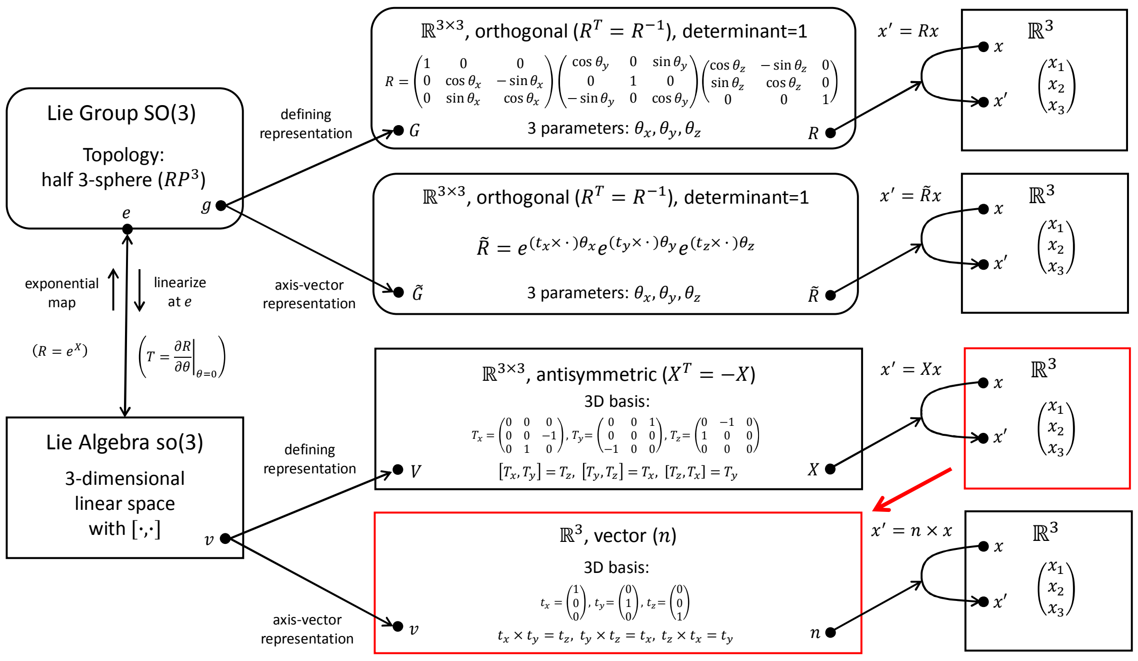
3.23 SO(3): Axis-Vector Representation; Cross Product


Lie Group SO(3)
Topology: half 3-sphere (RP^3)
e
g
defining representation
axis-vector representation
$\mathbb{R}^{3\times 3}$, orthogonal ($R^T = R^{-1}$), determinant=1
3 parameters: $\theta_x, \theta_y, \theta_z$
$\tilde{R} = e^{(t_x\times\cdot)\theta_x} e^{(t_y\times\cdot)\theta_y} e^{(t_z\times\cdot)\theta_z}$
$x' = Rx$
$x' = \tilde{R}x$
$\mathbb{R}^3$
exponential map
linearize at e
$(R = e^X)$
$\left(T = \frac{\partial R}{\partial \theta}\Big|_{\theta=0}\right)$
Lie Algebra so(3)
3-dimensional linear space with $[\cdot,\cdot]$
$\mathbb{R}^{3\times 3}$, antisymmetric ($X^T = -X$)
3D basis:
$[T_x, T_y] = T_z$, $[T_y, T_z] = T_x$, $[T_z, T_x] = T_y$
$x' = Xx$
$\mathbb{R}^3$, vector (n)
$t_x \times t_y = t_z$, $t_y \times t_z = t_x$, $t_z \times t_x = t_y$
$x' = n \times x$

We know that the Lie algebra of SO(3) is a 3-dimensional vector space. Therefore, we can represent its elements not only by the usual 3×3 matrices but also by simple 3-component column vectors. Now, the Lie-algebra elements do not only act *on* vectors, they *are* vectors (red arrow in the diagram). But how do the Lie-algebra vectors act on the representation-space vectors?

To represent a Lie-algebra matrix by a column vector, we expand the matrix X into a linear combination of the basis generators, $X = n_1 T_x + n_2 T_y + n_3 T_z$, and put the three coefficients into a column vector $n = (n_1, n_2, n_3)^T$. We thus "unpacked" the matrix into a vector. These vectors are now the new elements of the Lie algebra. Having a Lie algebra with lean vectors instead of bloated matrices is great, but now we need a vector-on-vector operation that produces the same result as the original matrix-vector multiplication $x' = Xx$. Amazingly enough, the good old *cross product* fits the bill: $x' = n \times x$. More explicitly, we can write

$$\begin{pmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} n_2x_3 - n_3x_2 \\ n_3x_1 - n_1x_3 \\ n_1x_2 - n_2x_1 \end{pmatrix}.$$

But we are not done yet, we also have the Lie-bracket operation, which in the case of matrix elements is the commutator $[X, Y] = XY - YX$. Now, we have to find the corresponding operation for our new vectors. Surprisingly, the cross product does this job too: $a \times b = c$ corresponds to $[X, Y] = Z$, where a, b, c are the vectors corresponding to the matrices X, Y, Z, respectively. More explicitly, we can write

$$\left[\begin{pmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & -b_3 & b_2 \\ b_3 & 0 & -b_1 \\ -b_2 & b_1 & 0 \end{pmatrix}\right] = \begin{pmatrix} 0 & a_2b_1 - a_1b_2 & a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 & 0 & a_3b_2 - a_2b_3 \\ a_1b_3 - a_3b_1 & a_2b_3 - a_3b_2 & 0 \end{pmatrix}.$$

What is the meaning of this new representation? An element of a Lie algebra corresponds to an infinitesimal transformation, in this case, an infinitesimal rotation. What information do we need to specify such a rotation? All we need is the axis of rotation, which is perfectly well described by a vector; there is no need for a matrix! Therefore, we will call this new representation the axis-vector representation. (Note that only in 3D space can we describe a rotation by an axis. In 2D space, we rotate about a point and in 4D space, we rotate about a 2D plane.)

Why does the cross product appear? We know that if we pick an element of the Lie algebra and let it act on all points (vectors) in the representation space, it produces a vector field. The flow of this vector field describes the transformations generated by the Lie-algebra element. Now, let's pick an axis vector n and let's take the cross product of n with all possible vectors x. What do we get? For points x on the axis, the product vectors are zero; for points x away from the axis, the product vectors point in the direction orthogonal to the plane spanned by the axis and the point; as the points x get farther away from the axis, the product vectors get larger. In conclusion, the vector field, $n \times x$, exactly describes rotations about the axis n! The operator $n \times \cdot$, where the dot is a place holder for the acted-upon vector, is the generator of rotation about the axis n.

Why does the cross product also come up for the Lie bracket? We know that the Lie bracket describes how small transformations in the Lie group fail to commute. In the case of 3D rotations, this means that if we "commute" two small rotations (e.g., we make a small rotation about the x axis, follow it by one about the y axis, follow it by the reverse about the x axis, and finally follow it by the reverse about the y axis), we end up with a small rotation in the orthogonal direction (e.g., about the z axis). The cross product is tailor-made for this! Taking the cross product of two vectors gives us a vector in the orthogonal direction.

How do we get from our new Lie-algebra representation to the corresponding Lie-group representation? We exponentiate the generators. For example, basis generator t_z produces the transformation $\tilde{R} = \exp[(t_z \times \cdot)\theta_z]$, which describes a rotation by θ_z about the z axis. But how are we to interpret the incomplete cross product in the exponent? We expand the exponential function into a power series $\tilde{R} = \exp[(t_z \times \cdot)\theta_z] = I + (t_z \times \cdot)\theta_z + \frac{1}{2}(t_z \times (t_z \times \cdot))\theta_z^2 + \cdots$ and then apply the series to the vector we want to transform $x' = \tilde{R}x = x + (t_z \times x)\theta_z + \frac{1}{2}(t_z \times (t_z \times x))\theta_z^2 + \cdots$. Now, all the cross products have two operands and make sense! This interpretation is completely analogous to what we did before (but for matrix generators we don't need the awkward dot notation): $R = \exp(T_z\theta_z)$ expands to $R = I + T_z\theta_z + \frac{1}{2}T_z^2\theta_z^2 + \cdots$ and when applied to a vector becomes $x' = Rx = x + (T_z x)\theta_z + \frac{1}{2}(T_z^2 x)\theta_z^2 + \cdots$.

Note how we were led from the dot product to the cross product. We started with a 3D vector space. We asked: "what (non-reflective) transformations preserve the dot product?". The SO(3) rotations. Then we asked "what generates the vector field that describes these rotations?". The cross product with the axis vector!