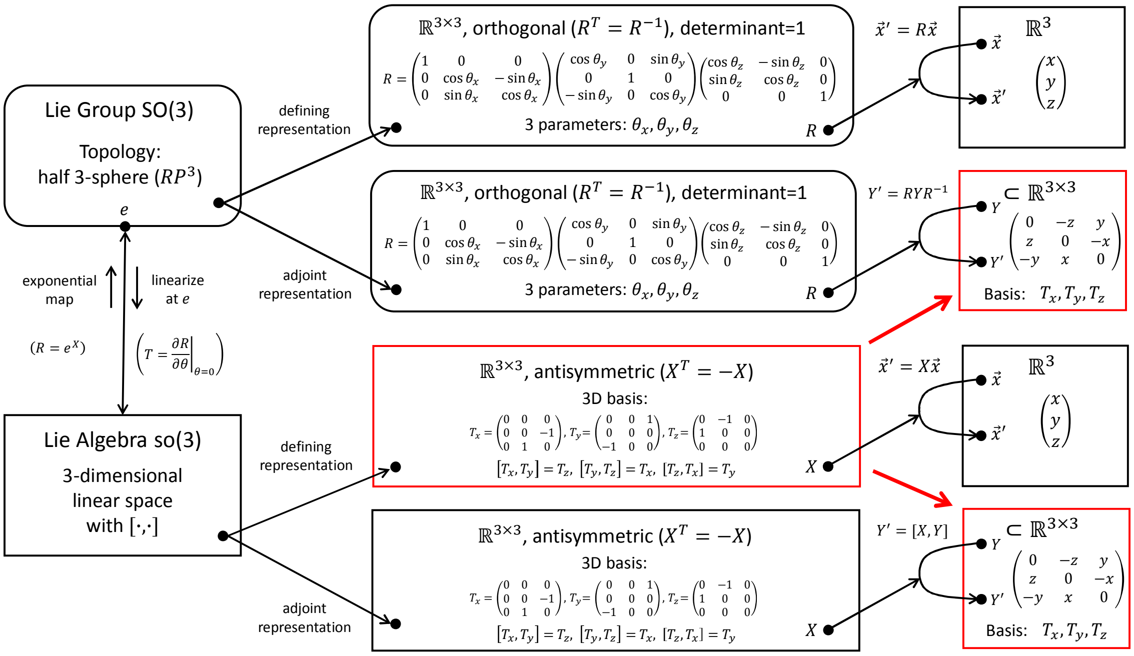
3.24 SO(3): Adjoint Representation



Let's construct the adjoint representation of SO(3) in the same way we did it for SU(2)! The upper branch of the diagram shows again the defining representation of SO(3). To construct the adjoint representation, we put the matrices of the defining Lie-algebra representation into the representation space (red arrows in the diagram) and let the representation act on these matrices instead of the 3D vectors. To avoid confusion, we call the matrices in the representation space Y, whereas the matrices in the Lie algebra are called X, as before. The basis of the Lie algebra and the basis of our new representation space are both given by the matrices T_x, T_y, and T_z. The Y matrices thus form a 3-dimensional vector space making the adjoint representation of SO(3) 3-dimensional. (Note that although the Y matrices have 3×3=9 components, they are determined by only 3 parameters.) How does the adjoint representation act on the Y matrices? From our work on SU(2) we know that the group elements act by conjugation, $Ad_R(Y) = RYR^{-1}$, and that the algebra elements act by commutation, $ad_X(Y) = [X, Y]$.

Next, we would like to "unpack" the Y matrix and write it as a conventional 3-component column vector, $\vec{x} = (x, y, z)^T$. How does the adjoint representation act on this vector? Let's take rotation about the z axis, $R_z(\theta_z)$, as an example. We pack the components of the column vector $\vec{x}$ into the matrix, $Y = xT_x + yT_y + zT_z$, and transform it like $Y' = R_z Y R_z^{-1}$:

$$\begin{pmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{pmatrix} \begin{pmatrix} \cos\theta_z & \sin\theta_z & 0 \\ -\sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{pmatrix}$$
$$= \begin{pmatrix} 0 & -z & x\sin\theta_z + y\cos\theta_z \\ z & 0 & -x\cos\theta_z + y\sin\theta_z \\ -x\sin\theta_z - y\cos\theta_z & x\cos\theta_z - y\sin\theta_z & 0 \end{pmatrix}.$$

After unpacking the matrix Y', we find the vector transformation $x' = x\cos\theta_z - y\sin\theta_z$, $y' = x\sin\theta_z + y\cos\theta_z$, and $z' = z$. But this is exactly what the defining representation does: $\vec{x}' = R_z(\theta_z)\vec{x}$!

Repeating this exercise for the remaining two rotation matrices, $R_x(\theta_x)$ and $R_y(\theta_y)$, gives analogous results. Thus, we conclude that the adjoint representation and the defining representation of SO(3) are *equivalent*.

Do we come to the same conclusion when working in the Lie algebra? Let's take the basis generator T_z as an example. We pack the components of the column vector $\vec{x}$ into the matrix, $Y = xT_x + yT_y + zT_z$, and transform it like $Y' = [T_z, Y] = [T_z, xT_x + yT_y + zT_z] = xT_y - yT_x + 0$:

$$\left[\begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{pmatrix}\right] = \begin{pmatrix} 0 & 0 & x \\ 0 & 0 & y \\ -x & -y & 0 \end{pmatrix}.$$

After unpacking the matrix Y', we find the vector transformation $x' = -y$, $y' = x$, and $z' = 0$. Again, this is exactly what the defining representation does: $\vec{x}' = T_z\vec{x}$! We thus come to the same conclusion when working in the Lie group or the Lie algebra, but the calculations are easier in the second case!

Why are the defining and adjoint representations of SO(3) equivalent? From our work on su(2) we know that the basis generators of the unpacked adjoint representation are made up of the structure constants: $[\tilde{T}_i]_{kj} = c_{ijk}$. For so(3), the structure constants are given by the 3-dimensional Levi-Civita symbol, $c_{ijk} = \varepsilon_{ijk}$, thus we have $[\tilde{T}_i]_{kj} = \varepsilon_{ijk}$. But, as we know, the basis generators of the defining representation of so(3) are also $[T_i]_{kj} = \varepsilon_{ijk}$. Same basis generators, same representation!

It is a rather special property of SO(3) that its defining and adjoint representations are equivalent. This "coincidence" is one of the reasons why we didn't start our explorations with the more familiar group SO(3), but instead chose SU(2) to begin with. Seeing the adjoint representation of SO(3) as a first example would be rather confusing!