Geometric Deep Learning

How to *"learn"* vector fields on manifolds

Francesco Ballerin

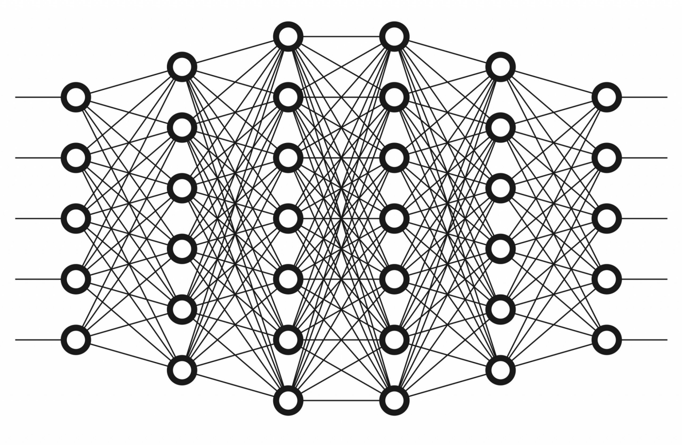
Joint work with **Erlend Grong** and **Nello Blaser**

Analysis & PDE seminar 2025

UNIVERSITY OF BERGEN

What is Geometric Deep Learning?

Neural Networks



Neural Networks

A feedforward neural network computes outputs $\hat{y}$ from inputs x through multiple layers of neurons. Each layer applies a linear transformation followed by a nonlinear activation function. Let $W^{(l)}$ be the weight matrix, $b^{(l)}$ the bias vector, and σ a non-linear activation function.

The computation at each layer is:

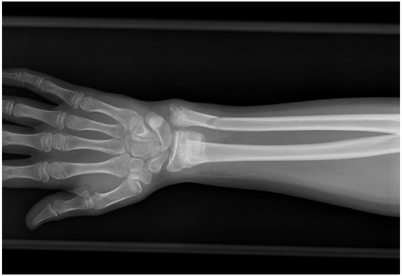
$$z^{(l)} = W^{(l)} a^{(l)} + b^{(l)}, \quad a^{(l+1)} = \sigma(z^{(l)})$$

where $a^{(0)} = x$. The output of the network after L layers is given by:

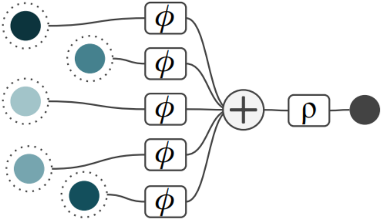
$$\hat{y} = \sigma(z^{(L)})$$

During training, the parameters (weight and bias) are updated through an optimization process according to a specific loss function.

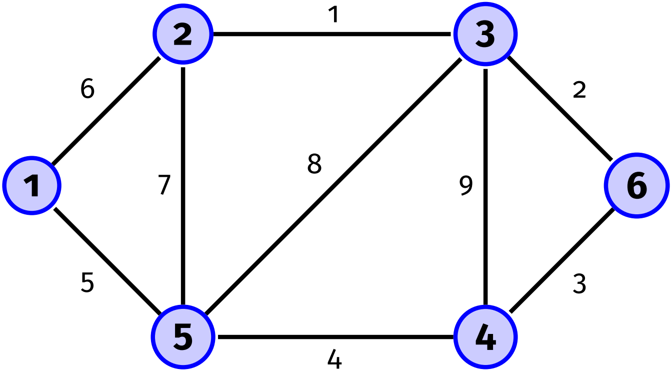
Symmetries in the data: images



Symmetries in the data: sets



Symmetries in the data: graphs



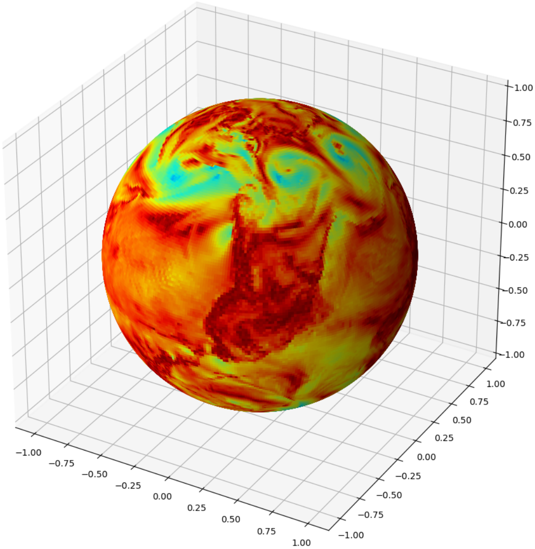
Symmetries in the data: graphs

Let (V, E) be a graph with nodes in V and edges in E. Then we can encode the graph as a vector and adjacency matrix:

$$V = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix} \qquad E = \begin{bmatrix} \cdot & 6 & \cdot & \cdot & 5 & \cdot \\ 6 & \cdot & 1 & \cdot & 7 & \cdot \\ \cdot & 1 & \cdot & 9 & 8 & 9 \\ \cdot & \cdot & 9 & \cdot & 4 & 3 \\ 5 & 7 & 8 & 4 & \cdot & \cdot \\ \cdot & \cdot & 9 & 3 & \cdot & \cdot \end{bmatrix}$$

$$V' = \begin{bmatrix} 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 1 \end{bmatrix} \qquad E' = \begin{bmatrix} \cdot & 1 & \cdot & 7 & \cdot & 6 \\ 1 & \cdot & 9 & 8 & 9 & \cdot \\ \cdot & 9 & \cdot & 4 & 3 & \cdot \\ 7 & 8 & 4 & \cdot & \cdot & 5 \\ \cdot & 9 & 3 & \cdot & \cdot & \cdot \\ 6 & \cdot & \cdot & 5 & \cdot & \cdot \end{bmatrix}$$

Symmetries in the data: manifolds



Symmetries

Definition

A *group* is a set G equipped with a binary operation $\cdot$ that satisfies the following properties:

1. **Closure:** For all $a, b \in G$, the element $a \cdot b$ is also in G.
2. **Associativity:** For all $a, b, c \in G$, $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. **Identity Element:** There exists an element $e \in G$, called the identity element, such that for all $a \in G$, $a \cdot e = e \cdot a = a$.
4. **Inverse Element:** For each $a \in G$, there exists an element $a^{-1} \in G$, called the inverse of a, such that $a \cdot a^{-1} = a^{-1} \cdot a = e$.

We can identify symmetries of a certain space as group operation acting on the domain.

Symmetries

Example ($\mathbb{R}^2$ - unbounded or periodic 2D images)

The symmetry group of 2D images is $(\mathbb{R}^2, +)$, i.e. the group of vertical and horizontal translations.

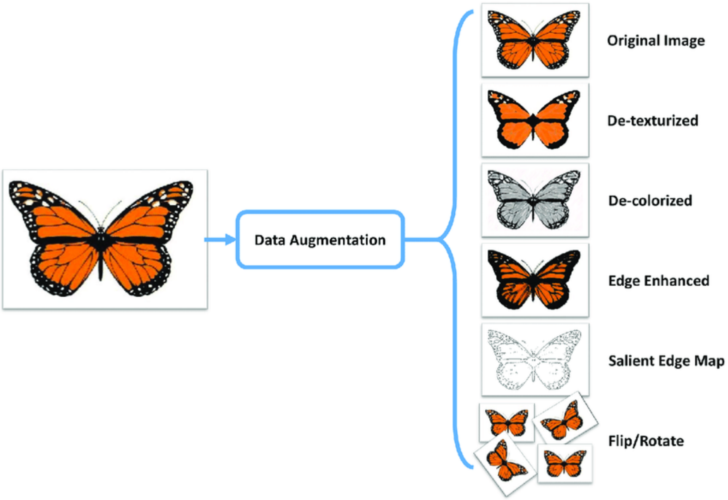
Example (Sets)

The symmetry group for sets with n elements is the permutation group S_n. A permutation acts on a set with fixed ordering by applying a permutation of the order.

Example (Graphs)

The symmetry group for graphs with n nodes is the permutation group S_n. We can indeed list all nodes in a vector V and edges as an $n \times n$ adjacency matrix E. Then a permutation $P \in S_n$ acts on the graph (V, E) as $V' = PV$ and $E' = PEP^T$.

Data augmentation: a naive solution to the problem of symmetries



Geometric Deep Learning

Geometric Deep Learning is a subfield of Deep Learning that focuses on developing algorithms capable of **natively and effectively handling data with a geometric structure**. Geometric Deep Learning aims to process data with an inherent non-Euclidean or geometric structure by identifying a **symmetry group** of interest, guaranteeing that the output of a GDL neural network is either **equivariant** or **invariant** depending on the problem of interest.

Geometric Deep Learning

Let $f : A \to B$ be a function and $x \in A$. Let G be the symmetric group of interest that defines an action on both A and B. Then

Definition

f is said to be **left-equivariant** if

$$f(g \cdot x) = g \cdot f(x)$$

Definition

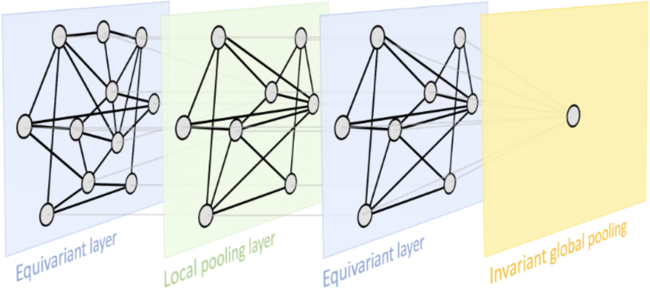
f is said to be **left-invariant** is

$$f(g \cdot x) = f(x)$$

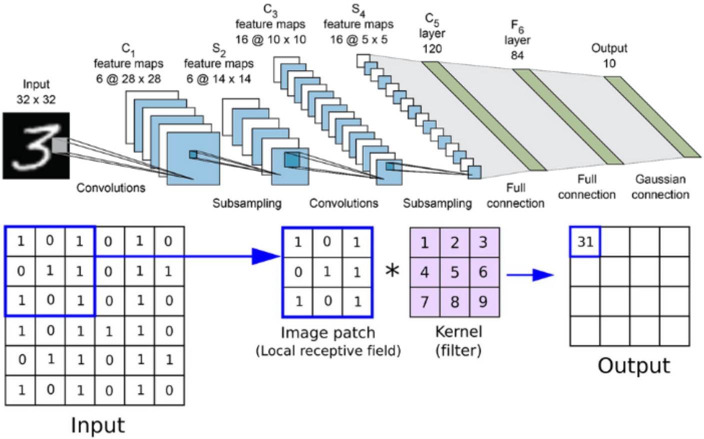
GDL blueprint

The ingredients of a GDL neural network with symmetry group G are:

- G-equivariant layer
- nonlinearity
- coarsening layer
- G-invariant layer (if required)

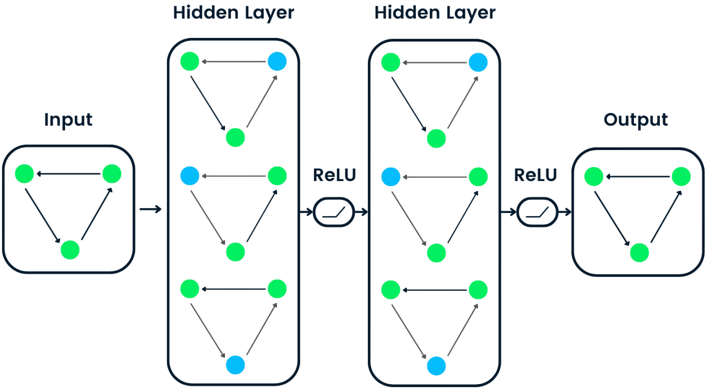


GDL blueprint in CNNs



GDL blueprint in graphs

Graph neural networks:



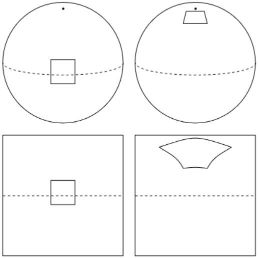
Problem of interest

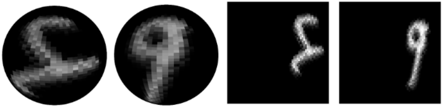
Learning functions on a sphere

Abstract: Spherical CNNs (Cohen T. et al, 2018)

Convolutional Neural Networks (CNNs) have become the method of choice for learning problems involving 2D planar images. However, a number of problems of recent interest have created a demand for models that can analyze spherical images. Examples include omnidirectional vision for drones, robots, and autonomous cars, molecular regression problems, and global weather and climate modelling. A naive application of convolutional networks to a planar projection of the spherical signal is destined to fail, because the space-varying distortions introduced by such a projection will make translational weight sharing ineffective.

Learning functions on a sphere





Learning functions on a sphere

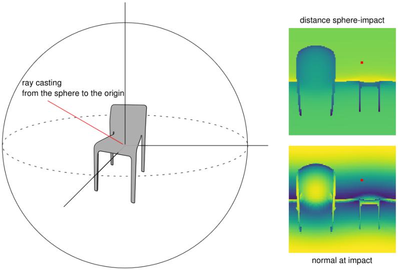
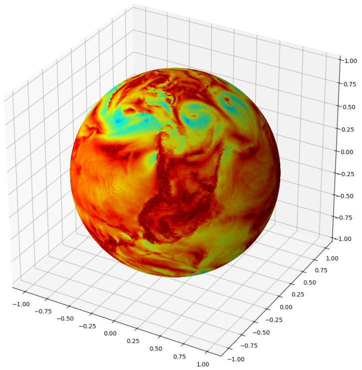


Figure 5: The ray is cast from the surface of the sphere towards the origin. The first intersection with the model gives the values of the signal. The two images of the right represent two spherical signals in (α, β) coordinates. They contain respectively the distance from the sphere and the cosine of the ray with the normal of the model. The red dot corresponds to the pixel set by the red line.

	NR / NR	R / R	NR / R
planar	0.98	0.23	0.11
spherical	0.96	0.95	0.94

Our task: to learn vector fields on a sphere

Learning vector fields on a sphere is not a trivial extension. In our case we are given wind data as a vector field in terms of $V(\alpha, \beta)$ and $U(\alpha, \beta)$ the north and east components of the wind at latitude α and longitude β.



Introduction to the geometry

Orthogonal frame bundle

Let $M = S^n$ embedded in $\mathbb{R}^{n+1}$. Then we can consider the principal G-bundle

$$\mathrm{SO}(n) \to \mathrm{SO}(n+1) \xrightarrow{\pi} S^n.$$

If we identify $\mathrm{SO}(n)$ as the rotation around the north pole $e_{n+1} = (0, \dots, 1)$, then we can consider $\pi : \mathrm{SO}(n+1) \to S^n$ simply as the map $\pi(A) = Ae_{n+1}$.

The group $\mathrm{SO}(n)$ acts on $\mathrm{SO}(n+1)$ as matrix multiplication by

$$\hat{B} = \begin{pmatrix} B & 0 \\ 0 & 1 \end{pmatrix}, \qquad B \in \mathrm{SO}(n).$$

Vector fields on the sphere

Theorem

There is a unique correspondence between real vector fields on the sphere $\xi : S^n \to \mathbb{R}^n$ *and equivariant maps* $\bar{\xi} : SO(n+1) \to \mathbb{R}^n$ *such that* $\bar{\xi}(A\hat{B}) = B^{-1}\bar{\xi}(A)$ *with*

$$\hat{B} = \begin{pmatrix} B & 0 \\ 0 & 1 \end{pmatrix}, \qquad B \in \mathrm{SO}(n), \qquad A \in \mathrm{SO}(n).$$

Vector fields on the sphere

Proof:

Let $\xi : S^n \to \mathbb{R}^{n+1}$ be a vector field $A = (A_1, \ldots, A_{n+1}) \in \mathrm{SO}(n+1)$. Then $\pi(A) = A_{n+1}$, while $\xi(A_{n+1})$ will be in the span of $A_1, \ldots, A_n$. We can then introduce an equivariant map $\bar{\xi}$ by

$$\bar{\xi}(A) = \begin{pmatrix} \langle \xi(A_{n+1}), A_1 \rangle \\ \vdots \\ \langle \xi(A_{n+1}), A_n \rangle \end{pmatrix}$$

Conversely, if we have an equivariant map $\bar{\xi} : \mathrm{SO}(n+1) \to \mathbb{R}^n$, then we can define a vector field by

$$\xi(\pi(A)) = A \begin{pmatrix} \bar{\xi}(A) \\ 0 \end{pmatrix}.$$

■

Vector fields on S^2 as complex functions

We consider the concrete case of a vector field ξ on the two-dimensional sphere S^2. If we identify $\mathbb{R}^2$ with $\mathbb{C}$, then the corresponding equivariance is given by

$$\bar{\xi}(A \cdot Z(\gamma)) = e^{-i\gamma}\bar{\xi}(A)$$

since for $B = e^{i\gamma}$ then $\hat{B} = Z(\gamma) = \begin{pmatrix} e^{i\gamma} & 0 \\ 0 & 1 \end{pmatrix}$.

For V being the wind component in the north direction and U the wind component in the east direction we can explicitly write:

$$\bar{\xi}(Z(\alpha)Y(\beta)Z(\gamma)) = -i(U(\alpha, \beta) + iV(\alpha, \beta))e^{-i\gamma}.$$

Fourier analysis on the sphere

The group SO(3)

By using **Euler's angles** we can describe any element of $\mathrm{SO}(3)$ in terms of rotations around the Z axis and rotations around the Y axis as

$$Z(\alpha)Y(\beta)Z(\gamma) \in \mathrm{SO}(3)$$

where $\alpha, \gamma \in [0, 2\pi)$ and $\beta \in [0, \pi)$.

The left Haar measure on $SO(3)$, i.e. the measure invariant by group transformations, is

$$d\mu = \frac{\sin \beta}{8\pi^2} d\alpha d\beta d\gamma$$

and let $\mathcal{X} := L^2 := L^2(\mathrm{SO}(3), \mathbb{C})$ be the space of square integrable functions on $\mathrm{SO}(3)$ with respect to $d\mu$.

Wigner D-matrices

L^2 admits a basis given by *Wigner D-matrices* $D^l := (D^l_{m,n})$ with indices laying in the indexing set

$$I = \left\{ (l,m,n) \ : \ \begin{matrix} l = 0,1,2,3,\dots, \\ m,n \in [-l,l] \cap \mathbb{Z} \end{matrix} \right\}.$$

For $(l,m,n) \in I$, define function $D^l_{m,n} : \mathrm{SO}(3) \to \mathbb{C}$ by

$$D^l_{m,n}(Z(\alpha)Y(\beta)Z(\gamma)) = e^{-im\alpha} d^l_{m,n}(\beta) e^{-in\gamma},$$

$$d^l_{m,n}(\beta) = \sqrt{(l+m)!(l-m)!(l+n)!(l-n)!} \cdot$$
$$\cdot \sum_{s=s_0}^{s_1} \frac{(-1)^{m-n+s} \cos\left(\frac{\beta}{2}\right)^{2(l-s)+n-m} \sin\left(\frac{\beta}{2}\right)^{m-n+2s}}{(l+n-s)!s!(m-n+s)!(l-m-s)!}$$

with $s_0 = \max\{0, n-m\}$, $s_1 = \min\{l-m, l+n\}$.

Wigner D-matrices

This is an orthonormal basis, so that for any $f \in L^2$, $A \in \mathrm{SO}(3)$

$$f(A) = \sum_{(l,m,n)\in I} \hat{f}^l_{m,n} D^l_{m,n}(A)$$

from which follows that

$$(f * \Psi)(A) = \int_{\mathrm{SO}(3)} f(B)\Psi(B^{-1}A)d\mu(B)$$

$$(f * \Psi)(A) = \sum_{l\in\mathbb{N}} \frac{1}{2l+1} \sum_{k,m,n=-l}^{l} \hat{f}^l_{m,k} \hat{\Psi}^l_{k,n} D^l_{m,n}(A)$$

$$\widehat{f * \Psi}^l_{n,m} = \frac{1}{2l+1} \sum_{k=-l}^{l} \hat{f}^l_{m,k} \hat{\Psi}^l_{k,n}$$

Wigner D-matrices and spherical harmonics

The sphere S^2 is not a group. However, a similar result is obtained by employing the well-known spherical harmonics Y_m^l, i.e. for a function $f \in L^2(S^2, \mathbb{C})$ with bandwidth b

$$f = \sum_{0 \le l \le b} \sum_{|m| \le l} \hat{f}_m^l Y_m^l$$

$$\hat{f}_m^l = \int_{S^2} f(x) \bar{Y}_m^l d\mu(x)$$

In fact

$$Y_m^l = D_{m,0}^l|_{S^2}$$

Wigner D-matrices and Equivariance

$$\mathcal{X}_n = span_{\mathbb{C}}\{D^l_{m,n}\}^{-l\leq m\leq l}_{l=n,n+1,n+2,\dots}$$

Consider the orthogonal decomposition $L^2 = \bigoplus_{n=-\infty}^{\infty} \mathcal{X}_n$.

Consider the subgroup $K \subseteq \mathrm{SO}(3)$ consisting of the matrices on the form $Z(\alpha)$.

We can define an action $\hat{\rho}$ of K on $\mathrm{SO}(3)$ by

$$\hat{\rho}(Z(\alpha))A = AZ(-\alpha) = AZ(\alpha)^{-1}$$

and a representation ρ_n on $\mathbb{C}$ by

$$\rho_n(Z(\alpha))z = e^{in\alpha}z.$$

Wigner D-matrices and Equivariance

Definition

We say that a function $f \in L^2$ is n-equivariant if

$$f(\hat{\rho}(Z(\alpha))A) = f(AZ(-\alpha)) = e^{in\alpha} f(A) = \rho_n(Z(\alpha)) f(A).$$

Direct computation shows that any $D^l_{m,n}$ is an n-equivariant function, and by using their orthogonallity, it follows that $f \in L^2$ is n-equivaraint if and only if $f \in \mathcal{X}_n$.

Corollary

Vector fields on the sphere, which are in unique correspondence with 1-equivariant functions, thus in unique correspondence with functions spanned by $\mathcal{X}_1$.

GDL blueprint in our project

GDL blueprint

Recall that the GDL blueprint for a neural network is the following:

- G-equivariant layer
- nonlinearity
- coarsening layer
- G-invariant layer (not needed in our case)

G-equivariant layer in previous work

For $f \in \mathcal{X}_1$ introduce weights Ψ^l for order $l \in \mathbb{N}$. An equivariant layer $f \mapsto f * \Psi$ can be defined as

$$\widehat{f * \Psi}^l_{m,1} = \frac{1}{2l+1} \hat{f}^l_{m,1} \hat{\Psi}^l$$

Advantages:

- The output is guaranteed to be in $\mathcal{X}_1$;
- Lightweight (few weights);
- Gradients of a tensor during training are manageable.

Disadvantages:

- Low expressivity (weights are only order-rescalings);
- Can only use equivariant nonlinearities.

The smoothing operator $\mathscr{S}_q$

Introduce the orthogonal projection $\mathscr{S}_q : \mathcal{X} \to \mathcal{X}_q$ defined by

$$\mathscr{S}_q(x) = \frac{1}{2\pi} \int_0^{2\pi} e^{iq\theta} r_{Z(-\theta)} x \, d\theta.$$

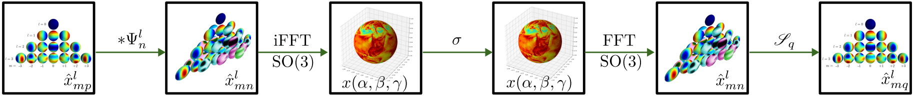
This can be rewritten in the spectral domain as

$$\widehat{\mathscr{S}_q x} = \hat{x}^l_{m,n} \delta_{n,q}.$$

G-equivariant layer in our work

$$\widehat{f * \Psi}^{l}_{m,n} = \frac{1}{2l+1} \sum_{s=-l}^{l} \hat{f}^{l}_{m,s} \hat{\Psi}^{l}_{s,n}$$

For $f \in \mathcal{X}_1$ we can restrict Ψ to coefficients $\hat{\Psi}^l_n$



Advantages:

- More expressive
- Can use any nonlinearity

Disadvantages:

- Gradients are heavier
- Slower

Nonlinearity: the $\mathbb{C}$-ReLU

The most common activation function is ReLU (Rectified Linear Unit)

$$\text{ReLU}(x) = \max(0, x)$$

- Complex neural networks deal with complex-valued inputs, weights, and activation fucntions.
- We need an activation function that can handle complex numbers effectively.

$$\mathbb{C}\text{-ReLU}(z) = \text{ReLU}(Re(z)) + i\text{ReLU}(Im(z))$$

This can then be modified to obtain weighted ReLU, complex leaky ReLU, ...

Coarsening layer

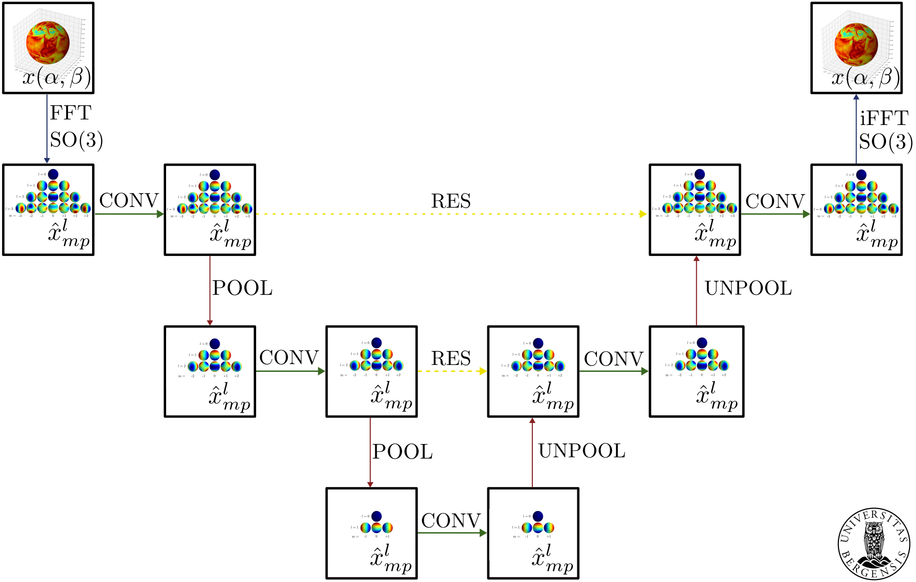
We implement coarsening by limiting the bandwidth of the function at a certain order.

We need to keep in mind that:

- group equivariance does not allow for the transfer of information between degrees (different values of l);
- in practice it cuts high-frequency information.

Therefore coarsening needs to be paired with an activation function in the spatial domain.

Our proposed architecture



Results

ERA5 Dataset

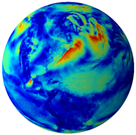
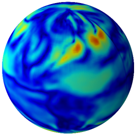
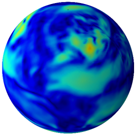
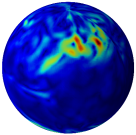
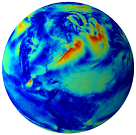
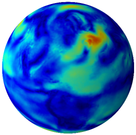
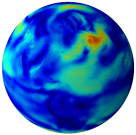
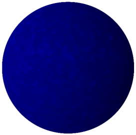
We use the ERA5 meteorological dataset.

It contains hourly global measurements of different quantities from 1940 to today.

In our experiments, we use wind data at 10m of elevation and temperature data at 2m of elevation.

For training and model selection we use a coarser dataset of 52 weekly datapoints per year for both wind and temperature. Years from 2000 to 2009 (included) have been used for training, while the years 2020 and 2021 have been used for validation and model selection. Years 2022 and 2023 have been used for testing.

Equivariance

Model	Ground truth	Pred. $\beta = 0$	Pred. $\beta = \frac{\pi}{4}$	Error
CNN				
Ours				

0 km/h  28 km/h

Wind to wind prediction

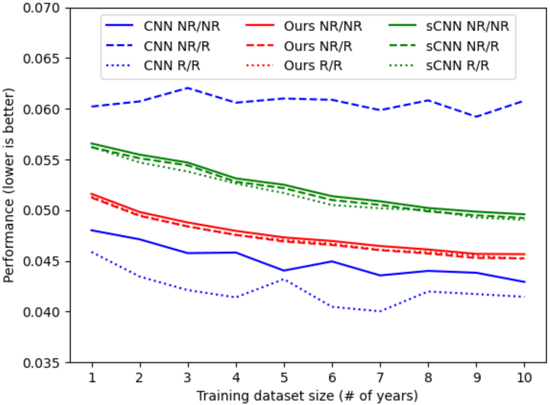


Figure: Wind to Wind t+24h prediction.

Temperature to wind estimation

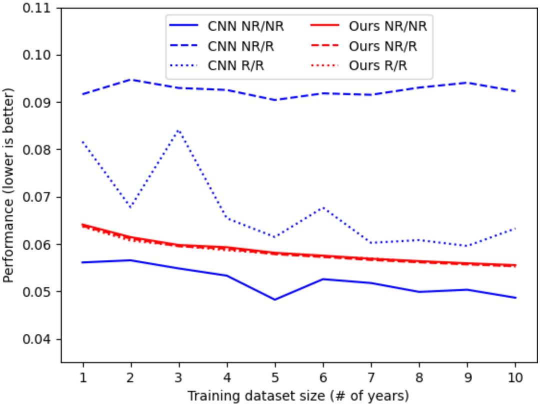


Figure: Temperature to Wind estimation.

Autoencoder compression

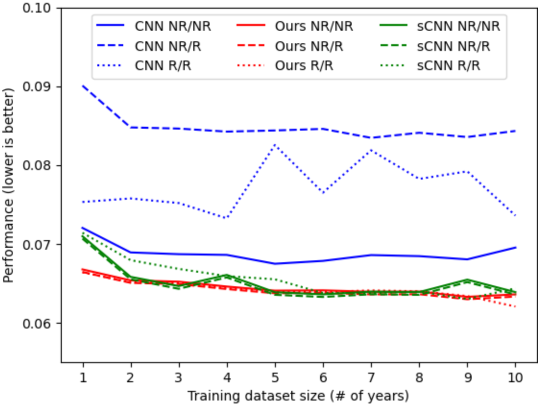


Figure: Autoencoder compression on wind data.

References

Bronstein M., Bruna J., Cohen T., Veličković P.
Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges, arXiv:2104.13478 (2021).

Cohen T., Geiger M., Koehler J., Welling M.
"Spherical CNNs", ICLR 2018.

Esteves C., Allen-Blanchette C., Makadia A., Daniilidis K.
Learning SO(3) Equivariant Representations with Spherical CNNs, Int. J. Comput. Vision, 128(3), 588–600, (2019).

Esteves C., Slotine J., Makadia A.
Scaling Spherical CNNs, ICML2023.

Ballerin F., Blaser N., Grong E.
SO(3)-Equivariant Neural Networks for Learning Vector Fields on Spheres, arXiv:2503.09456 (2025)

UNIVERSITAS BERGENSIS