Supporting Information:

Physical Constraint Frameworks in Chemistry: From Traditional Descriptors to Equivariant Neural Networks

Zhenxing Zhu,[†] Sirui Zhao,[‡] Diandong Tang,[*,¶] Jingyu Guo,[†] and Lin Shen[†]

[†]*College of Chemistry, Beijing Normal University, Beijing 100875, China;*

[‡]*College of Chemistry, Beijing University Of Technology, Beijing 100124, China;*

[¶]*Department of Chemistry, University of Washington, Seattle, Washington 98195, United States;*

E-mail: tangdd@uw.edu

Communications in Computational Chemistry

Contents

Communications in Computational Chemistry

S1 Mathematical background

In this section, we initially present the expansion of the three-dimensional coordinate space to spherical harmonic basis functions, subsequently introduce the equivariance of the Wigner-D matrix, followed by presenting the definition and properties of the Clebsch-Gordan coefficients in the tensor product. Finally, the definition of real spherical harmonic functions is provided, along with the application of their corresponding properties.

S1.1 Spherical harmonic basis function expansion

The common first-order tensors in three-dimensional space, such as coordinate vectors and the atomic force, etc., their vector directions can be expanded by spherical harmonic functions. Moreover, in the case of $l \geq 1$, the expansion is one-to-one and reversible. For example, for the spherical harmonic function with $l = 1$, its definition is:

$$Y_{1,-1}(x,y,z) = \sqrt{\frac{3}{8\pi}} \cdot \frac{x-iy}{r} \quad Y_{1,0}(x,y,z) = \sqrt{\frac{3}{4\pi}} \cdot \frac{z}{r} \quad Y_{1,1}(x,y,z) = -\sqrt{\frac{3}{8\pi}} \cdot \frac{x+iy}{r} \tag{S1}$$

Where $r = \sqrt{x^2+y^2+z^2}$ is the modulus length of the vector. Obviously, the coefficients $\mathbf{Y} \in \mathbb{C}^{2l+1}$ mapped from the coordinate space (real space $\mathbb{R}^3$) must satisfy the extra conditions of $Y_{1,0} \in \mathbb{R}$, $Re(Y_{1,-1}) = -Re(Y_{1,1})$and $Im(Y_{1,-1}) = Im(Y_{1,1})$. Therefore, if the coefficients of the spherical harmonic functions satisfy these conditions, there can be an inverse mapping.

$$x = r \cdot \frac{Y_{1,-1} - Y_{1,1}}{2}\sqrt{\frac{8\pi}{3}} \qquad y = r \cdot \frac{Y_{1,-1} + Y_{1,1}}{2} \cdot i \cdot \sqrt{\frac{8\pi}{3}} \qquad z = r \cdot Y_{1,0}\sqrt{\frac{4\pi}{3}} \tag{S2}$$

However, relatively, if the coefficients do not satisfy the above-mentioned restrictive conditions, they cannot be mapped back to the vectors in the three-dimensional coordinate space. Therefore, in neural networks, the spatial expansion of real spherical harmonic functions is frequently employed, which will be elaborated upon in subsection S1.4. At present, let's temporarily delve into the complex spherical harmonic function space. While both are mathematical constructs, a vector in three-dimensional Cartesian space and the space of spherical harmonic expansion coefficients differ fundamentally in their definition and geometric nature; consequently, they do not belong to the same coordinate space.

If we rotate the vectors in three-dimensional space, for example, using the Euler angle rotation matrix (a real matrix), then the coefficients of the corresponding spherical harmonic functions will have a corresponding transformation, and this transformation can be described by the Wigner-D matrix (a complex matrix). That is to say, the coefficients of the spherical harmonic basis functions upon which the Wigner-D matrix acts correspond to the three-dimensional vectors upon which the Euler angle rotation matrix acts, which precisely conforms to the definition of equivariance in Equation 2. We will further elaborate on this in the next subsection.

S1.2 Representation of a rotation matrix

The rotation matrix in the three-dimensional space, in the $Z-Y-Z$ sequence, the rotation matrix $\mathbf{R} \in \mathbb{R}_3^3$ (Represent the matrix using the (1, 1) tensor) corresponding to the angles is:

$$\mathbf{R}(\alpha,\beta,\gamma) = \begin{pmatrix} \cos\alpha\cos\beta\cos\gamma - \sin\alpha\sin\gamma & -\cos\alpha\cos\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta \\ \sin\alpha\cos\beta\cos\gamma + \cos\alpha\sin\gamma & -\sin\alpha\cos\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta \\ -\sin\beta\cos\gamma & \sin\beta\sin\gamma & \cos\beta \end{pmatrix} \tag{S3}$$

For the coefficients of spherical harmonic functions of different l orders, their dimensions are different. Therefore, the dimension of the Wigner-D matrix is accordingly $\mathbf{D}^l \in \mathbb{C}_{(2l+1)}^{(2l+1)}$. Regarding the mathematical general form of the Wigner matrix and why it can represent the three-dimensional rotation operation (α,β,γ) under spherical harmonic function space, we can refer to Appendix S2 or Chapter 20 of the relevant textbooks.[S1] We just give an example where $l = 1$.

$$\mathbf{D}^{l=1}(\alpha,\beta,\gamma) = \begin{pmatrix} \frac{1+\cos\beta}{2}e^{-i(\alpha+\gamma)} & -\frac{\sin\beta}{\sqrt{2}}e^{-i\alpha} & \frac{1-\cos\beta}{2}e^{-i(\alpha-\gamma)} \\ \frac{\sin\beta}{\sqrt{2}}e^{-i\gamma} & \cos\beta & -\frac{\sin\beta}{\sqrt{2}}e^{i\gamma} \\ \frac{1-\cos\beta}{2}e^{i(\alpha-\gamma)} & \frac{\sin\beta}{\sqrt{2}}e^{i\alpha} & \frac{1+\cos\beta}{2}e^{i(\alpha+\gamma)} \end{pmatrix} \tag{S4}$$

The equivariance of the rotation operation can now be expressed as the following relationship. It was introduced in subsection S1.1 that the mapping from the three-dimensional coordinate space to the spherical harmonic function space is invertible, as indicated by arrows ① ② and ⑤ ⑥. For a given rotation operation, it is represented as the Euler angle matrix multiplication in the coordinate space, as depicted by arrow ④. And it is represented as the Wigner-D matrix multiplication in the spherical harmonic function space, as shown by arrow ③.

$$\begin{array}{ccc} \mathbf{v} = (x,y,z) & \underset{\text{Inverse mapping to three-dimensional coordinate space ②(Eq S2)}}{\overset{\text{Mapped to the spherical harmonic function space ①(Eq S1)}}{\rightleftarrows}} & \mathbf{Y} = (Y_{-l}, Y_{-l+1}\cdots Y_l) \\ \Big\downarrow \mathbf{R}(\alpha,\beta,\gamma)\mathbf{v}^T \ ④(\text{Eq S3}) & & \mathbf{D}^l(\alpha,\beta,\gamma)\mathbf{Y}^T \ ③(\text{Eq S4}) \Big\downarrow \\ \mathbf{v}' = (x',y',z') & \underset{\text{Inverse mapping to three-dimensional coordinate space ⑥(Eq S2)}}{\overset{\text{Mapped to the spherical harmonic function space ⑤(EqS1)}}{\rightleftarrows}} & \mathbf{Y}' = (Y'_{-l}, Y'_{-l+1}\cdots Y'_l) \end{array}$$

The calculation of spherical harmonic functions coefficient and that of Wigner-D matrix have been implemented in many libraries and can be called directly.

S1.3 Definition and properties of Clebsch-Gordan coefficients

Given that we have two vectors, v_1 and v_2, of distinct dimensions within the spherical harmonic function space, in principle, many different operations can be carried out to define a new vector, $v_3 = v_1 \diamond v_2$. Nevertheless, if the two vectors, v_1 and v_2, are rotated by the Wigner-D matrix, it cannot be guaranteed that for any operation $\diamond$, the new vector v_3 transforms accordingly. A clear and illustrative example is the Hadamard (element-wise) product, which fails to preserve rotational equivariance despite its common use in neural networks.

Consider two vectors in the $l = 1$ representation space:

$$\mathbf{v}_1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \quad \mathbf{v}_2 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

Define the operation $\diamond$ as the Hadamard product:

$$\mathbf{v}_3 = \mathbf{v}_1 \diamond \mathbf{v}_2 = \mathbf{v}_1 \odot \mathbf{v}_2 = \begin{pmatrix} 1 \cdot 1 \\ 1 \cdot 0 \\ 0 \cdot 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

Now apply a rotation by $\pi/2$ around the x-axis, represented by the Wigner-D matrix $D^1(\pi/2, 0, 0)$. The rotated vectors are:

$$\mathbf{v}_1' = D^1(\pi/2, 0, 0)\, \mathbf{v}_1, \quad \mathbf{v}_2' = D^1(\pi/2, 0, 0)\, \mathbf{v}_2, \quad D^1(\pi/2, 0, 0) = \begin{pmatrix} -i & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & i \end{pmatrix}$$

Using the known form of the D^1 matrix, we find:

$$\mathbf{v}_1' \approx \begin{pmatrix} -i \\ 1 \\ 0 \end{pmatrix}, \quad \mathbf{v}_2' \approx \begin{pmatrix} -i \\ 0 \\ i \end{pmatrix}$$

Their Hadamard product is:

$$\mathbf{v}_3' = \mathbf{v}_1' \odot \mathbf{v}_2' \approx \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}$$

In contrast, rotating the original output gives:

$$D^1(\pi/2, 0, 0)\mathbf{v}_3 = D^1(\pi/2, 0, 0) \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} -i \\ 0 \\ 0 \end{pmatrix}$$

Clearly,

$$\mathbf{v}_1' \odot \mathbf{v}_2' \neq D^1(\pi/2, 0, 0)\, (\mathbf{v}_1 \odot \mathbf{v}_2)$$

which violates rotational equivariance. Therefore, we need to define a kind of tensor operation that satisfies rotational equivariance in the space of spherical harmonics like the Clebsch-Gordan tensor product rather than relying on arbitrary, symmetry-ignorant operations like the Hadamard product.

S1.3.1 The composition of angular momentum in quantum mechanics

In quantum mechanics, when contemplating physical processes entailing the coupling of multiple angular momenta, such as the coupling of the orbital angular momenta and spin angular momenta of multiple electrons within an atom, the CG coefficients are employed for operations. Supposing we have two angular momenta, $\vec{J}_1$ and $\vec{J}_2$, which respectively possess angular quantum numbers j_1, j_2 and magnetic quantum numbers m_1, m_2. Through coupling, the total angular momentum $\vec{J} = \vec{J}_1 + \vec{J}_2$ is attained, whose angular quantum number is j and magnetic quantum number is m.

Definition S1 (composition of angular momentum)**.** *Then the state $|j, m\rangle$ of the total angular momentum can be represented as a linear combination of the states $|j_1, m_1\rangle|j_2, m_2\rangle$ of the component angular momenta.*

$$|j, m\rangle = \sum_{m_1, m_2} \langle j_1, m_1; j_2, m_2 | j, m\rangle |j_1, m_1\rangle |j_2, m_2\rangle$$

Here, the Clebsch-Gordan coefficient, namely $\langle j_1, m_1; j_2, m_2 | j, m\rangle$, determines the weights in which the angular momentum states of the components combine to form the total angular momentum state. The total angular momentum can be expressed as a combination of the partial angular momenta, and the specific definition of the combination coefficients will be proved in Appendix S3. And symbol $\otimes$ is frequently omitted $|j_1, m_1\rangle \otimes |j_2, m_2\rangle \to |j_1, m_1\rangle|j_2, m_2\rangle$, which represents the tensor outer product.

Obviously, we expect that the composition of angular momentum satisfies the equivariance Equation 2 of rotation $R(\alpha, \beta, \gamma)$, that is to say, we expect the following equation to hold.

$$R(\alpha, \beta, \gamma)|j, m\rangle = \sum_{m_1, m_2} \langle j_1, m_1; j_2, m_2 | j, m\rangle (R(\alpha, \beta, \gamma)|j_1, m_1\rangle) \otimes (R(\alpha, \beta, \gamma)|j_2, m_2\rangle)$$

From a physics perspective, the reason why this equation holds is obvious: the total angular momentum after rotational is equivalent to first rotating all the sub-angular momentum and then coupling them. Note that the form of the action of $R(\alpha, \beta, \gamma)$ is inconsistent. Rewriting the operator that performs the rotation $R(\alpha, \beta, \gamma)$ on both sides of the equation with respect to states $|j, m\rangle$ as the $\sum_m D^{(l)}_m(\alpha, \beta, \gamma)$ operator in the corresponding space, we get:

$$\sum_{m'} D^{(l=j)m}_{m'} |j, m'\rangle = \sum_{m_1, m_2} \langle j_1, m_1; j_2, m_2 | j, m\rangle (\sum_{m'_1} D^{(l=j_1)m_1}_{m'_1} |j_1, m'_1\rangle) \otimes (\sum_{m'_2} D^{(l=j_2)m_2}_{m'_2} |j_2, m'_2\rangle) \tag{S5}$$

Then, using Definition S1, we can rewrite the left-hand side to obtain:

$$\sum_{m'} D^{(l=j)m}_{m'} |j, m'\rangle = \sum_{m'} D^{(l=j)m}_{m'} \sum_{m_1, m_2} \langle j_1, m'_1; j_2, m'_2 | j, m'\rangle (|j_1, m'_1\rangle \otimes |j_2, m'_2\rangle) \tag{S6}$$

Now, considering equations S5 and S6, we see that they actually give two different expressions for the same state. Therefore, their coefficients, expanded using a direct product basis $|j_1, m'_1\rangle \otimes |j_2, m'_2\rangle$, must be equal. Comparing the coefficients, we get:

Theorem S1. *The fundamental relationship between rotation matrices and Clebsch-Gordan coefficients is*

$$\sum_{m'} D_{m'}^{(l=j)m}(R)\langle j_1, m_1'; j_2, m_2'|j, m'\rangle = \sum_{m_1,m_2} \langle j_1, m_1; j_2, m_2|j, m\rangle D_{m_1'}^{(l=j_1)m_1}(R) D_{m_2'}^{(l=j_2)m_2}(R) \tag{S7}$$

The above equation represents the rotational equivariance of the tensor product of the Clebsch-Gordan coefficients in the angular momentum space, where the angular momentum takes the value of j as a half-integer and m has $2j+1$ values. On the other hand, we note that for the spherical harmonics, l is a natural number and m has $2l+1$ values. That is, l of the spherical harmonics can be regarded as a special case of j of the angular momentum. Therefore, the above equation holds similarly, and this will be discussed specifically in the next section.

S1.3.2 Tensor composition of the spherical harmonic function space

For the input sub-angular momenta j_1 and j_2, the CG coefficient is valid only when $|j_1 - j_2| < j < j_1 + j_2$; otherwise, the CG coefficient must return a value of 0, resulting in the final synthesized total angular momentum j being a zero vector.

Therefore, in the space of spherical harmonics, the first-order tensor $\mathbf{Y} = (Y_{-l}, Y_{-l+1} \cdots Y_l)^T$ that we previously defined has $2l+1$ values for a given l value. Expressed in terms of the contravariant tensor (column vector), the tensor corresponding to $Y^{(l_{in})}$ is written as $Y^{(l_{in})m_{in}}$. Thus, analogously to angular momentum, we could define

Definition S2. *A tensor operation in a spherical harmonic function space that satisfies equivariance is*

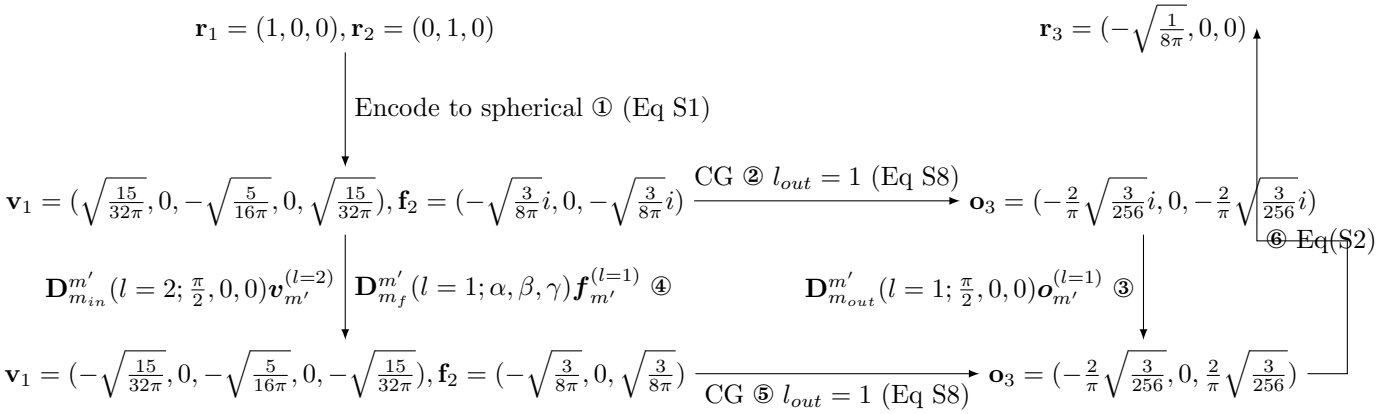
$$\boldsymbol{Y}^{(l_{out})m_{out}} = \sum_{m_f, m_{in}} C_{m_{in}, m_f}^{m_{out}} \boldsymbol{Y}^{(l_f)m_f} \otimes \boldsymbol{Y}^{(l_{in})m_{in}} \tag{S8}$$

Here, the subscript m indicates the $m-th$ component corresponding to l. That is to say, two first-order tensors $Y^{(l_f)} \in \mathbb{C}^{2l_f+1}, Y^{l_{(in)}} \in \mathbb{C}^{2l_{in}+1}$ are acted upon by the CG coefficient matrix $C \in \mathbb{R}^{(2l_{out}+1)}_{(2l_{in}+1)\times(2l_f+1)}$ to obtain the output tensor $Y^{(l_{out})} \in \mathbb{C}^{2l_{out}+1}$. Among them, the dimension of the output tensor l_{out} can be arbitrarily selected within $|l_{in} - l_f| < l_{out} < l_{in} + l_f$. So now, the Equation S5 and S6 now naturally becomes the rotational equivariance of the tensor product of the Clebsch-Gordan coefficients in the spherical harmonic function space.

$$\sum_{m'} \mathbf{D}_{m_{out}}^{m'}(l = l_{out}; \alpha, \beta, \gamma)\boldsymbol{Y}_{m'}^{(l_{out})} = \sum_{m_f, m_{in}, m', m''} C_{m_{out}}^{m_{in}, m_f} (\mathbf{D}_{m_f}^{m''}(l = l_f; \alpha, \beta, \gamma)\boldsymbol{Y}_{m''}^{(l_f)}) \otimes (\mathbf{D}_{m_{in}}^{m'}(l = l_{in}; \alpha, \beta, \gamma)\boldsymbol{Y}_{m'}^{(l_{in})}) \tag{S9}$$

In fact, since a large number of l_{out} can be composed by l_{in} and l_f, generally we choose l_{out} artificially, such as the following example:

Using the table from S1 about CG coefficients, and after defining the rotation matrix $\mathbf{R}(\pi/2, 0, 0)$, we present a process example as follows to verify the equivariance:



- Firstly, as indicated by arrow ①, we arbitrarily define two vectors r_1, r_2within the coordinate space and subsequently encode them as first-order tensors in the spherical harmonic function space. Here, v_1 is defined as $l = 2$, and f_2 is defined as $l = 1$.

- Secondly, in accordance with arrow ②, two tensors of distinct orders l can be combined to constitute a new tensor o_3 using CG coefficients.

- Thirdly, the old vectorv_1, f_2 acted on by the Wigner-D matrix and subsequently undergoes the composition of the CG coefficient vector, namely arrow ④ ⑤, is equivalent to first carrying out the composition of the CG coefficient and then being acted on by the Wigner-D matrix, namely arrow ② ③. This is the equivariance represented by equation S9. It should be noted that the Wigner-D matrix of v_1 here is distinct from the other two f_2, o_3 .

- Fourthly, arrow ①② indicates that the vector in the old three-dimensional coordinate space is initially mapped and then rotated by the Wigner-D matrix. In fact, this process can also be modified to perform the Euler angle matrix rotation first and subsequently the mapping. This is the equivariance discussed earlier in S1.2.

- Fifthly, the arrows ①②③⑥ (or ①④⑤⑥) signify that the old vector has transformed into a new vector. Although r_3 herein appears to be in the same direction as r_2 rotated by $\pi/2$ rad ($r_3 \sim \mathbf{R}(\pi/2, 0, 0)r_2$). However, actually under the influence of more complex continuous CG coefficients (and the introduction of new convolutions), r_3 can indeed point in any direction. That is to say, by adjusting certain coefficients that satisfy the equivariance, the input of the coordinates r_1, r_2 can lead to any output r_3 (such as the atomic force vector).

- Sixthly, it should be noted that if v_1 and f_2 are eventually synthesized into o_3 through a very complex process that fulfills equivariance, then o_3 does not necessarily satisfy the constraint conditions of Equation S2 and thus may not be mapped back to the real three-dimensional coordinate space. If we aim to predict real vectors, this is not what we desire. The real spherical harmonic function space can address this issue and will be discussed in subsection S1.4 later.

S1.4 The properties of the space of real spherical harmonic functions

Employing complex spherical harmonic functions to encode vectors in three-dimensional space gives rise to additional issues. Specifically, when synthesizing new vectors by operating with CG coefficients on the encoded vectors, if l_1+l_2+l is odd, it will result in the inability of the new vector to be mapped back to the three-dimensional coordinate space. Taking $l = 1$ as an illustration, the spherical harmonic functions that can correspond to the vectors in the three-dimensional coordinate space should fulfill $Y_{1,0} \in \mathbb{R}$, $Re(Y_{1,-1}) = -Re(Y_{1,1})$and $Im(Y_{1,-1}) = Im(Y_{1,1})$. Nevertheless, when l_1+l_2+l is odd, what is satisfied is $Y_{1,0} \in \mathbb{C} \setminus \mathbb{R}$, $Re(Y_{1,-1}) = Re(Y_{1,1})$and $Im(Y_{1,-1}) = -Im(Y_{1,1})$. Similar circumstances also arise for other l. Hence, we need to guarantee that $l_1 + l_2 + l$ is even to accomplish the final mapping back to the three-dimensional coordinate space, which is not what we desire. On the other hand, the degree of freedom of spherical harmonic functions in the complex space is indeed higher. The three components corresponding to $l = 1$ possess six degrees of freedom, but the spherical harmonic functions mapped from the coordinate space have constraints, thus there are only three degrees of freedom. Evidently, we do not require six degrees of freedom, and therefore real spherical harmonic functions can be contemplated.

Definition S3. *Real spherical harmonic functions are linear combinations of complex spherical harmonic functions*

$$Y_l^m(\theta,\varphi) = \begin{cases} Y_{l,0}(\theta,\varphi), & m = 0 \\ \frac{1}{\sqrt{2}}(Y_{l,m}(\theta,\varphi) + (-1)^m Y_{l,-m}(\theta,\varphi)), & m > 0 \\ \frac{i}{\sqrt{2}}((-1)^m Y_{l,-m}(\theta,\varphi) - Y_{l,m}(\theta,\varphi)), & m < 0 \end{cases} \tag{S10}$$

Since the vectors mapped from the three-dimensional coordinate space to the complex spherical harmonic functions satisfy certain restrictions, this restriction can ensure that the coefficients of the real spherical harmonic functions are all real numbers after the above-mentioned linear transformation. For real spherical harmonic functions, they also have properties similar to the Wigner-D matrix of complex spherical harmonic functions. At this time, the corresponding rotation matrix is called the real spherical harmonic function Wigner-D matrix. The same is true for the CG coefficients. At this time, the CG coefficients are called real CG coefficients.

Owing to the space constraints, we merely present the conclusion herein. The detailed proof is provided in Appendix S4.

Defining in the form of tensors: The transformation matrix $T_{m'}^{(l)m}$ from the contravariant tensor complex spherical harmonic function $Y_c^{(l)m'}$ to the real spherical harmonic function $Y_r^{(l)m}$ act as $Y_r^{(l)m} = \sum_{m'=-l}^{l} T_{m'}^{(l)m} Y_c^{(l)m'}$. Then the action of the Wigner-D matrix $R(\alpha,\beta,\gamma)$ in the real spherical harmonic functions and its definition are as follows:

$$Y_r^{m'''} = \sum_{m=-l}^{l} R_m^{m'''}(\alpha,\beta,\gamma) Y_r^m \qquad , R_m^{m'''}(\alpha,\beta,\gamma) = \sum_{m''=l}^{l} \sum_{m'=l}^{l} T_{m''}^{(l)m'''} D_{m'}^{(l)m''}(\alpha,\beta,\gamma)(T^{-1})_m^{(l)m'}$$

The role and definition of the CG coefficients of the real spherical harmonics $C_{(r)m_1m_2}^{m_3}$are presented as follows:

$$Y_r^{(l_3)m_3''} = \sum_{m_1',m_2'} C_{(r)m_1'm_2'}^{m_3''} Y_r^{(l_1)m_1'} Y_r^{(l_2)m_2'} \qquad , C_{(r)m_1'm_2'}^{m_3''} = \sum_{m_1,m_2,m_3} T_{m_3}^{(l_3)m_3''} C_{m_1m_2}^{m_3} (T^{-1})_{m_1'}^{(l_1)m_1} (T^{-1})_{m_2'}^{(l_2)m_2}$$

The Wigner-D matrix of the aforementioned real spherical harmonic functions and the CG coefficients are all constructed on the basis of complex spherical harmonic functions, and thus also satisfy the corresponding equivariance. In neural networks, the corresponding operations can still be performed. Therefore, in contrast to the neural network of complex spherical harmonic functions, only the process of vector encoding in the three-dimensional coordinate space and the final decoding process exhibit significant differences, while the forms of other parts are similar. Considering that the coordinate space mapping of real spherical harmonic functions does not need to comply with the restriction of $l1 + l2 + l$ being even, in reality, the commonly utilized e3nn [S2] library and a considerable number of equivariant neural networks are constructed based on real spherical harmonic functions.

S2 The spherical harmonic basis representation of the rotation operator is the Wigner-D matrix

S2.1 The rotation operator can be expressed in terms of the angular momentum operator.

Consider the infinitesimal rotation operator $R(\alpha, 0, 0)$ about the Z-axis, where$\alpha \to \delta\theta$ infinitely small. Expanding the trigonometric function $-sin\delta$ to the first order yields $-\delta\theta$ and $cos\delta \to 1$. Hence, the coordinate transformation from Equation S3 can be expressed as $x' = x - y\delta\theta$, $y' = x\delta\theta + y$, $z' = z$

Consider the infinitesimal coordinate transformation of the wave function $\psi'(x, y, z) = R(\alpha, 0, 0)\psi(x, y, z)$, which can now be expressed to $\psi'(x, y, z) = \psi(x - y\delta\theta, x\delta\theta + y, z)$. Carry out Taylor expansion to obtain $\psi'(x, y, z) = \psi(x, y, z) + \frac{\partial\psi}{\partial x}y\delta\theta - \frac{\partial\psi}{\partial y}x\delta\theta$

Review the basic concepts of quantum mechanics. The form of the angular momentum operator $\hat{L}_z$ is: $\hat{L}_z = -i\hbar(x\frac{\partial}{\partial y} - y\frac{\partial}{\partial x})$. The above rotation operator $R(\delta\theta, 0, 0)$is similar to this form and can be rewritten to obtain $\hat{R}(\delta\theta, 0, 0) = 1 - \frac{i}{\hbar}\hat{L}_z\delta\theta$. Same for x and y axis.

Since a rotation of a limited angle can be decomposed to the sum of infinitesimal rotations (the rotation group is closed and the elements of the rotation group are associative): $R(\theta, 0, 0) = \lim_{N\to\infty} R^N(\frac{\theta}{N}, 0, 0) = R^N(\frac{\theta}{N}, 0, 0) = (1 - \frac{i}{\hbar}\hat{L}_z\frac{\theta}{N})^N$. It should be noted that the operator expressed in the form of an exponent is not necessarily identical to the exponent operation rules of numbers. However, in this case, it holds true, and we obtain $R(\theta, 0, 0) = e^{-\frac{i}{\hbar}\hat{L}_z\theta}$.

The rotations in the x and y directions are analogous as well. Hence, in accordance with the Z-Y-Z rotation convention, we have:

$$R(\alpha, \beta, \gamma) = e^{-\frac{i}{\hbar}\hat{L}_z\alpha}e^{-\frac{i}{\hbar}\hat{L}_y\beta}e^{-\frac{i}{\hbar}\hat{L}_z\gamma} \tag{S11}$$

S2.2 Act the angular momentum operator on the spherical harmonic functions

The spherical harmonic functions $Y_l^m(\hat{r})$ (where $l = 0, 1, 2, \cdots$; $m = -l, \cdots, l$) fulfill the eigenvalue equations of angular momentum, namely, $\hat{L}^2Y_l^m(\hat{n}) = l(l+1)\hbar^2Y_l^m(\hat{n})$ and $\hat{L}_zY_l^m(\hat{n}) = m\hbar Y_l^m(\hat{n})$, where $\hat{L}^2$ represents the square operator of total angular momentum and $\hat{L}_z$ stands for the operator of the z component of angular momentum.

Further considering the specific effect of the rotation operator, the rotation in the spherical harmonic function space $RY_l^m(\hat{n})$is equivalent to the reverse rotation of the spatial vector $\{Y_l^m(R^{-1}\hat{n})\}$. For any rotation operator R, $Y_l^m(R^{-1}\hat{n})$ can be expanded as $Y_l^m(R^{-1}\hat{n}) = \sum_{m'} c_{m'm}Y_l^{m'}(\hat{n})$ because $\{Y_l^m(\hat{n})\}$ constitutes a complete orthogonal basis of the eigenfunction space of the l-th order angular momentum. That is to say, for any rotation operation R, $RY_l^m(\hat{n}) = \sum_{m'} c_{m'm}Y_l^{m'}(\hat{n})$ holds. Then our goal is to find $c_{m'm}$.

Multiply both sides of the above equation by $Y_l^{m''}$ on the left simultaneously and integrate for all space Ω. By using the orthogonality of spherical harmonic functions, we obtain $\langle Y_l^{m''}(\hat{n})|R|Y_l^m(\hat{n})\rangle = c_{m''m}$. Then we expand the rotation operator R into the form of the angular momentum operator :$\langle Y_l^{m''}(\hat{n})|e^{-\frac{i}{\hbar}\hat{L}_z\alpha}e^{-\frac{i}{\hbar}\hat{L}_y\beta}e^{-\frac{i}{\hbar}\hat{L}_z\gamma}|Y_l^m(\hat{n})\rangle = c_{m''m}$.

When R is applied to the spherical harmonic function, the $\hat{L}_z$ operator in the exponent can act on the eigenfunc-

tion Y directly to obtain the eigenvalue, but $\hat{L}_y$ cannot act directly and needs to be retained: $Y_l^{*m''}(\hat{n})\hat{R}Y_l^m(\hat{n}) = e^{-im''\alpha}Y_l^{*m''}(\hat{n})e^{-i\beta\hat{L}_y}Y_l^m(\hat{n})e^{-im\gamma}$, where $Y_l^{*m''}(\hat{n})$ represents the conjugate function corresponding to $Y_l^{m''}(\hat{n})$.

So now $c_{m''n}$ can be written in a new form, we replace m'' by m:

$$c_{m'm} = e^{-\frac{i}{\hbar}m'\alpha}\langle Y_l^{m'}(\hat{n})|e^{-\frac{i}{\hbar}\hat{L}_y\beta}|Y_l^m(\hat{n})\rangle e^{-\frac{i}{\hbar}m\gamma}$$

Replace Y_l^m with the abstract state vector in Hilbert space, we get the definition of Wigner-D matrix:

$$RY_l^m(\hat{n}) = \sum_{m'} D_{m'm}Y_l^{m'}(\hat{n}) \quad \text{where} \quad D_{m'm} = e^{-\frac{i}{\hbar}m'\alpha}\langle l,m'|e^{-\frac{i}{\hbar}\hat{L}_y\beta}|l,m\rangle e^{-\frac{i}{\hbar}m\gamma} \tag{S12}$$

Sometimes we will define the Wigner-d matrix additionally:

$$D_{m'm} = e^{-\frac{i}{\hbar}m'\alpha}d_{m'm}e^{-\frac{i}{\hbar}m\gamma} \quad \text{where} \quad d_{m'm} = \langle l,m'|e^{-\frac{i}{\hbar}\hat{L}_y\beta}|l,m\rangle \tag{S13}$$

So far, we have proved that the Wigner-D matrix is the representation of the rotation matrix in the angular momentum space, but the calculation formula of the Wigner-d matrix has not been given yet.

S2.3 The calculation of Wigner-d matrix

Although the definition of the above Wigner-D operator is based on the orbital angular momentum $\hat{L}$, yet as the rotation operator in the $SO(3)$space, it is a "relatively obvious" conclusion that it is applicable to the coupled angular momentum $J = L + S$ of both orbital and spin. The strict proof can be referred to the textbooks of group theory. Here, we refer to textbook[S3] chapter 30, we directly deal with the case where its j takes the value of $1/2$, that is:

$$d_m^{(1/2)m'} = \langle 1/2,m'|e^{-\frac{i}{\hbar}\hat{S}_y\beta}|1/2,m\rangle = \langle 1/2,m'|e^{-i\frac{\beta}{2}\sigma_y}|1/2,m\rangle$$

Since the spin is $1/2$, the Pauli matrix was used to replace the operator $\hat{S}_y$ above. Then, by utilizing the properties of the Pauli matrix $\sigma_y^2 = I$, the exponential operator was expanded in a Taylor series.

$$\begin{aligned}
e^{-i\frac{\beta}{2}\sigma_y} &= \sum_{k=0}^{\infty}\frac{(-i\frac{\beta}{2}\sigma_y)^{2k}}{(2k)!} + \sum_{k=0}^{\infty}\frac{(-i\frac{\beta}{2}\sigma_y)^{2k+1}}{(2k+1)!} \\
&= \sum_{k=0}^{\infty}\frac{(-1)^k(\frac{\beta}{2})^{2k}}{(2k)!}I + \sum_{k=0}^{\infty}\frac{(-1)^k(-i)(\frac{\beta}{2})^{2k+1}}{(2k+1)!}\sigma_y \\
&= \cos\frac{\beta}{2}I - i\sin\frac{\beta}{2}\sigma_y \\
&= \begin{pmatrix} \cos\frac{\beta}{2} & -\sin\frac{\beta}{2} \\ \sin\frac{\beta}{2} & \cos\frac{\beta}{2} \end{pmatrix}
\end{aligned}$$

The third line above utilizes the definition of trigonometric functions, and the fourth line substitutes the specific forms of the Pauli matricesσ_y and the unit matrix I. Then, the rotation operator operates on the particles with upward spin

$|\alpha\rangle$ and downward spin $|\beta\rangle$ in the following:

$$d_m^{(1/2)m'}(\beta)|\alpha\rangle = \cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|\beta\rangle \quad , \quad d^{1/2}(\beta)|\beta\rangle = -\sin\frac{\theta}{2}|\alpha\rangle + \cos\frac{\theta}{2}|\beta\rangle$$

Then, in the case where $j \geq 1$, it is considered that it can invariably be decomposed into the coupling of $2j$ identical particles with spin 1/2. For any corresponding m, among these particles, there are $j+m$ in the upward spin $|\alpha\rangle$ and $j-m$ in the downward spin $|\beta\rangle$.

$$|jm\rangle = \sqrt{\frac{(j+m)!(j-m)!}{(2j)!}} \sum_P P \underbrace{|\alpha\rangle|\alpha\rangle\cdots|\alpha\rangle}_{(j+m)} \underbrace{|\beta\rangle|\beta\rangle\cdots|\beta\rangle}_{(j-m)}$$

Among them, the coefficient in the front is to ensure the normalization under the assumption of identical particles, and P is the permutation operator, which is used to ensure traversing all possibilities. Rotation of the entire system is equivalent to that of each identical particle. Hence, by directly applying the rotation operator of spin 1/2 to each particle, we obtain:

$$\begin{aligned} d_m^{(j)m'}(\beta)|jm\rangle &= \sqrt{\frac{(j+m)!(j-m)!}{(2j)!}} \times \sum_P P\left[\cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|\beta\rangle\right]^{j+m}\left[-\sin\frac{\theta}{2}|\alpha\rangle + \cos\frac{\theta}{2}|\beta\rangle\right]^{j-m} \\ &= \sqrt{\frac{(j+m)!(j-m)!}{(2j)!}} \sum_P P \sum_r \sum_s \frac{(j+m)!}{r!(j+m-r)!}\frac{(j-m)!}{s!(j-m-s)!} \\ &\times \left(\cos\frac{\beta}{2}\right)^r \left(\sin\frac{\beta}{2}\right)^{j+m-r} \left(-\sin\frac{\beta}{2}\right)^s \left(\cos\frac{\beta}{2}\right)^{j-m-s} |\alpha\rangle^r|\beta\rangle^{j+m-r}|\alpha\rangle^s|\beta\rangle^{j-m-s} \end{aligned}$$

Among them, the second equal sign uses the binomial theorem to expand $j+m$ and $j-m$, and the expansion coefficients are r and s respectively.

Let $r+s = j+m'$, and changing the summation from r to m', the above equation become:

$$\begin{aligned} d_m^{(j)m'}(\beta)|jm\rangle &= \sum_{m'}\left[\sqrt{\frac{(j+m)!(j-m)!}{(2j)!}}\sum_P P|\alpha\rangle^{j+m'}|\beta\rangle^{j-m'}\right] \\ &\times \sum_s (-1)^s \sqrt{\frac{(j+m)!(j-m)!(j+m')!(j-m')!}{(j+m-s)!(j-m'-s)!s!(m'-m+s)!}} \\ &\times \left(\cos\frac{\beta}{2}\right)^{2j-m+m'-2s}\left(\sin\frac{\beta}{2}\right)^{m-m'+2s} \end{aligned}$$

Considering that the content in the square brackets can be rewritten as $|j,m'\rangle$, thus by contrast, the Wigner-d matrix definition for l can be written as:

$$d_{m'm}^l(\beta) = \sum_{k=\max(0,m-m')}^{\min(l+m,l-m')} \frac{(-1)^k\sqrt{(l+m)!(l-m)!(l+m')!(l-m')!}}{k!(l+m-k)!(l-m'-k)!(m'-m+k)!}\left(\cos\frac{\beta}{2}\right)^{2l-m+m'-2k}\left(\sin\frac{\beta}{2}\right)^{m-m'+2k}$$

So far we have obtained the specific form of the Wigner-d matrix, and then the Wigner-D matrix is naturally derived.

S3 The total angular momentum can be represented as the composition of the partial angular momenta

S3.1 The total angular momentum and the direct product of sub-angular momenta reside in the same space.

First of all, we need to prove that the spatial dimension of the total angular momentum is the same as that of the direct product of the sub-angular momenta.

Let $\mathcal{H}_1$ be the Hilbert space spanned by the eigenstates $\{|j_1m_1\rangle\}$ of the angular momentum $\hat{\mathbf{J}}_1$, and its dimension is $(2j_1+1)$, since the range of values of m_1 extends from $-j_1$ to j_1, totaling $2j_1+1$ values. Similarly, $\mathcal{H}_2$ is the Hilbert space spanned by the eigenstates $\{|j_2m_2\rangle\}$ of the angular momentum $\hat{\mathbf{J}}_2$, and its dimension amounts to $(2j_2+1)$

When contemplating the combination of two angular momentum systems, the state space of the combined system is the tensor product space of $\mathcal{H}_1$ and $\mathcal{H}_2$, designated as $\mathcal{H}=\mathcal{H}_1\otimes\mathcal{H}_2$. Within this tensor product space, the basis vectors can be represented as $|j_1m_1\rangle\otimes|j_2m_2\rangle$, simply expressed as $|j_1m_1\rangle|j_2m_2\rangle$. The dimension of this tensor product space amounts to $(2j_1+1)(2j_2+1)$.

For the result of the total angular momentum of angular momentum coupling, we define that the range of the value of the total angular momentum quantum number j is from $|j_1-j_2|$ to j_1+j_2, that is, $j=|j_1-j_2|,|j_1-j_2|+1,\cdots,j_1+j_2$. For each determined value of j, the range of the quantum number m, which is the projection of the total angular momentum in the z direction, is from $-j$ to j, with a total of $2j+1$ values. According to the mathematical summation formula $\sum_{k=a}^{b}(2k+1)=(b+1)^2-a^2$, where $a=|j_1-j_2|$ and $b=j_1+j_2$, it follows that $\sum_{j=|j_1-j_2|}^{j_1+j_2}(2j+1)=(j_1+j_2+1)^2-|j_1-j_2|^2$. Expanding this expression, we have $(j_1+j_2+1)^2-|j_1-j_2|^2=(j_1^2+2j_1j_2+2j_1+2j_2+j_2^2+1)-(j_1^2-2j_1j_2+j_2^2)=4j_1j_2+2j_1+2j_2+1=(2j_1+1)(2j_2+1)$.

This means that the total angular momentum has the same dimension as the coupled angular momentum.

S3.2 Find out the specific form of the CG coefficient.

Since both $\{|j_1m_1\rangle|j_2m_2\rangle\}$ and $\{|jm\rangle\}$ are complete bases of the same tensor product space $\mathcal{H}=\mathcal{H}_1\otimes\mathcal{H}_2$ (because the dimensions of the spaces they span are the same and both satisfy the completeness condition), the states in one basis set can be expressed as linear combinations of the states in the other basis set. Specifically, for the eigenstate of the total angular momentum $|jm\rangle$, it can be expressed as:

$$|jm\rangle=\sum_{m_1,m_2}C^{jm}_{j_1m_1j_2m_2}|j_1m_1\rangle|j_2m_2\rangle \tag{S14}$$

Here, the coefficient $C^{jm}_{j_1m_1j_2m_2}$ is the Clebsch - Gordan coefficient $\langle j_1m_1j_2m_2|jm\rangle$. This representation is based on the basic theory of vector spaces, that is, in the same vector space (here $\mathcal{H}_1\otimes\mathcal{H}_2$), there exists a linear transformation relationship between different complete bases. Now that we have proved the existence of the linear transformation, the problem becomes to find out the specific form of the transformation coefficient, the CG coefficient $C^{jm}_{j_1m_1j_2m_2}$.

First, similar to the content in Appendix S2.3, we define the angular momentum operators and raising and lowering operators in the total angular momentum space and the sub-angular momentum space as :$\hat{\mathbf{J}}_1$ and $\hat{\mathbf{J}}_2$ are two angular momentum operators, and $\hat{\mathbf{J}} = \hat{\mathbf{J}}_1 + \hat{\mathbf{J}}_2$ is the total angular momentum operator. $\hat{\mathbf{j}}$ can represent $\hat{j_z}$ or $\hat{j^2}$. $|j_1 m_1\rangle$ is the common eigenstate of $\hat{J}_1^2$ and $\hat{J}_{1z}$, $|j_2 m_2\rangle$ is the common eigenstate of $\hat{J}_2^2$ and $\hat{J}_{2z}$, and $|jm\rangle$ is the common eigenstate of $\hat{J}^2$ and $\hat{J}_z$.

According to the basic properties of the tensor product, the action of the total angular momentum operator satisfies:

$$\hat{\mathbf{J}}|jm\rangle = (\hat{\mathbf{J}}_1 + \hat{\mathbf{J}}_2) \sum_{m_1,m_2} C^{jm}_{j_1 m_1 j_2 m_2} (|j_1 m_1\rangle \otimes |j_2 m_2\rangle) = \sum_{m_1,m_2} C^{jm}_{j_1 m_1 j_2 m_2} \hat{\mathbf{J}}_1 |j_1 m_1\rangle |j_2 m_2\rangle + |j_1 m_1\rangle \hat{\mathbf{J}}_2 |j_2 m_2\rangle \tag{S15}$$

Let the above operator $\hat{\mathbf{J}}$ be $\hat{j}_z$ acting on the corresponding eigenstate, and obtain:

$$m\hbar|jm\rangle = \sum_{m_1,m_2} C^{jm}_{j_1 m_1 j_2 m_2} (m_1 + m_2)\hbar |j_1 m_1\rangle |j_2 m_2\rangle$$

The left side of the equation can be replaced according to Equation S14:

$$m\hbar \sum_{m_1,m_2} C^{jm}_{j_1 m_1 j_2 m_2} |j_1 m_1\rangle |j_2 m_2\rangle = \sum_{m_1,m_2} C^{jm}_{j_1 m_1 j_2 m_2} (m_1 + m_2)\hbar |j_1 m_1\rangle |j_2 m_2\rangle$$

Since the basis vectors $|j_1 m_1\rangle|j_2 m_2\rangle$ are obviously linearly independent, it must satisfy $m = m_1 + m_2$; otherwise, the corresponding Clebsch-Gordan coefficient must be 0.

Secondly, we set the value of j to the maximum, that is, $j = j_1 + j_2$, and let $m = j$. Then on the right side of the equation, m_1, m_2 can only take the maximum values j_1 and j_2 to satisfy $m_1 + m_2 = m$. At this point, there is:

$$m\hbar C^{j_1+j_2, j_1+j_2}_{j_1 j_1 j_2 j_2} |j_1 m_1\rangle |j_2 m_2\rangle = C^{j_1+j_2, j_1+j_2}_{j_1 j_1 j_2 j_2} m\hbar |j_1 m_1\rangle |j_2 m_2\rangle$$

Therefore, the coefficient $C^{j_1+j_2, j_1+j_2}_{j_1 j_1 j_2 j_2} = 1$, corresponds to state $j_1 + j_2, j_1 + j_2\rangle = |j_1, j_1\rangle \otimes |j_2, j_2\rangle$

Thirdly, according to Equation S15, apply the lowering operator $j_- = j_{1-} + j_{2-}$, and also let j be the maximum value, $j = j_1 + j_2$:

$$\hbar\sqrt{j(j+1) - m(m-1)}|j_1 + j_2, m-1\rangle = \sum_{m_1,m_2} \langle j_1 m_1 j_2 m_2 | j_1 + j_2, m\rangle (\hbar\sqrt{j_1(j_1+1) - m_1(m_1-1)}|j_1, m_1 - 1\rangle |j_2 m_2\rangle + \hbar\sqrt{j_2(j_2+1) - m_2(m_2-1)}|j_1 m_1\rangle |j_2, m_2 - 1\rangle)$$

Next, we set the value of m to be the largest $m = j$, then the value of m_1, m_2 also needs to be $m_1 + m_2 = m$ (lead to $m_1 = j_1, m_2 = j_2$), so the equation now becomes:

$$\hbar\sqrt{2(j_1+j_2)}|j_1+j_2, j_1+j_2-1\rangle = \langle j_1 j_1 j_2 j_2|j_1+j_2, j_1+j_2\rangle(\hbar\sqrt{2j_1}|j_1, j_1-1\rangle|j_2 m_2\rangle + \hbar\sqrt{2j_2}|j_1 m_1\rangle|j_2, m_j-1\rangle)$$

By substituting in the CG coefficient $C^{j_1+j_2,j_1+j_2}_{j_1 j_1 j_2 j_2} = 1$ that has been computed before, we get the new CG coefficient by lowering the operator:

$$|j_1+j_2, j_1+j_2-1\rangle = \sqrt{\frac{j_1}{j_1+j_2}}|j_1, j_1-1\rangle|j_2 j_2\rangle + \sqrt{\frac{j_2}{j_1+j_2}}|j_1 j_1\rangle|j_2, j_2-1\rangle$$

From the above equation, obviously we have new CG coefficient $C^{j_1+j_2,j_1+j_2-1}_{j_1 j_1-1 j_2 j_2} = \sqrt{\frac{j_1}{j_1+j_2}}$, $C^{j_1+j_2,j_1+j_2-1}_{j_1 j_1 j_2 j_2-1} = \sqrt{\frac{j_2}{j_1+j_2}}$

Fourthly, by applying the L_- twice or more to Equation S15, we can obtain the result similarly until the $j-$th L_- is applied. Up to this point, we can calculate the CG coefficient for all cases where $j = j_1 + j_2$:

$$\langle j_1, m_1; j_2, m_2|j_1+j_2, m\rangle = \delta_{m_1+m_2,m}\sqrt{\frac{(j_1+j_2+m)!(j_1+j_2-m)!}{(j_1+m_1)!(j_1-m_1)!(j_2+m_2)!(j_2-m_2)!}}$$

Fifthly, next we deduce the Clebsch-Gordan coefficients for the situation where $j < j_1 + j_2$. Consider the expansions of angular momenta for $j = j_1 + j_2$ and $j = j_1 + j_2 - 1$:

$$|j_1+j_2, m\rangle = \sum_{m_1,m_2}\langle j_1 m_1 j_2 m_2|j_1+j_2, m\rangle|j_1 m_1\rangle|j_2 m_2\rangle$$

$$|j_1+j_2-1, m\rangle = \sum_{m_1',m_2'}\langle j_1 m_1' j_2 m_2'|j_1+j_2-1, m\rangle|j_1 m_1'\rangle|j_2 m_2'\rangle$$

Taking the inner product, and making use of orthogonality $\langle jm|j'm\rangle = \delta_{jj'}$, $\langle jm|jm'\rangle = \delta_{mm'}$ can obtain:

$$\begin{aligned}
&\langle j_1+j_2, m|j_1+j_2-1, m\rangle = 0 \\
&= \left(\sum_{m_1,m_2}\langle j_1 m_1 j_2 m_2|j_1+j_2, m\rangle^*\langle j_1 m_1|\langle j_2 m_2|\right)\left(\sum_{m_1',m_2'}\langle j_1 m_1' j_2 m_2'|j_1+j_2-1, m\rangle|j_1 m_1'\rangle|j_2 m_2'\rangle\right) \\
&= \sum_{m_1,m_2}\sum_{m_1',m_2'}\langle j_1 m_1 j_2 m_2|j_1+j_2, m\rangle^*\langle j_1 m_1' j_2 m_2'|j_1+j_2-1, m\rangle\langle j_1 m_1|j_1 m_1'\rangle\langle j_2 m_2|j_2 m_2'\rangle \\
&= \sum_{m_1,m_2}\langle j_1 m_1 j_2 m_2|j_1+j_2, m\rangle^*\langle j_1 m_1 j_2 m_2|j_1+j_2-1, m\rangle
\end{aligned}$$

For $j = j_1 + j_2 - 1$, the maximum value of m is $m = j_1 + j_2 - 1$. At this point, there are only two cases where $m = m_1 + m_2$ is satisfied: $(m_1 = j_1, m_2 = j_2 - 1)$ and $(m_1 = j_1 - 1, m_2 = j_2)$. Therefore, the summation expansion of

the above formula $\sum_{m_1 m_2}$is carried out if and only if these two situations occur.

$$
\begin{aligned}
0 &= \langle j_1 j_1 - 1 j_2 j_2 | j_1 + j_2, j_1 + j_2 - 1 \rangle^* \langle j_1 j_1 - 1 j_2 j_2 | j_1 + j_2 - 1, j_1 + j_2 - 1 \rangle \\
&+ \langle j_1 j_1 j_2 j_2 - 1 | j_1 + j_2, j_1 + j_2 - 1 \rangle^* \langle j_1 j_1 j_2 j_2 - 1 | j_1 + j_2 - 1, j_1 + j_2 - 1 \rangle \\
&= \sqrt{\frac{j_1}{j_1 + j_2}}^* \langle j_1 j_1 - 1 j_2 j_2 | j_1 + j_2 - 1, j_1 + j_2 - 1 \rangle + \sqrt{\frac{j_2}{j_1 + j_2}}^* \langle j_1 j_1 j_2 j_2 - 1 | j_1 + j_2 - 1, j_1 + j_2 - 1 \rangle \\
&= \sqrt{\frac{j_1}{j_1 + j_2}}^* C_{j_1 j_1 - 1 j_2 j_2}^{j_1 + j_2 - 1, j_1 + j_2 - 1} + \sqrt{\frac{j_2}{j_1 + j_2}}^* C_{j_1 j_1 j_2 j_2 - 1}^{j_1 + j_2 - 1, j_1 + j_2 - 1}
\end{aligned}
$$

On the other hand, since only these two cases ($m_1 = j_1, m_2 = j_2 - 1$) and ($m_1 = j_1 - 1, m_2 = j_2$) can result in $j = j_1 + j_2 - 1$ and $m = j_1 + j_2 - 1$, it is obvious that these two coefficients are normalized: $|C_{j_1 j_1 j_2 j_2 - 1}^{j_1 + j_2 - 1, j_1 + j_2 - 1}|^2 + |C_{j_1 j_1 - 1 j_2 j_2}^{j_1 + j_2 - 1, j_1 + j_2 - 1}|^2 = 1$. Then at this time, the CG coefficients can be solved. Generally, it is conventional that $C_{j_1 j_1 - 1 j_2 j_2}^{j_1 + j_2 - 1, j_1 + j_2 - 1} = \sqrt{\frac{j_2}{j_1 + j_2}}$ ($m_2 = j_2$) is positive, and $C_{j_1 j_1 j_2 j_2 - 1}^{j_1 + j_2 - 1, j_1 + j_2 - 1} = -\sqrt{\frac{j_1}{j_1 + j_2}}$ ($m_1 = j_1$) is negative.

Sixth, through the two CG coefficients we obtained above, combined with Equation S14, taking $j = j_1 + j_2 - 1$ and applying the L_z operator once or more (up to $j_1 + j_2 - 1$ times), all the CG coefficients corresponding to the m values of $j = j_1 + j_2 - 1$ can be obtained through a similar iterative method.

Seventh, cycle the above steps. The CG coefficient of the highest order m of the new j is obtained by employing the orthogonal relation, and the CG coefficient of the corresponding secondary m is acquired by utilizing the descending operator L_z. Eventually, the general formula of the CG coefficient can be derived:

$$
\begin{aligned}
\langle j_1 m_1 j_2 m_2 | j m \rangle = \delta_{m, m_1 + m_2} &\sqrt{\frac{(2j+1)(j_1 + j_2 - j)!(j + j_1 - j_2)!(j + j_2 - j_1)!}{(j_1 + j_2 + j + 1)!}} \\
&\times \sqrt{(j_1 + m_1)!(j_1 - m_1)!(j_2 + m_2)!(j_2 - m_2)!(j + m)!(j - m)!} \\
&\times \sum_z \frac{(-1)^z}{z!(j_1 + j_2 - j - z)!(j_1 - m_1 - z)!(j_2 + m_2 - z)!(j - j_2 + m_1 + z)!(j - j_1 - m_2 + z)!}
\end{aligned}
$$

Among them, the summation $\sum_z$ is performed for all integers z ensuring that the parameters within the factorial remain non-negative.

Table S1: The CG coefficient table for the composition of $l_1 = 2$ and $l_2 = 1$ to $l_{out} = 1$. The columns represent the components m_{in} of the sub-angular momentum, while the rows represent the components m_{out} of the total angular momentum.

(m_1, m_2)	(-2,-1)	(-2,0)	(-2,1)	(-1,-1)			⋯			(1,-1)		⋯				(2,1)
m_{out}=-1	0	0	$\frac{\sqrt{15}}{5}$	0	$-\frac{\sqrt{30}}{10}$	0	$\frac{\sqrt{10}}{10}$	0	0	0	0	0	0	0	0	0
m_{out}=0	0	0	0	0	0	$\frac{\sqrt{30}}{10}$	0	$-\frac{\sqrt{10}}{5}$	0	$\frac{\sqrt{30}}{10}$	0	0	0	0	0	0
m_{out}=1	0	0	0	0	0	0	0	0	$\frac{\sqrt{10}}{10}$	0	$-\frac{\sqrt{30}}{10}$	0	$\frac{\sqrt{15}}{5}$	0	0	0

Table S2: Clebsch - Gordan coefficient ($l_1 = 1$, $l_2 = 1$, $l = 2$)

	$(-1,-1)$	$(-1,0)$	$(-1,1)$	$(0,-1)$	$(0,0)$	$(0,1)$	$(1,-1)$	$(1,0)$	$(1,1)$
$m=-2$	1	0	0	0	0	0	0	0	0
$m=-1$	0	$\frac{\sqrt{2}}{2}$	0	$\frac{\sqrt{2}}{2}$	0	0	0	0	0
$m=0$	0	0	$\frac{\sqrt{6}}{6}$	0	$\frac{\sqrt{6}}{3}$	0	$\frac{\sqrt{6}}{6}$	0	0
$m=1$	0	0	0	0	0	$\frac{\sqrt{2}}{2}$	0	$\frac{\sqrt{2}}{2}$	0
$m=2$	0	0	0	0	0	0	0	0	1

S4 Real spherical harmonic function

S4.1 Real spherical harmonic function Wigner-D matrix

The conversion from the coefficients of the contravariant tensor complex spherical harmonic functions $Y_r^{(l)m}$ to those of the contravariant tensor real spherical harmonic functions $Y_c^{(l)m}$ is accomplished via the $(1,1)$-type tensor $T^m_{(l)m'}$, and the conversion relationship is expressed as:

$$Y_r^{(l)m} = \sum_{m'=-l}^{l} T_{m'}^{(l)m} Y_c^{(l)m'}$$

The elements of the transformation matrix $T_{m'}^{(l)m}$ are defined as follows:

$$T_{m'}^{(l)m} = \begin{cases} 1, & m=0, m'=0 \\ \frac{1}{\sqrt{2}}, & m>0, m'=m \\ \frac{(-1)^m}{\sqrt{2}}, & m>0, m'=-m \\ -\frac{i}{\sqrt{2}}, & m<0, m'=m \\ \frac{i(-1)^m}{\sqrt{2}}, & m<0, m'=-m \\ 0, & \text{otherwise} \end{cases} , \mathbf{T}_2 = \begin{pmatrix} -\frac{i}{\sqrt{2}} & 0 & 0 & 0 & \frac{i}{\sqrt{2}} \\ 0 & -\frac{i}{\sqrt{2}} & 0 & \frac{i}{\sqrt{2}} & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & 0 & 0 & 0 & \frac{1}{\sqrt{2}} \end{pmatrix}$$

Here we give $l = 2$ as an example T_2, where the rows are labeled as $m \in (-2, 2)$ and the columns are labeled as $m' \in (-2, 2)$. The transformation matrix is a unitary matrix. Therefore, its inverse is equal to its conjugate transpose, that is, $(U^{-1})_{ij} = (U_{ji})^*$. Then, similarly, it is possible to define the expansion of complex spherical harmonic functions using real spherical harmonic functions:

$$Y_c^{(l)m'} = \sum_{m=-l}^{l} (T^{-1})_m^{(l)m'} Y_r^{(l)m}$$

Then, the rotation under the original Wigner-D definition can expand the complex spherical harmonic functions inside into real spherical harmonic functions:

$$\sum_{m=-l}^{l} (T^{-1})_m^{(l)m''} Y_r^m = \sum_{m'=-l}^{l} D_{m'}^{(l)m''}(\alpha,\beta,\gamma) \sum_{m=-l}^{l} (T^{-1})_m^{(l)m'} Y_r^m$$

Finally, multiply both sides of the equation on the left by the matrix T, that is, perform the tensor contraction operation $\sum_{m''} T_{m''}^{m'''}$:

$$\sum_{m''=-l}^{l}\sum_{m=-l}^{l} T_{m''}^{(l)m'''}(T^{-1})_{m}^{(l)m''}Y_r^m = \sum_{m''=-l}^{l}\sum_{m'=-l}^{l}\sum_{m=-l}^{l} T_{m''}^{(l)m'''}D_{m'}^{(l)m''}(\alpha,\beta,\gamma)(T^{-1})_{m}^{(l)m'}Y_r^m$$

The contraction of the tensor on the left side of the equation can be directly cancelled as the $\delta_m^{m'''}$ because the transformation matrix is a unitary matrix. Now on the right side of the equation, a new matrix $R_m^{m'''}$ can be defined to act on Y_r^m:

$$Y_r^{m'''} = \sum_{m=-l}^{l} R_m^{m'''}(\alpha,\beta,\gamma)Y_r^m \qquad , R_m^{m'''}(\alpha,\beta,\gamma) = \sum_{m''=l}^{l}\sum_{m'=l}^{l} T_{m''}^{(l)m'''}D_{m'}^{(l)m''}(\alpha,\beta,\gamma)(T^{-1})_{m}^{(l)m'}$$

The left side of the above equation is the action of the Wigner-D of the real spherical harmonic function, and the right side is its definition.

S4.2 Real spherical harmonic function CG coefficient

The same holds true for the CG coefficients. We expand the complex spherical harmonic functions in terms of the real spherical harmonic functions.

$$\sum_{m_3'}(T^{-1})_{m_3'}^{(l_3)m_3}Y_r^{(l_3)m_3'} = \sum_{m_2,m_1,m_1',m_2'} C_{m_1m_2}^{m_3}(T^{-1})_{m_1'}^{(l_1)m_1}Y_r^{(l_1)m_1'} \otimes (T^{-1})_{m_2'}^{(l_2)m_2}Y_r^{(l_2)m_2'}$$

Similarly, multiply both sides of the equation on the left by $T_{m_3}^{(l_3)m_3''}$ for contraction to $\delta_{m_3'}^{m_3''}$.

$$Y_r^{(l_3)m_3''} = \sum_{m_2,m_1,m_1',m_2',m_3} T_{m_3}^{(l_3)m_3''}C_{m_1m_2}^{m_3}(T^{-1})_{m_1'}^{(l_1)m_1}(T^{-1})_{m_2'}^{(l_2)m_2}Y_r^{(l_1)m_1'}Y_r^{(l_2)m_2'}$$

Then, the left side of the below equation is the action of the CG of the real spherical harmonic function, and the right side is its definition.

$$Y_r^{(l_3)m_3''} = \sum_{m_1',m_2'} C_{(r)m_1'm_2'}^{m_3''}Y_r^{(l_1)m_1'}Y_r^{(l_2)m_2'} \qquad , C_{(r)m_1'm_2'}^{m_3''} = \sum_{m_1,m_2,m_3} T_{m_3}^{(l_3)m_3''}C_{m_1m_2}^{m_3}(T^{-1})_{m_1'}^{(l_1)m_1}(T^{-1})_{m_2'}^{(l_2)m_2}$$

The aforementioned real CG coefficients can invoke the pre-existing libraries and be computed and stored in advance. Particularly, for the real CG coefficients with $l1 = l2 = l = 1$, they can be expressed more succinctly by the Levi-Civita symbol: $C_{l_1=1,m_1=j,l_2=1,m_2=k}^{l=1,m=i} = \epsilon_{ijk}$

S5 The prove of the rotate equivariance of the convolution layer

Now we simply prove that the convolution layer conforms to rotational equivariance.

1. Firstly, apply the three-dimensional Euler rotation R to both sides of the equation and expand the convolution kernel:

$$R(\alpha,\beta,\gamma)\mathcal{L}^{(l_o)}_{ac,m_o}(\vec{r}_a, V^{(l_i)}) = R(\alpha,\beta,\gamma)\ [\ \sum_{m_f,m_i} C^{m_i,m_f}_{m_o} \sum_{b\in S} R^{(l_f,l_i)}_c(|r_{ab}|)\left(Y^{(l_f)}_{m_f}(\hat{r}_{ab})\otimes V^{(l_i)}_{bc,m_i}\right)\ ]$$

2. Secondly, the Euler rotation matrix acts directly on the three-dimensional vector. When the rotation matrix acts on the tensor in the spherical harmonic function space, it follows the form of the Wigner-D matrix corresponding to the l-order (See subsection S1.2) :

$$\mathcal{L}^{(l_o)}_{ac,m_o}\left(R(\alpha,\beta,\gamma)\vec{r}_a, D^{(l_i)}(\alpha,\beta,\gamma)V^{(l_i)}\right) = \sum_{m_f,m_i} C^{m_i,m_f}_{m_o} \sum_{b\in S} R^{(l_f,l_i)}_c(|r_{ab}|)\left(Y^{(l_f)}_{m_f}(R(\alpha,\beta,\gamma)\hat{r}_{ab})\otimes D^{(l_i)}(\alpha,\beta,\gamma)V^{(l_i)}_{bc,m_i}\right)$$

3. Thirdly, the rotation R of the vector $\vec{r}_{ab}$ is equivalent to the action of the Wigner-D matrix upon mapping the spherical harmonic functions. Furthermore, R being a scalar can be placed on the leftmost side of the expression without influencing the conclusion.

$$\mathcal{L}^{(l_o)}_{ac,m_o}\left(R(\alpha,\beta,\gamma)\vec{r}_a, D^{(l_i)}(\alpha,\beta,\gamma)V^{(l_i)}\right) = \sum_{b\in S} R^{(l_f,l_i)}_c(|r_{ab}|) \sum_{m_f,m_i,m',m''} C^{m_i,m_f}_{m_o}\left(D^{m''}_{m_f}(\alpha,\beta,\gamma)Y^{(l_f)}_{m''}(\hat{r}_{ab})\otimes D^{m'}_{m_i}(\alpha,\beta,\gamma)V^{(l_i)}_{bc,m}\right.$$

4. Fourth, by using the property of Equation S9, the Wigner-D matrix can be extracted. Furthermore, reposition R once more.

$$\mathcal{L}^{(l_o)}_{ac,m_o}\left(R(\alpha,\beta,\gamma)\vec{r}_a, D^{(l_i)}(\alpha,\beta,\gamma)V^{(l_i)}\right) = \sum_{m_o} D^{m_o}_{m'''} \sum_{m_f,m_i} C^{m_i,m_f}_{m_o}\left(\sum_{b\in S} R^{(l_f,l_i)}_c(|r_{ab}|)Y^{(l_f)}_{m_f}(\hat{r}_{ab})\otimes V^{(l_i)}_{bc,m_i}\right)$$

5. Fifth, and the final step, substitute the right side of the equation with the one given by Equation 14, That is, it conforms to the definition of equivariance in Equation 2:

$$\mathcal{L}^{(l_o)}_{ac,m'''}\left(R(\alpha,\beta,\gamma)\vec{r}_a, D^{(l_i)}(\alpha,\beta,\gamma)V^{(l_i)}\right) = \sum_{m_o} D^{m_o}_{m'''}\mathcal{L}^{(l_o)}_{ac,m_o}(\vec{r}_a, V^{(l_i)})$$

S6 Hands-on Implementation of Equivariant Neural Networks for Molecular Dynamics

In this section, we provide a step-by-step explanation of the equivariant neural network architecture, built with `e3nn`[S2] and `MACE`.[S4] The goal is to bridge the gap between the theoretical derivations and the practical implementation details, following the reviewer's suggestion. All code snippets are based on PyTorch and the relevant geometric deep learning libraries.

S6.1 Irreducible Representations and Feature Layout

All features in the network are expressed as direct sums of irreducible representations (irreps). For instance, a tensor with irreps specification `"4x0e + 3x1o + 2x2e + 1x3o"` corresponds to a one-dimensional array of length

$$4 + 3 \times 3 + 2 \times 5 + 1 \times 7 = 4 + 9 + 10 + 7 = 30.$$

The order of the entries must follow the sequence: first the 4 scalars (irrep $0e$, even parity), then the 9 components of the three vectors ($1o$, odd parity, $3 \times 3 = 9$), then the 10 components of the two rank-2 tensors ($2e$, 5 components each), and finally the 7 components of the single rank-3 tensor ($3o$, 7 components). Any deviation from this ordering will lead to dimension mismatches or, even worse, corrupted equivariance when multiplying by Clebsch-Gordan coefficients later on. The numbers $4, 3, 2, 1$ denote the *channel multiplicities*, analogous to the number of feature maps in conventional neural networks. However, unlike ordinary dense layers, these multiplicities can only be changed through equivariant operations (tensor products, self-interaction layers) that act on whole irrep blocks.

S6.2 Graph Construction and Data Representation

S6.2.1 Node and Edge Definitions

In our framework, each atom in a molecule is represented as a node in a graph. Let N be the number of atoms in a single molecule. Nodes are connected by edges based on spatial proximity. We define a cutoff radius R_{cut} (typically 6.0 Å); any pair of atoms (i, j) with distance $\|\mathbf{r}_i - \mathbf{r}_j\| < R_{\text{cut}}$ is considered connected. The connectivity is stored in an edge index tensor, denoted as `edge_index`, with shape $[2, E]$, where E is the total number of directed edges. The first row `edge_index[0, :]` contains the source node indices, and the second row `edge_index[1, :]` contains the target node indices. For example, consider a 5-atom system where atoms 0, 1, and 2 form a fully connected triangle, and atoms 3 and 4 form a dimer. The corresponding `edge_index` would be:

$$\texttt{edge\_index} = \begin{bmatrix} 0 & 0 & 1 & 1 & 2 & 2 & 3 & 4 \\ 1 & 2 & 0 & 2 & 0 & 1 & 4 & 3 \end{bmatrix} \tag{S16}$$

During message passing, the model iterates over this list. For node 0, the algorithm identifies neighbors by finding all columns where the first row equals 0 (indices 0 and 1 in the example above), revealing that nodes 1 and 2 are its neighbors. This neighbor list is critical; since molecular geometries change during dynamics simulations, the `edge_index` must be recomputed whenever atomic positions are updated. In practice, we use `torch_cluster.radius_graph` to efficiently generate these edges given atomic coordinates `pos` and the cutoff distance `self.cutoff`.

S6.2.2 Batch Processing in Graph Neural Networks

Unlike traditional neural networks that process data in batches of shape $[B, N, F]$ (where B is batch size, N is sequence length, and F is feature dimension), Graph Neural Networks (GNNs) typically flatten the batch and node dimensions into a single axis. A batch of B molecules, each with potentially different numbers of atoms, is concatenated into a single large graph.

The node features are stacked as:

$$[\texttt{Node}_1^{(1)}, \ldots, \texttt{Node}_{N_1}^{(1)}, \texttt{Node}_1^{(2)}, \ldots, \texttt{Node}_{N_B}^{(B)}]$$

resulting in a tensor of shape $[\sum_{b=1}^{B} N_b, F]$. To prevent information leakage between different molecules in the same batch, the `edge_index` is adjusted by adding offsets. If the first molecule has N_1 atoms, the node indices for the second molecule in the batch are shifted by N_1.

For instance, if we batch two copies of the 5-atom system described earlier, the combined `edge_index` becomes:

Listing 1: Batched edge index for two 5-atom systems

```
# Original edges for Molecule 1 (nodes 0-4):
# [[0,0,1,1,2,2,3,4], [1,2,0,2,0,1,4,3]]
# Original edges for Molecule 2 (nodes 5-9, offset by 5):
# [[5,5,6,6,7,7,8,9], [6,7,5,7,5,6,9,8]]
# Combined batched edge_index:
[[0, 0, 1, 1, 2, 2, 3, 4, 5, 5, 6, 6, 7, 7, 8, 9],
 [1, 2, 0, 2, 0, 1, 4, 3, 6, 7, 5, 7, 5, 6, 9, 8]]
```

This mechanism ensures that message passing operations remain strictly intra-molecular. While the number of edges E may vary across molecules, the `torch_geometric` library handles variable-sized graphs seamlessly via the `Batch` object, which also stores a `batch` vector indicating which molecule each node belongs to. This allows for efficient parallel computation while preserving the structural integrity of individual molecular systems.

S6.3 Input Embeddings

S6.3.1 Data Preparation

In a typical training step, the batch object contains all the raw physical quantities that describe the molecular system. These quantities are stored in Euclidean space and must be transformed into equivariant representations before entering the equivariant layers. The relevant fields are:

- `batch.pos`: atomic Cartesian coordinates, shape $[B \cdot N, 3]$;
- `batch.z`: atomic numbers (core types), shape $[B \cdot N]$;
- `batch.force`: forces acting on each atom, shape $[B \cdot N, 3]$;
- `batch.veloc`: atomic velocities, shape $[B \cdot N, 3]$;
- `batch.energy`: potential energy of the whole molecule, shape $[B, 1]$

The graph connectivity is defined by `batch.edge_index` as described in Section S6.2. The following subsections explain how each type of physical quantity is embedded into the irreps-based feature space.

S6.3.2 Edge Vectors and Distances

Given the atomic positions `pos` ($[B \cdot N, 3]$) and the edge index, we first compute the relative displacement vectors and their lengths:

```
from mace import get_edge_vectors_and_lengths
vectors, lengths = get_edge_vectors_and_lengths( pos, edge_index )
# vectors: [B*E, 3], lengths: [B*E, 1]
```

These geometric quantities are fundamental for constructing the convolution kernel in the message-passing framework (see Definition 5 in the main text).

S6.3.3 Gaussian Expansion of Edge Distances

To enhance the representational capacity of interatomic distances, we do not feed the raw scalar distance $r_{ij} = \|\mathbf{r}_i - \mathbf{r}_j\|$ directly into the neural network. Instead, we project it onto a set of Radial Basis Functions (RBFs). This transformation maps a single scalar value to a high-dimensional feature vector, allowing the subsequent Multi-Layer Perceptron (MLP) in the convolution kernel $\mathcal{F}$ to learn complex, non-linear distance-dependent interactions more effectively than a linear layer could with a raw scalar input.

We uniformly discretize the cutoff range $[0.0, R_{\text{cut}}]$ (where $R_{\text{cut}} = 6.0\,\text{Å}$) into N_b centers μ_k ($k = 1, \ldots, N_b$), where N_b corresponds to the hyperparameter `num_edge_bases`. For each center, a Gaussian kernel is defined as:

$$\phi_k(r) = \exp\left(-\frac{(r - \mu_k)^2}{2\sigma^2}\right), \tag{S17}$$

where σ is the width parameter, typically determined by the spacing between adjacent centers to ensure smooth overlap between basis functions. Given an atomic distance r_{ij}, the edge scalar feature vector $\mathbf{e}_{ij} \in \mathbb{R}^{N_b}$ is obtained by evaluating all basis functions:

$$\mathbf{e}_{ij} = [\phi_1(r_{ij}), \phi_2(r_{ij}), \ldots, \phi_{N_b}(r_{ij})]^\top. \tag{S18}$$

For example, with $N_b = 7$ and $r_{ij} = 3.0\,\text{Å}$, the components of $\mathbf{e}_{ij}$ corresponding to centers near 3.0 Åwill exhibit high activation values (e.g., ≈ 0.20), while those far from this value decay rapidly to near zero (e.g., $< 10^{-5}$). This "soft one-hot" encoding provides two key advantages: it ensures smooth gradients with respect to atomic positions, avoiding numerical instabilities associated with hard cutoffs, and it transforms the distance into a rich feature space that enables the downstream MLP to capture intricate physical potentials.

In practice, this expansion is performed using the `soft_one_hot_linspace` function from the `e3nn` library:

Listing 2: Gaussian expansion of edge distances using e3nn

```
from e3nn.math import soft_one_hot_linspace
# lengths: [B*E, 1], containing r_ij for each edge
edge_scalars = soft_one_hot_linspace(
    lengths.squeeze(-1), start=0.0, end=self.cutoff,
    number=num_edge_bases,
    basis='gaussian', ) # Output shape: [B*E, num_edge_bases]
```

The resulting $\mathbf{e}_{ij}$ are later concatenated with other edge features and passed to the MLP that generates the message weights.

S6.3.4 Spherical Harmonic Embedding of Vectors

The direction vector $\mathbf{r}_{ij}$ (or any other Cartesian vector quantity such as forces or velocities) is transformed into equivariant features via real spherical harmonics. This operation maps a 3D Cartesian vector $\mathbf{v} = [x, y, z]^\top$ into a direct sum of irreducible representations up to a maximum angular momentum $l_{\max}$. In the implementation, we use the `spherical_harmonics` function from `e3nn`, which computes the real spherical harmonics $Y_l^m(\theta, \phi)$ expressed in terms of Cartesian coordinates. For a given vector $\mathbf{v}$ with magnitude $r = \|\mathbf{v}\|$ and direction (θ, ϕ), the embedding generates components for each l from 0 to $l_{\max}$. For illustration, consider the case where $l_{\max} = 2$. The resulting feature vector consists of:

- $l = 0$ **(Scalar, even parity):** A single component proportional to 1 (often normalized by $1/\sqrt{4\pi}$). In many implementations, this is simply treated as a constant scalar channel if the input is a unit vector, or scaled by $r^0 = 1$.
- $l = 1$ **(Vector, odd parity):** Three components corresponding to the Cartesian axes themselves:

$$Y_1^{-1} \propto y, \quad Y_1^0 \propto z, \quad Y_1^1 \propto x.$$

Thus, the $l = 1$ block is essentially the original vector $[x, y, z]$

- $l = 2$ **(Rank-2 Tensor, even parity):** Five components constructed from quadratic combinations of x, y, z. Using the standard real spherical harmonic basis, these are proportional to:

$$Y_2^{-2} \propto xy, \quad Y_2^{-1} \propto yz, \quad Y_2^0 \propto 3z^2 - r^2, \quad Y_2^1 \propto xz, \quad Y_2^2 \propto x^2 - y^2.$$

Note that specific normalization factors (such as $\sqrt{15}$ or $\sqrt{5/16\pi}$) depend on the convention used by the library.

Consequently, for $l_{\max} = 3$, the edge feature becomes a direct sum of one scalar ($0e$), one vector ($1o$), one rank-2 tensor ($2e$), and one rank-3 tensor ($3o$), yielding $1+3+5+7 = 16$ components per edge. The same `spherical_harmonics` function can be applied to any other vector quantity entering the equivariant network, such as atomic forces and velocities:

```
# Example: Embedding edge vectors
edge_sh = spherical_harmonics(vectors)
# Output shape: [B*E, 16] for lmax=3
# Irreps: "1x0e + 1x1o + 1x2e + 1x3o"
# Example: Embedding atomic forces and velocities
atomforce_sh = spherical_harmonics(atomforce)   # [B*N, 16]
atomveloc_sh = spherical_harmonics(atomveloc)   # [B*N, 16]
```


This step converts ordinary Cartesian vectors into the structured irreducible blocks required by the geometric tensor operations, ensuring that subsequent layers can process them while strictly preserving rotational equivariance.

S6.3.5 Embedding of Scalar Features

Scalar quantities, such as atomic numbers z_i, are invariant under rotations and thus can initially be processed with ordinary neural network layers. A typical embedding module is:

```
self.node_core_embedding = nn.Sequential(
    nn.Embedding(num_embeddings=100, embedding_dim=32),
    nn.Linear(32, 64),
    nn.SiLU(),
    nn.Linear(64, 128),
    nn.SiLU(),
    nn.Linear(128, z_embedding_dim)
)
node_core_feat = self.node_core_embedding(z)  # [B*N, z_embedding_dim]
```


At this stage the features are pure scalars (i.e., irreps `"z_embedding_dim` $\times$ `0e"`). This means they can be safely transformed by dense layers and activation functions without breaking equivariance.

Crucial transition: Once these scalar node features participate in a tensor product with the edge spherical harmonics, they acquire higher-order irreducible components. For example, after the first MACE interaction, the node features

are typically expanded to an irreps specification such as `"16x0e + 8x1o + 4x2e + 2x3o"`. From that point onward, ordinary `Linear` layers are *no longer allowed*, and all subsequent transformations must use equivariant operations (tensor products, self-interaction layers, etc.) that act on whole irrep blocks.

S6.4 Core Equivariant Operations

S6.4.1 Tensor Products and the Self-Interaction Layer

The central operation in an equivariant message-passing block is the tensor product between the incoming node features and the edge features. Consider a node feature with irreps `"24x0e"` (i.e., 24 scalar channels, the above node features `"z_embedding_dim × 0e"`) and an edge feature with irreps `"1x0e + 1x1o + 1x2e + 1x3o"`. According to the Clebsch-Gordan rules, the full tensor product yields:

$$\texttt{"24x0e"} \otimes \texttt{"1x0e + 1x1o + 1x2e + 1x3o"} = \texttt{"24x0e + 24x1o + 24x2e + 24x3o"}.$$

If the desired output node feature is, say, `"8x1o"`, we **cannot** simply apply a dense layer to the $24 \times 3 = 72$ components of the $1o$ block, because that would mix components of the vector and destroy rotation equivariance. Instead, we use a *self-interaction layer*: a single linear weight matrix that acts on the *entire* irrep block. For the $1o$ block specifically,

$$\text{Output}_{1o}^{(c)} = \sum_{c'=1}^{24} W_{c,c'} \, \text{Input}_{1o}^{(c')},$$

where $c = 1, \ldots, 8$ indexes the output channels and $c' = 1, \ldots, 24$ indexes the input channels. The same weight $W_{c,c'}$ multiplies all three Cartesian components (x, y, z) of the c'-th input vector. This operation reduces the channel multiplicity from 24 to 8 without breaking equivariance. The same principle applies to any irrep: a self-interaction layer can compress or expand the number of channels, while the angular momentum label l and parity are preserved. This mechanism is the key to controlling the dimension of the feature space in an equivariant network.

S6.4.2 CG Interactions Between Different Feature Streams

When the model needs to combine information from two different physical quantities (e.g., forces and velocities), a direct concatenation followed by a dense layer is not allowed if we want to preserve equivariance. Instead, we use a Clebsch-Gordan tensor product. Assume the force stream has been encoded with irreps `"1x0e + 1x1o + 1x2e"` and the velocity stream with irreps `"1x0e + 1x1o"`. The Cartesian product of their irreps, respecting parity, gives all possible couplings:

$$\begin{aligned} &\text{For the } 0e \text{ component of velocity:} \quad 0e \otimes (0e, 1o, 2e) = 0e, 1o, 2e, \\ &\text{For the } 1o \text{ component of velocity:} \quad 1o \otimes (0e, 1o, 2e) = 1o,\ (0e, 1e, 2e),\ (1o, 2o, 3o). \end{aligned}$$

Thus the full output can contain irreps up to $3o$. The maximum available l in the output must satisfy $l \leq l_1 + l_2$, a constraint enforced by the Clebsch-Gordan rules. Within this limit, the channel multiplicities for each allowed irrep can

be freely chosen through self-interaction layers, giving the user control over the expressiveness and computational cost.

In practice, this interaction is implemented as a `TensorProduct` layer with instruction-based path selection:

Listing 3: Fully connected tensor product with automatic instruction generation

```
cg_energy_z = TensorProduct(
    irreps_in1=self.energy_edge_maceout_irreps,
    irreps_in2=self.z_edge_maceout_irreps,
    irreps_out=self.cg_energy_z_out_irreps,
    instructions=[
        (i1, i2, i_out, "uvw", True)
        for i1, (mul1, ir1) in enumerate(irreps_in1)
        for i2, (mul2, ir2) in enumerate(irreps_in2)
        for i_out, (mul_out, ir_out) in enumerate(irreps_out)
        if ir_out in ir1 * ir2 and ir1.p * ir2.p == ir_out.p
    ],
    shared_weights=True
)
```

The list comprehension generates an instruction for every physically allowed combination of input irreps that contributes to a desired output irrep. Each instruction specifies a contraction mode (`"uvw"`) and creates a set of trainable parameters. The fully connected version shown here maximizes the model's expressive power at the expense of higher computational cost; the user may manually restrict the instructions to enforce physically motivated interactions or to reduce memory usage.

S6.5 Output Readout and Global Pooling

After the final equivariant layers, each atom is typically described by a feature vector that has been reduced to a pure scalar (e.g., `"1x0e"` representing an atomic contribution to the potential energy). The resulting tensor is of shape $[B \cdot N, 1]$. To obtain a global molecular property, we need to aggregate over all atoms belonging to the same molecule.

When all molecules have the same number of atoms N:

```
prepare_out = final_node_feat.view(B, N)    # [B, N]
out = prepare_out.sum(dim=1, keepdim=True) # [B, 1]
```

This is mathematically safe because the features are scalars (invariant under rotations), and the summation is a linear permutation-invariant operation.

General case of variable-sized molecules: If the batch contains molecules with different atom counts, a simple `view` is impossible. We instead use a scatter-based pooling, which relies on a `batch_index` vector that assigns each of the $B \cdot N$ nodes to its parent molecule:

```
from torch_scatter import scatter
```

```
out = scatter(final_node_feat, batch_index, dim=0, reduce='sum')
# shape [B, 1]
```

For example, if the first molecule has 12 atoms, the second 17, and the third 33, `batch_index` starts with 12 zeros, followed by 17 ones, then 33 twos. The scatter operation sums the contributions within each group, correctly recovering a $[B, 1]$ output. This general scheme also applies to other global properties such as total electronic density or total energy, as long as the per-atom predictions are scalars.

S6.6 Representation of Temporal Dimensions in Data Structures

As this work is a review, we do not propose new architectures for modeling temporal dynamics. However, to illustrate how time-dependent data can be represented within the equivariant framework, we describe a standard data construction strategy. When dealing with T consecutive snapshots from molecular dynamics trajectories, the node features are typically flattened into a single dimension following the order: Batch $\rightarrow$ Time Step $\rightarrow$ Atom. Specifically, the node ordering becomes:

$$\texttt{[B1T1N1, B1T1N2, ..., B1T1N}_x\texttt{, B1T2N1, ..., B1T2N}_x\texttt{, ..., B1T}_y\texttt{N}_x\texttt{, B2T1N1, ...]}$$

In this representation, graph construction (edge index generation) and message passing are performed independently for each snapshot. This ensures that spatial interactions are computed correctly within each time frame without introducing artificial temporal correlations during the spatial convolution steps. The batch index is adjusted to group nodes by both molecule and time step, allowing for correct aggregation if needed.

For readers interested in explicitly modeling temporal dependencies, such as learning time-correlation functions or predicting future states via spatio-temporal convolutions, we refer to specialized literature.[S5] These approaches introduce additional layers on top of the spatial equivariant blocks, which is beyond the scope of the current implementation details presented here. Our focus remains on the fundamental spatial equivariant operations and their data representations.

References

(S1) Richtmyer, R. D. *Principles of Advanced Mathematical Physics*; Springer-Verlag, 1981.

(S2) Geiger, M. et al. Euclidean neural networks: e3nn. 2022.

(S3) Ka, X. *Advanced Quantum Mechanics*; Higher Education Press, 2001; pp 420–422.

(S4) Batatia, I.; Kovács, D. P.; Simm, G. N. C.; Ortner, C.; Csányi, G. MACE: Higher Order Equivariant Message Passing Neural Networks for Fast and Accurate Force Fields. 2023; `https://arxiv.org/abs/2206.07697`.

(S5) Keller, T. A. Flow Equivariant Recurrent Neural Networks. 2025; `https://arxiv.org/abs/2507.14793`.