# Automatic Symmetry Discovery with Lie Algebra Convolutional Network

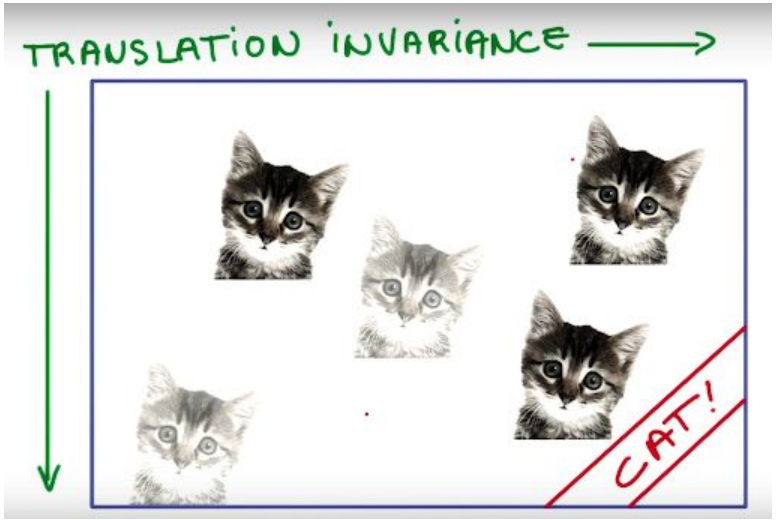
Nima Dehmamy[1], Robin Walters[2], Yanchen Liu[2],
Dashun Wang[1], Rose Yu[2,3]
*1. Northwestern University, 2. Northeastern University , 3. UCSD*

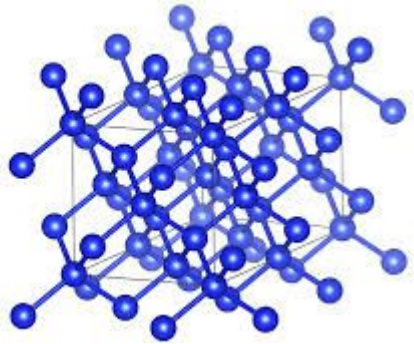
NeurIPS 2021
https://proceedings.neurips.cc/paper/2021/file/148148d62be67e0916a833931bd32b26-Paper.pdf

# Symmetry

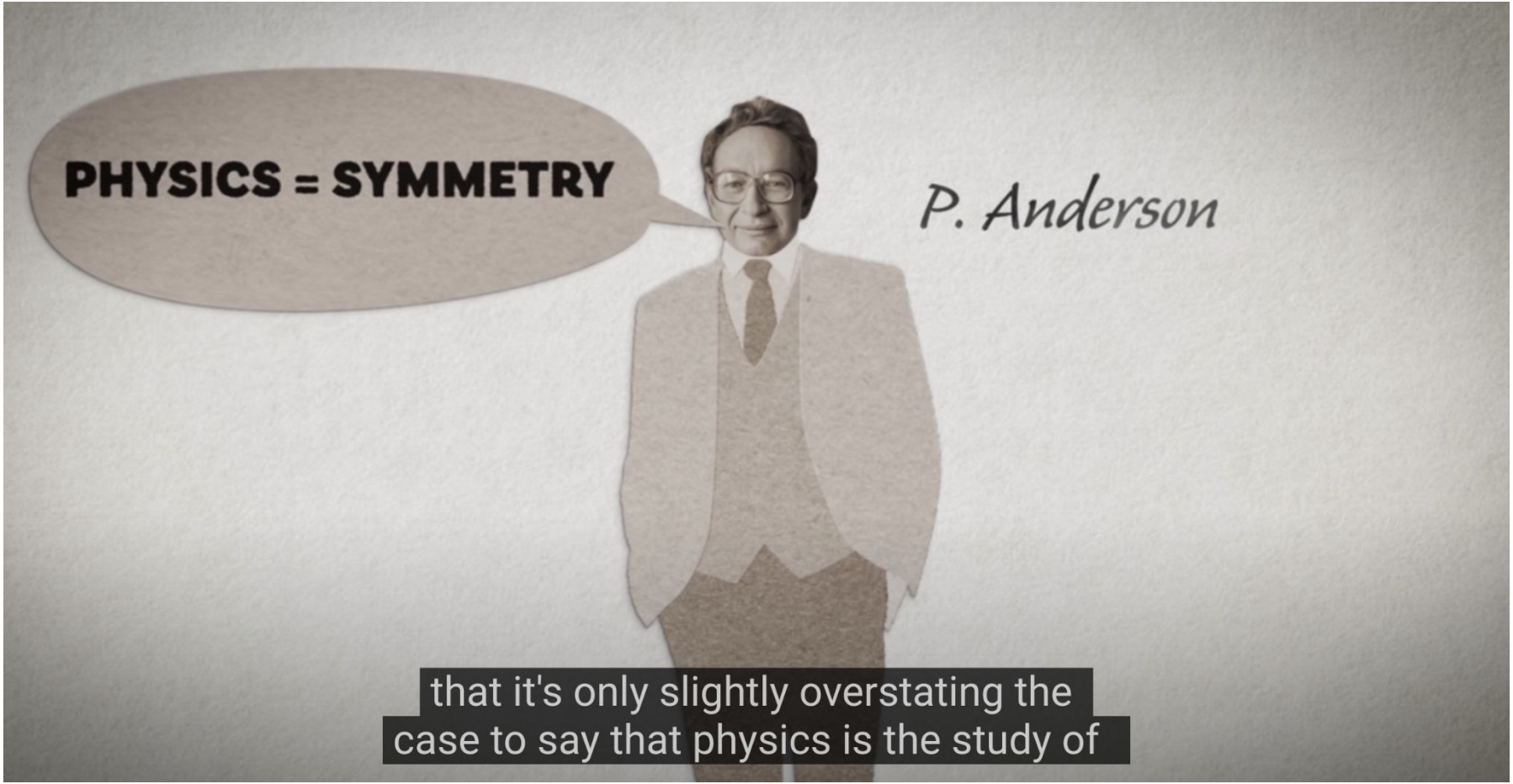
Many real-world domains have “symmetries”

- In natural images (shifting doesn’t change the objects)
- Crystals



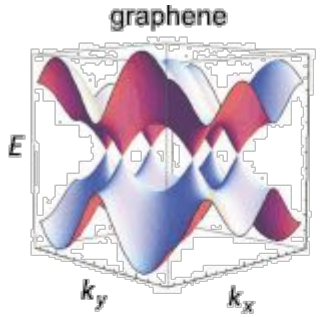



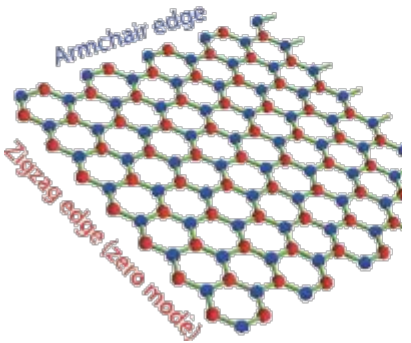
# Michael Bronstein Keynote ICLR 2021



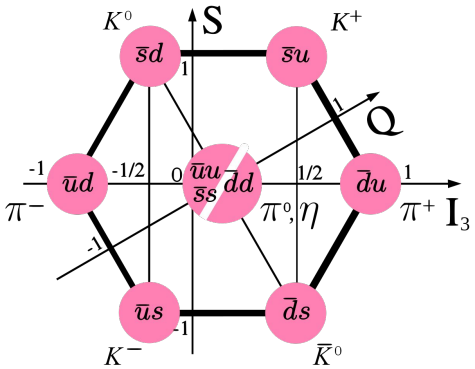
# Symmetry in Physics

- **All known forces of nature are a result of symmetries**
  - Electroweak: U(1)xSU(2)
  - Strong nuclear (QCD): SU(3)
  - Gravity: SO(3,1)
- **Physical states (e.g. electrons on graphene)**



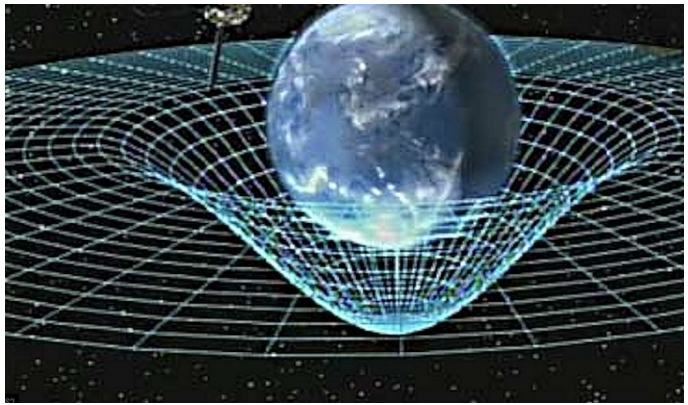




## Ex: Particle Physics



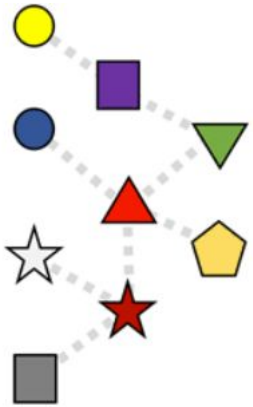

Wikimedia

## General Relativity

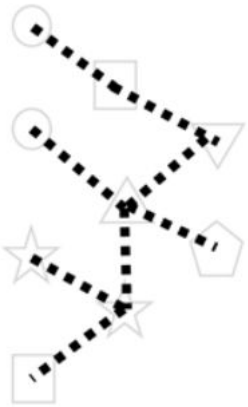


# Philosophy: Reductionism in science

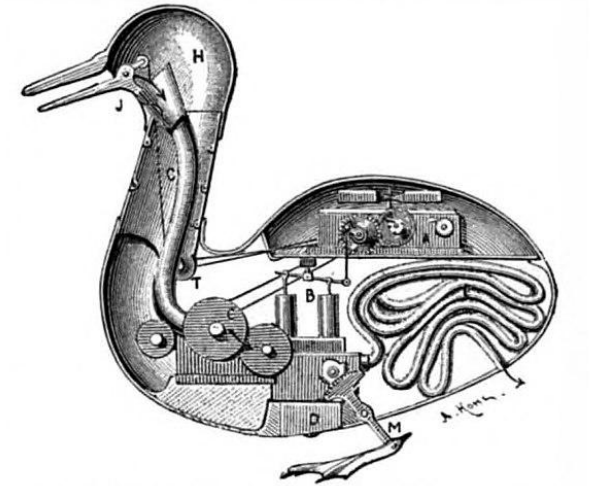
Break down to basic parts



Reductionist Approach



Holistic Approach




INTERIOR OF VAUCANSON'S AUTOMATIC DUCK.

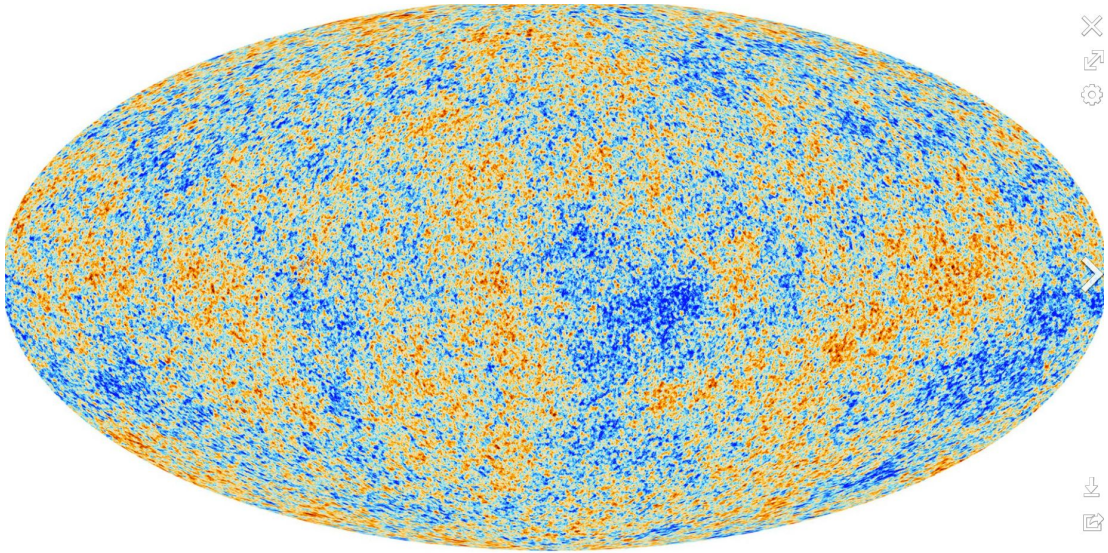
*A*, clockwork; *B*, pump; *C*, mill for grinding grain; *F*, intestinal tube; *J*, bill; *H*, head; *M*, feet.

Geza Szollosi

https://complementarytraining.net/strength-training-categorization/figure-8-complementary-aspects-of-reductionist-approach-vs-holistic-system-approach/

# Symmetry in Physical Laws

**Ex: Cosmological principle** states: 'Viewed on a sufficiently large scale, the properties of the universe are the same for all observers.'



The Cosmic Microwave Background as seen from the Planck satellite. Credit: ESA
https://www.esa.int/ESA_Multimedia/Images/2013/03/Planck_CMB

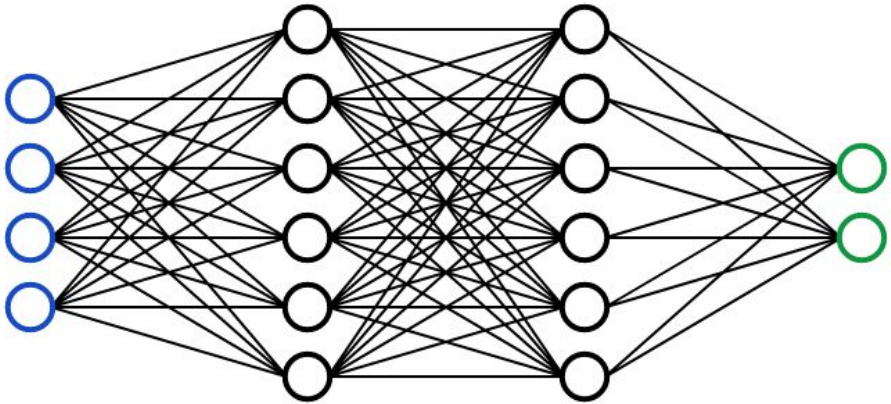
More details

*(Credit: NASA; ESA; G. Illingworth, D. Magee, and P. Oesch, University of California, Santa Cruz; R. Bouwens, Leiden University; and the HUDF09 Team)*

# Using AI in Science

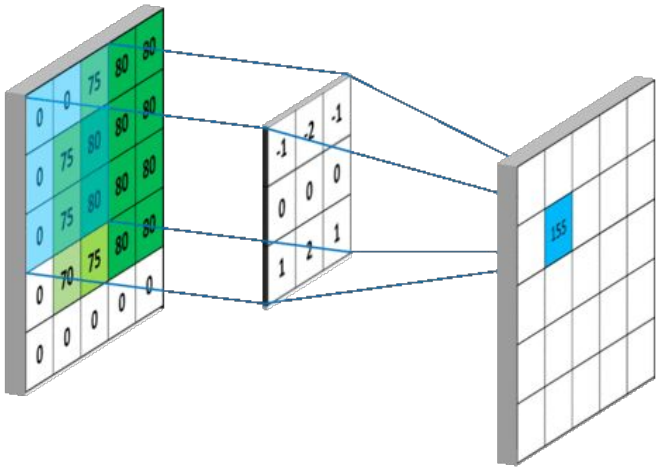
**Laws of physics** look the same everywhere

- How do we design AI architectures which use the same **thinking process as scientists**?
- How do we encode the assumption that **laws of nature are minimal** and **look the same everywhere**?



# Symmetry in Machine Learning

Convolutional neural networks use translation (shift) symmetry




https://www.pinterest.com/pin/628885535445259486/

https://tvirdi.github.io/2017-10-29/cnn/

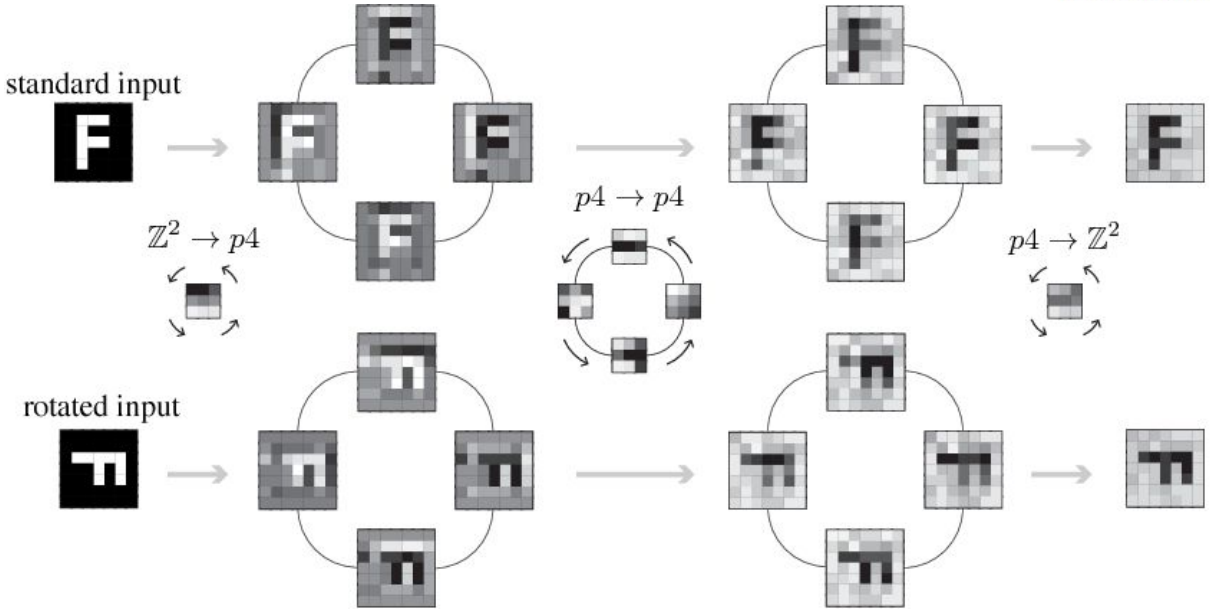
# General symmetry encoding neural nets

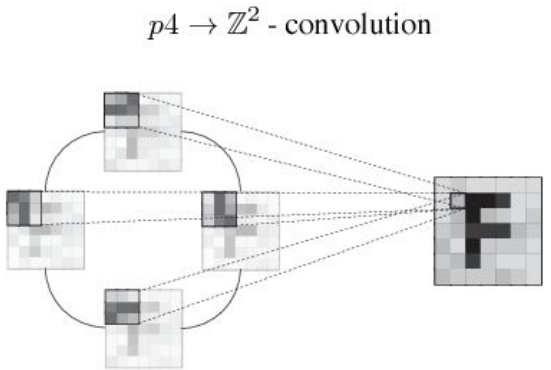
**Group equivariant convolutional neural networks:** In recent years, a number of other symmetry encoding neural networks have been invented

**Group Equivariant Convolutional Networks**

**Taco S. Cohen**
University of Amsterdam

**Max Welling**
University of Amsterdam
University of California Irvine
Canadian Institute for Advanced Research

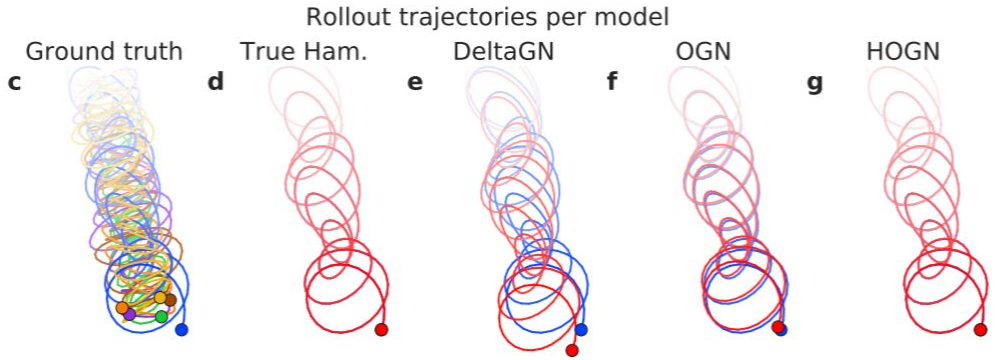



https://medium.com/intel-student-ambassadors/rotation-equivariant-convolutional-neural-network-16e279924fa4

# Benefits of Encoding Symmetries

- Encoding symmetries into the architecture can reduce data requirements by
  - reducing **learnable parameters** (e.g. CNN, Group equivariant CNN), and
  - may help with **generalization**




Sanchez-Gonzalez et al 2019 Hamiltonian GN

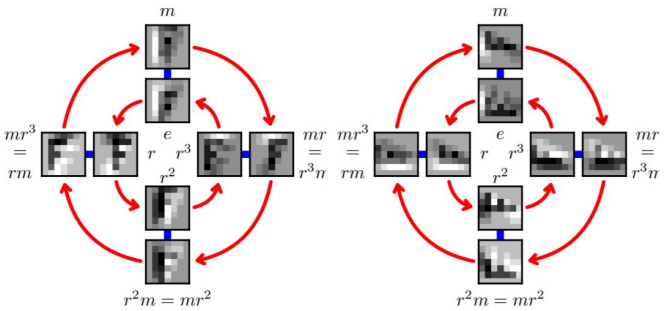


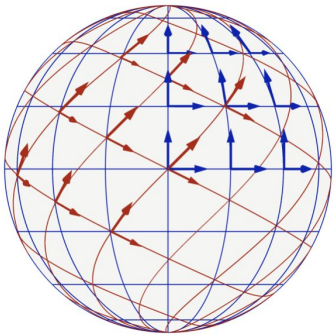

*Figure 2.* A p4m feature map and its rotation by $r$.

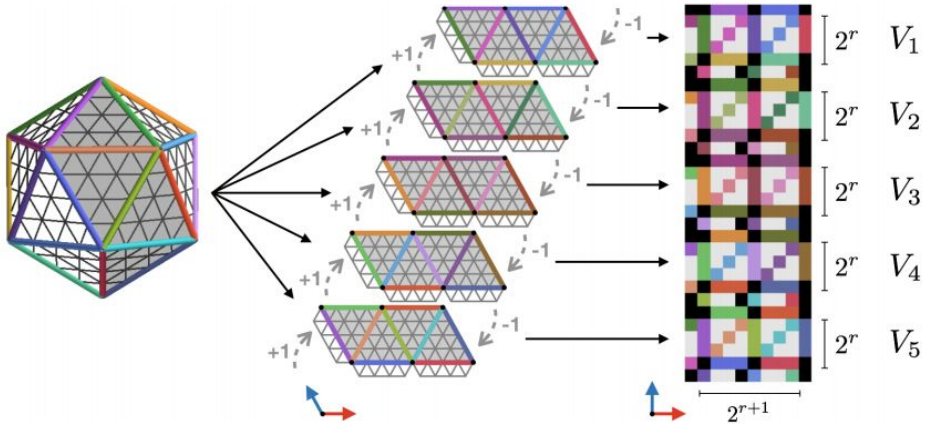
Cohen & Welling 2016

# Existing Literature for Continuous Symmetries

To deal with continuous symmetries, current methods use

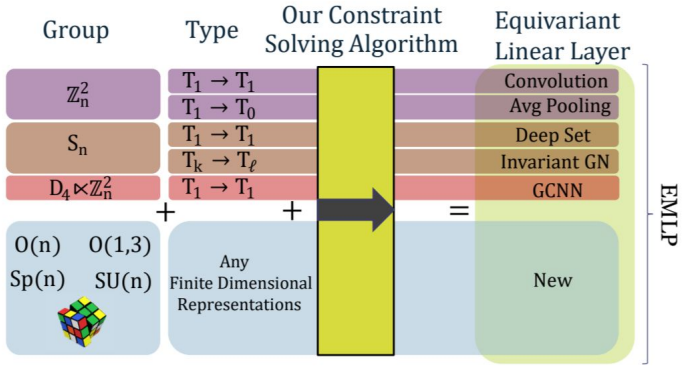
- **Discretization** of group (G-CNN and follow-ups)
- **Truncated sum over irreducible representations** (irreps) (Clebsch-Gordon Nets)
- **Implementing Exp map** and sampling from G (LieConv, Augerino)



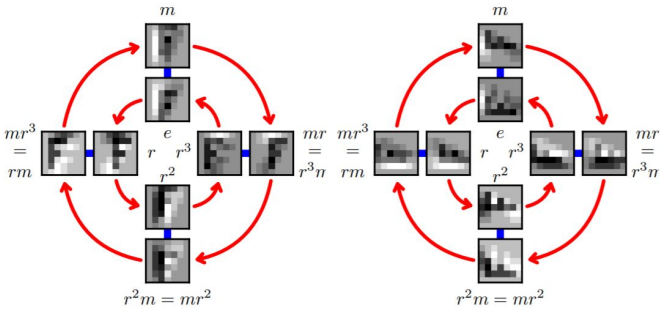


# Why we need improvement?

- **Discretization error** (requires many group elements and parametrization of group)
- Sum over **irreps is tedious** and involves complicated functions
- Each group requires **reimplementing the architecture**
- Cannot deal with entirely **unknown symmetries** (LieConv only searches within subgroups of known groups s.a. rotations, translations and scaling)



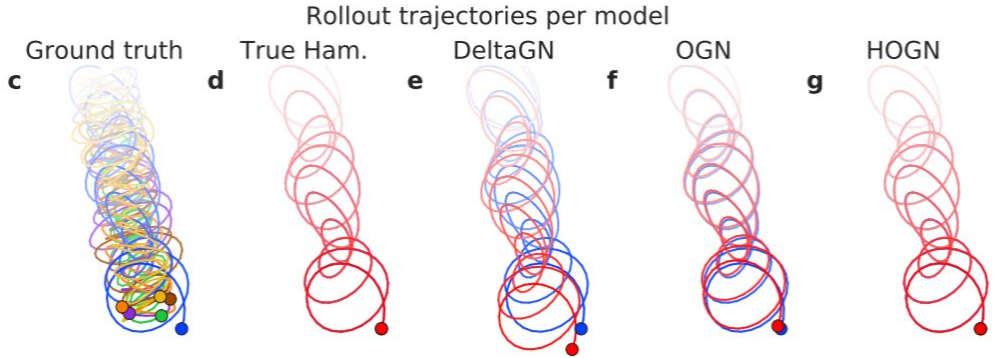

Finzi et al 2021




Cohen & Welling 2016

# Discover Symmetry Automatically

- Many systems may harbor **hidden symmetries**
- Identifying the correct symmetries can help learn compact model

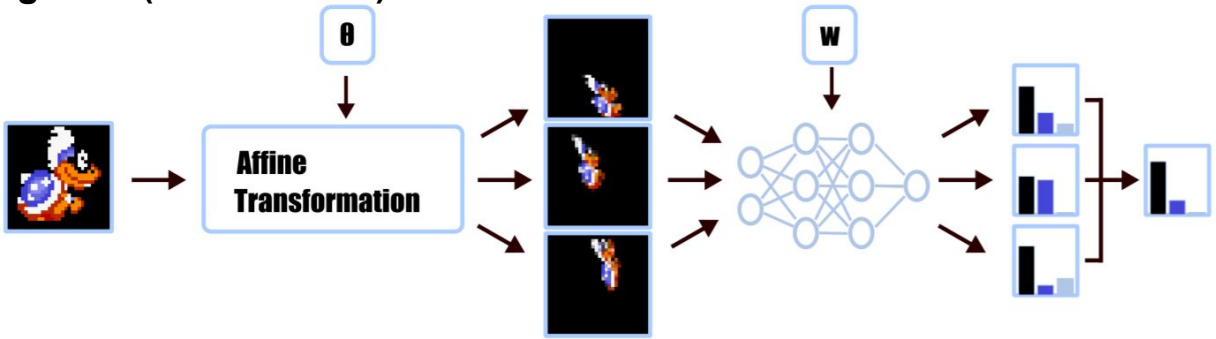



$$\frac{d\,\mathbf{z}}{dt} = J\nabla\mathcal{H} \text{ with } J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}$$

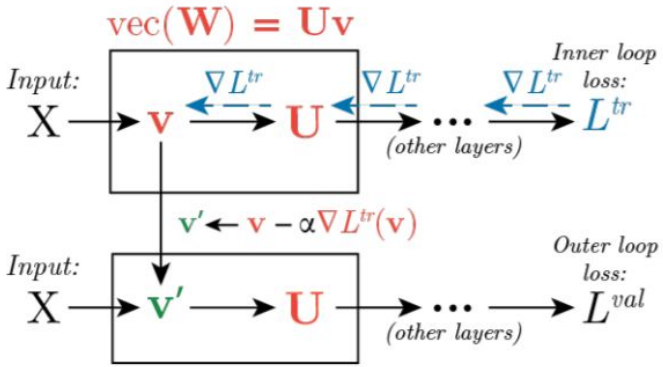
Sanchez-Gonzalez et al 2019 Hamiltonian GN

# Literature on Symmetry discovery

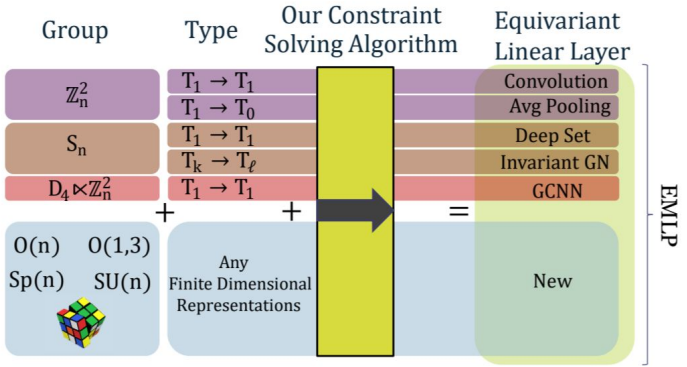
**Augerino (Benton 2020)**




**Meta-learning Rep (Zhou 2021)**

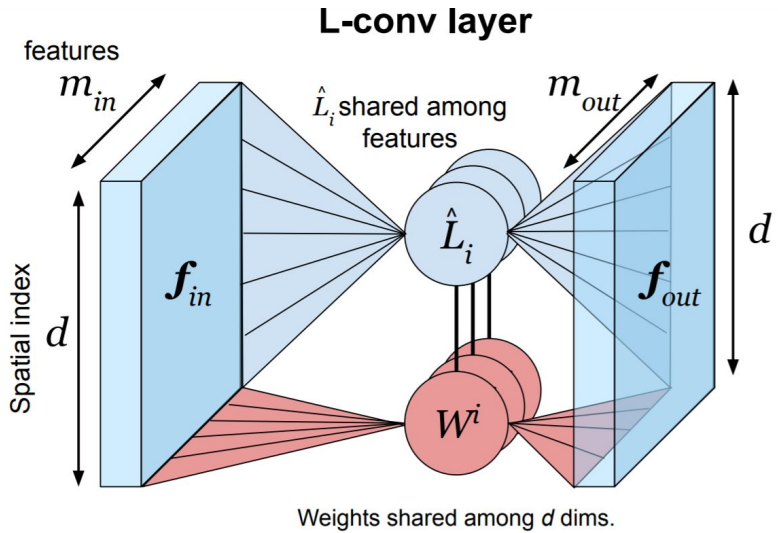



**Practical method (Finzi 2021)**



# Contribution

- Use the fact that all Lie groups can be constructed from their **Lie algebra**
- Find a **basic building block** for approximating any G-conv
- Instead of large numbers of group elements or irreps, encode or learn a **Lie algebra basis**
- We find **relations with physics** which may allow for new ways to discover symmetries
- It can also aid in modeling highly **nonlinear physical systems**



# Symmetry Group

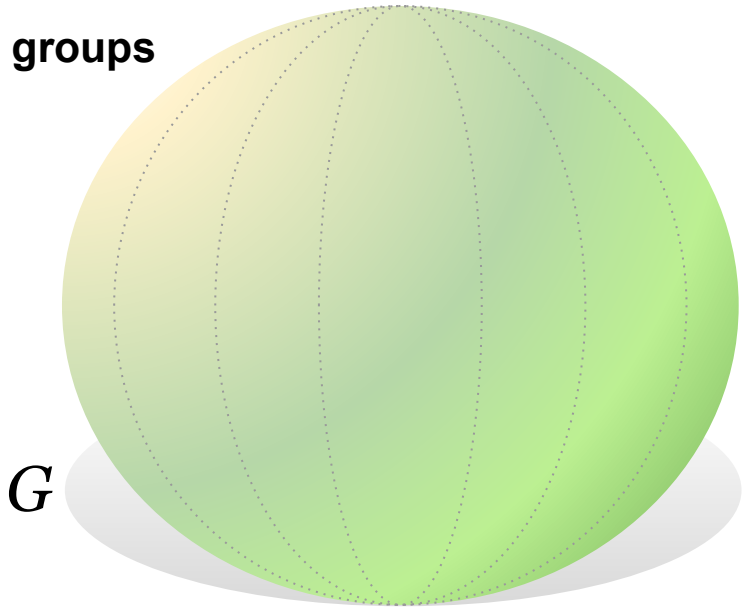
- Let *G* be a group (i.e. a set closed under an operation)

$$I \in G, \quad \text{(identity)}$$

$$\forall g_1, g_2 \in G, \quad g_1 g_2 \in G$$

$$\forall g \in G, \quad \exists g^{-1} \in G \quad \text{s.t.: } g g^{-1} = I$$

- We focus on certain continuous groups called **Lie groups**
- Each Lie group has a **group manifold** (i.e. a topological space which locally looks like $\mathbb{R}^m$ )



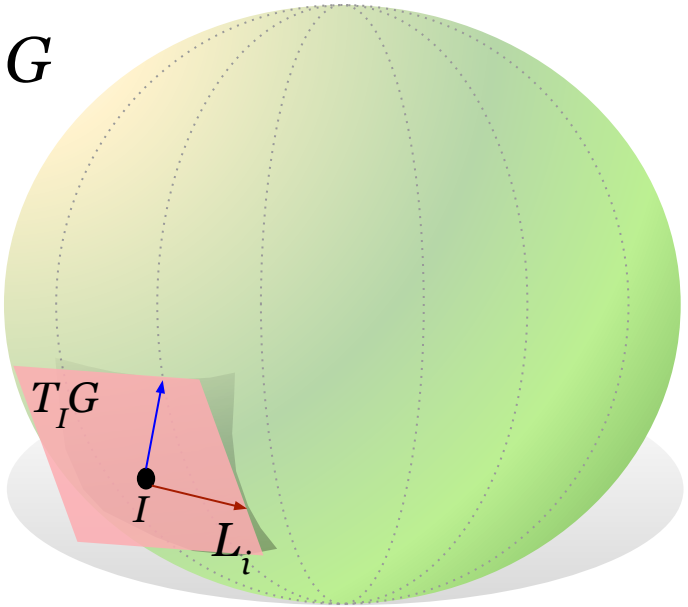
# Lie Groups and Lie Algebras

**Lie algebra:** group elements infinitesimally close to the identity (Einstein summation)

$$u \approx I + \epsilon^i L_i$$

i.e. the tangent space $\mathfrak{g} = T_I G$ , form an algebra, closed under the Lie bracket

$$[L_i, L_j] = c_{ij}{}^k L_k$$



# Lie Groups and Lie Algebras

**Exponential Map:** Any group element on the component of $G$ containing $I$ can be constructed from the Lie algebra using the exponential map

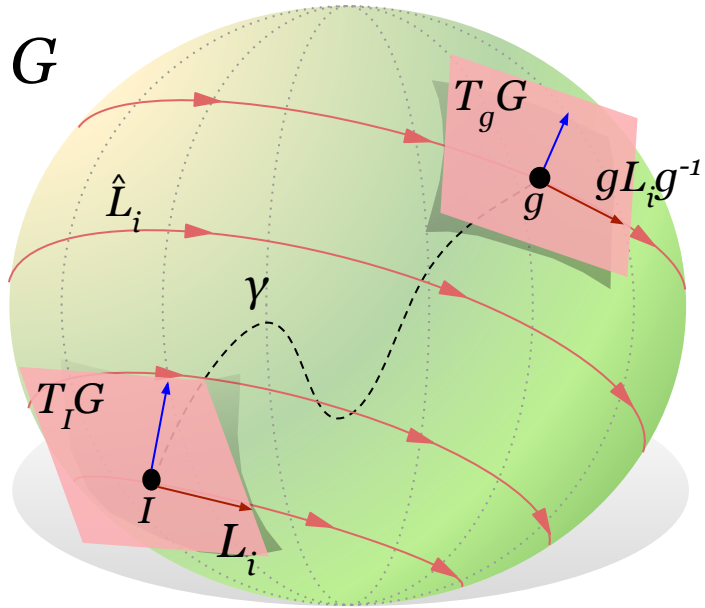
$$g = \prod_a \exp[t_a^i L_i]$$

**Pushforward:** The Lie algebra can be transformed into the tangent space at any group element $g$ as

$$\hat{L}_i(g) = g L_i g^{-1} \in T_g G$$

$$[\hat{L}_i(g), \hat{L}_j(g)] = c_{ij}{}^k \hat{L}_k(g)$$

which is a vector field over $G$

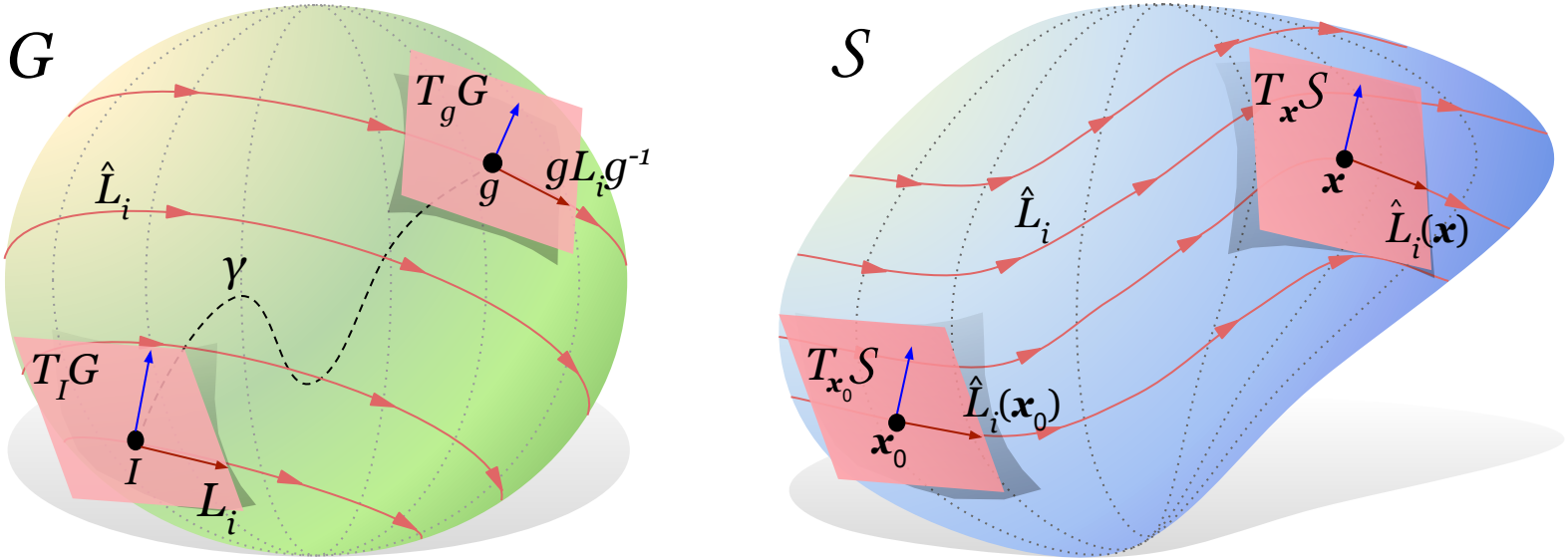


# Lie Group Action on Space $S$

- Let $S$ be a space on which the group acts, meaning

$$x \in \mathcal{S} \text{ and } g \in G,\ gx \in \mathcal{S}$$

- The group action induces similar maps on the space $S$
- The tangent spaces on $G$ and $S$ get mapped to each other
- The vector field $\hat{L}_i$ also generates flows on $S$



# Group Action and Equivariance

Let $f : \mathcal{S} \to \mathcal{F} = \mathbb{R}^m$ be a *scalar* feature map, meaning under the group action it transforms as

$$u \cdot f(\boldsymbol{x}) = f(u^{-1}\boldsymbol{x}) \qquad u \in G$$

Let $F$ be a mapping to a new feature space, such that $F(f) : \mathcal{S} \to \mathcal{F}' = \mathbb{R}^{m'}$

We say $F$ is **equivariant** if

$$u \cdot (F(f)) = F(u \cdot f) \qquad u \in G$$

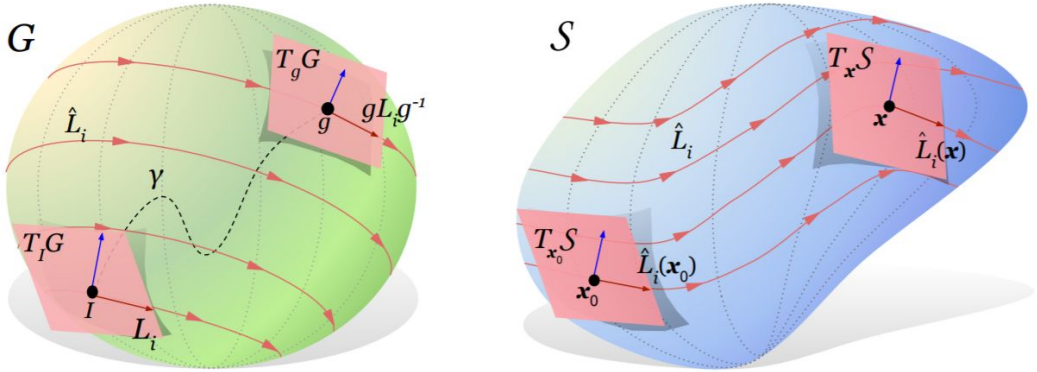
# Group Convolution (G-conv)

$$u \cdot (F(f)) = F(u \cdot f) \qquad u \in G$$

Kondor & Trivedi (2018) showed that a **linear** map *F* is **equivariant** iff it implements group convolution

$$F(f)(g) = [\kappa \star f](g) = \int_G \kappa(g^{-1}v) f(v) dv$$

Where $f(g) \equiv f(g\boldsymbol{x}_0)$, with $\boldsymbol{x}_0 \in \mathcal{S}$ being a chosen origin and kernel $\kappa : G \to \mathbb{R}^{m'} \otimes \mathbb{R}^m$



# Equivariance of G-conv

$$F(f)(g) = [\kappa \star f](g) = \int_G \kappa(g^{-1}v) f(v) dv$$

**Equivariance of G-conv.** G-conv in equation 3 is equivariant (Kondor & Trivedi, 2018). By definition, for $w \in G$ we have

$$\begin{aligned}[\kappa \star w \cdot f](g) &= \int_G \kappa(v) w \cdot f(gv) dv = \int_G \kappa(v) f(w^{-1}gv) dv \\ &= [\kappa \star f](w^{-1}g) = w \cdot [\kappa \star f](g) \end{aligned} \tag{4}$$

# G-conv and Lie algebras

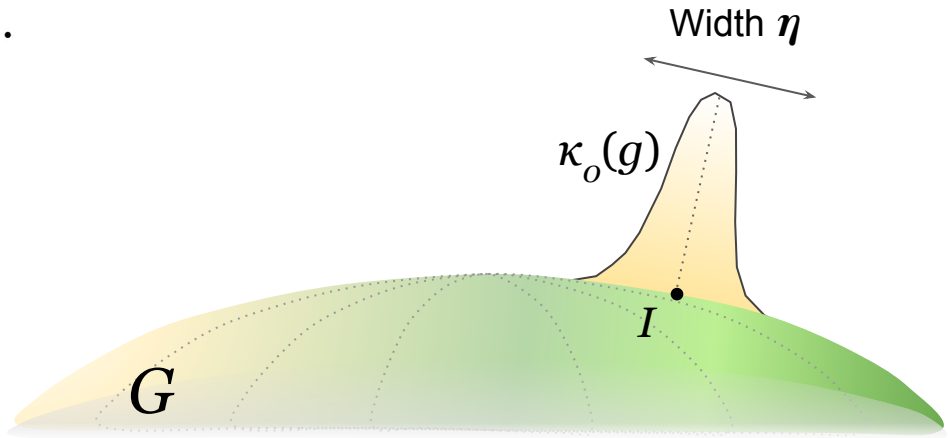
If the kernel is localized near identity, G-conv simplifies as

$$\kappa_0(u) = W^0 \delta_\eta(u)$$

$$Q[f](g) = [\kappa_0 \star f](g) = \int_G dv \kappa_0(v) f(gv) \; = W^0 \left[ I + \overline{\epsilon}^i g L_i \cdot \frac{d}{dg} \right] f(g) + O(\eta^2)$$

$$\overline{\epsilon}^i = \int d\epsilon \delta_\eta(v_\epsilon) \epsilon^i \in \mathbb{R}^m \otimes \mathbb{R}^m.$$

$$\hat{L}_i f(\boldsymbol{x}) \equiv g L_i \cdot \frac{df}{dg} = [g L_i \boldsymbol{x}_0] \cdot \nabla f(\boldsymbol{x})$$

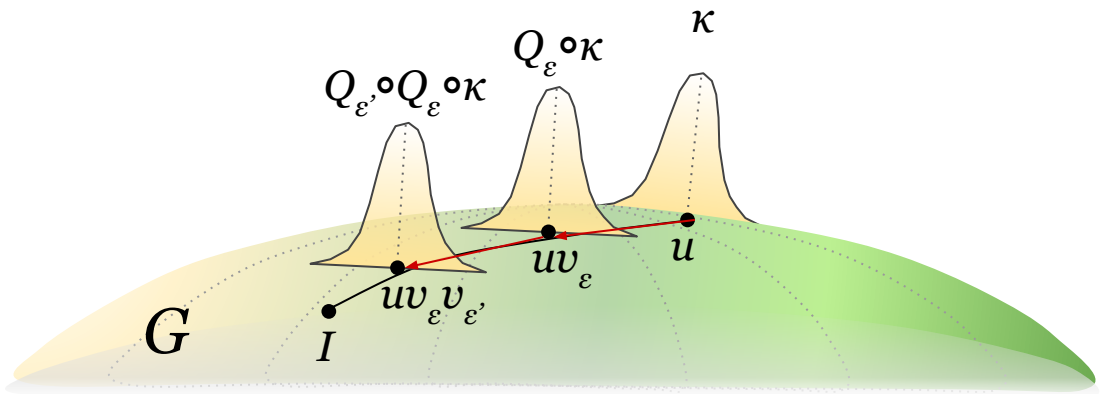


# Lie Algebra Convolution (L-conv)

- **L-conv** = G-conv with localized kernel near identity

$$Q[f](\boldsymbol{x}) = W^0 \left[ I + \bar{\epsilon}^i \hat{L}_i \right] f(\boldsymbol{x})$$
$$= W^0 \left[ I + \bar{\epsilon}^i [g L_i \boldsymbol{x}_0]^\alpha \partial_\alpha \right] f(\boldsymbol{x})$$

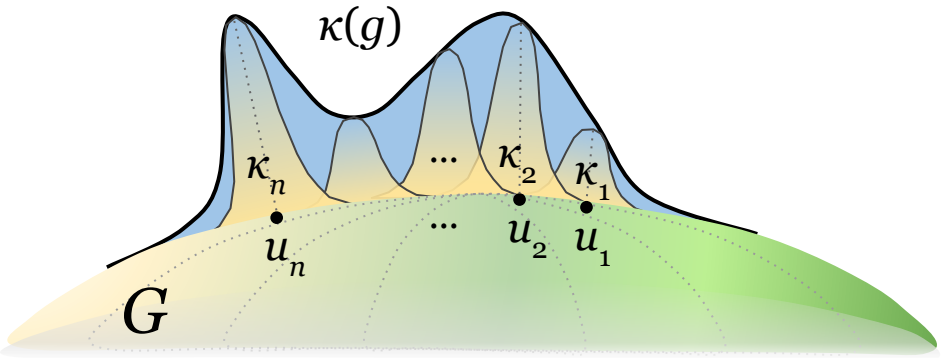
- Each L-conv Layer moves $f(\boldsymbol{x})$ along $\hat{L}_i$ by $W^0 \bar{\epsilon}^i$
- Multi-layer L-conv can be used to shift a kernel toward or away from identity



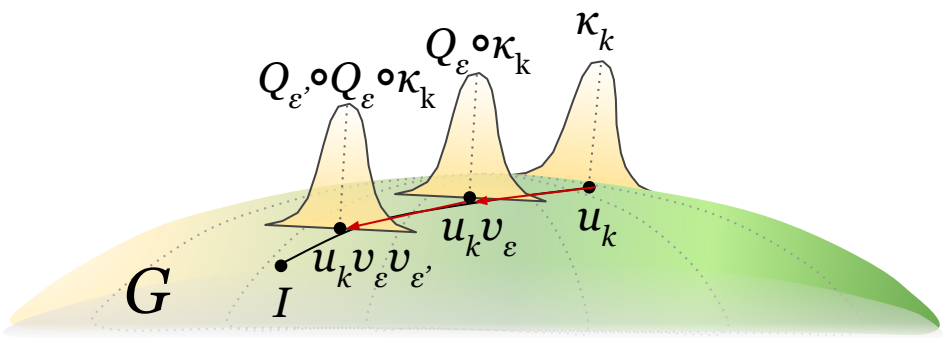
# Approximating any G-conv Using L-conv

- The idea is a combination of the Taylor expansion and the universal approximation theorem for neural networks.
- To approximate G-conv, we need to approximate the kernel

1) Approximate kernel with set of L-convs at $u_k$




2) move $u_k$ to $I$ using multi-layer L-conv



# Universal approximation of Equivariant NN

**Equivariance of nonlinearity.** Pointwise nonlinearities give equivariant maps between scalar feature maps. To see this, let $\sigma : \mathbb{R} \to \mathbb{R}$. We extend $\sigma : \mathcal{F} \to \mathcal{F}$ by applying $\sigma$ component-wise. Let $f : \mathcal{S} \to \mathcal{F}$ be a scalar feature map (i.e., $g \cdot f(\boldsymbol{x}) = f(g^{-1}\boldsymbol{x})$). Then

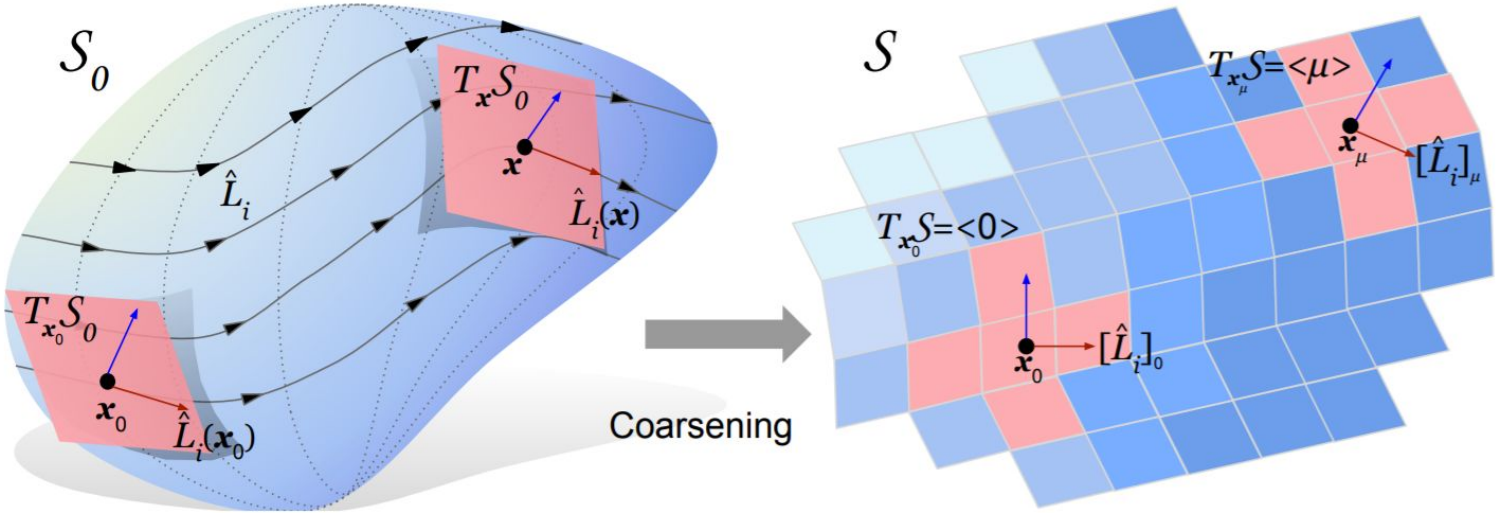
$$g \cdot (\sigma \circ (f))(\boldsymbol{x}) = \sigma \circ (f)(g^{-1}\boldsymbol{x}) = \sigma \circ (g \cdot f)(\boldsymbol{x}).$$

Since the composition of equivariant maps is equivariant, given equivariant linear mapping $Q : \mathcal{F}^{\mathcal{S}} \to \mathcal{F}'^{\mathcal{S}}$ (i.e. $g \cdot Q[f] = Q[g \cdot f]$), the layer $f \mapsto \sigma \circ Q[f]$ is equivariant. Hence we have the corollary:

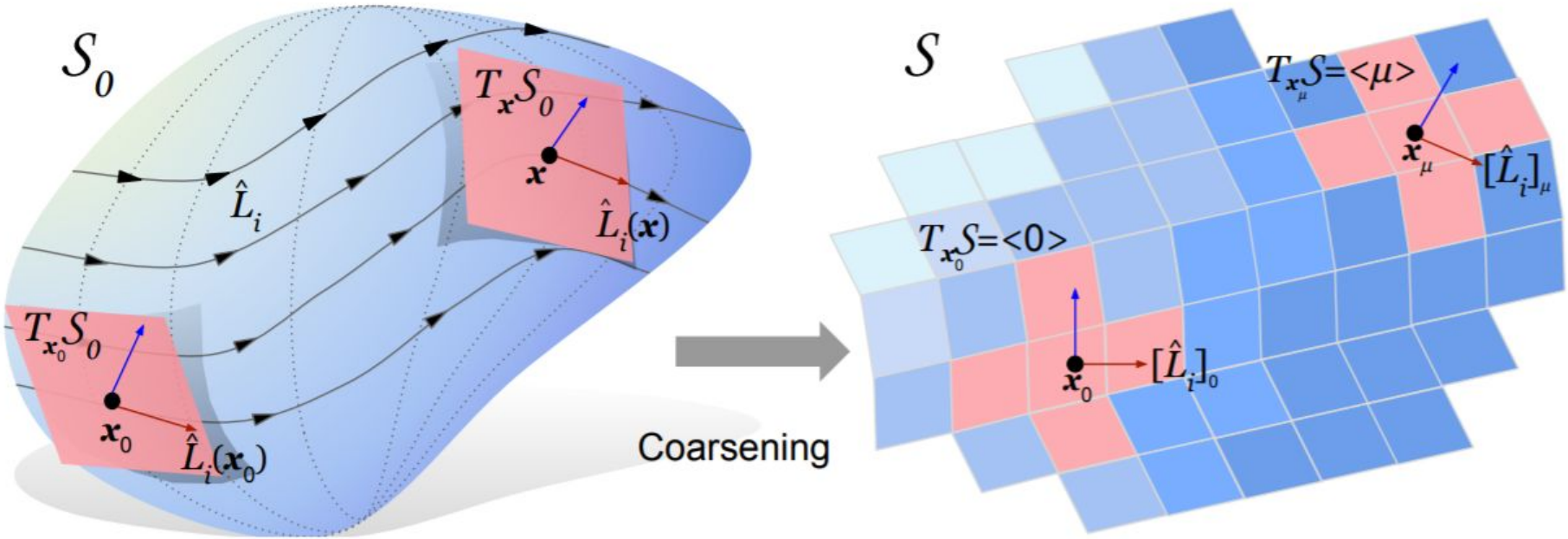
**Corollary 1.** *Assume $G$ is compact and acts on $\mathcal{S}$ transitively. Then any equivariant feedforward neural network (FNN) can be approximated using multilayer L-conv with point-wise nonlinearities.*

# Discretized Version and Implementation

- The space becomes *d* nodes $\mathcal{S} = \{\boldsymbol{x}_0, \dots \boldsymbol{x}_{d-1}\}$
- The feature maps become matrices $\boldsymbol{f} \in \mathcal{F} = \mathbb{R}^d \otimes \mathbb{R}^m$
- **Note:** Even when the space is discretized, the **symmetry group** is still **continuous** (subgroup of linear transformations $G \subseteq \mathrm{GL}_d(\mathbb{R})$ )
- The vector fields $\hat{L}_i$ become *d*x*d* matrices
- **Tangent spaces** become **neighbors** of nodes



# Using L-conv on Discretized data



# Discrete space L-conv with continuous symmetries

**Feature maps** To define features $f(\boldsymbol{x}_\mu) \in \mathbb{R}^m$ for $\boldsymbol{x}_\mu \in \mathcal{S}$, we embed $\boldsymbol{x}_\mu \in \mathbb{R}^d$ and encode them as the canonical basis (one-hot) vectors with components $[\boldsymbol{x}_\mu]^\nu = \delta_\mu^\nu$ (Kronecker delta), e.g. $\boldsymbol{x}_0 = (1, 0, \ldots, 0)$. The feature space becomes $\mathcal{F} = \mathbb{R}^d \otimes \mathbb{R}^m$, meaning feature maps $\boldsymbol{f} \in \mathcal{F}$ are $d \times m$ tensors, with $f(\boldsymbol{x}_\mu) = \boldsymbol{x}_\mu^T \boldsymbol{f} = \boldsymbol{f}_\mu$.

**G-conv and L-conv in tensor notation** Writing G-conv equation 3 in the tensor notation we have

$$[\kappa \star f](g\boldsymbol{x}_0) = \int_G \kappa(v) f(gv\boldsymbol{x}_0) dv = \boldsymbol{x}_0^T \int_G v^T g^T \boldsymbol{f} \kappa^T(v) dv \equiv \boldsymbol{x}_0^T [\boldsymbol{f} \star \kappa](g) \tag{66}$$

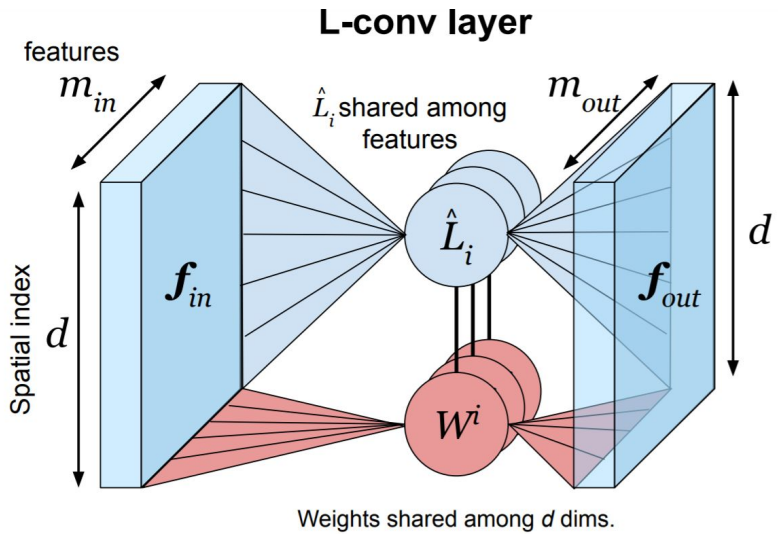
where we moved $\kappa^T(v) \in \mathbb{R}^m \otimes \mathbb{R}^{m'}$ to the right of $\boldsymbol{f}$ because it acts as a matrix on the output index of $\boldsymbol{f}$. The equivariance of equation 66 is readily checked with $w \in G$

$$w \cdot [\boldsymbol{f} \star \kappa](g) = [\boldsymbol{f} \star \kappa](w^{-1}g) = \int_G v^T g^T w^{-1T} \boldsymbol{f} \kappa^T(v) dv = [(w \cdot \boldsymbol{f}) \star \kappa](g) \tag{67}$$

# L-conv Implemented as Graph Convolution (GCN)

- Each $\hat{L}_i$ is a graph adjacency matrix or aggregation function
- L-conv can be implemented as multiple GCN
- The $\hat{L}_i$ can also be made trainable to learn symmetries

$$Q[\boldsymbol{f}] = \left(\boldsymbol{f} + \hat{L}_i \boldsymbol{f} \bar{\epsilon}^i\right) W^{0T}$$

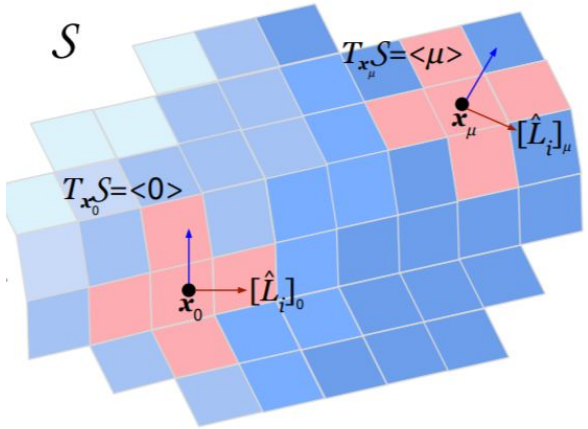


# Structure of Discretized vector field

Using th incidence matrix ***B***

$$<\mu>=\left\{\alpha\in\mathcal{E}\middle|\bar{\boldsymbol{B}}^{\mu}_{\alpha}=-1\right\}$$

$$\alpha\in<\mu>,\boldsymbol{B}_{\alpha}\boldsymbol{f}=\boldsymbol{f}_{\mu}-\boldsymbol{f}_{\nu}\sim\partial_{\alpha}\boldsymbol{f}$$




$$[\hat{L}_i]^{\nu}_{\mu}=[\hat{\ell}_i]^{\alpha}_{\mu}\boldsymbol{B}^{\nu}_{\alpha}=[g_{\mu}L_i\boldsymbol{x}_0]^{\nu}=\sum_{\rho}[g_{\mu}\boldsymbol{x}_{\rho}\boldsymbol{x}^T_{\rho}L_i\boldsymbol{x}_0]^T\boldsymbol{x}_{\nu}$$

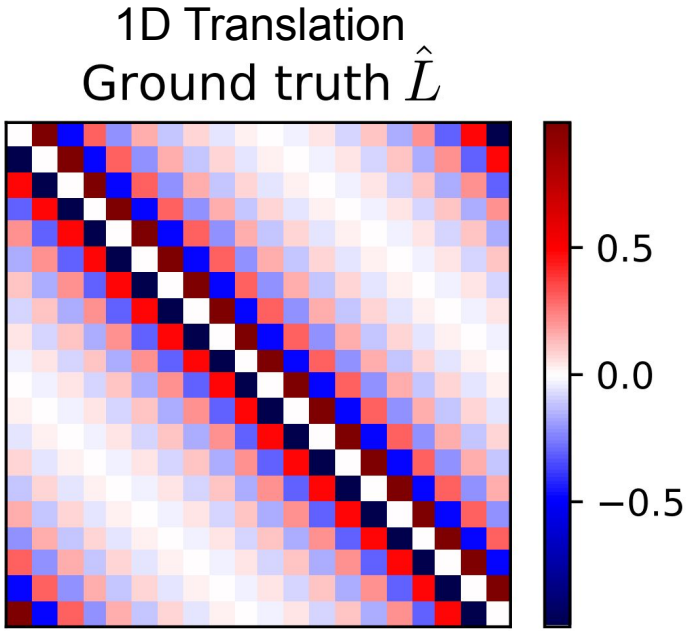
$$=\sum_{\rho\in<0>}[L_i]^{\rho}_0[\boldsymbol{x}^T_{\nu}g_{\mu}\boldsymbol{x}_{\rho}]^T=[L_i]^{\rho}_0[g_{\mu}]^{\nu}_{\rho}=[\hat{\ell}_i]^{\alpha}_0\boldsymbol{B}^{\rho}_{\alpha}[g_{\mu}]^{\nu}_{\rho}\qquad(72)$$

This $\hat{L}_i\equiv\hat{\ell}^{\alpha}_i\boldsymbol{B}_{\alpha}$ is the discrete $\mathcal{S}$ version of the vector field $\hat{L}_i(\boldsymbol{x})=[gL_i\boldsymbol{x}_0]^{\alpha}\partial_{\alpha}$ in equation 8.

# Example of Continuous Symmetry Generator on Discrete Space

Using Whittaker-Shannon interpolation, continuous translations (e.g. half a pixel) are possible on discrete spaces

$$\hat{L}_{\rho}^{\nu} = \sum_{p} \frac{2\pi p}{d^2} \sin\left(\frac{2\pi p}{d}(\rho - \nu)\right)$$



# Learning 1D translation Lie algebra

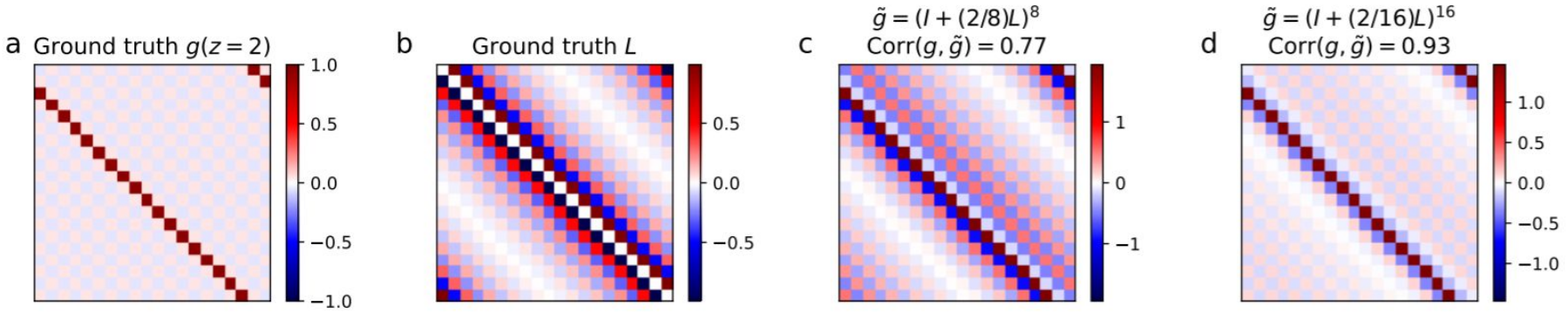
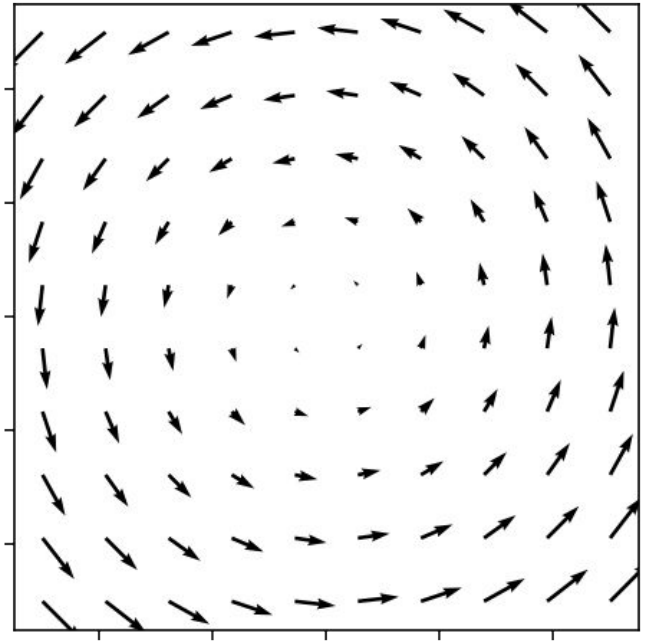
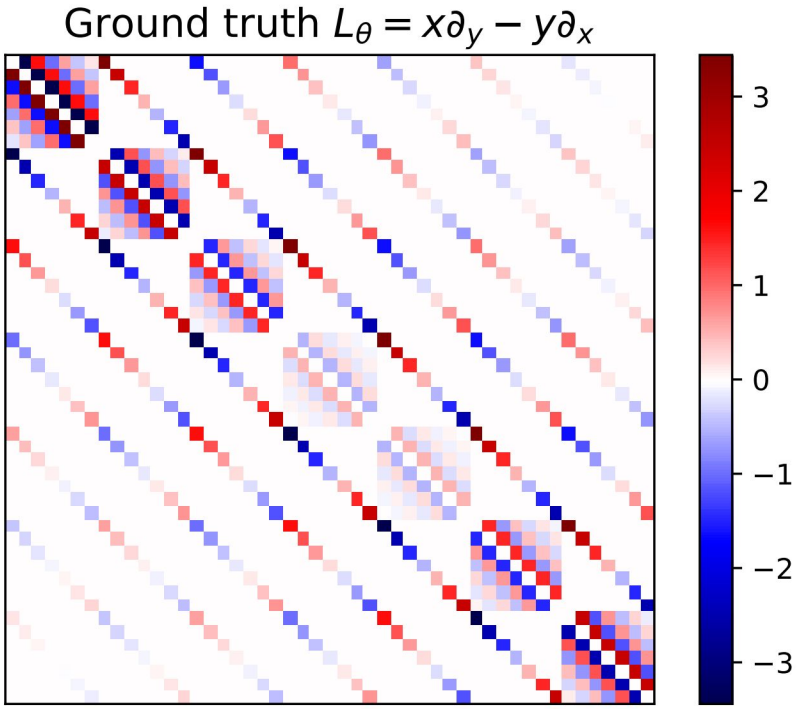



Figure 6: 1D Translation: Using the Shannon-Whittaker Interpolation (SWI) one can generate continuous shifts on discrete data. These include integer shifts (a, ground truth). (SWI) also yields an infinitesimal generator $L$ for shifts (b). This $L$ can be used to approximate finite shifts using $\tilde{g}_n(z) = (I + z/nL)^n$, with $n \to \infty$ yielding $\exp[zL]$. (c) and (d) show the approximation of a shift by two pixels using $n = 8$ and $n = 16$.

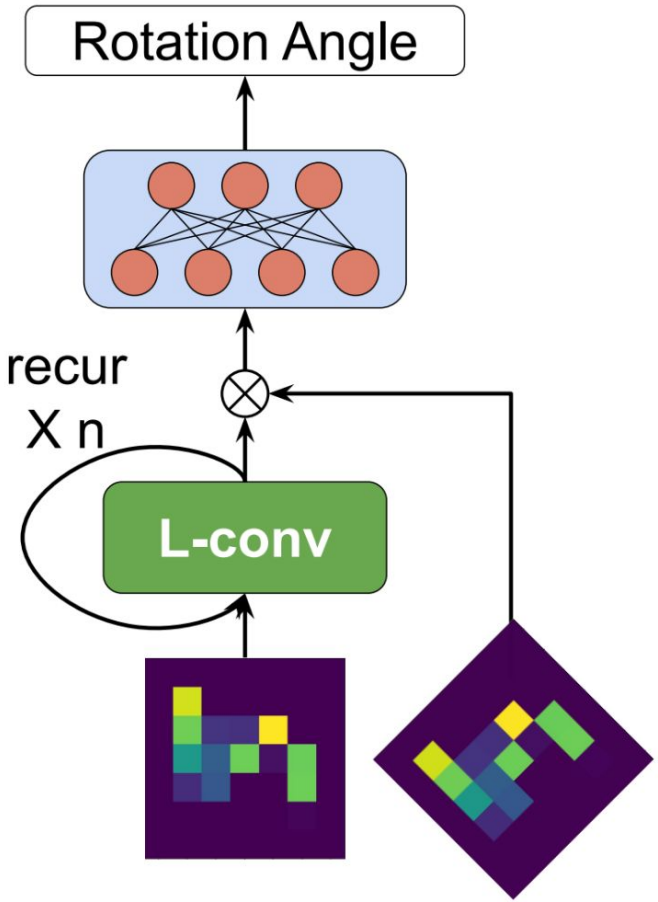
# Example: 2D Rotations

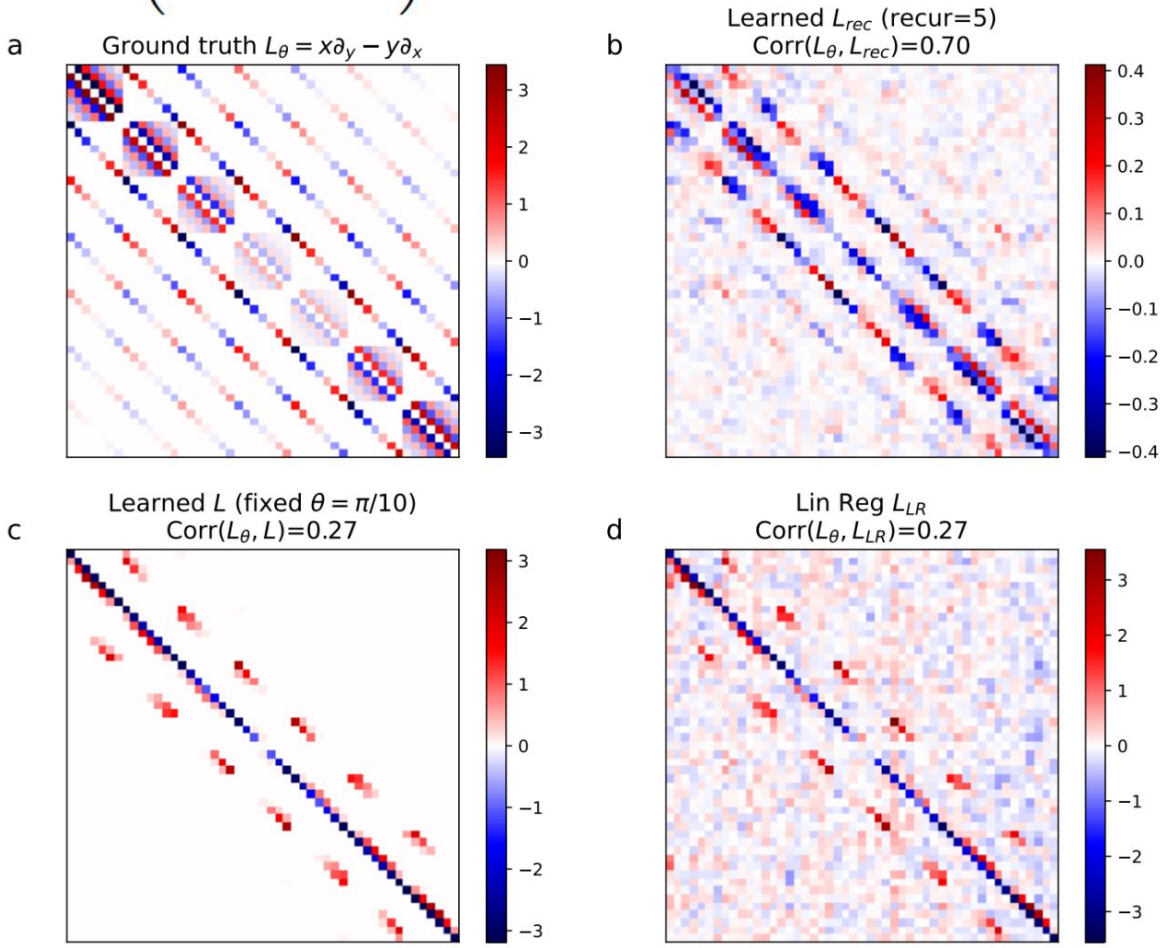




# Experiment

$$Q[\boldsymbol{f}] = \left(\boldsymbol{f} + \hat{L}_i \boldsymbol{f} \overline{\epsilon}^i\right) W^{0T}$$



# On Small Image datasets

More experiments in paper

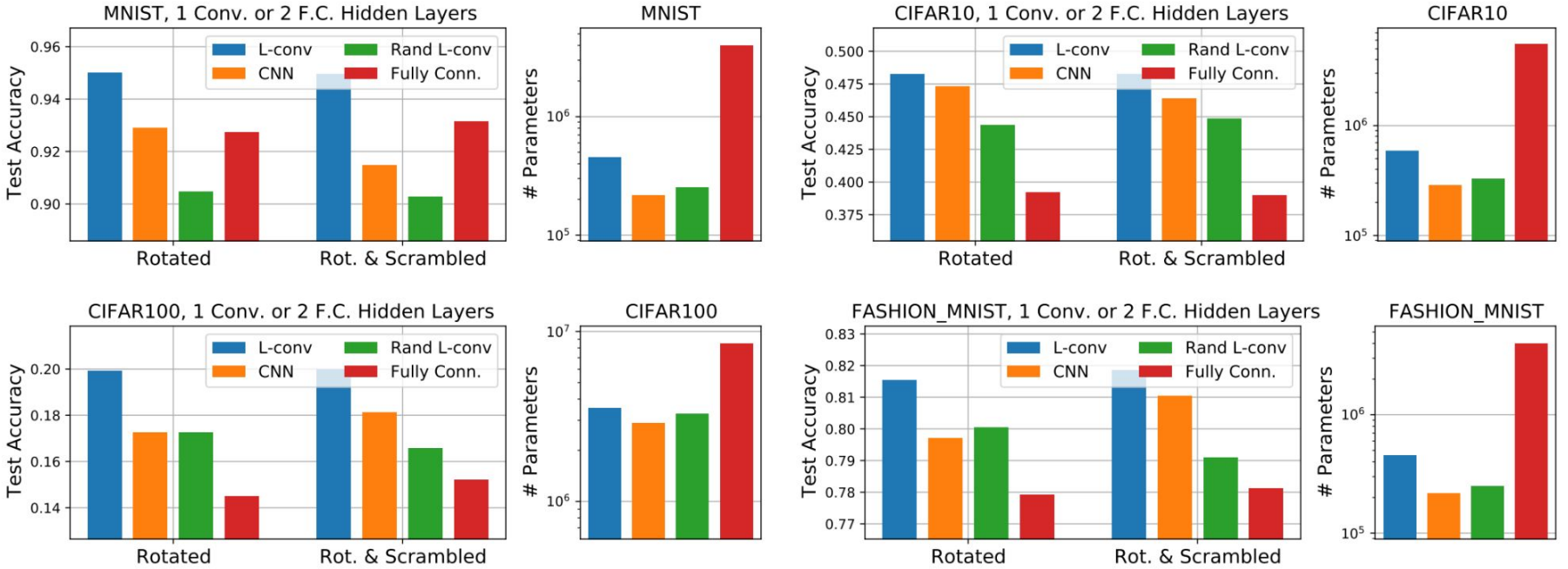



Figure 9: Results on four datasets with two variant: “Rotated” and “Rotated and scrambled”. In all cases L-conv performs best. On MNIST, FC and CNN come close, but using 5x more parameters.

# Loss Function and Relation with Physics Lagrangians

How do Group Equivariant NN relate to physics?

- Physicist have long used models encoding continuous symmetries
- In Physics, the symmetry is encoded in the objective function
- **Concrete Relation:** The MSE loss for a single layer L-conv is known in physics as the "free field theory Lagrangian"

$$I[\Phi; W] = \int_G dg \mathcal{L}[\Phi; W] = \int_G dg \left\| W^0 \left[ I + \bar{\epsilon}^i [\hat{L}_i]^\alpha \partial_\alpha \right] \Phi(g) \right\|^2$$
$$= \int_{\mathcal{S}} \frac{d^n x}{\left| \frac{\partial x}{\partial g} \right|} \left[ \Phi^T \mathbf{m_2} \Phi + \partial_\alpha \Phi^T \mathbf{h}^{\alpha\beta} \partial_\beta \Phi + [\hat{L}_i]^\alpha \partial_\alpha \left( \Phi^T \mathbf{v}^i \Phi \right) \right]$$

Where $\phi_n \equiv [f_n | y_n]$ are the input and output features. In physics, **h** is a Riemannian metric for space *S* and $\mathbf{m_2}$ is the mass matrix. The last term usually becomes a boundary term and vanishes.

# Why it matters

For ML, the connection with physics allows us to

1) Use existing physics methods (e.g. noether's theorem) for discovering and learning symmetries in ML
2) Euler-Lagrange equations may help assess robustness of the ML model

For physics, potential benefits could be to

3) Build new physics models which use the flexibility of NN, while being equivariant. Possible uses
    a) Find Lagrangians for strongly interacting systems (e.g. confinement in QCD)
    b) Highly nonlinear systems (Sine-Gordon, Kuramoto, turbulence)

# Thank you