to automation. Computers are still not very good at answering *declarative* statements like "what is the volume of a sphere" (and here I don't mean using a search engine or a neural net to find a solution published by a human). However, Mathematica will swiftly provide the symbolic solution to a very *imperative* statement

$$V_{\text{sphere}} = \int_0^R \mathrm{d}r \int_0^\pi \mathrm{d}\theta \int_0^{2\pi} \mathrm{d}\phi \, r^2 \sin(\theta) = \frac{4}{3}\pi \, R^3 \, .$$

So when I find myself with a system of equations which I can ask the computer to solve (nearly instantaneously), I use the computer — that's what computers are for.