



Group Equivariant Deep Learning

Lecture 3 - Equivariant graph neural networks

Lecture 3.4 - Group Theory | $SO(3)$ irreps (Wigner-D matrices), Clebsch-Gordan TP

Preliminaries for 3D steerable g-convs

Irreps of SO(3): Wigner-D matrices

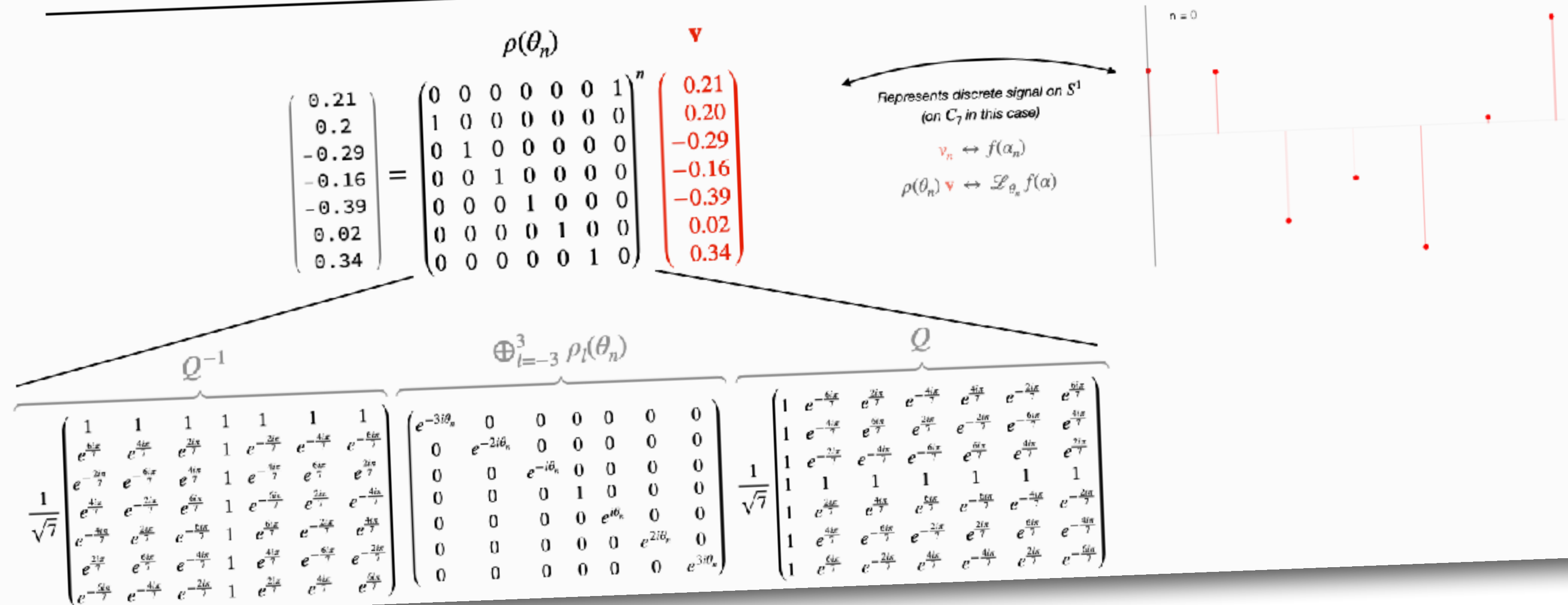
Lecture 2.3

Equivalence of group representations

A (matrix) representation is called reducible if it can be written as


$$\rho(g) = Q^{-1} (\rho_1(g) \oplus \rho_2(g)) Q = Q^{-1} \begin{pmatrix} \rho_1(g) & 0 \\ 0 & \rho_2(g) \end{pmatrix} Q$$

If the blocks $\rho_1(g), \rho_2(g)$ are not reducible they are called **irreducible representations (irreps)**



Irreps of $SO(3)$: Wigner-D matrices

Wigner-D matrices of type l are the irreducible matrix representations of $SO(3)$.
We will denote these $(2l + 1) \times (2l + 1)$ dimensional matrices with $\mathbf{D}^{(l)}(\mathbf{R})$



$$\mathbf{D}^{(l)}(\mathbf{R}) = [D_{mn}^{(l)}(\mathbf{R})]_{m,n=-l}^l$$

Wigner-D functions
form an orthogonal basis for $\mathbb{L}_2(SO(3))!!!$

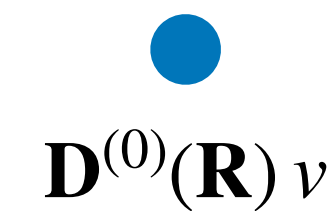


The $(2l + 1)$ dimensional vector space on which $\mathbf{D}^{(l)}(\mathbf{R})$ acts will be called a **steerable vector space of type l** and denoted with $V_l = \mathbb{R}^{2l+1}$. A vector $\mathbf{v} \in V_l$ will be called a **type- l vector**.

Irreps of $SO(3)$: Wigner-D matrices

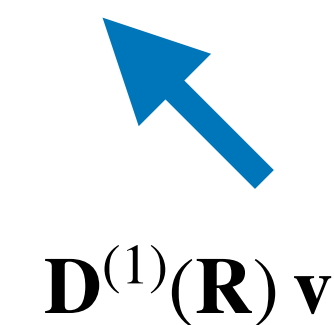
Example (type 0): A type-0 vector $v \in V_0$ is just a scalar that trivially transforms by a 1×1 dimensional “matrix”

$$\mathbf{D}^{(0)}(\mathbf{R}) v = 1 v = v$$



Example (type 1): A type-1 vector $\mathbf{v} \in V_1$ is a 3D vector (e.g. velocity, force, displacement) that transforms directly via the rotation matrix $\mathbf{R} \in SO(3)$

$$\mathbf{D}^{(1)}(\mathbf{R}) \mathbf{v} = \mathbf{R} \mathbf{v}$$



$$D_{mn}^{(l)}(\mathbf{R}_{\alpha,\beta,\gamma}) = \begin{pmatrix} \cos(\alpha)\cos(\gamma) - \sin(\alpha)\cos(\beta)\sin(\gamma) & \sin(\beta)\sin(\gamma) & \cos(\alpha)\cos(\beta)\sin(\gamma) + \sin(\alpha)\cos(\gamma) \\ \sin(\alpha)\sin(\beta) & \cos(\beta) & -\cos(\alpha)\sin(\beta) \\ \sin(\alpha)(-\cos(\beta))\cos(\gamma) - \cos(\alpha)\sin(\gamma) & \sin(\beta)\cos(\gamma) & \cos(\alpha)\cos(\beta)\cos(\gamma) - \sin(\alpha)\sin(\gamma) \end{pmatrix}$$

Wigner-D functions: complete orthogonal basis for functions on $SO(3)$

The **Wigner-D functions** $D_{mn}^{(l)} : SO(3) \rightarrow \mathbb{R}$ form a **complete orthogonal basis** for functions on $SO(3)$.

Thus any function can be represented in such an $SO(3)$ Fourier series:

$$\begin{aligned} f(\mathbf{R}) &= \sum_l \sum_{m=-l}^l \sum_{n=-l}^l \hat{f}_{mn}^{(l)} D_{mn}^l(\mathbf{R}) \\ &= \sum_l \text{tr} \left(\hat{f}^{(l)} \mathbf{D}^{(l)}(\mathbf{R}^{-1}) \right) \end{aligned}$$

General form Fourier trafo on G (Peter-Weyl)

Forward $[\mathcal{F}_G f]_l = \int_G f(g) \rho_l(g) dg$

Inverse $\mathcal{F}^{-1}[\hat{f}](g) = \sum_l d_{\rho_l} \text{tr} \left[\hat{f}(\rho_l) \rho_l(g^{-1}) \right]$

The **central columns** $D_{\cdot 0}$ is invariant to rotations \mathbf{R}_α around chosen reference axis (e.g. \mathbf{e}_x):

$$\forall_{\alpha \in [0, 2\pi)} : D_{m0}^{(l)}(\mathbf{R} \mathbf{R}_\alpha) = D_{m0}^{(l)}(\mathbf{R})$$

These $SO(2)$ -invariant functions coincide with functions on the sphere $S^2 \equiv SO(3)/SO(2)$: the **spherical harmonics** $Y : S^2 \rightarrow \mathbb{R}$!

Homogeneous space: the sphere S^2

The sphere S^2 is a homogeneous space of 3D rotations $SO(3)$

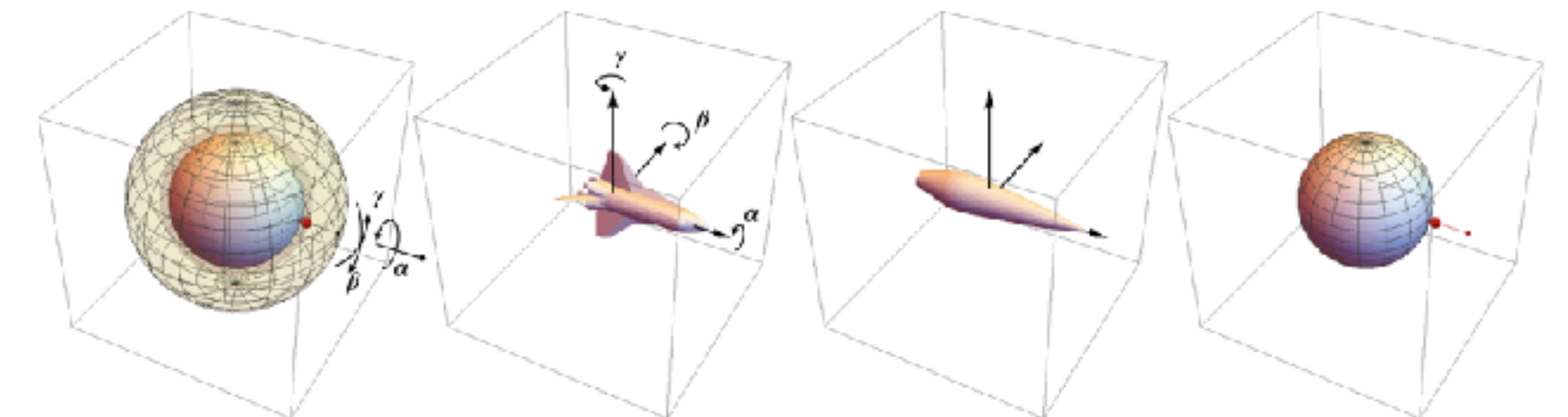
The 3D rotation group $SO(3)$

The 2-sphere as a quotient space

Representation in parameter space (XYZ-Euler angles)

Rotation by $R \in SO(3)$
 $R = R_{e_z, \gamma} R_{e_y, \theta} R_{e_x, \alpha}$

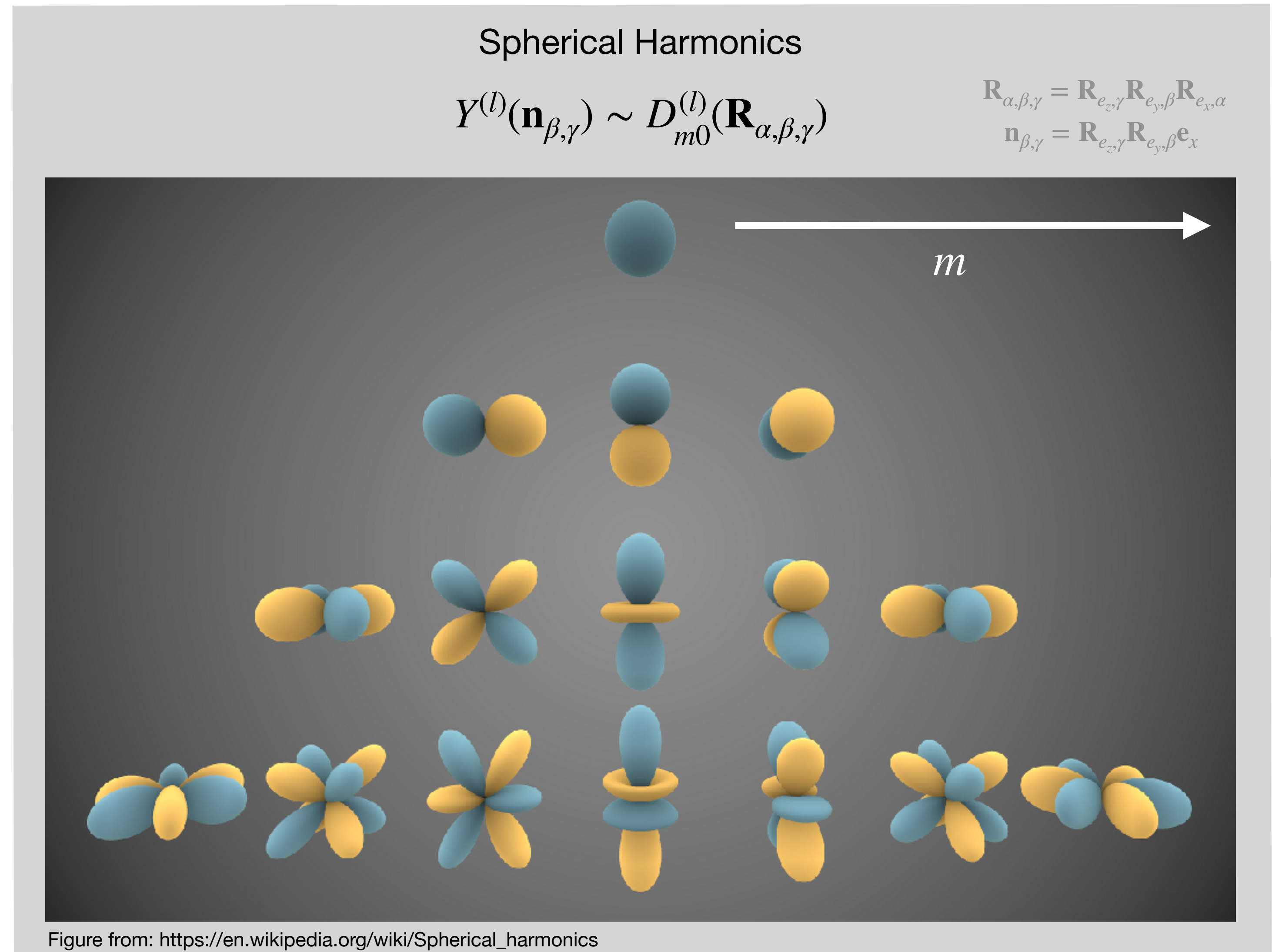
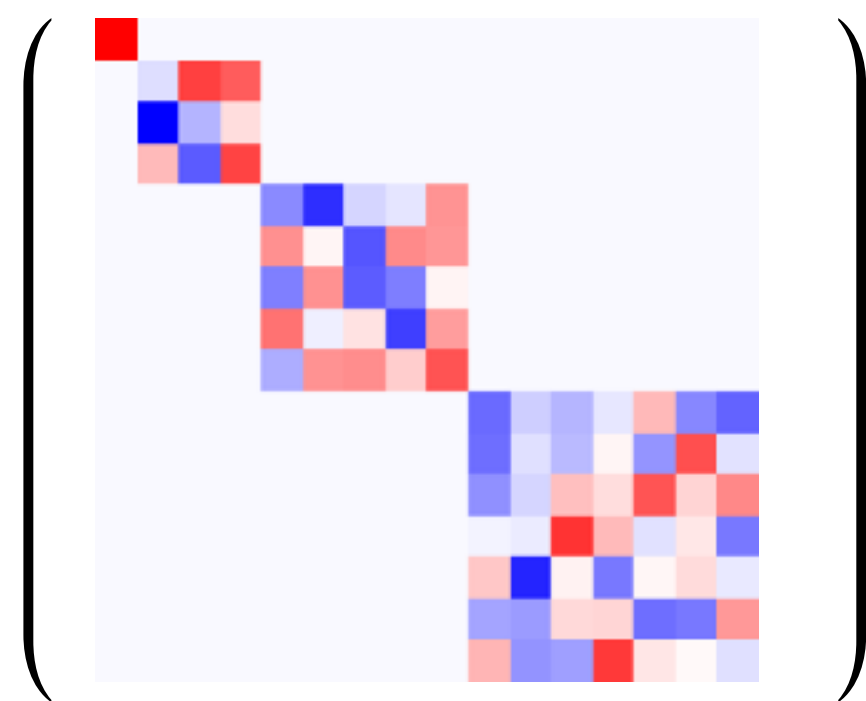
$S^2 \equiv SO(3)/SO(2)$



Spherical harmonics: complete orthogonal basis for functions on S^2

- Solutions of Laplace equation (hence “harmonics”)
- A Fourier basis on the sphere S^2
- Fourier coefficients transform via block-diagonal representations

$$f(\mathbf{Rn}) = [\mathcal{F}_{S^2}^{-1} \mathbf{D}(\mathbf{R}) \mathcal{F}_{S^2} f](\mathbf{n})$$



Spherical harmonics are $SO(3)$ steerable

Since $Y(\mathbf{n}_{\beta,\gamma}) \sim \mathbf{D}_{:0}(\mathbf{R}_{\alpha,\beta,\gamma})$ and $\mathbf{D}^{(l)}$ are (irreducible) representations ($\mathbf{D}^{(l)}(\mathbf{R}\mathbf{R}') = \mathbf{D}^{(l)}(\mathbf{R})\mathbf{D}^{(l)}(\mathbf{R}')$) it follows

$$\forall \mathbf{R} \in SO(3), \mathbf{n} \in S^2 : Y^{(l)}(\mathbf{R}\mathbf{n}) = \mathbf{D}^{(l)}(\mathbf{R}) Y^{(l)}(\mathbf{n})$$

Lecture 2.1

Steerable basis

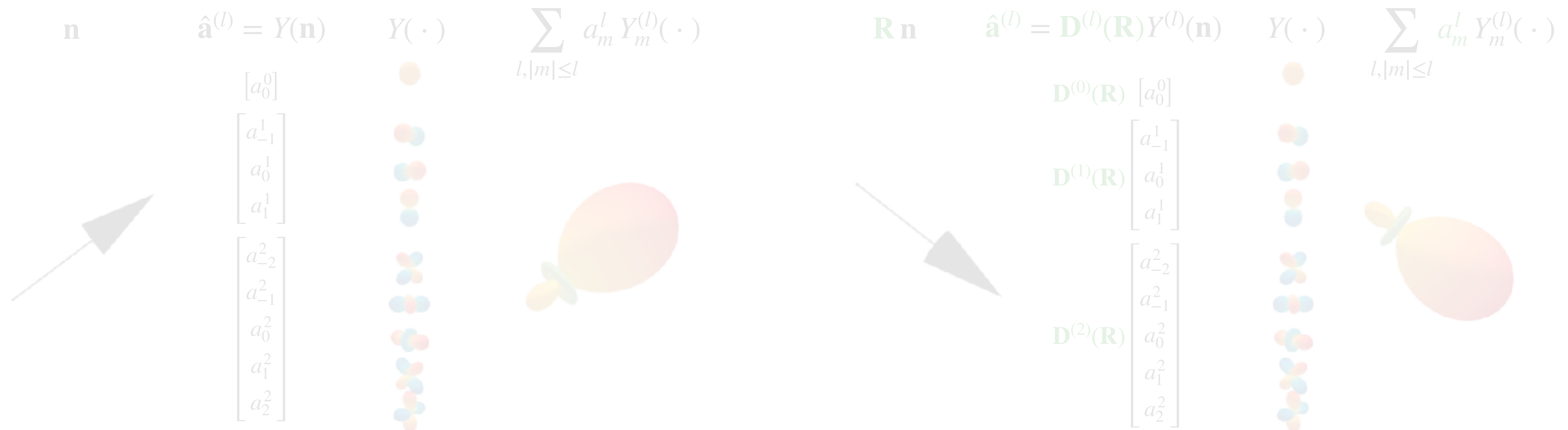
A vector $Y(x) = \begin{pmatrix} \vdots \\ Y_l(x) \\ \vdots \end{pmatrix} \in \mathbb{K}^L$ with (basis) functions $Y_l \in \mathbb{L}_2(X)$ is steerable if

$$\forall g \in G : Y(gx) = \rho(g)Y(x),$$

where gx denotes the action of G on X and $\rho(g) \in \mathbb{K}^{L \times L}$ is a representation of G .

I.e., we can transform all basis functions simply by taking a linear combination of the original basis functions.

3



Spherical harmonics are $SO(3)$ steerable

Since $Y(\mathbf{n}_{\beta,\gamma}) \sim \mathbf{D}_{:0}(\mathbf{R}_{\alpha,\beta,\gamma})$ and $\mathbf{D}^{(l)}$ are (irreducible) representations ($\mathbf{D}^{(l)}(\mathbf{R}\mathbf{R}') = \mathbf{D}^{(l)}(\mathbf{R})\mathbf{D}^{(l)}(\mathbf{R}')$) it follows

Equivariant attribute embedding

$\forall \mathbf{R} \in SO(3), \mathbf{n} \in S^2 : Y^{(l)}(\mathbf{R}\mathbf{n}) = \mathbf{D}^{(l)}(\mathbf{R}) Y^{(l)}(\mathbf{n})$ Functional representation of a geometric quantity

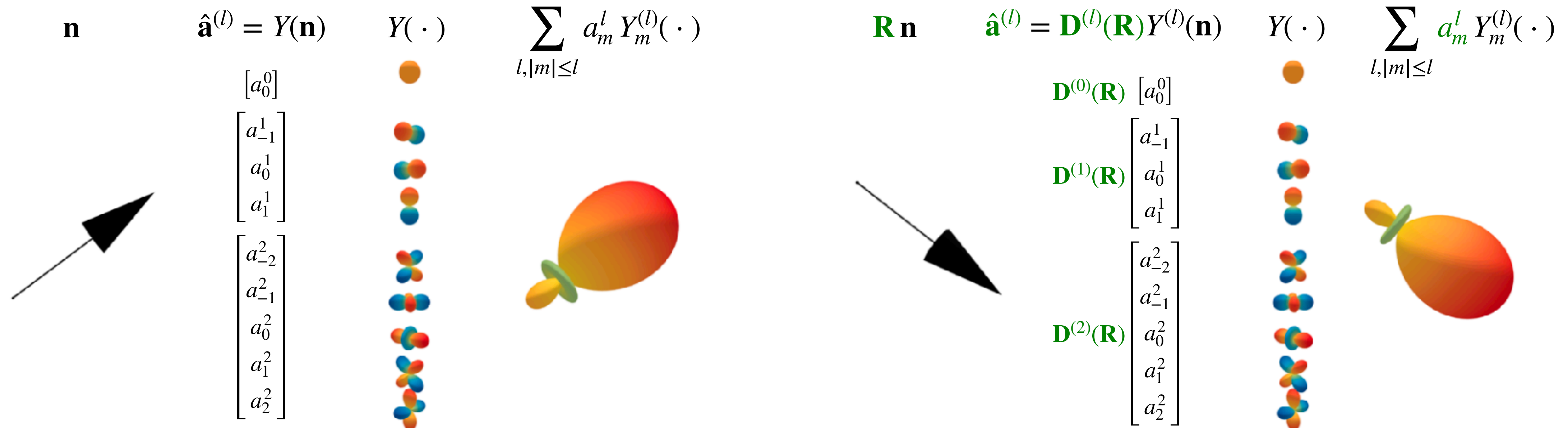
Lecture 2.1

Steerable basis

A vector $Y(x) = \begin{pmatrix} \vdots \\ Y_l(x) \\ \vdots \end{pmatrix} \in \mathbb{K}^L$ with (basis) functions $Y_l \in L_2(X)$ is steerable if

$$\forall g \in G : Y(g.x) = \rho(g)Y(x),$$

where $g.x$ denotes the action of G on X and $\rho(g) \in \mathbb{K}^{L \times L}$ is a representation of G .
i.e., we can transform all basis functions simply by taking a linear combination of the original basis functions.



Clebsch-Gordan Tensor Product

Consider the tensor product of two steerable vectors $\mathbf{a} \in V_{l_a}$ and $\mathbf{b} \in V_{l_b}$

$$\mathbf{a} \otimes \mathbf{b} = \mathbf{a} \mathbf{b}^T = \begin{pmatrix} a_1 b_1 & a_1 b_2 & \dots \\ a_2 b_1 & a_2 b_2 & \\ \vdots & & \ddots \end{pmatrix}$$

The tensor product rotates via

$$(\mathbf{a} \mapsto \mathbf{D}^{(l_a)}(\mathbf{R})\mathbf{a}, \quad \mathbf{b} \mapsto \mathbf{D}^{(l_b)}(\mathbf{R})\mathbf{b})$$

$$\mathbf{a} \otimes \mathbf{b} \mapsto \mathbf{D}^{(l_a)}(\mathbf{R}) \mathbf{a} \otimes \mathbf{b} \mathbf{D}^{(l_b)}(\mathbf{R})^T$$

Vectorized tensor products are steerable via:

$$(\text{using identity } \text{vec}(\mathbf{A}\mathbf{X}\mathbf{B}) = (\mathbf{B}^T \otimes \mathbf{A})\text{vec}(\mathbf{X}))$$

$$\text{vec}(\mathbf{a} \otimes \mathbf{b}) \mapsto (\mathbf{D}^{(l_b)}(\mathbf{R}^{-1}) \otimes \mathbf{D}^{(l_a)}(\mathbf{R})) \text{vec}(\mathbf{a} \otimes \mathbf{b})$$

Its representation is block-diagonalizable:

$$\mathbf{Q}^{-1} \begin{pmatrix} \mathbf{D}^{l_1}(\mathbf{R}) & 0 & 0 & 0 \\ 0 & \mathbf{D}^{l_2}(\mathbf{R}) & 0 & 0 \\ 0 & 0 & \mathbf{D}^{l_3}(\mathbf{R}) & 0 \\ 0 & 0 & 0 & \ddots \end{pmatrix} \mathbf{Q} \text{vec}(\mathbf{a} \otimes \mathbf{b})$$

Clebsch-Gordan Tensor Product

Consider the tensor product of two steerable vectors $\mathbf{a} \in V_{l_a}$ and $\mathbf{b} \in V_{l_b}$

$$\mathbf{a} \otimes \mathbf{b} = \mathbf{a} \mathbf{b}^T = \begin{pmatrix} a_1 b_1 & a_1 b_2 & \dots \\ a_2 b_1 & a_2 b_2 & \\ \vdots & & \ddots \end{pmatrix}$$

The tensor product rotates via

$$(\mathbf{a} \mapsto \mathbf{D}^{(l_a)}(\mathbf{R})\mathbf{a}, \quad \mathbf{b} \mapsto \mathbf{D}^{(l_b)}(\mathbf{R})\mathbf{b})$$

$$\mathbf{a} \otimes \mathbf{b} \mapsto \mathbf{D}^{(l_a)}(\mathbf{R}) \mathbf{a} \otimes \mathbf{b} \mathbf{D}^{(l_b)}(\mathbf{R})^T$$

Vectorized tensor products are steerable via:

$$(\text{using identity } \text{vec}(\mathbf{A}\mathbf{X}\mathbf{B}) = (\mathbf{B}^T \otimes \mathbf{A})\text{vec}(\mathbf{X}))$$

$$\text{vec}(\mathbf{a} \otimes \mathbf{b}) \mapsto (\mathbf{D}^{(l_b)}(\mathbf{R}^{-1}) \otimes \mathbf{D}^{(l_a)}(\mathbf{R})) \text{vec}(\mathbf{a} \otimes \mathbf{b})$$

Its representation is block-diagonalizable:

Clebsch-Gordan tensor product
includes change of basis!

$$\begin{pmatrix} \mathbf{D}^{l_1}(\mathbf{R}) & 0 & 0 & 0 \\ 0 & \mathbf{D}^{l_2}(\mathbf{R}) & 0 & 0 \\ 0 & 0 & \mathbf{D}^{l_3}(\mathbf{R}) & 0 \\ 0 & 0 & 0 & \ddots \end{pmatrix} \text{vec}(\mathbf{a} \otimes_{cg} \mathbf{b})$$

Clebsch-Gordan Tensor Product

Consider two steerable vectors $\mathbf{v}^{(l_1)} = \begin{pmatrix} \vdots \\ v_{m_1}^{(l_1)} \\ \vdots \end{pmatrix} \in V_{l_1}$ and $\mathbf{v}^{(l_2)} = \begin{pmatrix} \vdots \\ v_{m_2}^{(l_2)} \\ \vdots \end{pmatrix} \in V_{l_2}$ of type l_1 and l_2 respectively

The **Clebsch-Gordan tensor product** is defined as

$$\underbrace{(\mathbf{v}^{(l_1)} \otimes_{cg}^w \mathbf{v}^{(l_2)})}_m^{(l)} = \sum_{m_1=-l_1}^{l_1} \sum_{m_2=-l_2}^{l_2} C_{(l_1, m_1)(l_2, m_2)}^{(l, m)} v_{m_1}^{(l_1)} v_{m_2}^{(l_2)}$$

A steerable output vector of type l

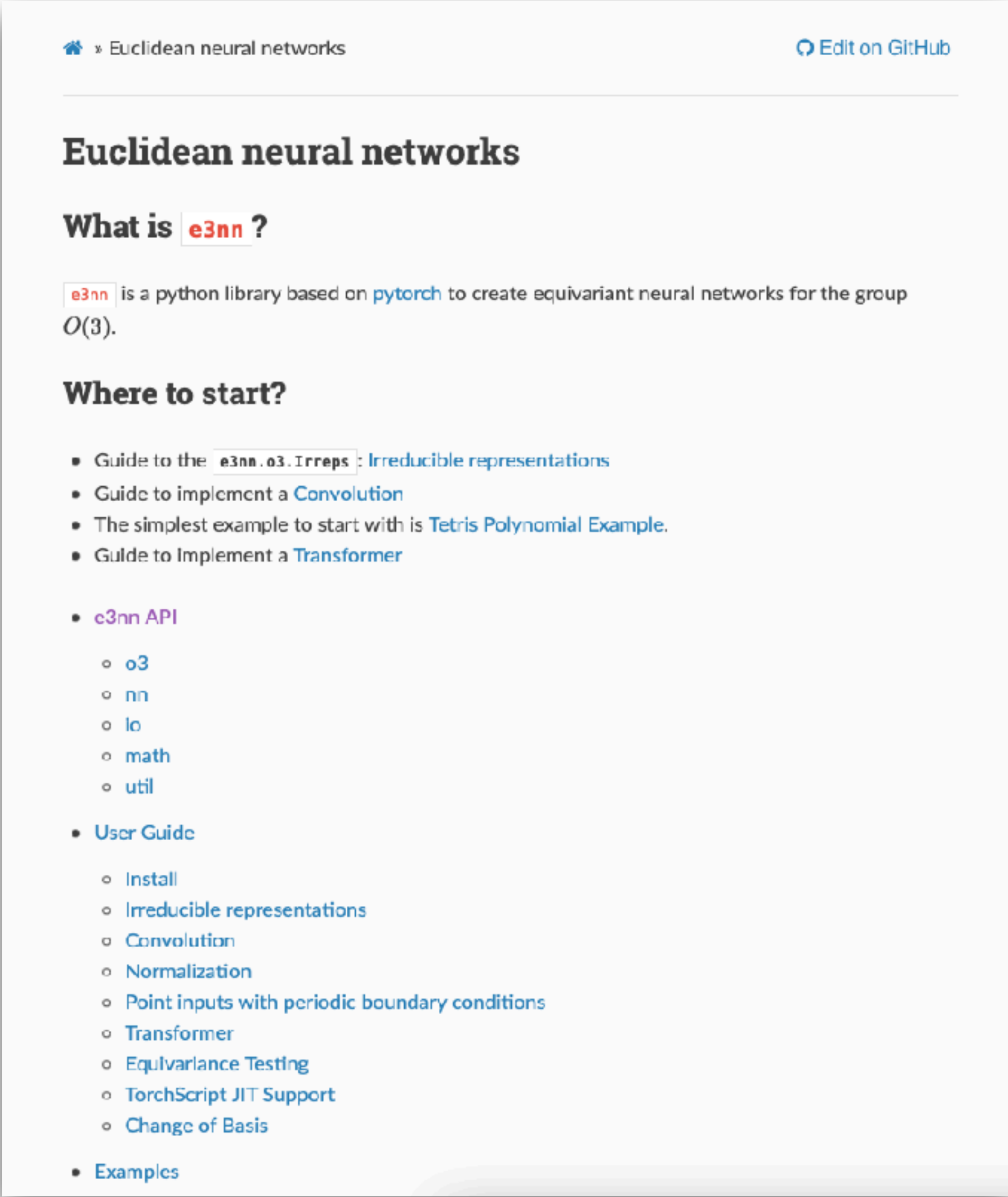
- The Clebsch-Gordan tensor product is highly sparse (many $C_{(l_1, m_1)(l_2, m_2)}^{(l, m)} = 0$)
- In particular for all $l < |l_1 - l_2|$ and $l > l_1 + l_2$ the CG coefficients are zero.

Familiar Examples:

- Product of two scalars $(l_1 = 0, l_2 = 0, l = 0)$
- The scalar-vector product $(l_1 = 0, l_2 = 1, l = 1)$
- The dot product $(l_1 = 1, l_2 = 1, l = 0)$
- The cross product $(l_1 = 1, l_2 = 1, l = 1)$

Clebsch-Gordan Tensor Product with the e3nn library

<https://docs.e3nn.org/en/stable/>



The screenshot shows the top portion of the e3nn documentation website. At the top, there is a navigation bar with a home icon and the text 'Euclidean neural networks' on the left, and a 'Edit on GitHub' link on the right. Below this is a main heading 'Euclidean neural networks'. Underneath, a section titled 'What is e3nn?' explains that e3nn is a Python library based on PyTorch for creating equivariant neural networks for the group $O(3)$. A 'Where to start?' section follows, listing several guides: 'Guide to the e3nn.o3.Irreps: Irreducible representations', 'Guide to implement a Convolution', 'The simplest example to start with is Tetris Polynomial Example', and 'Guide to Implement a Transformer'. Below these are two main categories: 'e3nn API' and 'User Guide'. The 'e3nn API' category includes sub-items: 'o3', 'nn', 'lo', 'math', and 'util'. The 'User Guide' category includes: 'Install', 'Irreducible representations', 'Convolution', 'Normalization', 'Point inputs with periodic boundary conditions', 'Transformer', 'Equivariance Testing', 'TorchScript JIT Support', and 'Change of Basis'. Finally, there is an 'Examples' category at the bottom.

Euclidean neural networks

[Edit on GitHub](#)

Euclidean neural networks

What is e3nn?

e3nn is a python library based on [pytorch](#) to create equivariant neural networks for the group $O(3)$.

Where to start?

- Guide to the `e3nn.o3.Irreps`: [Irreducible representations](#)
- Guide to implement a [Convolution](#)
- The simplest example to start with is [Tetris Polynomial Example](#).
- Guide to Implement a [Transformer](#)

- **e3nn API**
 - [o3](#)
 - [nn](#)
 - [lo](#)
 - [math](#)
 - [util](#)
- **User Guide**
 - [Install](#)
 - [Irreducible representations](#)
 - [Convolution](#)
 - [Normalization](#)
 - [Point inputs with periodic boundary conditions](#)
 - [Transformer](#)
 - [Equivariance Testing](#)
 - [TorchScript JIT Support](#)
 - [Change of Basis](#)
- **Examples**

Lecture 2.7

Harmonic networks

counter-clockwise rotation of an image $F(r, \phi)$ about the origin by an angle θ is $F(r, \phi + \theta) = F(r, \phi - \theta)$. As a shorthand we denote $F^\theta := F(r, \phi + \theta)$. It is a well-known result [23, 7] (proof in Supplementary Material) that

$$[W_m * F^\theta] = e^{im\theta} [W_m * F^0], \quad (5)$$

where we have written W_m in place of $W_m(r, \phi; R, \theta)$ for brevity. We see that the response to a θ -rotated image F^θ with a circular harmonic of order m is equivalent to the cross-correlation of the unrotated image F^0 with the harmonic, followed by multiplication by $e^{im\theta}$. While the rotation is done in feature space, multiplication by $e^{im\theta}$ is performed in feature space,

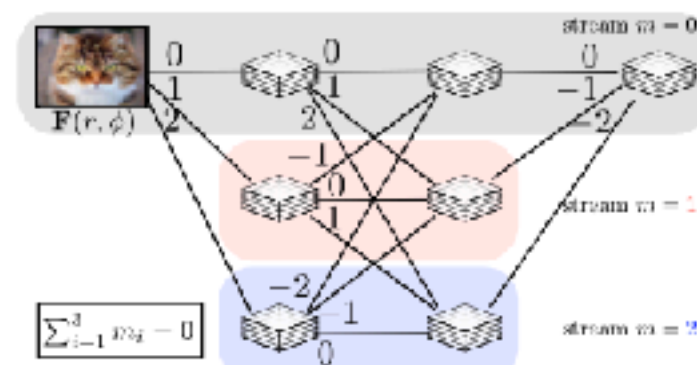
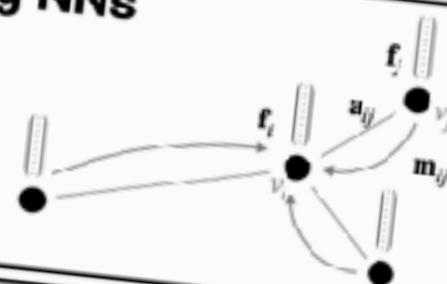


Figure 4. An example of a 2 hidden layer H-Net with $m=0$ output, input-output left-to-right. Each horizontal stream represents a series of feature maps (circles) of constant rotation order. The edges represent cross-correlations and are numbered with the rotation order of the cross-correlation. The sum of rotation orders along any path of length M is $M=0$, to maintain

Lecture 3.2

Linear vs non-linear (group) convolutions

Message passing NNs



Compute messages:

$$m_{ij} = \phi_m(f_i, f_j, a_{ij})$$

Aggregate and update:

$$f'_i = \phi_f\left(f_i, \sum_{j \in \mathcal{N}(i)} m_{ij}\right)$$

Classic point convolutions

(Lecture 1.7: regular g-convs on homogeneous spaces)

$$m_{ij} = W(\|x_j - x_i\|) f_j$$

$$m_{ij} = W(g_i^{-1} g_j) f_j$$

Steerable G-CNNs

(Lecture 2: steerable g-convs)

$$m_{ij} = W_{\hat{a}_{ij}}(\|x_j - x_i\|) \hat{f}_j$$

$$:= \hat{f}_j \otimes_{cg}^{W(\|x_j - x_i\|)} \hat{a}_{ij}$$

Invariant Message Passing NNs

(Lecture 3)

$$m_{ij} = \text{MLP}(f_i, f_j, \|x_j - x_i\|)$$

Equivariant (Steerable) Message Passing NNs

(Lecture 3)

$$\hat{m}_{ij} = \widehat{\text{MLP}}(\hat{f}_i, \hat{f}_j, x_j - x_i)$$

With steerable MLP:

$$\widehat{\text{MLP}}_{\hat{a}_{ij}}(\hat{f}_i, \hat{f}_j, \|x_j - x_i\|) := \sigma(W_{\hat{a}_{ij}}^{(n)}(\dots(\sigma(W_{\hat{a}_{ij}}^{(1)} \hat{h}_i))))$$

28

Steerable G-CNNs as Clebsch-Gordan networks

$$\hat{\mathcal{H}}(\hat{f})(\mathbf{x}) = \int_{\mathbb{R}^d} \sum_l \sum_{j=-l}^l \hat{w}_j(\|\mathbf{x}' - \mathbf{x}\|) Y_j(\alpha_{\mathbf{x}' - \mathbf{x}}) \hat{f}_l(\mathbf{x}') d\mathbf{x}'$$

Lecture 3.3

Conditional linear layers



Linear layer (matrix-vector multiplication)

$$f' = W f$$

Conditional linear layer

$$f' = W(x_b - x_a) f$$

$$f' = f \otimes_{cg}^{W} Y_j(x_b - x_a)$$

$$f \mapsto f' = W(x_b - x_a) f \mapsto f'' = \sigma(f')$$

linear layer

activation

Conditional linear layers are (partially evaluated) tensor products!!!

$$f' = f \otimes^W Y_j(x_b - x_a)$$

\Leftrightarrow

with a basis $Y(x) = \begin{pmatrix} Y_0(x) \\ Y_1(x) \\ Y_2(x) \end{pmatrix}$

- Basis (coordinate embedding) functions $Y_j: \mathbb{R}^3 \rightarrow \mathbb{R}$
- Matrix-valued weights W_j with elements w_{jl}^j

$$W(x_b - x_a) = \sum_j W_j Y_j(x_b - x_a)$$



35