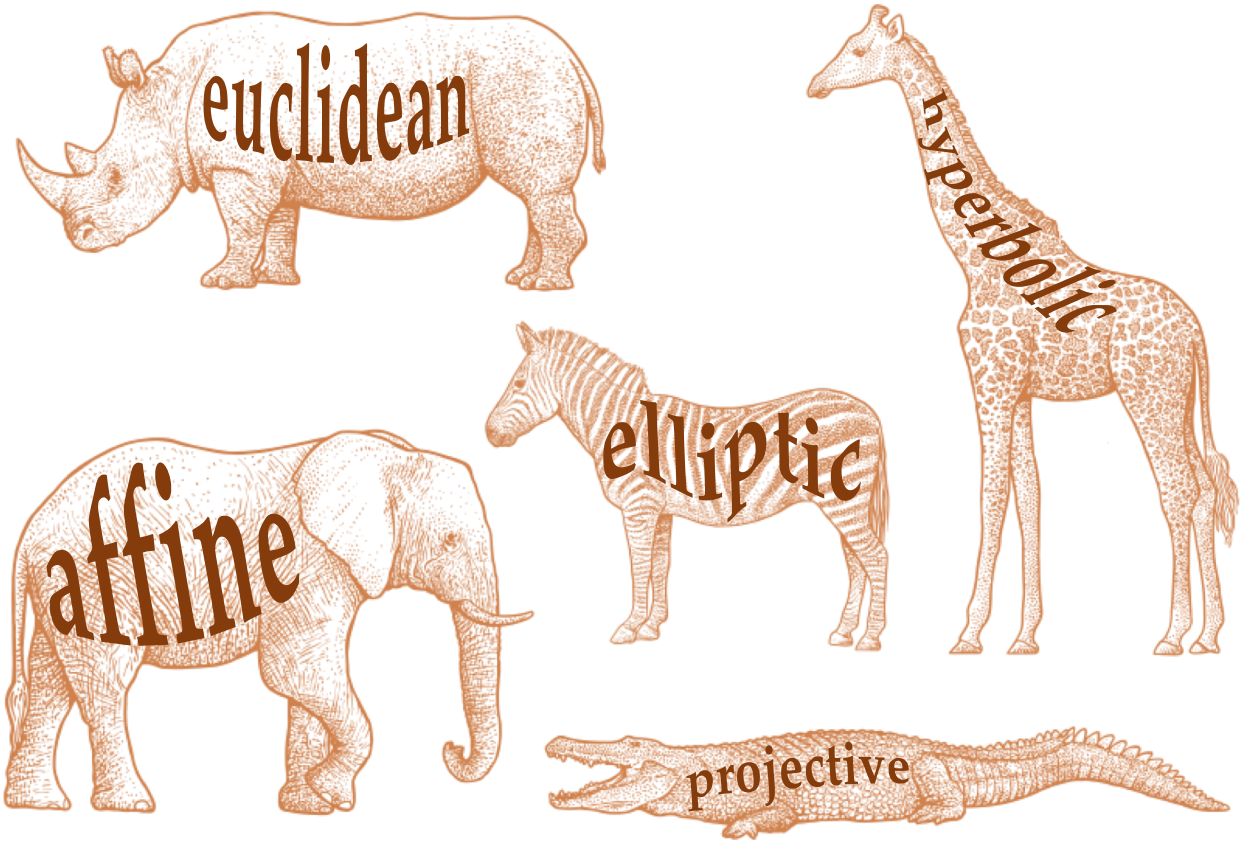
These slides and topics are extracted from the course "Geometric Deep Learning" taught in the African Master in Machine Intelligence in 2022 by Michael Bronstein (Imperial College/Twitter), Joan Bruna (NYU), Taco Cohen (Qualcomm), and Petar Veličković (DeepMind).

Original ideas are theirs, errors are mine.

https://www.youtube.com/playlist?list=PLn2-dEmQeTfSLXW8yXP4q_Ii58wFdxb3C.

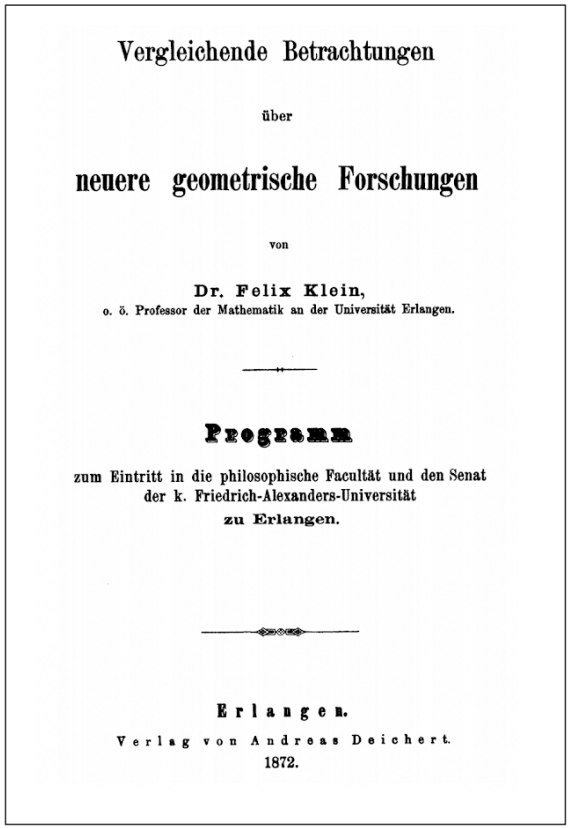
Highly recommended.

19th Century Zoo of Geometries



The Erlangen Programme

"Given a [homogeneous] manifold and a transformation group acting [transitively] on it, to investigate those properties of figures on that manifold which are invariant under transformations of that group"



Vergleichende Betrachtungen

über

neuere geometrische Forschungen

von

Dr. Felix Klein,
o. ö. Professor der Mathematik an der Universität Erlangen.

Programm

zum Eintritt in die philosophische Facultät und den Senat
der k. Friedrich-Alexanders-Universität
zu Erlangen.

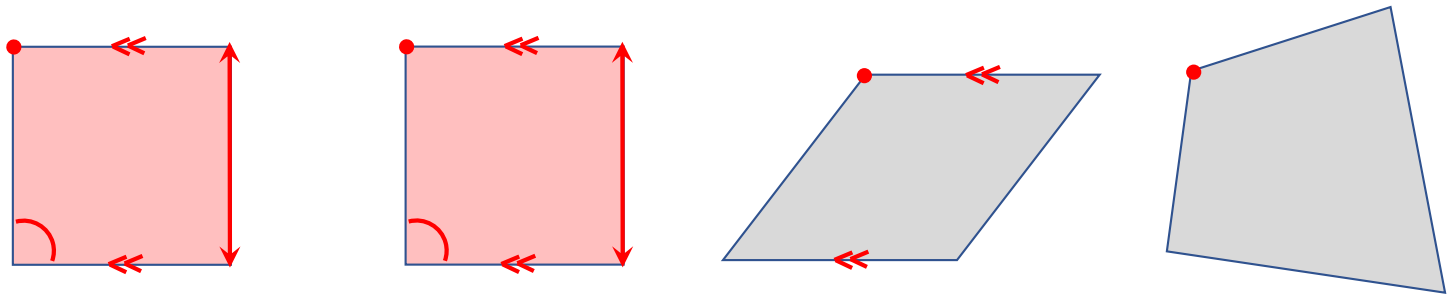
Erlangen.
Verlag von Andreas Deichert.
1872.

F. Klein

1872

Klein 1872

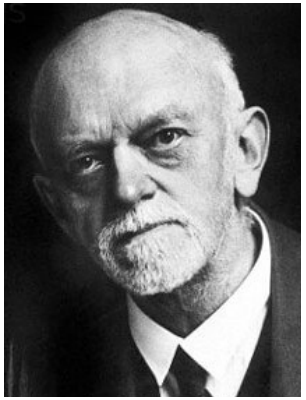
The Erlangen Programme



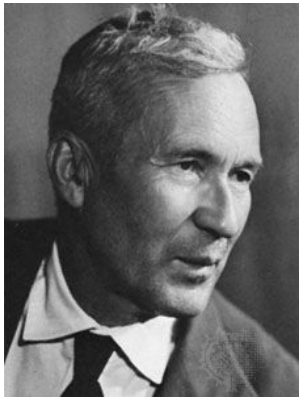
	Euclidean	**Affine**	**Projective**
angle	+	—	—
distance	+	—	—
area	+	—	—
parallelism	+	+	—
intersection	+	+	+

?

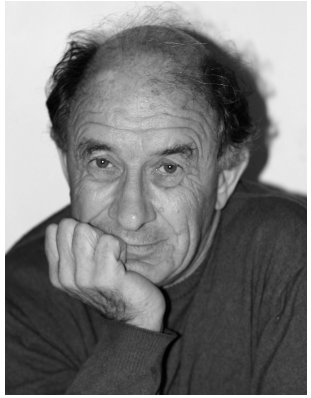
Universal approximation



D. Hilbert



A. Kolmogorov



V. Arnold

G. Cybenko

K. Hornik

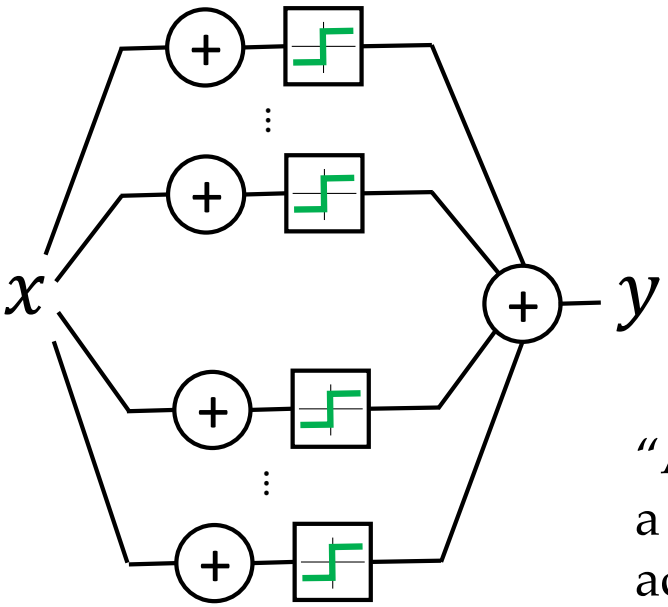
13th Problem

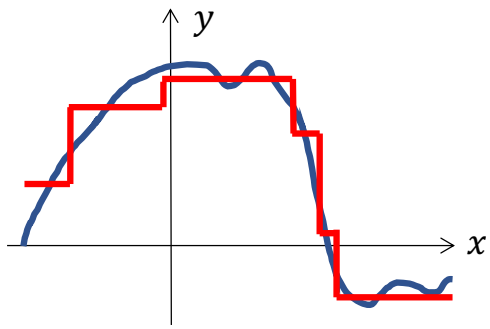
$$f(x_1, \ldots, x_n) = \sum_{q=0}^{2n} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right)$$

Results specific to multilayer neural networks

Hilbert 1900; Arnold 1956; Kolmogorov 1957; Cybenko 1989; Hornik 1991

Universal approximation



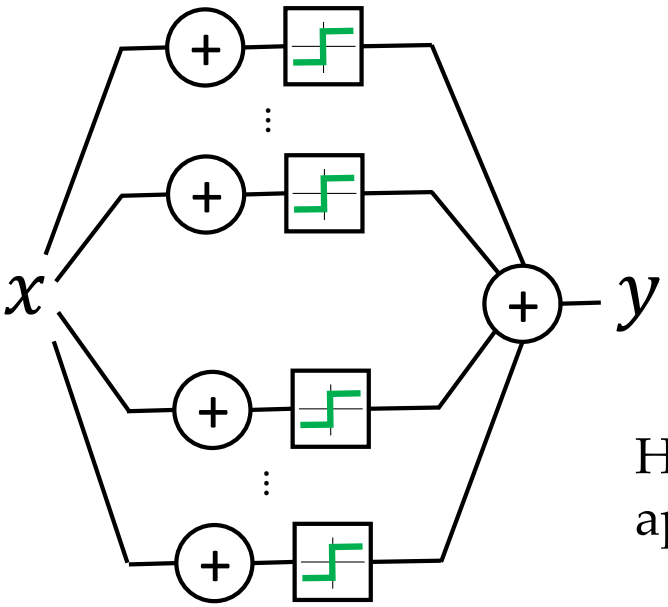


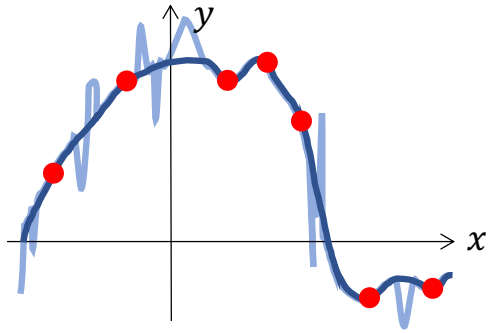
"A 2-layer perceptron can approximate a continuous function to any desired accuracy"

Cybenko 1989; Hornik 1991; Barron 1993; Leshno et al 1993; Maiorov 1999; Pinkus 1999

Deep learning = glorified curve fitting

Universal approximation





How many samples are needed to approximate to accuracy ε ?

Cybenko 1989; Hornik 1991; Barron 1993; Leshno et al 1993; Maiorov 1999; Pinkus 1999

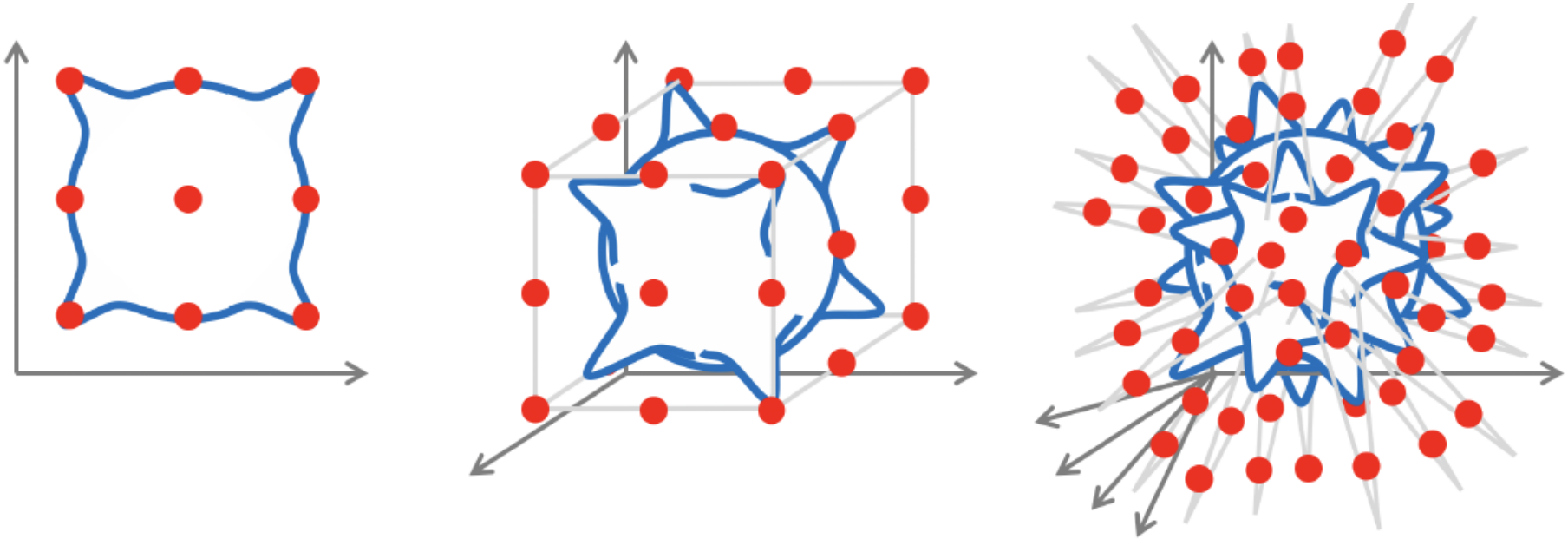


Figure 2: We consider a Lipschitz function $f(x) = \sum_{j=1}^{2^d} z_j \phi(x - x_j)$ where $z_j = \pm 1$, $x_j \in \mathbb{R}^d$ is placed in each quadrant, and ϕ a locally supported Lipschitz 'bump'. Unless we observe the function in most of the 2^d quadrants, we will incur in a constant error in predicting it. This simple geometric argument can be formalised through the notion of *Maximum Discrepancy* (von Luxburg and Bousquet, 2004), defined for the Lipschitz class as $\kappa(d) = \mathbb{E}_{x,x'} \sup_{f \in \mathrm{Lip}(1)} \left| \frac{1}{N} \sum_l f(x_l) - \frac{1}{N} \sum_l f(x'_l) \right| \simeq N^{-1/d}$, which measures the largest expected discrepancy between two independent N-sample expectations. Ensuring that $\kappa(d) \simeq \epsilon$ requires $N = \Theta(\epsilon^{-d})$; the corresponding sample $\{x_l\}_l$ defines an ϵ-net of the domain. For a d-dimensional Euclidean domain of diameter 1, its size grows exponentially as ϵ^{-d}.

First "geometric" machine learning

Group Invariance Theorem: "if a neural network is invariant to a group, then its output can be expressed as functions of the orbits of the group"

M. Minsky **S. Papert**

1969

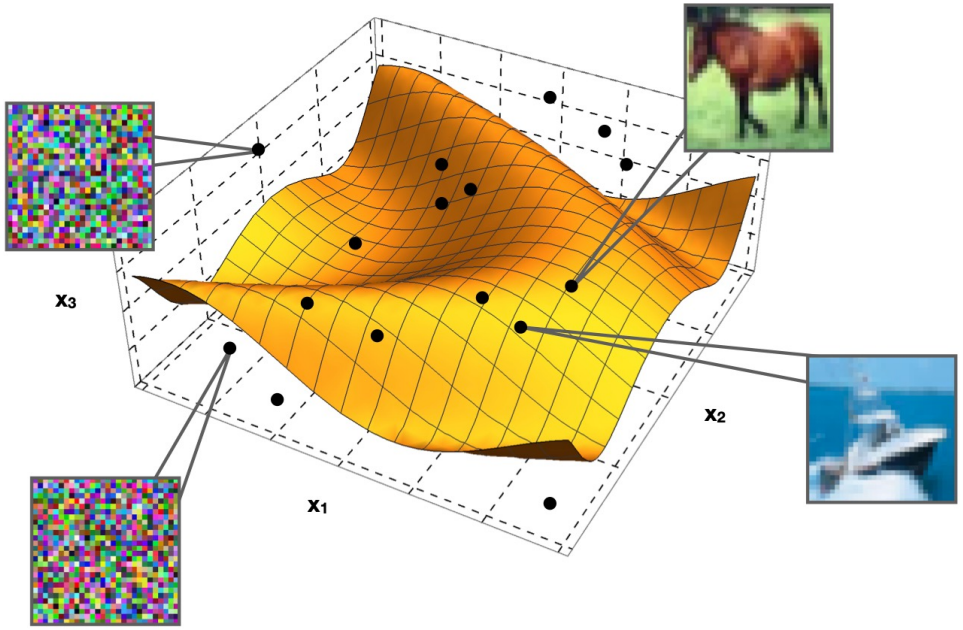
Minsky, Papert 1969

Data

$\{(x_i, y_i)\}_i, with\ x \in \mathcal{X}: data\ domain, y \in \mathbb{R}$

$x_i \sim \nu$: data distribution defined over $\mathcal{X}$.

$y_i = f^*(x_i)$ for some unknown $f^*: \mathcal{X} \to \mathbb{R}$



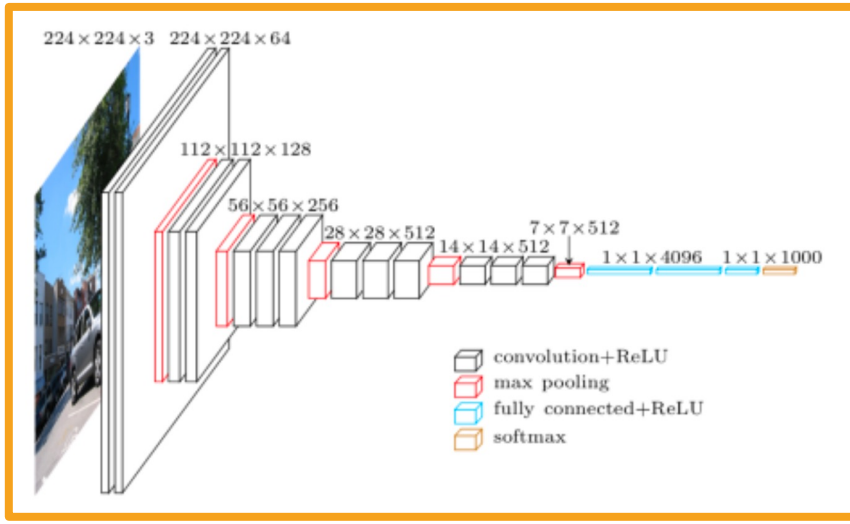
[Goldt, Zdeborova, Krzakala et al]

Examples:

- f*(x) : excitation energy of molecule x.
- f*(x) = P(y=1 | x) : logit in a binary classification task.

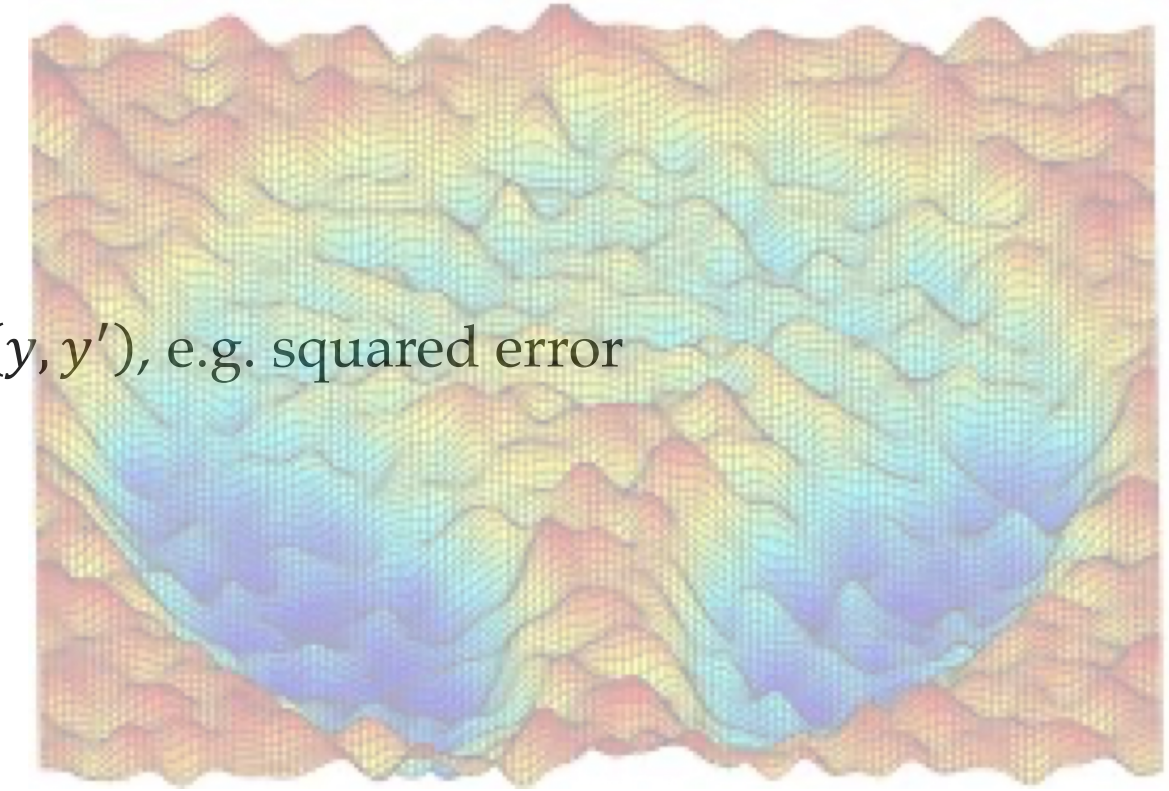
In order to analyse ML algorithms and provide guarantees, we need assumptions on both ν and f^*.

Model



- The model or *hypothesis class* is a subset $\mathcal{F} \subset \{f: \mathcal{X} \to \mathbb{R}\}$.
- Examples:
  - $\mathcal{F} = \{\, polynomials\ of\ degree\ up\ to\ k\}$
  - $\mathcal{F} = \{\, Neural\ networks\ of\ given\ architecture\ \}$
- The hypothesis class is *organized* in terms of a (non-negative) *complexity measure* $\gamma: \mathcal{F} \to \mathbb{R}$, *e.g.* a norm.
- Examples:
  - $\gamma(f)$: number of neurons in a NN.
  - $\gamma(f) = \int (1+\omega^2)^s |\hat{f}(\omega)|^2 d\omega$ *Sobolev Norm*
- **Remark**: Often this complexity γ(f) is implicitly defined through the algorithm!

Error Metric

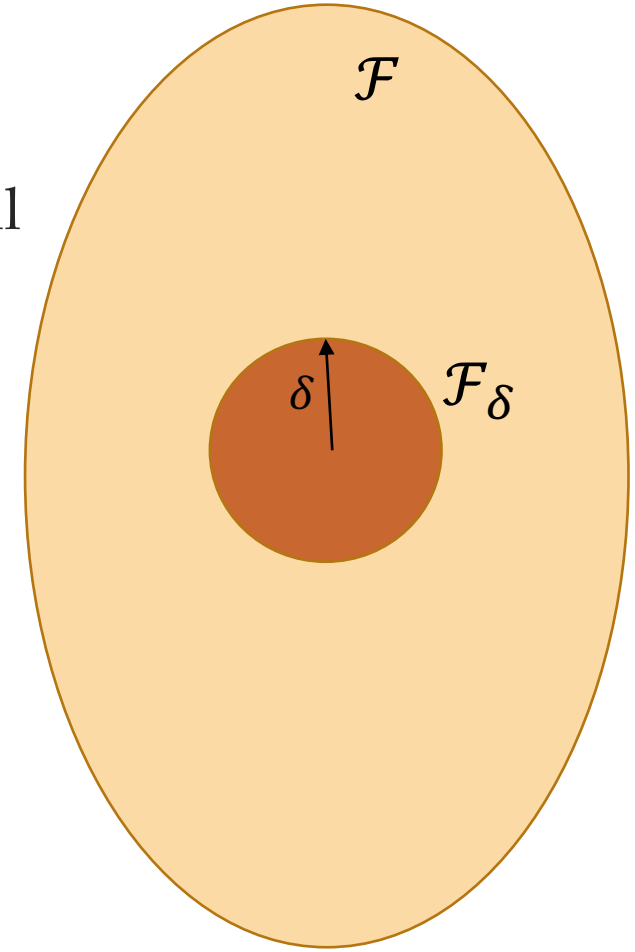


- Given point-wise convex error measure $\ell(y, y')$, e.g. squared error $\ell(y, y') = |y - y'|^2$, we consider:
  - *Population Loss:* $\mathcal{R}(f) = \mathbb{E}_{\nu}[\ell\big(f(x), f^*(x)\big)]$
  - *Empirical Loss:* $\hat{\mathcal{R}}(f) = \frac{1}{n}\sum_i \ell\left(f(x_i), f^*(x_i)\right)$

- **<u>Fact</u>**: for each $f \in \mathcal{F}$, $\hat{\mathcal{R}}(f)$ is an unbiased estimator of $\mathcal{R}(f)$, with variance $\sigma^2(f) = \frac{1}{n}\mathbb{E}\left[\ell\big(f(x), f^*(x)\big) - \mathcal{R}(f)\right]^2$
  - However, this *point-wise* variance bound is not very useful in ML, since hypothesis f depends on training set!
  - We use *uniform* bounds instead, e.g. Rademacher complexities.

Empirical Risk Minimization

- <u>Underlying Goal</u>: Minimise $\mathcal{R}(f)$ having only access to $\hat{\mathcal{R}}(f)$.
- In order to control fluctuations uniformly, we consider the ball
$$\mathcal{F}_\delta = \{f \in \mathcal{F}; \gamma(f) \leq \delta\}$$
- **Empirical Risk Minimisation** (ERM), constrained form:
$$\hat{f}_\delta = arg \min_{f \in \mathcal{F}_\delta} \hat{\mathcal{R}}(f)$$
  - Penalized form: $arg \min_{f \in \mathcal{F}} \hat{\mathcal{R}}(f) + \lambda\gamma(f)$
  - Interpolation form: $arg \min_{f \in \mathcal{F}} \gamma(f) \;\; s.t. \hat{\mathcal{R}}(f) = 0.$



$$\hat{f} = \arg\min_{f \in \mathcal{F}_\delta} \hat{R}(f)$$

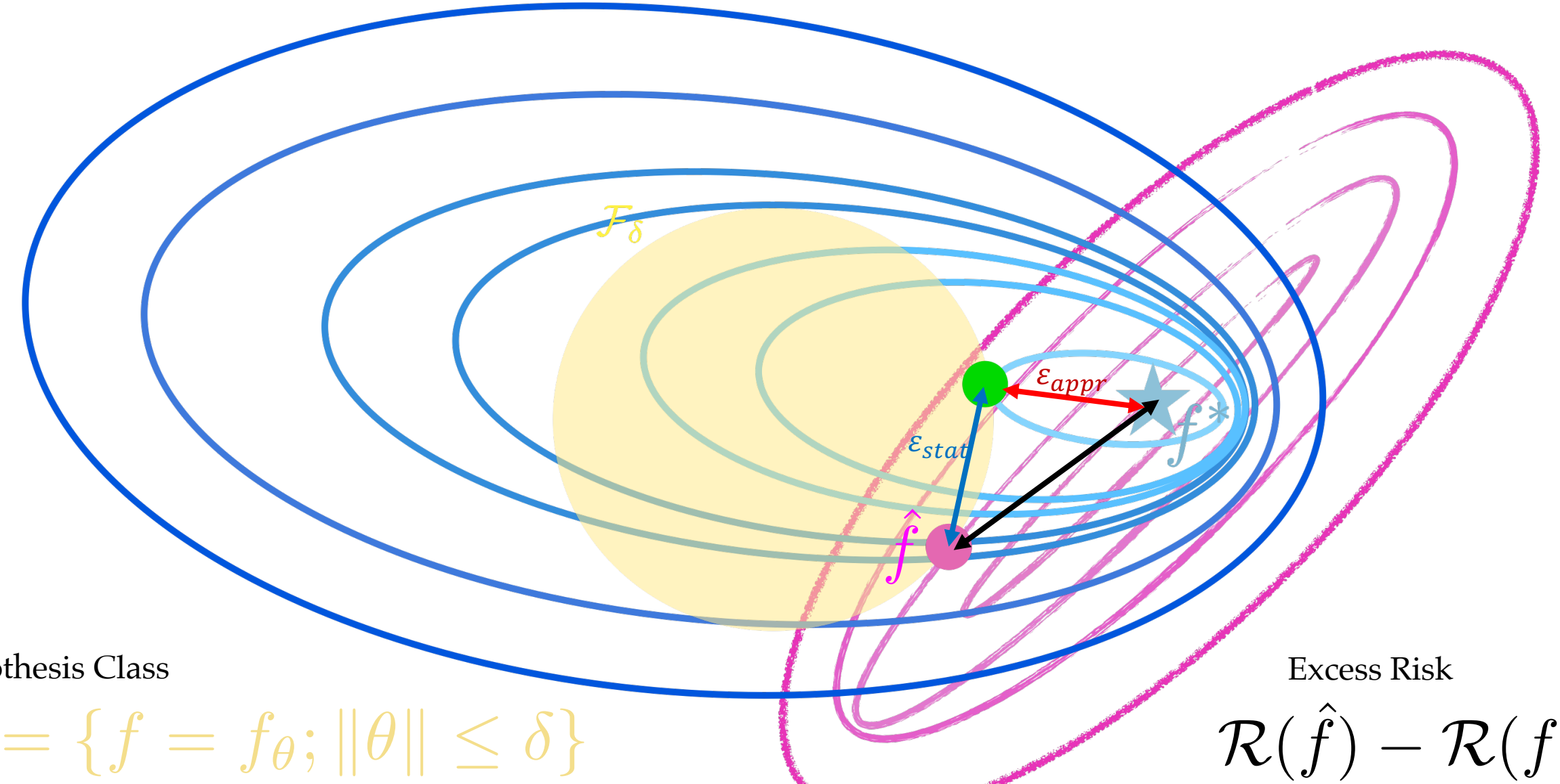
ERM

$$\mathcal{R}(f) = \mathbb{E}[\ell(f(x), y)]$$

Population Loss

$$\hat{\mathcal{R}}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i)$$

Empirical Loss



Hypothesis Class

$$\mathcal{F}_\delta = \{f = f_\theta; \|\theta\| \leq \delta\}$$

Excess Risk

$$\mathcal{R}(\hat{f}) - \mathcal{R}(f^*)$$

The Challenge of High-dimensional Learning

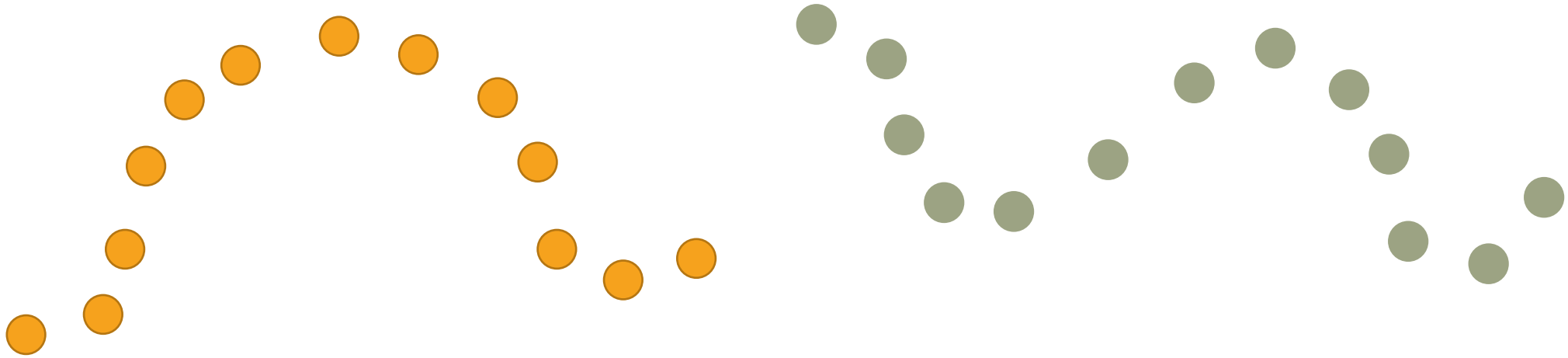
[Bottou, Bousquet]

$$\mathcal{R}(\hat{f}) \leq inf_{f \in \mathcal{F}} \mathcal{R}(f) + \varepsilon_{opt} + \varepsilon_{stat} + \varepsilon_{appr}$$

- $inf_{f \in \mathcal{F}} \mathcal{R}(f) = 0$ if $\mathcal{F}$ is dense, e.g. neural networks with non-polynomial activation (Universal Approximation Theorems).
- Approximation error $\varepsilon_{appr} = inf_{f \in \mathcal{F}_\delta} \mathcal{R}(f)$: Small if hypothesis space is such that target f^* has small complexity. Decreases with δ.
- Statistical error $\varepsilon_{stat} = 2sup_{f \in \mathcal{F}_\delta} |\mathcal{R}(f) - \hat{\mathcal{R}}(f)|$: small when $\mathcal{F}_\delta$ can be covered with 'few' small balls. Increases with δ.
- Optimization error $\varepsilon_{opt} = \hat{\mathcal{R}}(\hat{f}) - inf_{f \in \mathcal{F}_\delta} \hat{\mathcal{R}}(f)$ small when ERM can be efficiently solved.

How to simultaneously control all sources of error in the high-dimensional regime?

Basic Principle of Learning: Interpolation



- Inherent low-dimensional property: based on proximity between empirical and population densities.
  - Similarity search / Nearest-Neighbor / Kernel methods.

How does this learning principle behave in higher-dimensions?

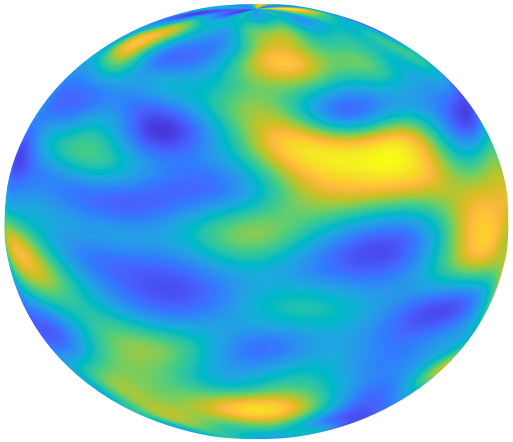
Learning Lipschitz Functions

- Recall that a function $f: \mathcal{X} \subseteq \mathbb{R}^d \to \mathbb{R}$ is β-Lipschitz if $|f(x) - f(x')| \leq \beta||x - x'||$.
  - Regularity class associated with a locality prior.

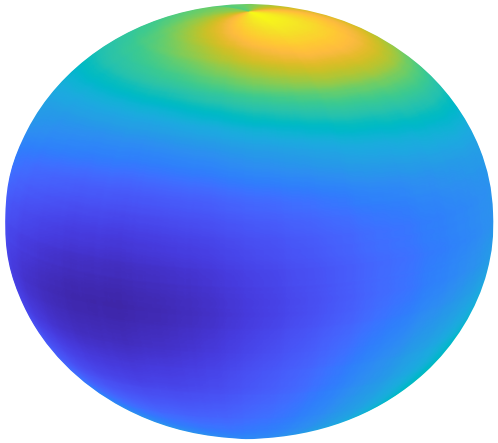
- Assume $f^*$is 1-Lipschtiz, and ν the $\mathcal{N}(0, I_d)$ distribution. How many samples are needed to estimate f^* up to error ε?

Summary so far

- Lipschitz class is too large: statistical error cursed by dimension.



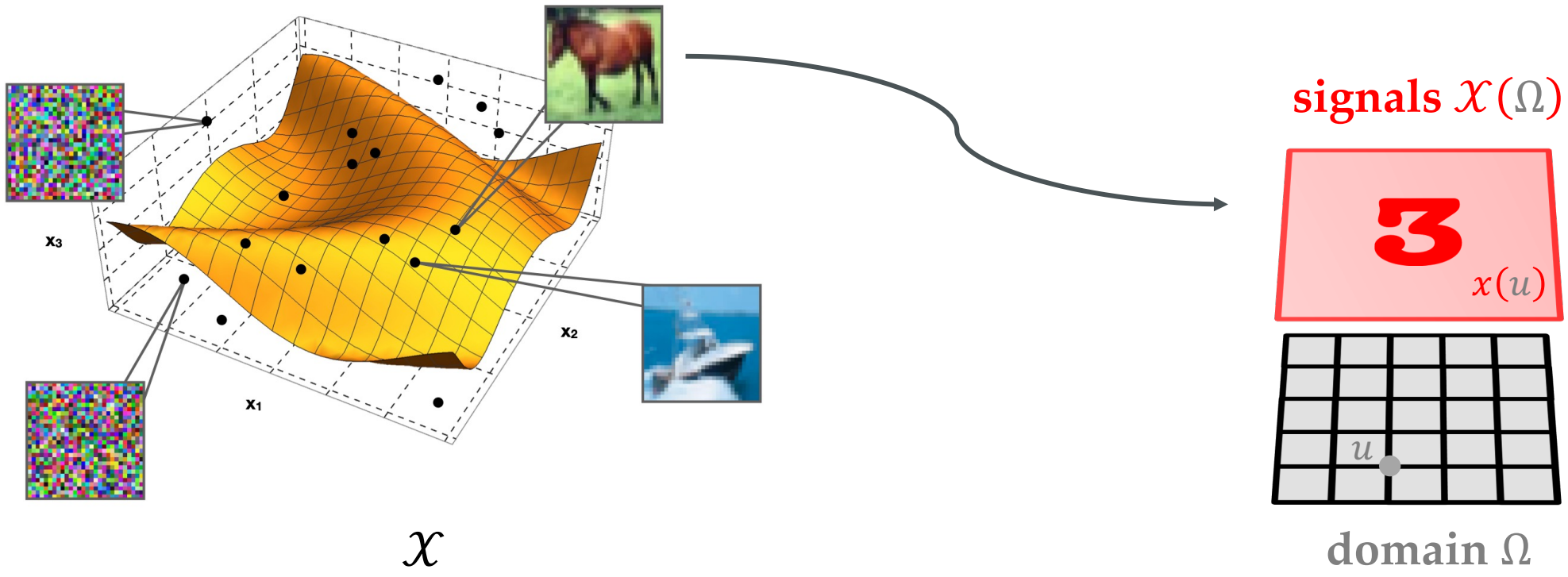
- Smooth Sobolev/Barron classes are too small: approximation error cursed by dimension.



- We need to define more adapted function spaces. How?

Towards Geometric Function Spaces

- Exploit the underlying low-dimensional structure of the input high-dimensional space $\mathcal{X}$:

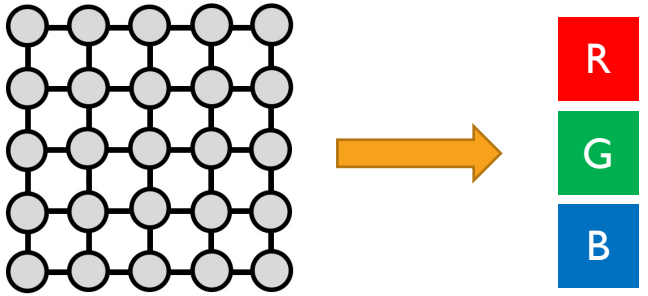


$\mathcal{X}$

The Space of Signals on a Geometric Domain

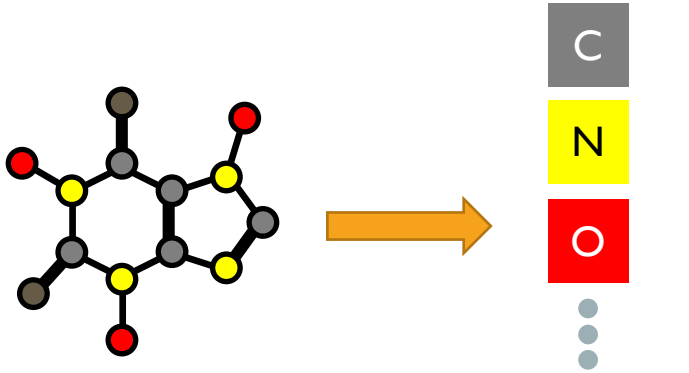
- A *signal* on Ω is just a function $x : \Omega \rightarrow \mathcal{C}$, where:
  - Ω is the domain
  - $\mathcal{C}$ is a vector space, whose dimensions are called *channels*
- The space $\mathcal{C}$-valued signals on Ω is defined as

$$\mathcal{X}(\Omega, \mathcal{C}) = \{ x : \Omega \rightarrow \mathcal{C} \}$$



$\Omega = \mathbb{Z}_n \times \mathbb{Z}_n$ $\mathcal{C} = \mathbb{R}^3$

Example: $n \times n$ RGB image



$\Omega = \{1, \dots, n\}$ $\mathcal{C} = \mathbb{R}^m$

Example: molecular graph

Hilbert Space Structure

- We can add signals and multiply by scalars:

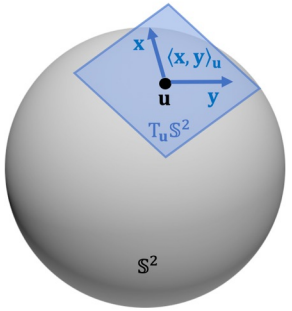
$$(\alpha x + \beta y)(u) = \alpha x(u) + \beta y(u), \qquad \text{where } \alpha, \beta \in \mathbb{R} \text{ and } u \in \Omega$$

- → The space of signals is a vector space! (possibly infinite dimensional)
- Moreover, given an inner product $\langle\, , \rangle_{\mathcal{C}}$ on $\mathcal{C}$ and a measure μ on Ω, we obtain an inner product on $\mathcal{X}(\Omega, \mathcal{C})$:

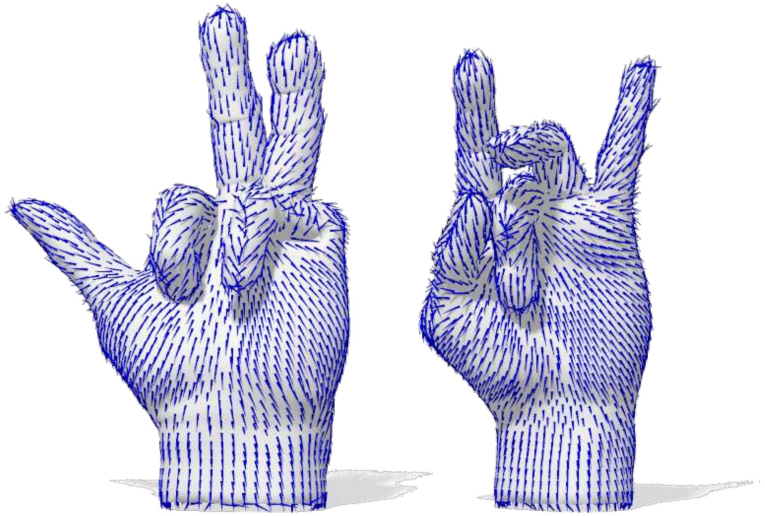
$$\langle x, y \rangle = \int_{\Omega} \langle x(u), y(u) \rangle_{\mathcal{C}} \, \mathrm{d}\mu(u).$$

- → The space of signals is a Hilbert space!
  Verify that the above satisfies the axioms of an inner product

Fields of Geometric Features



The tangent space



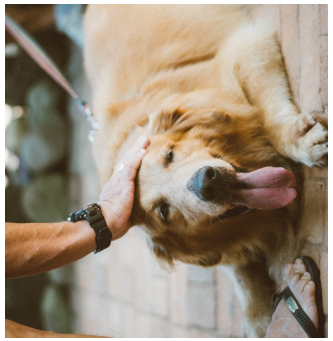
A *vector field* assigns to each $u \in \Omega$ a vector $x(u)$ in the tangent space $\mathcal{C}_u = T_u\Omega$ at u.

- Function:
  - maps $u \in \Omega$ to $x(u) \in \mathcal{C}$
- Field ("section of a bundle"):
  - maps $u \in \Omega$ to $x(u) \in \mathcal{C}_u$
  - Where $\mathcal{C}_u$ is a feature space ("fiber") attached to $u \in \Omega$
  - The feature spaces are assumed to be isomorphic but not identical!
  - Only once we choose a way to identify the fibers, we can write a field as a function

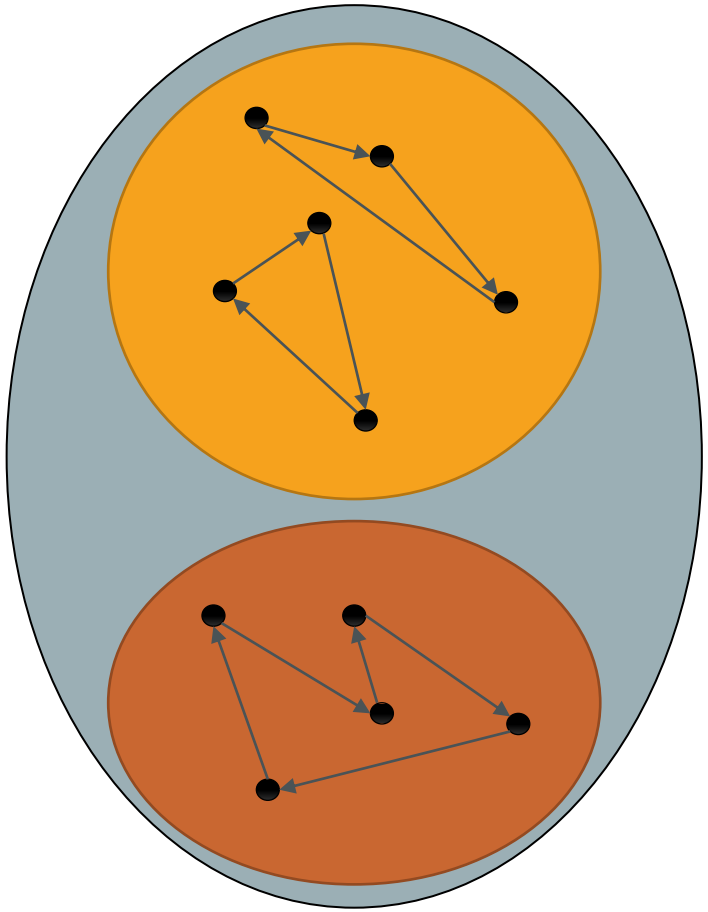
For simplicity we will only work with function spaces $\mathcal{X}(\Omega, \mathcal{C})$ for now

Symmetries of the Label Function

- Let $\mathcal{X}$ denote the input space and $\mathcal{Y}$ the label space
- Let $L : \mathcal{X} \to \mathcal{Y}$ be the ground-truth label function
- A transformation $\mathfrak{g} : \mathcal{X} \to \mathcal{X}$ is a symmetry if $L \circ \mathfrak{g} = L$

L( ) = L() = “dog”

Learning Classes ≅ Learning Symmetries



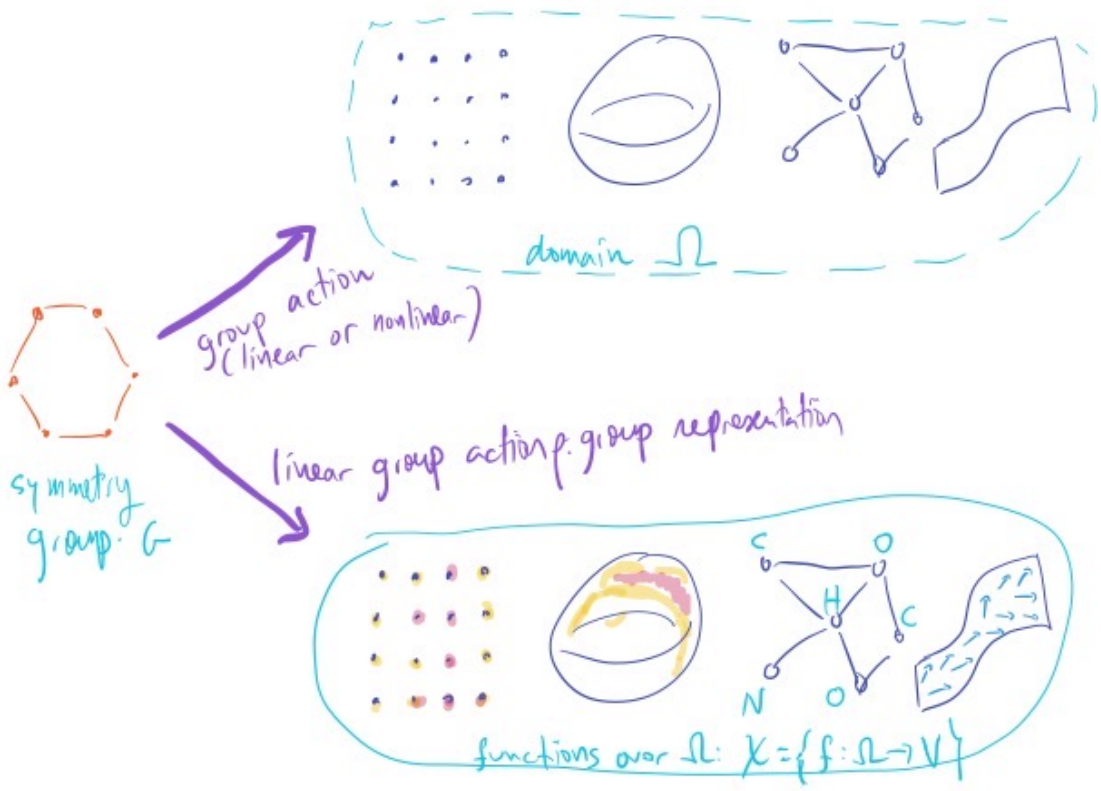
$\mathcal{X}$

- *Any* invertible map that respects the class boundaries is a symmetry of the label function L
- If we knew *all* the symmetries of L, we only need one label per class!
  - Explain why
- If the learning problem is non-trivial, we should not expect to be able to find the full symmetry group a priori.

Symmetries of Ω acting on Signals $\mathcal{X}(\Omega, \mathcal{C})$

Given an action of $\mathfrak{G}$ on Ω, we automatically obtain an action of on the space of signals $\mathcal{X}(\Omega)$:

$$(\mathfrak{g}\, x)(u) = x(\mathfrak{g}^{-1}u).$$

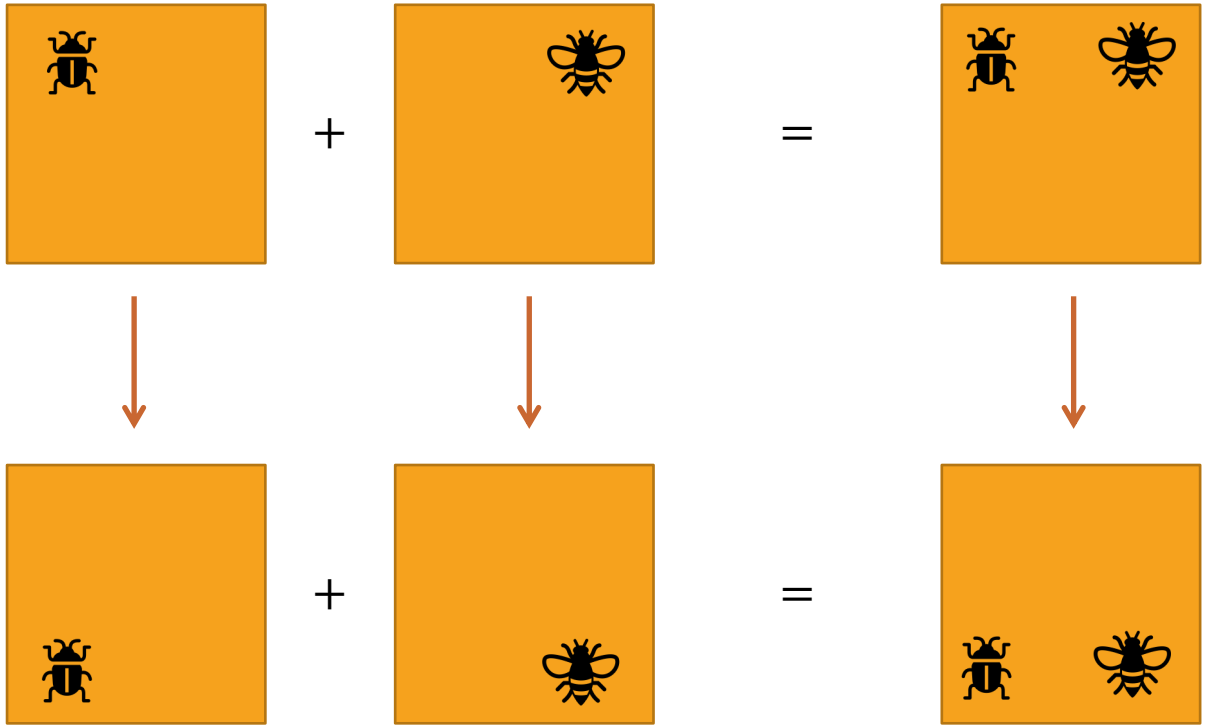


Linearity of the Group Action on Signals

Action on signals: $(\mathfrak{g}\, x)(u) = x(\mathfrak{g}^{-1}u)$.

Linearity: $\mathfrak{g}\,(\alpha x + \beta y) = \alpha \mathfrak{g} x + \beta \mathfrak{g} y$

Apply $\mathfrak{g} = (t_x, t_y)$



Group Representations

An n-dimensional real representation of a group $\mathfrak{G}$ is a map $\rho : \mathfrak{G} \to \mathbb{R}^{n \times n}$, assigning to each $\mathfrak{g} \in \mathfrak{G}$ an invertible matrix $\rho(\mathfrak{g})$, and satisfying

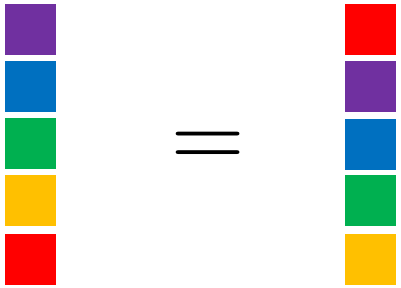
$$\rho(\mathfrak{g}\mathfrak{h}) = \rho(\mathfrak{g})\rho(\mathfrak{h}), \qquad \forall \mathfrak{g}, \mathfrak{h} \in \mathfrak{G}$$

Example:

- The group $\mathfrak{G} = (\mathbb{Z}, +)$
- The domain $\Omega = \mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$ (e.g. short audio signal)
- The action of $\mathfrak{g} = n$ on $\mathrm{u} \in \Omega$: $(n, u) \mapsto n + u \pmod 5$
- The representation on $\mathcal{X}(\Omega)$: (verify that this is a representation)

$$\rho(n) = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}^n$$

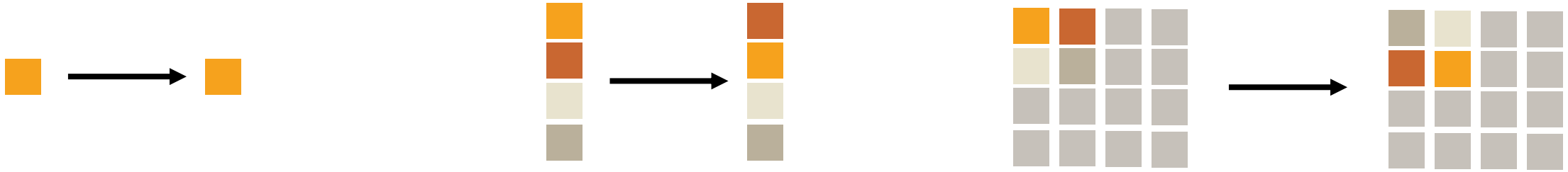
$\rho(1)$



- Now derive the representation for the group C4 of discrete rotations on a square grid of size 3x3

Symmetries: Sets & Graphs

- The group: $\mathfrak{G} = \mathbb{S}_n$, the group of permutations ("symmetric group")
- The domain: $\Omega = V$, the set of vertices (and possibly edges)
- Three kinds of graph features (representations of $\mathbb{S}_n$):



$$\rho_0(P_{12})s = 1 \cdot s$$

Scalar feature
e.g network output in graph classification

$$\rho_1(P_{12})v = P_{12}v$$

Vector feature
e.g. one feature per node

$$\rho_2(P_{12})M = P_{12}MP_{12}^T$$

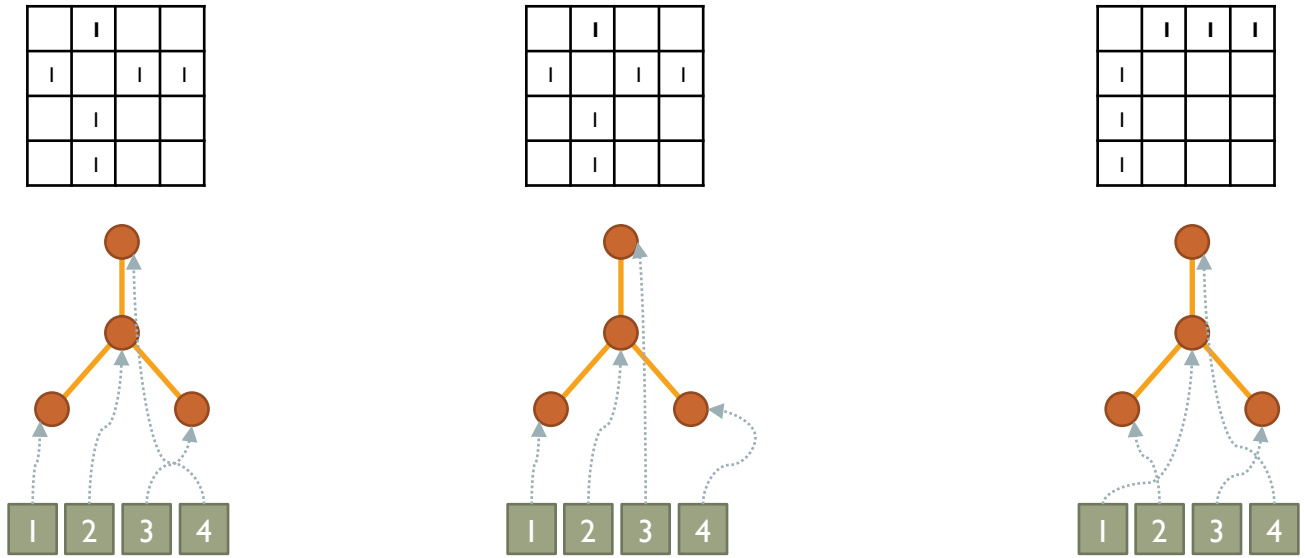
Tensor feature
e.g. one feature per edge or node pair

Maron, H., Ben-Hamu, H., Shamir, N., & Lipman, Y. Invariant and Equivariant Graph Networks. *ICLR 2019*.
Maron, H., Ben-Hamu, H., Serviansky, H., & Lipman, Y. Provably Powerful Graph Networks. *NeurIPS 2019*
Kondor, R., Son, H. T., Pan, H., Anderson, B., & Trivedi, S. *Covariant Compositional Networks For Learning Graphs, 2018*
De Haan, P., Cohen, T., Welling, M., Natural Graph Networks. *NeurIPS 2020*

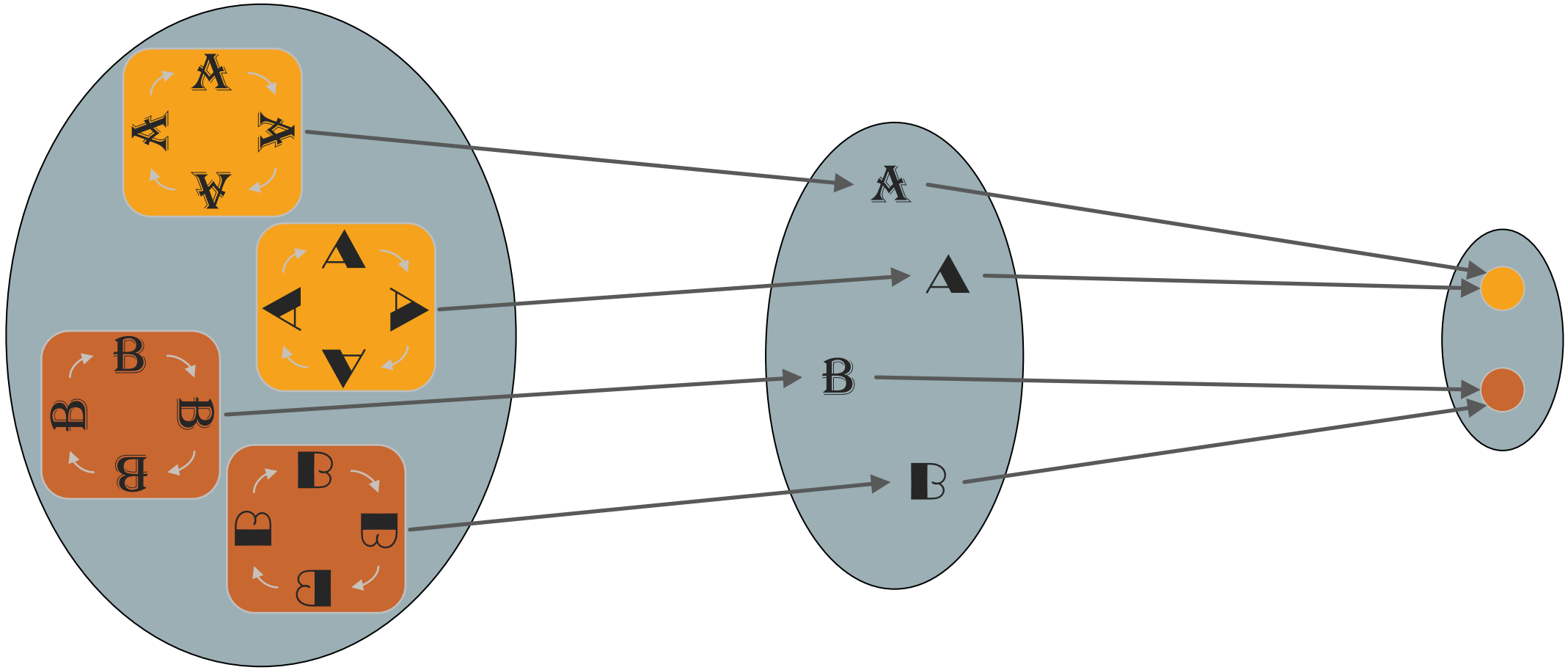
For each, verify that they are group representations

Symmetries of Graphs: Objects & their Descriptions

- A graph (or even a set) is an abstract object
  - All we have access to in practice is a description of it in computer memory
  - The description has properties (such as order) that are not intrinsic to the object
  - Usually we are interested in the symmetries of the description, rather than the symmetries of the object itself



Invariant Representations



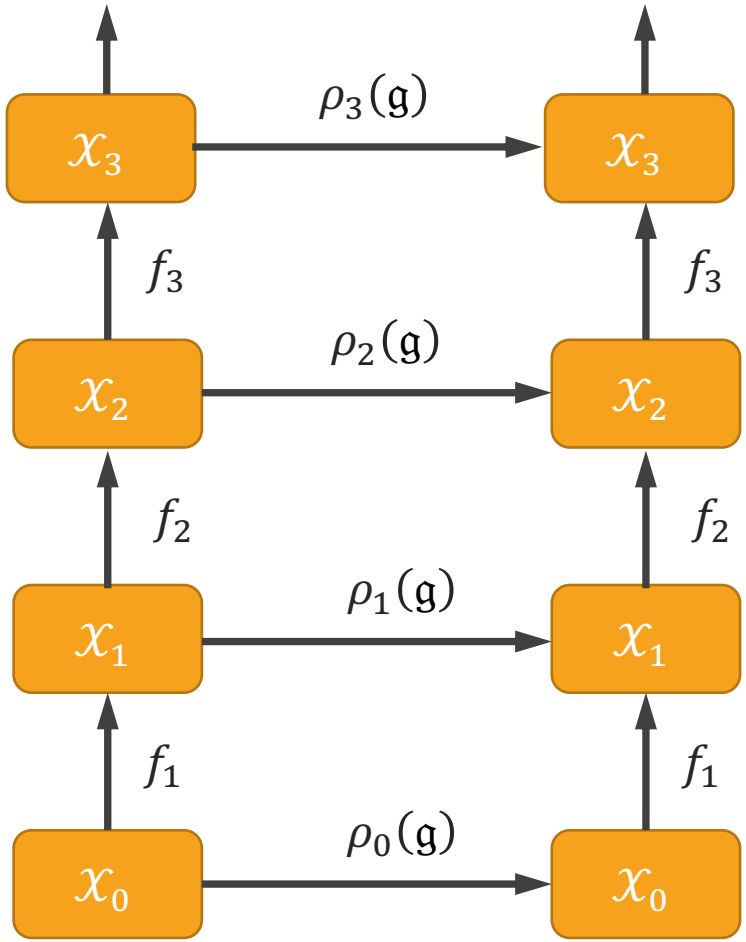
The Problem with Invariance in Deep Learning

- To recognize whole objects, we need to first recognize parts
  - This is why neural networks should be deep
- If we make the intermediate representations invariant, we lose critical information:

- The *relative pose* of object parts contains critical information [Hinton]

Hinton et al., Transforming Auto-Encoders, 2011

Equivariant Networks



Ingredients:

- Feature vector spaces $\mathcal{X}_i$
- Maps f_i between them ("Layers")
- A symmetry group $\mathfrak{G}$
- Group representations ρ_i of $\mathfrak{G}$ for each $\mathcal{X}_i$

Equivariance

$$f_i \circ \rho_{i-1}(\mathfrak{g}) = \rho_i(\mathfrak{g}) \circ f_i$$

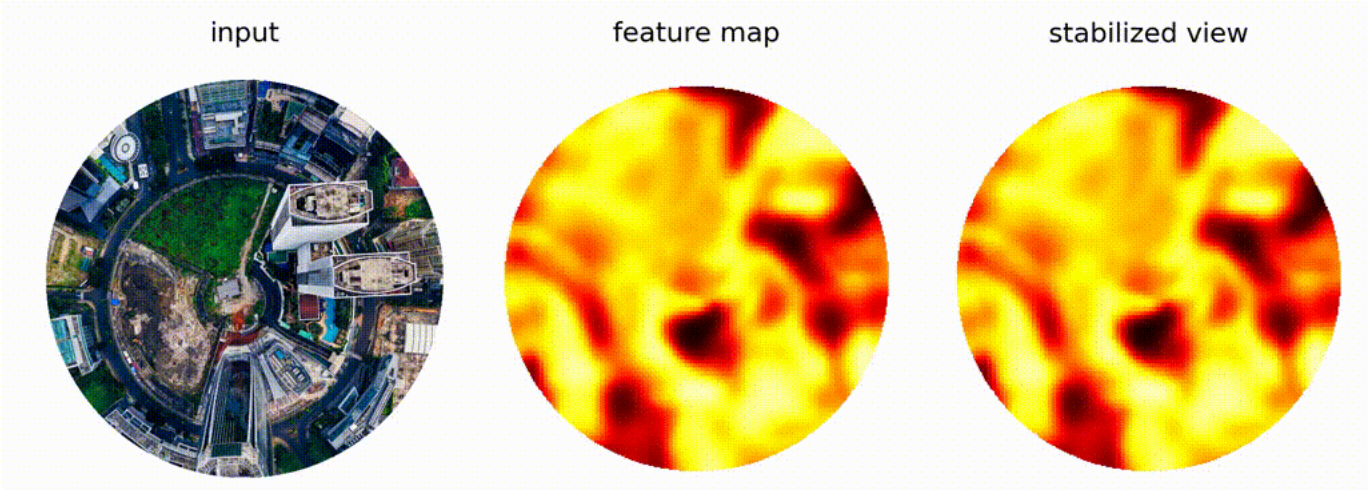
Show that if f_i are all equivariant, so is their composition

Wood, J., & Shawe-Taylor, J. (1996). Representation theory and invariant neural networks. *Discrete Applied Mathematics* , *69*(1), 33–60.

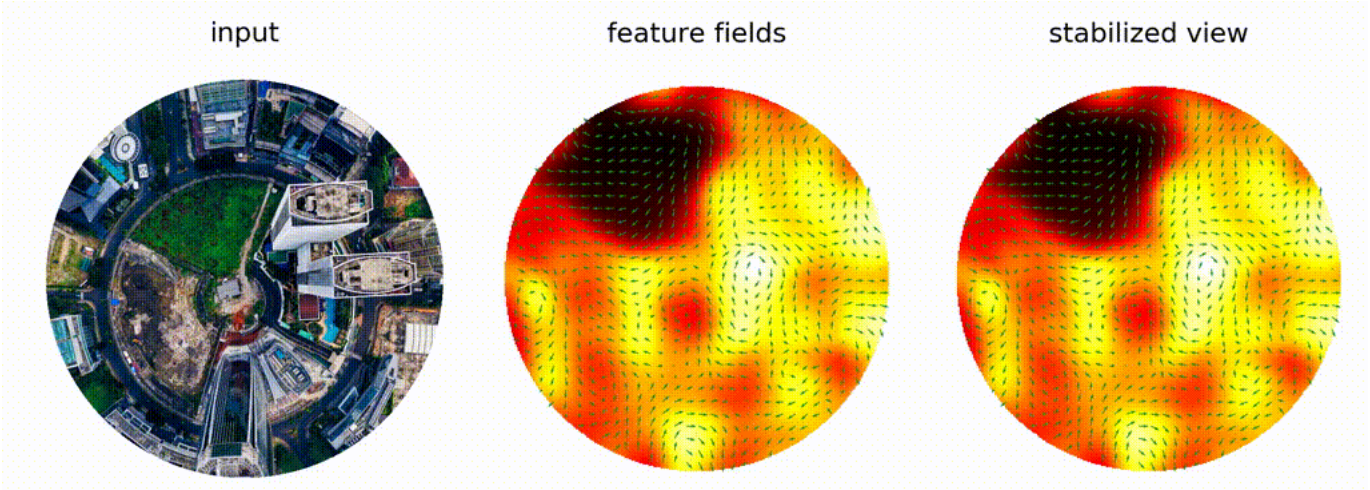
Translation Equivariance in CNNs

Rotation Equivariance in CNNs

CNN



G-CNN

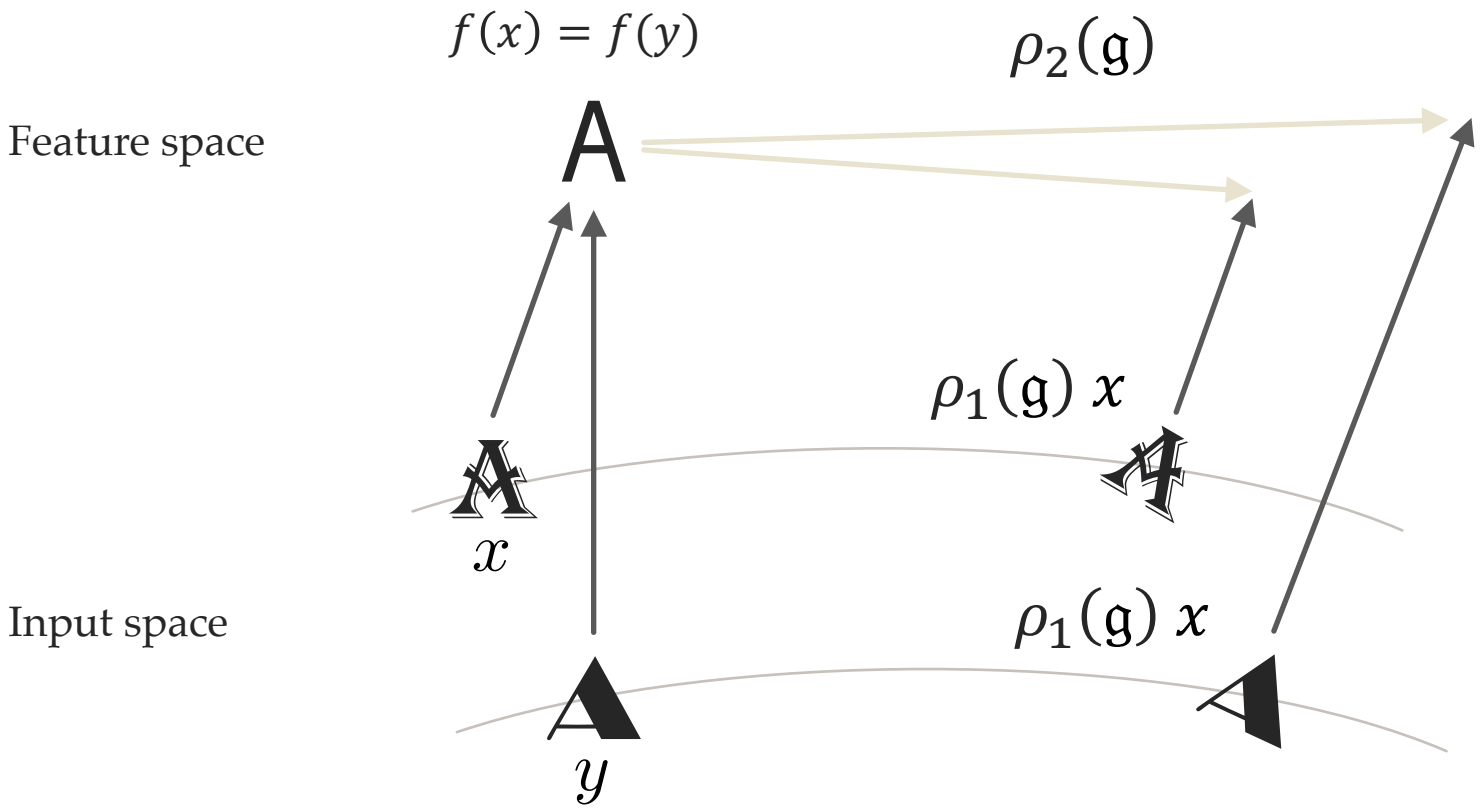


Show that convolution is translation equivariant, but not rotation equivariant.

Graphics courtesy of Gabriele Cesa

Equivariance as Symmetry-consistent generalization

Equivariance: $f(\rho_{i-1}(\mathfrak{g})\, x) = \rho_i(\mathfrak{g})\, f(\mathrm{x})$

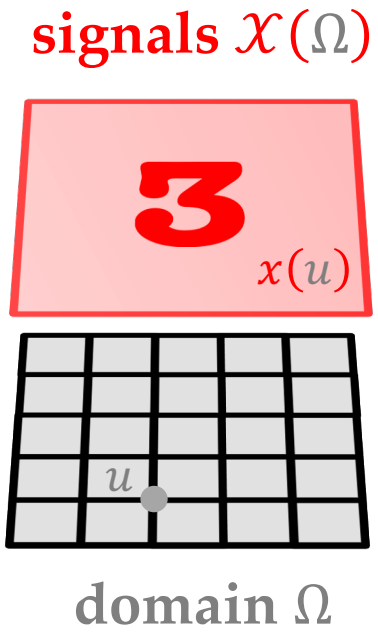


This cannot happen!
Equivariant net must generalize consistently across the whole orbit.

Equivariance vs Data augmentation

Property	**Train augmentation**	**Test augmentation**	**Equivariance**
Layerwise or whole-net constraint	Whole-net	Whole-net	Layerwise
Easy to implement, simple			
Guaranteed in/equivariance at training data			
Guaranteed in/equivariance at test data			
Works for large (e.g S_n) / infinite (e.g. SE(n)) groups			
Efficient at train time			
Efficient at test time			

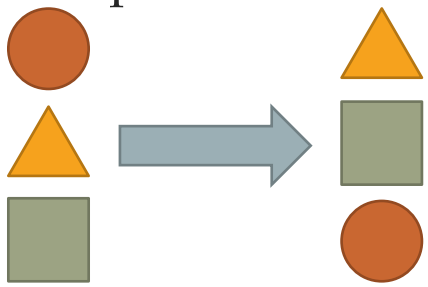
Invariant Function Classes



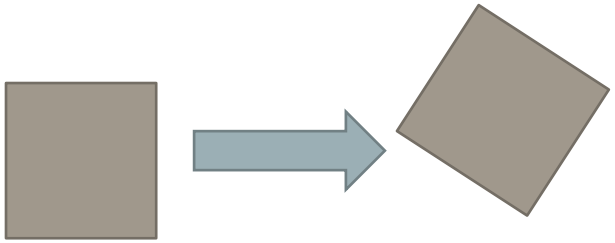
- ML Goal: Learn an unknown function $f^*: \mathcal{X} \to \mathbb{R}$
- Recap: Our input space is `opened' as

$$\mathcal{X} = \{x: \Omega \to \mathcal{C}\}$$

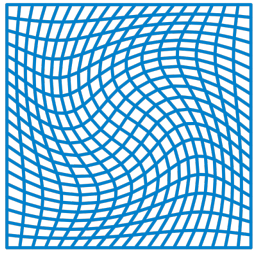
- Geometric domain Ω defines transformations $\mathfrak{g} : \Omega \to \Omega$
  - Examples:



Permutation



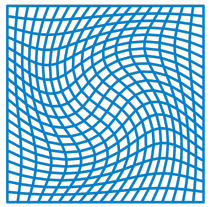
Euclidean isometry



Diffeomorphism of $\mathbb{R}^2$

  - Transformations form a *Group* $\mathfrak{G}$ by composition.

Invariant Function Classes



- These domain transformations are *lifted* to <u>linear</u> transformations (a.k.a. *group representations*):

$$\mathfrak{g} : \Omega \to \Omega \quad \Rightarrow \quad \mathfrak{g} : \mathcal{X}(\Omega) \to \mathcal{X}(\Omega)$$

$$(\mathfrak{g}\, x)(u) = x(\mathfrak{g}^{-1} u)$$

Invariant Function Classes

- Recall our learning setup:
  - $f^*: \mathcal{X}(\Omega) \to \mathbb{R}$ unknown function to be learnt.
  - $\mathcal{F}$: hypothesis class (parametrized e.g. with neural nets).
- <u>Promise</u>: f^* is $\mathfrak{G}$-invariant:

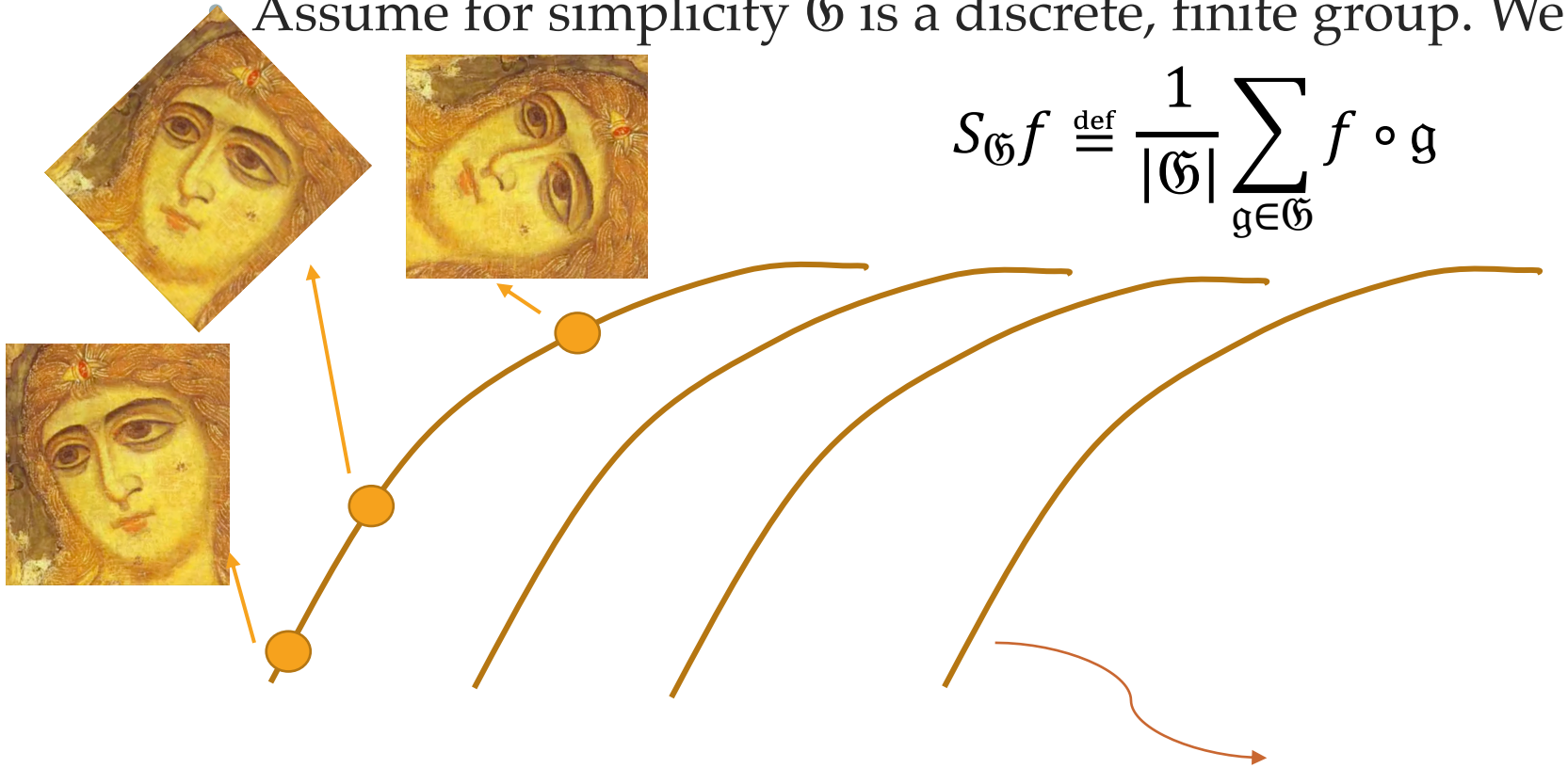
$$for\ all\ x \in \mathcal{X}(\Omega), \mathfrak{g} \in \mathfrak{G}, f^*(\mathfrak{g}.x) = f^*(x)$$

- Under this promise, we may ask:
  1. How to leverage it in our hypothesis class?
  2. It it generally sufficient to break the curse of dimensionality?

Invariant Function Classes

- We may first consider an abstract (non-algorithmic) option, using $\mathfrak{G}$-*smoothing operators:*
- Assume for simplicity $\mathfrak{G}$ is a discrete, finite group. We define

$$S_{\mathfrak{G}} f \stackrel{\text{def}}{=} \frac{1}{|\mathfrak{G}|} \sum_{\mathfrak{g} \in \mathfrak{G}} f \circ \mathfrak{g}$$



$$S_{\mathfrak{G}} f(x) = \frac{1}{|\mathfrak{G}|} \sum_{\mathfrak{g} \in \mathfrak{G}} f(\mathfrak{g}.x)$$

$S_{\mathfrak{G}}$ thus averages a function over group orbits.

Observe that $S_{\mathfrak{G}} f^* = f^*$.

Group Orbits $\mathfrak{G}.x = \{\mathfrak{g}.x; \mathfrak{g} \in \mathfrak{G}\}$

Invariant Function Classes

- Given a hypothesis class $\mathcal{F}$, we can make it $\mathfrak{G}$-invariant by applying the group smoothing operator:

$$S_{\mathfrak{G}}\mathcal{F} \stackrel{\text{def}}{=} \{S_{\mathfrak{G}}f; f \in \mathcal{F}\}$$

Learning under Invariance

- Approximation error not affected by $\mathfrak{G}$-*smoothing:*

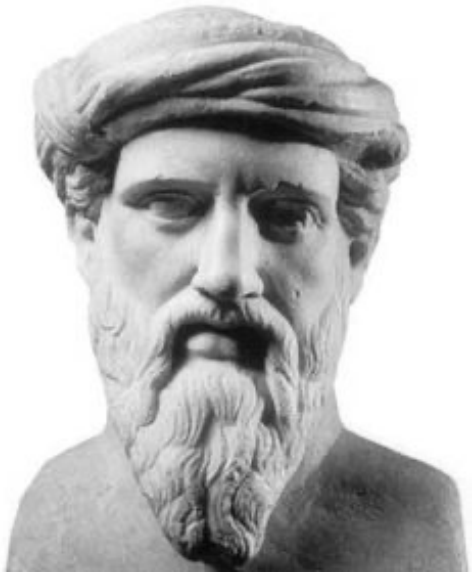
$$inf_{f\in\mathcal{F}} \|f - f^*\|^2 = inf_{f\in S_{\mathfrak{G}}\mathcal{F}} \|f - f^*\|^2$$

Indeed, $S_{\mathfrak{G}}\mathcal{F}$ is an *orthogonal projection* operator in $L^2(\mathcal{X})$

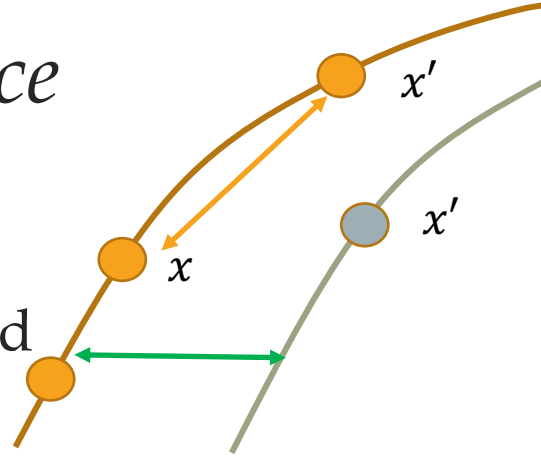
$$\|f - g\|^2 = \|S_{\mathfrak{G}}f - S_{\mathfrak{G}}g\|^2 + \|(I - S_{\mathfrak{G}})f - (I - S_{\mathfrak{G}})g\|^2$$

$$\|f - f^*\|^2 = \|S_{\mathfrak{G}}f - f^*\|^2 + \|(I - S_{\mathfrak{G}})f\|^2$$

- Statistical error reduced..
- .. But by *how* much?



On the Sample Complexity of Learning under Invariance



- We can consider, as before, the Lipschitz class $\mathcal{F}$, and its $\mathfrak{G}$-smoothed version.

$f \in \mathcal{F}: |f(x) - f(x')| \leq \beta \|x - x'\|$; $f \in S_{\mathfrak{G}}\mathcal{F}: |f(x) - f(x')| \leq \beta \, inf_{\mathfrak{g}} \|x - \mathfrak{g}.x'\|$

- **Theorem [BVB'21]:** Using a $\mathfrak{G}$-*invariant kernel ridge regression, the generalization error of learning a Lipschitz,* $\mathfrak{G}$-*invariant function* f^* *satisfies*

$$\mathbb{E}\mathcal{R}(\tilde{f}) \lesssim (|\mathfrak{G}| \, n)^{\frac{-1}{d}}$$

  - Group size $|\mathfrak{G}|$ can be exponential in dimension (local translations, or $\mathcal{S}_d$).
  - Rate is still cursed – indicating that group invariance is insufficient.
  - Gains are sharp w.r.t. non-invariant kernel [Caponetto & DeVito]

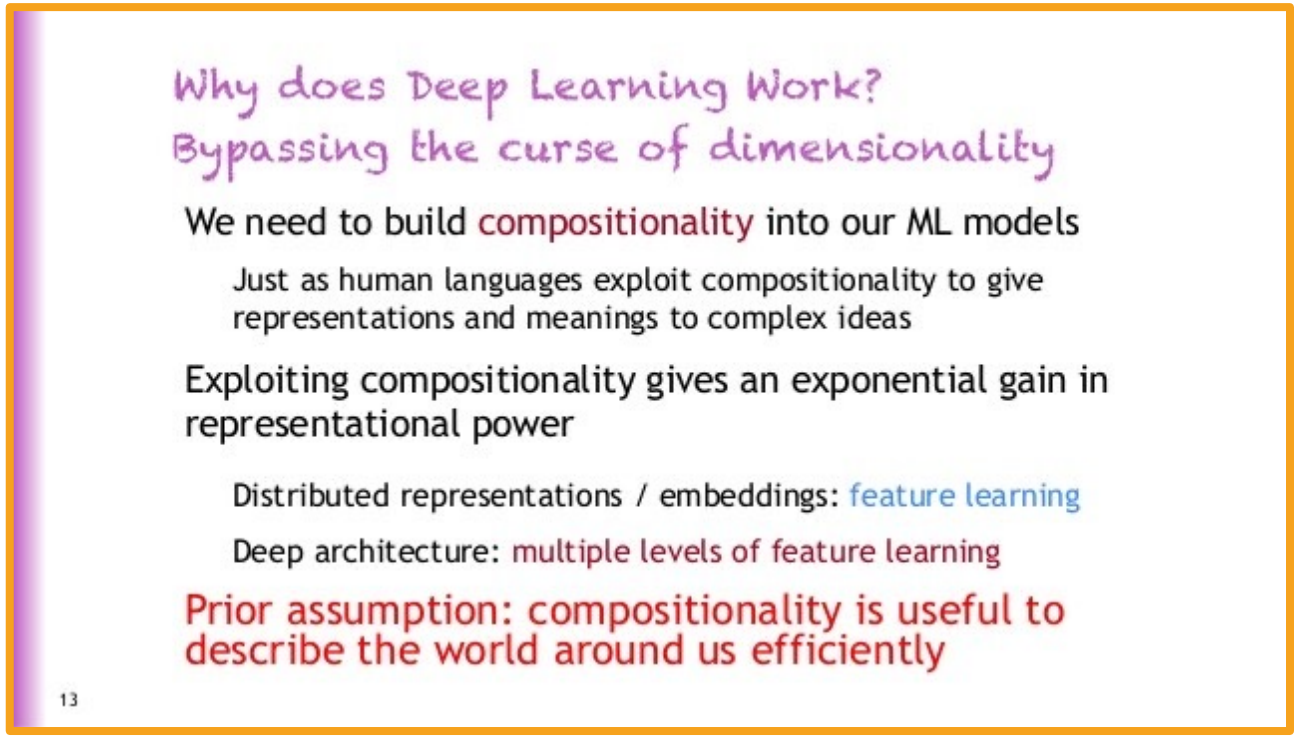
[Mei, Misiakiewicz, Montanari, 21][Bietti, Venturi, **B**.'21]

Conclusions so far

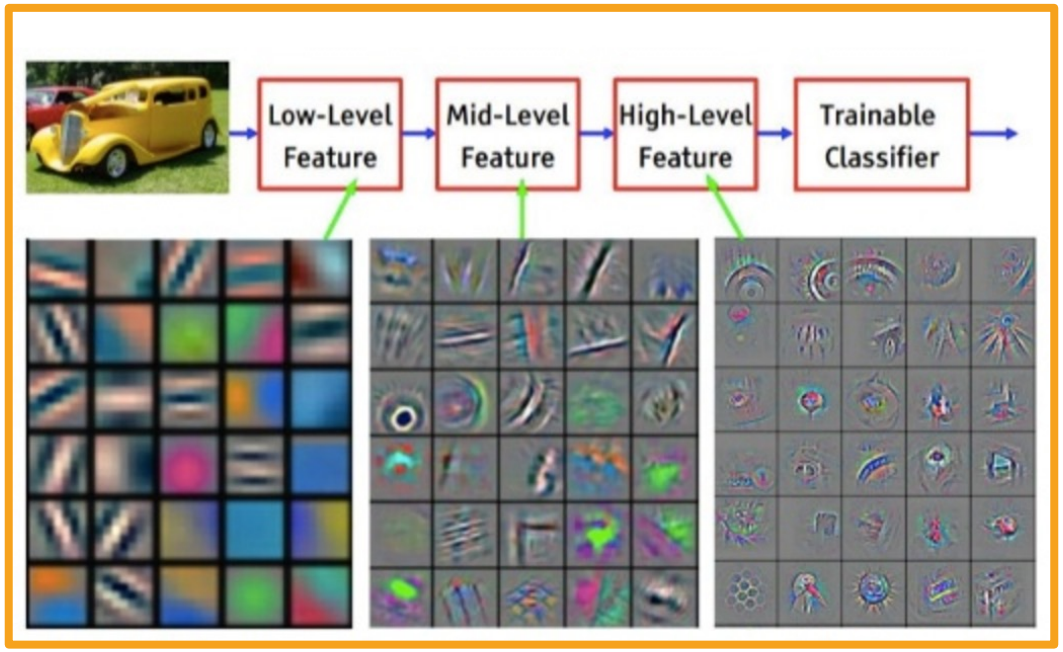
- Known global symmetries: using them in our hypothesis class is a *no-brainer:* guaranteed improvements in sample complexity.

- However, two important issues remain:
  1. Likely not sufficient to break the curse of dimensionality. What is missing?
  2. How to build such invariant classes efficiently? From first-principles?

Deep Learning "Inductive Bias": Compositionality

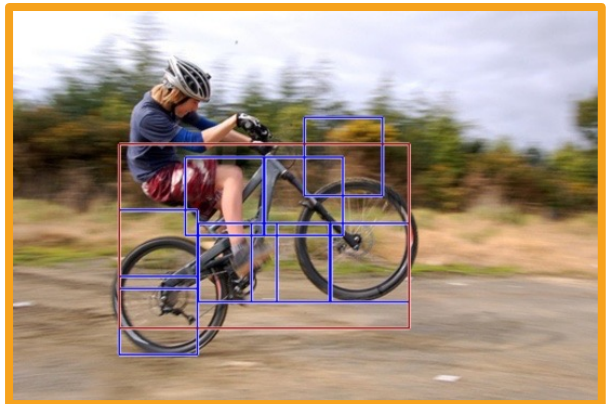


Yoshua Bengio

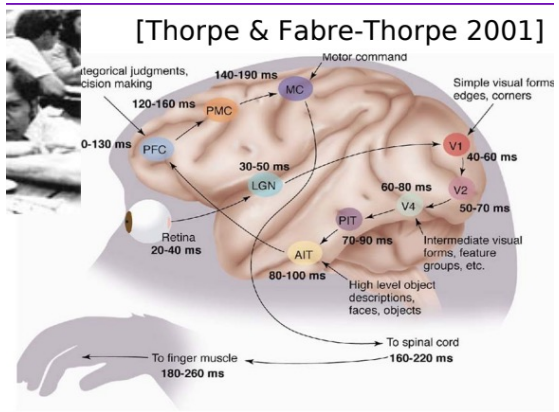


Yann LeCun

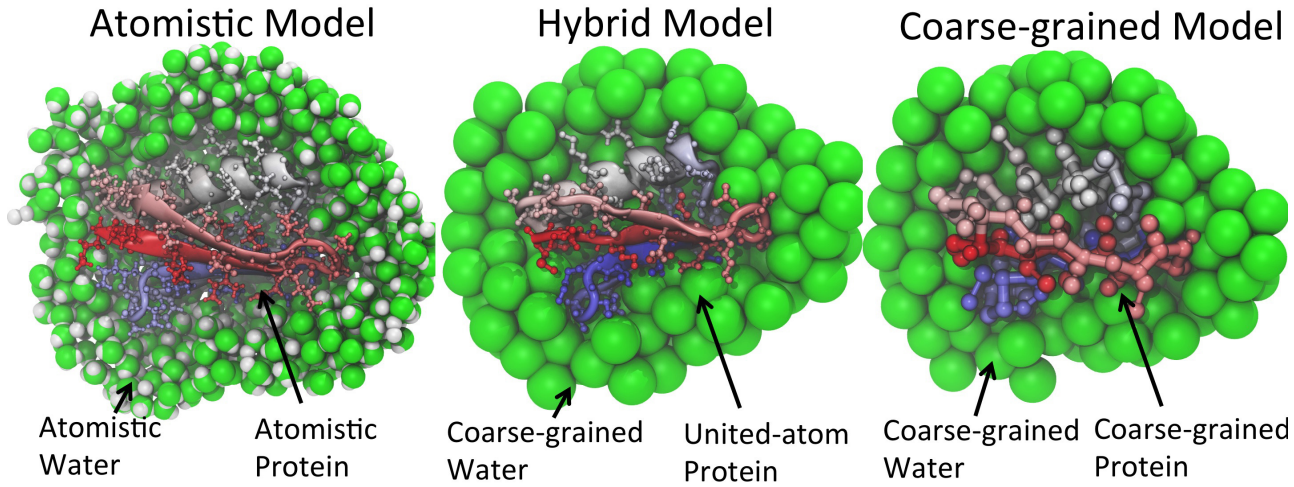
How to formalize this intuition?

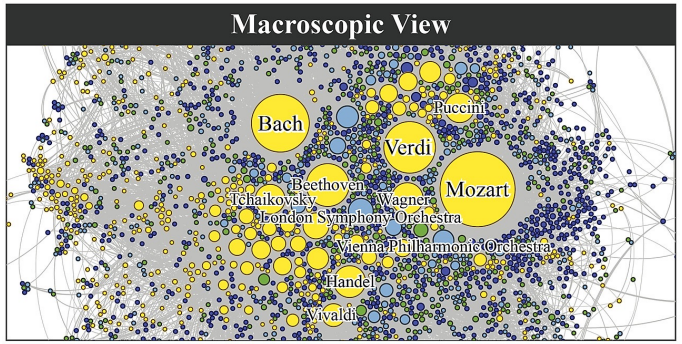


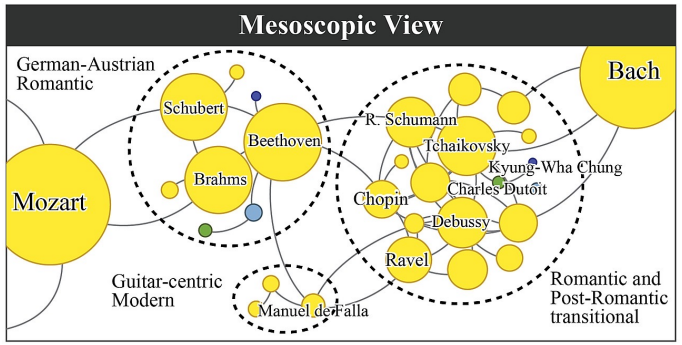
Deformable Parts Model, Felzenswalb et al.

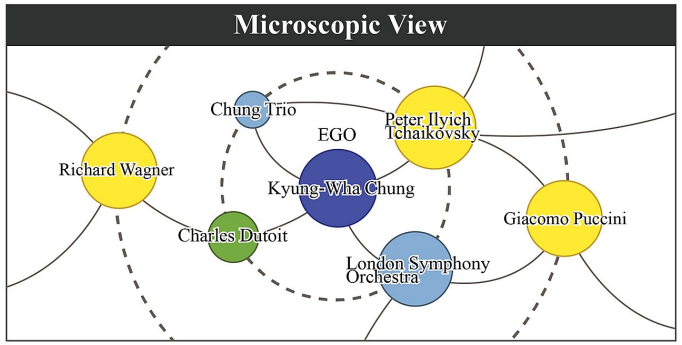


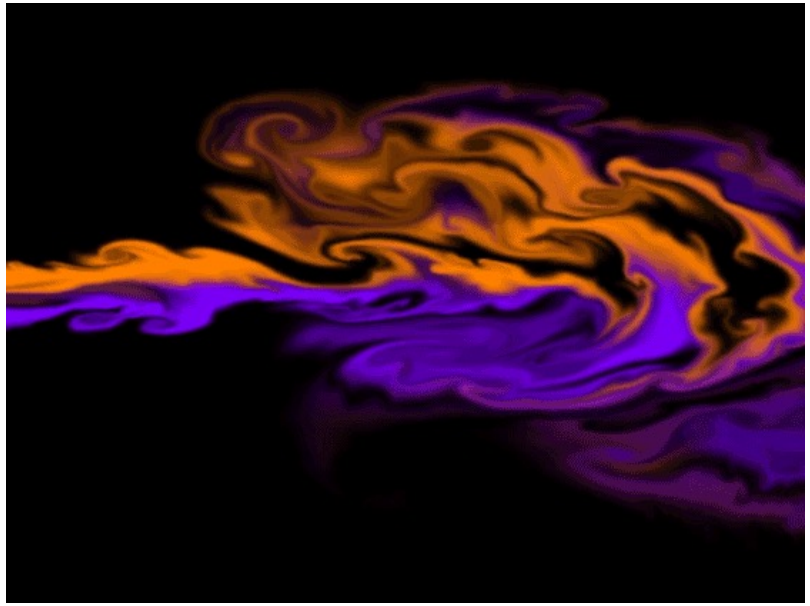
Multiscale Structures: Prevalent Across Science

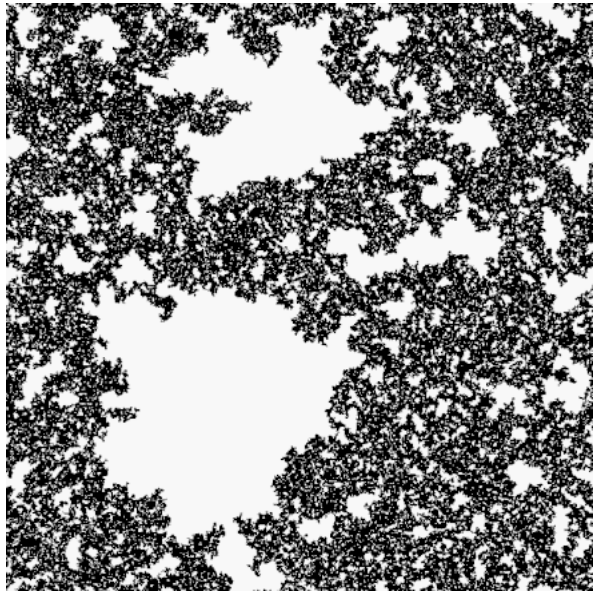




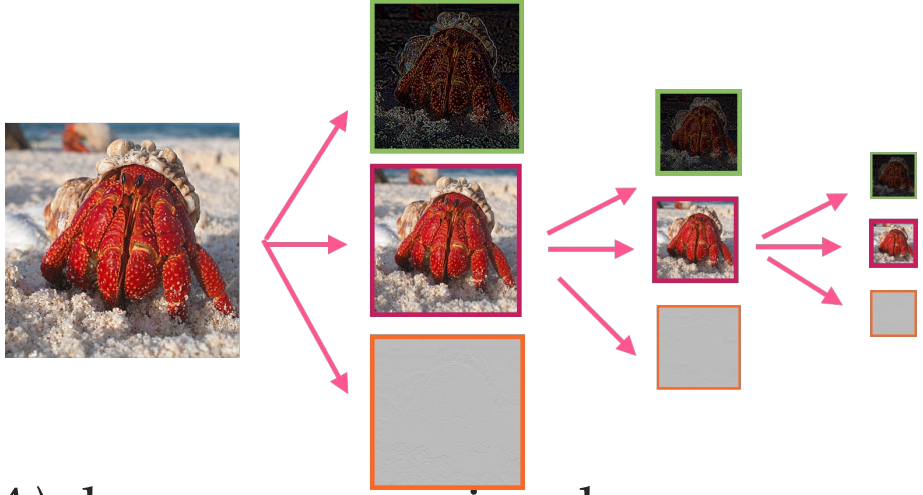








Basics of Multiresolution Analysis

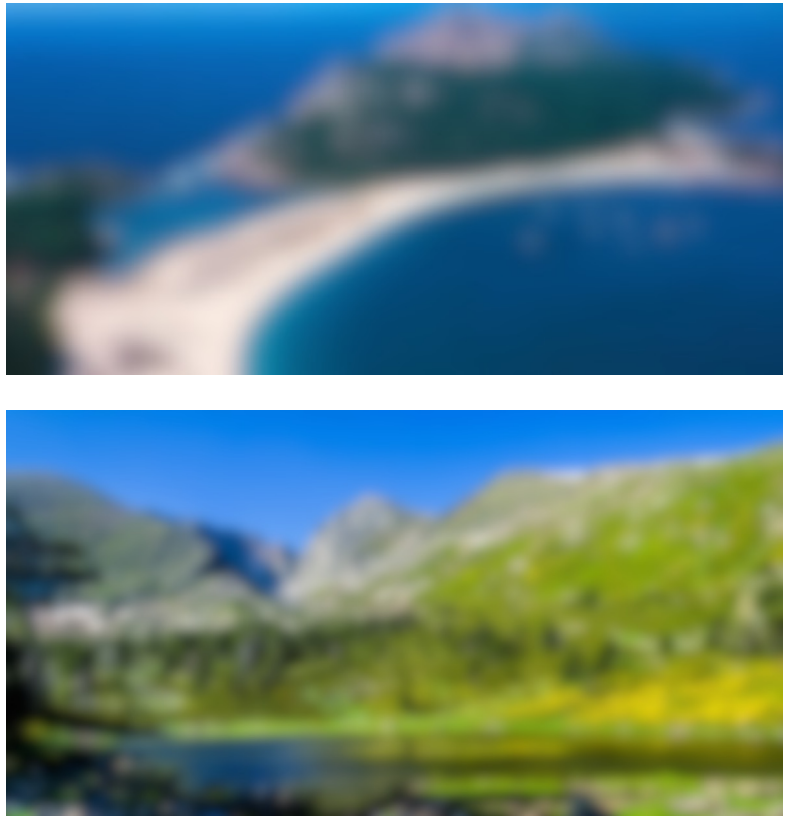


- For simplicity, fix Ω to be a 2D-grid.
- Roughly speaking, a *Multiresolution Analysis (MRA)* decomposes a signal $x \in L^2(\Omega)$ in terms of
  - a signal $\tilde{x} \in L^2(\tilde{\Omega})$ defined on a *coarser* grid $\tilde{\Omega}$, and
  - *details* at resolution Ω needed to reconstruct x from $\tilde{x}$.
- An MRA can be implemented with a *wavelet* filter bank [Meyer, Mallat], cf Lecture 7.
- Key property: filters are **localized** in space.
- Fundamental tool in signal processing that complements Fourier analysis, widespread applications (compression, denoising, inverse problems).

What is the link with our high-dimensional learning problem?

Scale Separation Prior

- Suppose first that the unknown target f^* is such that $f^*(x) \approx \tilde{f}^*(\tilde{x})$; e.g.



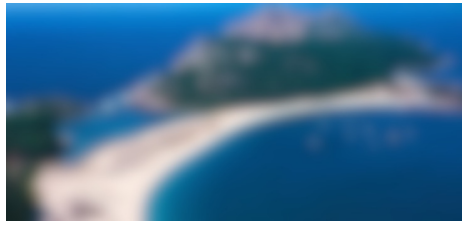
$\tilde{x}$

$$f^*(x) = P(y = \{'beach', 'mountain'\} | \, x)$$

Coarse scales dominate

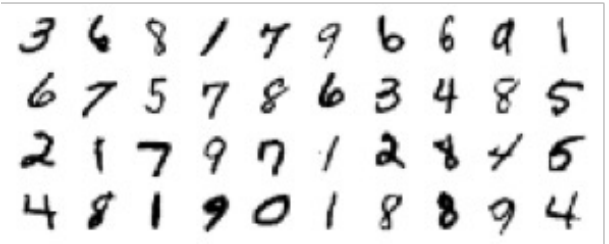
Scale Separation Prior

- Suppose first that the unknown target f^* is such that $f^*(x) \approx \tilde{f}^*(\tilde{x})$; e.g.



$$f^*(x) = P(y = \{'beach', 'mountain'\} \mid x)$$

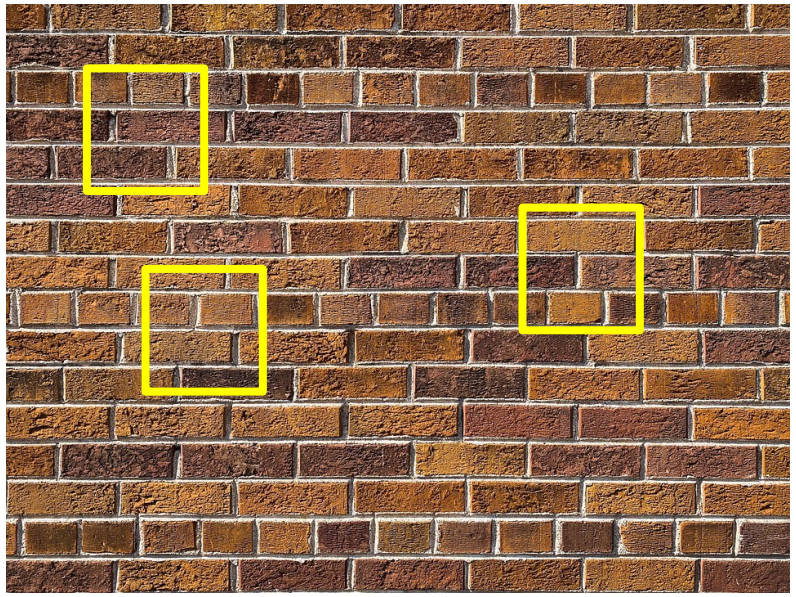
- Since $\dim(\mathcal{X}) \propto |\Omega|$ and $|\tilde{\Omega}| \ll |\Omega|$, coarsening reduces curse of dimensionality in this case.
- Strong assumption in general:
  - Fine-scale details are necessary!

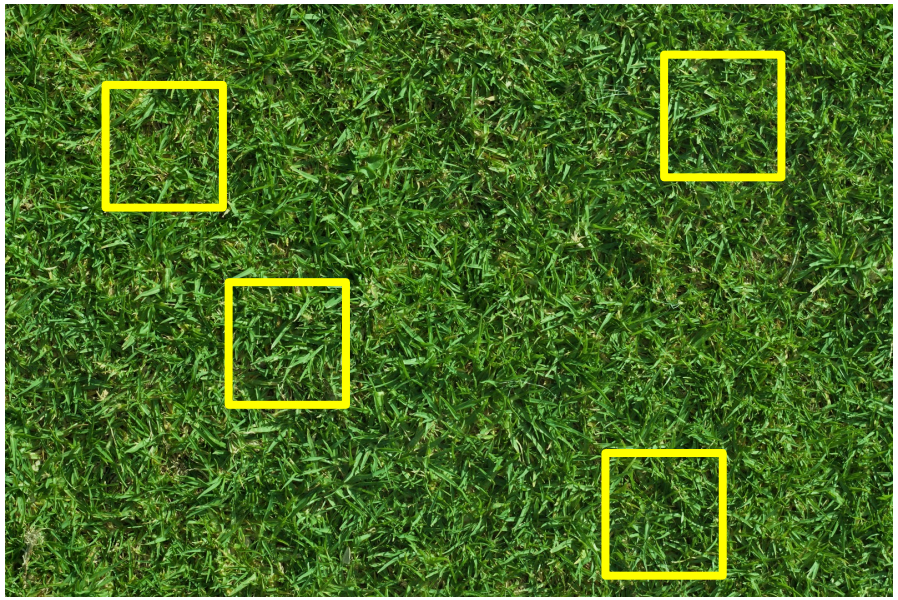


Scale Separation Prior

- Alternatively, suppose target f^* is such that $f^*(x) \approx \sum_u g(x_u)$.

x_u: image patch centered at u.



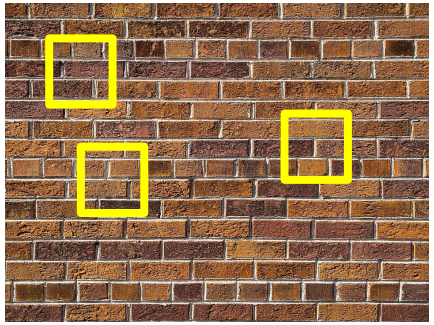


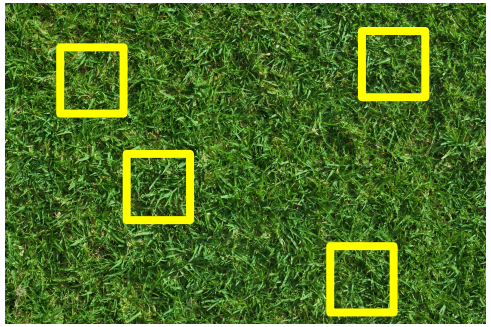
Local scales dominate

Scale Separation Prior

- Alternatively, suppose target f^* is such that $f^*(x) \approx \sum_u g(x_u)$.

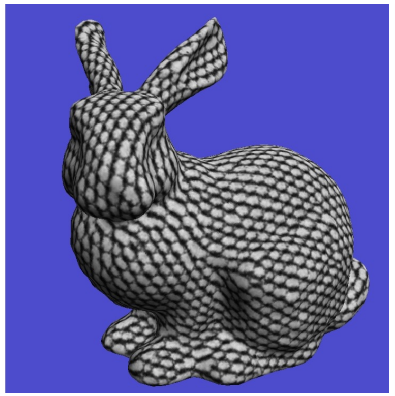
x_u: image patch centered at u.

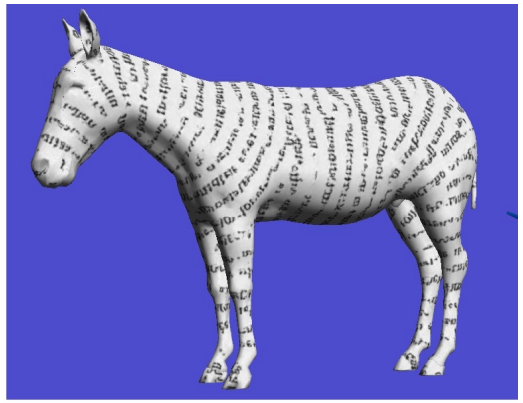


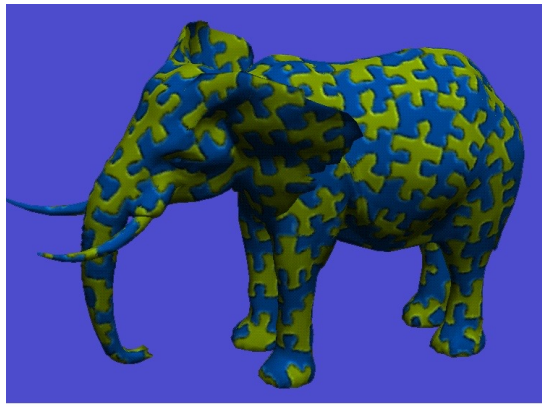


- In that case, curse is also avoided, since relevant dimension is the *patch* dimension [Favero et al'21,Bietti'22].
- Local Markov Random Fields, weakly interacting systems.
- Strong assumption in general:

[Turk,'01]

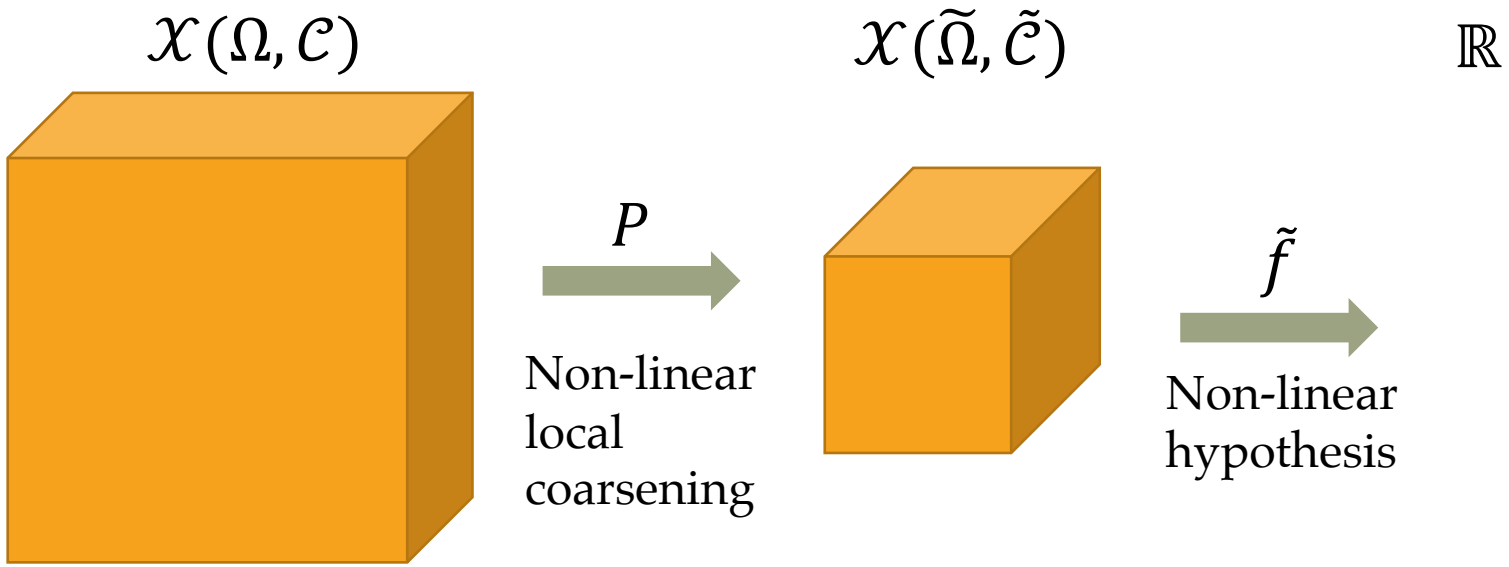






Scale Separation Prior

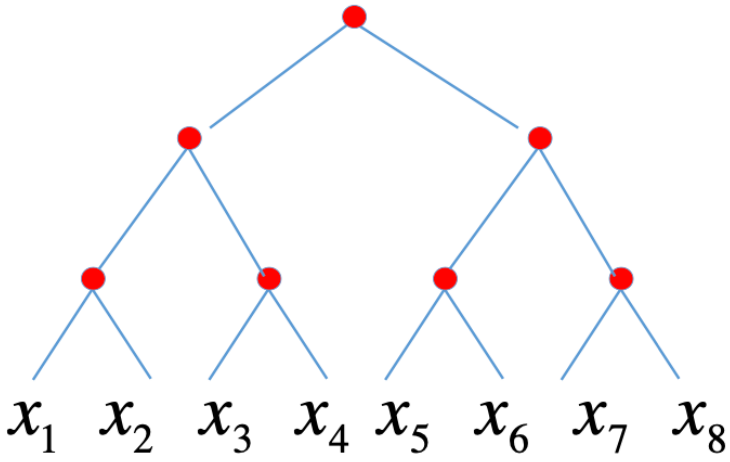
- More generally, we consider a compositional model of the form

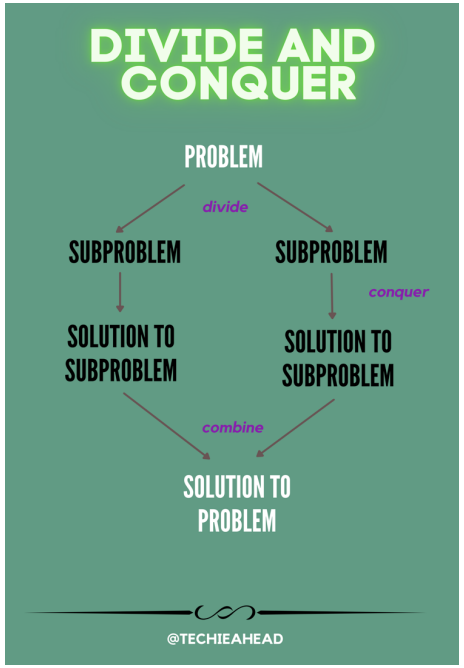


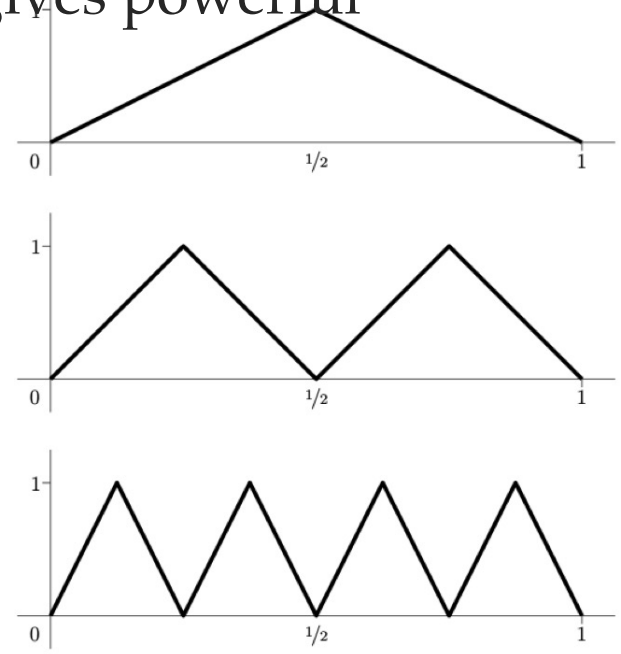
- Both P and $\tilde{f}$ learnt and non-linear.
- When is composition more efficient?

Benefits of composition

- Provable approximation, computational & estimation benefits in specific contexts.
- General structure of multiscale hypothesis space $\mathcal{F}$ still poorly understood theoretically.
- Yet, combining scale separation with $\mathfrak{G}$-symmetry prior gives powerful model from first-principles.







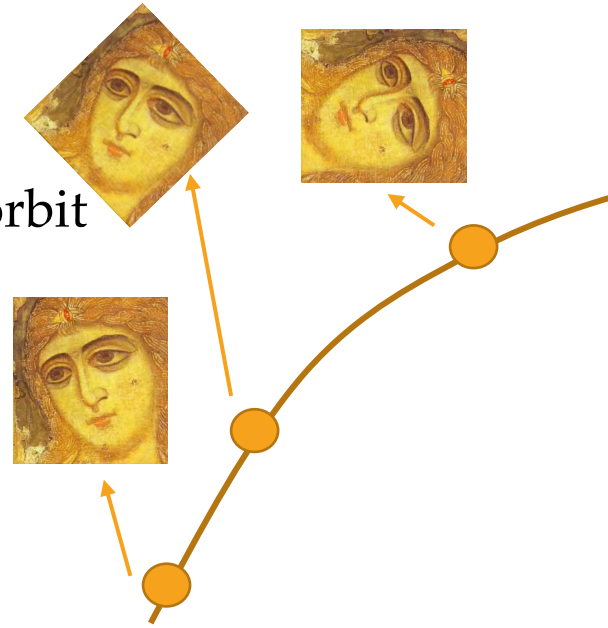
[Telgarsky,'15] [Cohen&Shashua'16][Poggio]

Combining Invariance with Scale Separation

- Function Hypothesis *wish-list:* $\mathfrak{G}$-invariant, multiscale structure, rich approximation.
- Architecture built from first-principles that satisfies these?

- Start with *linear* $\mathfrak{G}$-invariants: How many are there?
- $f(x) = \frac{1}{|\mathfrak{G}|}\sum_{\mathfrak{g}} f(\mathfrak{g}.x) = f\left(\frac{1}{|\mathfrak{G}|}\sum_{\mathfrak{g}} \mathfrak{g}.x\right) = f(\bar{x})$
- Group average $Ax \stackrel{\text{def}}{=} \bar{x} = \frac{1}{|\mathfrak{G}|}\sum_{\mathfrak{g}} \mathfrak{g}.x$
- Loses a lot of information!
- How to complement?

$\bar{x}$: average over orbit



Combining Invariance with Scale Separation

- Consider instead *linear $\mathfrak{G}$-equivariants:* if $\mathfrak{G}$ is acting on both $\mathcal{X}$ and $\mathcal{Y}$, a mapping $B: \mathcal{X} \to \mathcal{Y}$ *is $\mathfrak{G}$-equivariant* if for all $\mathfrak{g} \in \mathfrak{G}$, $x \in \mathcal{X}$,

$$B(\mathfrak{g}.x) = \mathfrak{g}.B(x)$$

  - We will see many examples throughout the course (convolutions, graph diffusions, etc.)
- How to extract invariants from equivariant features?
- Directly composing a linear equivariant with linear invariant does not work (**why?**).
- Alternative?

Combining Invariance with Scale Separation

- Introduce an *element-wise* non-linear function:

$$\boldsymbol{\rho}: \mathcal{X} \to \mathcal{X}, with\ \boldsymbol{\rho} x(u) = \rho(x(u))$$

- Composing a linear $\mathfrak{G}$-*equivariant* B with an element-wise non-linear function $\boldsymbol{\rho}$ yields a non-linear $\mathfrak{G}$-equivariant $\tilde{B} = \boldsymbol{\rho} \circ B$

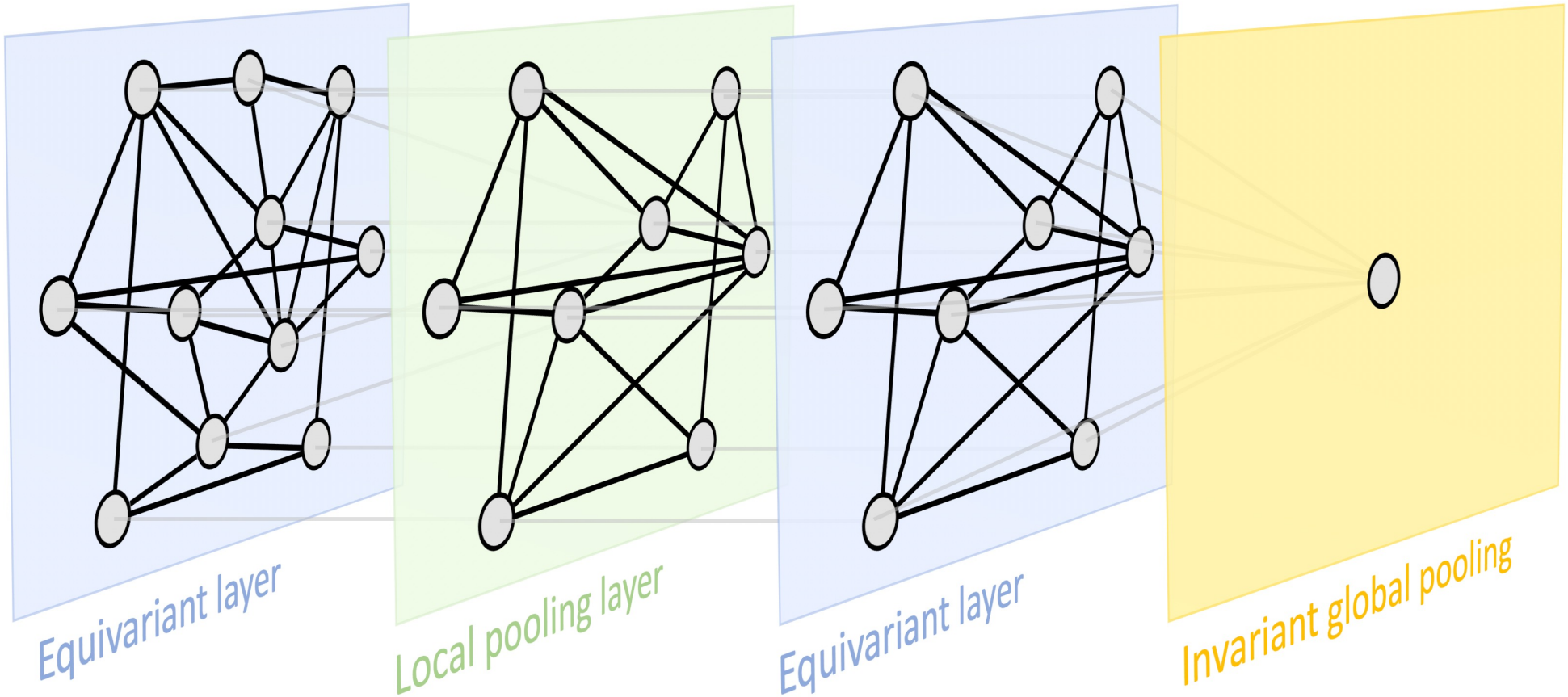
- Moreover, if B is *local*, $\tilde{B}$ is also local.

The Geometric DL Blueprint

- This suggests a constructive approach to build rich invariants with multiscale structure, with building blocks:
  - *Linear $\mathfrak{G}$-equivariant layer* $B : \mathcal{X}(\Omega, \mathcal{C}) \to \mathcal{X}(\Omega', \mathcal{C}')$, satisfying $B(\mathfrak{g}.x) = \mathfrak{g}.B(x)$ for all $\mathfrak{g} \in \mathfrak{G}$ and $x \in \mathcal{X}(\Omega, \mathcal{C})$.
  - *Nonlinearity* $\sigma : \mathcal{C} \to \mathcal{C}'$ applied element-wise as $\big(\boldsymbol{\sigma}(x)\big)(u) = \sigma(x(u))$.
  - *Local pooling (coarsening)* $P : \mathcal{X}(\Omega, \mathcal{C}) \to \mathcal{X}(\Omega', \mathcal{C})$, such that $\Omega' \subseteq \Omega$.
  - *$\mathfrak{G}$-invariant layer (global pooling)* $A : \mathcal{X}(\Omega, \mathcal{C}) \to \mathcal{Y}$, satisfying $A(\mathfrak{g}.x) = A(x)$for all $\mathfrak{g} \in \mathfrak{G}$ and $x \in \mathcal{X}(\Omega, \mathcal{C})$.

- These blocks can be defined under very mild conditions on the geometric domain Ω.

The GDL Blueprint



The GDL Blueprint

Architecture	**Domain Ω**	**Symmetry Group $\mathfrak{G}$**
CNN	Grid	Translation
Spherical CNN	Sphere / $\mathrm{SO}(3)$	Rotation $\mathrm{SO}(3)$
Intrinsic / Mesh CNN	Manifold	Isometry $\mathrm{Iso}(\Omega)$ / Gauge Symmetry $\mathrm{SO}(2)$
GNN	Graph	Permutation Σ_n
Deep Sets	Set	Permutation Σ_n
Transformer	Complete Graph	Permutation Σ_n
LSTM	1D Grid	Time warping