CS 6501: Constrained-Aware Generative AI

Lecture Notes 5
Architectures for Control

Geometric deep learning, equivariance, and inductive bias as architectural constraints

Prof. Ferdinando Fioretto
Department of Computer Science, University of Virginia

Tuesday, January 27, 2026

Abstract

This lecture instantiates the "architecture-level" mode of constraint injection: instead of adding penalties, projecting samples, or searching over outputs, we *restrict the hypothesis class* so that the model *automatically respects* a desired symmetry. The technical language for this restriction is *equivariance*: a map f is equivariant to a group action if transforming the input and then applying f is equivalent to applying f and then transforming the output. Equivariance captures a large class of control-relevant inductive biases, including translation equivariance in CNNs, permutation equivariance in graph neural networks, and roto-translation equivariance in models over 3D point clouds, molecules, and rigid-body trajectories. We formalize group actions and equivariant maps, connect equivariance to weight tying and sample efficiency, and then study three concrete realizations: (i) permutation-equivariant message passing on graphs; (ii) $E(n)$-equivariant graph neural networks (EGNNs) that update both node features and coordinates while respecting Euclidean symmetries; and (iii) higher-order SE(3)-equivariant architectures based on tensor representations, including Tensor Field Networks and SE(3)-Transformers. Throughout, we emphasize why these inductive biases are best interpreted as *hard architectural constraints* that complement the probabilistic and optimization-based mechanisms developed elsewhere in the course.

1 Why geometry is "architecture-level control"

The course has so far emphasized constrained generation as inference over a target distribution formed by combining a base model with soft potentials and hard feasibility sets (Lecture 1), and showed how constraints can enter at inference time through decoding and search (Lecture 4). This lecture focuses on a different injection point: *the architecture itself.*

Goal. The goal is to encode *symmetry constraints* directly into a neural network so that the network's outputs transform predictably under known transformations of the input. In scientific and engineering problems, these symmetries are often consequences of first principles. For example, molecular energies do not change when we translate or rotate the entire molecule; a rigid-body state transforms under the SE(3) group; and many multi-agent and mesh-based representations are inherently permutation symmetric.

Challenge. If a model ignores symmetry and is trained on finite data, it typically learns a function class that *breaks* the symmetry: it can assign different outputs to physically identical inputs related by a transformation. Data augmentation can reduce this mismatch, but it does not guarantee exact symmetry and can be inefficient, especially when the symmetry group is continuous.

Solution pattern. Equivariance implements symmetry as a hard constraint on the function class. Instead of learning an arbitrary function $f : \mathcal{X} \to \mathcal{Y}$, we restrict to those f satisfying an equivariance condition relative to a group $\mathcal{G}$. This is a form of architectural control: the model is "steered" not by modifying the objective, but by modifying the space of functions that can be represented. The unifying survey of this viewpoint is the geometric deep learning program (**?**).

2 Groups, actions, invariance, and equivariance

2.1 Group actions as transformations of data

Definition 1 (Group and group action)**.** *A* group $(\mathcal{G}, \cdot)$ *is a set* $\mathcal{G}$ *with an associative binary operation, an identity element* $e \in \mathcal{G}$*, and inverses* g^{-1} *for each* $g \in \mathcal{G}$*. A (left)* action *of* $\mathcal{G}$ *on a set* $\mathcal{X}$ *is a map* $\mathcal{G} \times \mathcal{X} \to \mathcal{X}$*, written* $(g, \mathbf{x}) \mapsto g \cdot \mathbf{x}$*, satisfying* $e \cdot \mathbf{x} = \mathbf{x}$ *and* $g \cdot (h \cdot \mathbf{x}) = (gh) \cdot \mathbf{x}$ *for all* $g, h \in \mathcal{G}$ *and* $\mathbf{x} \in \mathcal{X}$*.*

In learning problems, $\mathcal{X}$ is the space of inputs (signals on a grid, graphs, point clouds, trajectories) and $g \cdot \mathbf{x}$ denotes a known transformation. Examples include translations acting on images, permutations acting on node orderings, and rigid motions acting on 3D coordinates.

Euclidean groups. In $\mathbb{R}^n$, a key symmetry group is the Euclidean group $E(n) = \mathbb{R}^n \rtimes \mathrm{O}(n)$ consisting of translations and orthogonal transformations. For orientation-preserving rigid motions in 3D, the group is $\mathrm{SE}(3) = \mathbb{R}^3 \rtimes \mathrm{SO}(3)$.

2.2 Equivariance as a hard constraint on maps

Let $\mathcal{G}$ act on $\mathcal{X}$ and $\mathcal{Y}$. A map $f : \mathcal{X} \to \mathcal{Y}$ can either ignore the action or be structured to respect it.

Equivariance and invariance

Definition 2 (Equivariance and invariance)**.** *Let* $\mathcal{G}$ *act on* $\mathcal{X}$ *and* $\mathcal{Y}$*. A function* $f : \mathcal{X} \to \mathcal{Y}$ *is* $\mathcal{G}$-equivariant *if, for all* $g \in \mathcal{G}$ *and* $\mathbf{x} \in \mathcal{X}$*,*

$$f(g \cdot \mathbf{x}) = g \cdot f(\mathbf{x}). \tag{1}$$

If $\mathcal{G}$ *acts trivially on* $\mathcal{Y}$ *(that is,* $g \cdot \mathbf{y} = \mathbf{y}$ *for all* $\mathbf{y} \in \mathcal{Y}$*), then* (**??**) *reduces to* $\mathcal{G}$-invariance*:*

$$f(g \cdot \mathbf{x}) = f(\mathbf{x}). \tag{2}$$

Interpretation for control. Equivariance is often the desired property for predictors of *geometric* quantities (forces, vector fields, poses), because the output should transform with the input. Invariance is often desired for *scalar* objectives (energy, stability, reward), because those should not change under a global frame transformation.

Lemma 1 (Closure under composition)**.** *If* $f : \mathcal{X} \to \mathcal{Y}$ *is* $\mathcal{G}$*-equivariant and* $h : \mathcal{Y} \to \mathcal{Z}$ *is* $\mathcal{G}$*-equivariant, then* $h \circ f$ *is* $\mathcal{G}$*-equivariant. If* f *is* $\mathcal{G}$*-equivariant and* h *is* $\mathcal{G}$*-invariant, then* $h \circ f$ *is* $\mathcal{G}$*-invariant.*

This lemma is one reason equivariance is an architectural primitive: if each layer is equivariant, then the full network is equivariant.

2.3 Equivariance versus data augmentation

A recurring design choice is whether to impose symmetry by augmentation or by architecture. Augmentation enforces the symmetry *in expectation over the training set* by exposing transformed samples to the learner. In contrast, equivariant architectures enforce the symmetry *for all inputs*, including out-of-distribution samples. Group equivariance in convolutional networks is a canonical example (**?**), and in modern geometric settings the benefits are pronounced because the relevant group is often continuous and high dimensional (**?**).

3 From CNNs to group convolutions: a fast geometric recap

The success of standard CNNs can be reframed as a symmetry statement: convolution is translation-equivariant, meaning that shifting the input shifts the feature map. In the geometric deep learning view, CNNs are instances of group equivariant architectures on the translation group (**?**).

Group convolution. To extend translation equivariance to a richer group $\mathcal{G}$ (for example, translations plus discrete rotations), one replaces standard convolution with *group convolution*. The core idea is to define features on $\mathcal{G}$ and convolve by integrating (or summing) over the group. In discrete settings this yields practical architectures with stronger weight sharing than standard CNNs, improving sample efficiency and robustness (**?**). More generally, steerable CNNs characterize equivariant kernels via group representation theory and yield systematic constructions for continuous groups such as $E(2)$ (**?**).

For this course, the main conceptual point is that group convolution is not "regularization": it hard-codes a constraint (**??**) into the layer definition.

4 Graphs: permutation symmetry as the first geometric constraint

Many control and generation problems are expressed over graphs: molecules, meshes, social or multi-agent interaction graphs, and factor graphs. The basic symmetry is node relabeling. Let S_n be the permutation group on n elements, acting on node-indexed representations by permuting indices.

Permutation equivariance. A graph layer is permutation equivariant if permuting node order permutes outputs in the same way. Message passing networks achieve this by aggregating neighbor messages through a commutative operator (typically summation). This is closely related to the characterization of permutation-invariant set functions (**?**).

Definition 3 (Message passing layer (one-step form))**.** *Let $G = (V, E)$ with nodes $V = \{1, \dots, n\}$, node features $h_i \in \mathbb{R}^d$, and edge features a_{ij} for $(i, j) \in E$. A generic message passing update has the form*

$$m_{ij} = \phi_e(h_i, h_j, a_{ij}), \tag{3}$$

$$\bar{m}_i = \sum_{j \in \mathcal{N}(i)} m_{ij}, \tag{4}$$

$$h_i^+ = \phi_h(h_i, \bar{m}_i), \tag{5}$$

where ϕ_e and ϕ_h are shared functions (typically MLPs) and the sum is taken over neighbors.

Proposition 1 (Permutation equivariance of message passing)**.** *Assume the aggregation operator is commutative and associative (for example, sum or mean) and the same functions ϕ_e, ϕ_h are shared across nodes and edges. Then the message passing update is equivariant to permutations of node indices.*

This proposition is an explicit architectural constraint: the model cannot "cheat" by depending on an arbitrary indexing of nodes.

5 $E(n)$-equivariant GNNs: learning over features and coordinates

Graphs with geometry attach coordinates $x_i \in \mathbb{R}^n$ to nodes: atoms in a molecule, residues in a protein backbone, particles in an N-body system, or agents in a spatial environment. In these settings we want both permutation equivariance and equivariance to the Euclidean group $E(n)$ acting on coordinates.

5.1 The EGNN layer

The E(n)-Equivariant Graph Neural Network (EGNN) of **?** is a widely used construction because it achieves Euclidean equivariance without explicitly manipulating higher-order tensor features.

$E(n)$-Equivariant Graph Neural Network (EGNN) update

For node features $h_i^\ell \in \mathbb{R}^d$ and coordinates $x_i^\ell \in \mathbb{R}^n$, define

$$m_{ij}^\ell = \phi_e\Big(h_i^\ell, h_j^\ell, a_{ij}, \|x_i^\ell - x_j^\ell\|_2^2\Big), \tag{6}$$

$$h_i^{\ell+1} = \phi_h\left(h_i^\ell, \sum_{j\in\mathcal{N}(i)} m_{ij}^\ell\right), \tag{7}$$

$$x_i^{\ell+1} = x_i^\ell + \sum_{j\in\mathcal{N}(i)} (x_i^\ell - x_j^\ell)\,\phi_x(m_{ij}^\ell), \tag{8}$$

where ϕ_e, ϕ_h, ϕ_x are shared learnable functions (typically MLPs), and $\phi_x(m_{ij}^\ell)$ is scalar-valued.

The design choices in (**??**)–(**??**) are tightly coupled to equivariance. The message depends on coordinates only through squared distances, which are invariant to $E(n)$. The coordinate update is a sum of relative displacement vectors $(x_i - x_j)$ scaled by invariant scalars, yielding an equivariant update.

5.2 Equivariance guarantee

We state the key property informally; proofs appear in **?**.

Theorem 1 ($E(n)$-equivariance of EGNN layers)**.** *Let $g \in E(n)$ act on coordinates by $g{\cdot}x = Rx+t$ with $R \in \mathrm{O}(n)$ and $t \in \mathbb{R}^n$, and act trivially on scalar node features. Then the EGNN update* (**??**)–(**??**) *is equivariant to $E(n)$: if we transform the input coordinates by g, the output coordinates transform by the same g, and the updated node features are invariant.*

Why this matters. In molecular and robotic settings, the global coordinate frame is arbitrary. Equivariant architectures eliminate spurious dependence on that frame and thereby reduce the burden on data and training. In practice, EGNN-style models are used both as predictive components (property prediction, dynamics) and as score networks inside diffusion models for 3D generation (**?**).

6 Higher-order equivariance: Tensor Field Networks and SE(3) attention

EGNN achieves equivariance with scalar features and coordinate updates, but many scientific tasks benefit from *higher-order* geometric features (vectors and tensors) that carry directional information. This motivates architectures grounded in representation theory for SO(3) and SE(3).

6.1 Tensor Field Networks (TFNs)

Tensor Field Networks (TFNs) represent node features as collections of geometric tensors that transform under irreducible representations (irreps) of SO(3), and define convolutional filters using spherical harmonics (**?**). At a high level, TFNs enforce equivariance by ensuring that every linear map between features respects the representation structure, and by combining features using tensor products followed by Clebsch–Gordan decompositions.

Practical intuition. One can view TFNs as learning with "typed" channels: scalars (type 0), vectors (type 1), and higher-order tensors. The network is only allowed to combine these channels in ways permitted by SO(3) representation theory, which acts as a set of hard constraints on the layer parameterization.

6.2 SE(3)-Transformers

The SE(3)-Transformer extends self-attention to 3D point clouds and graphs while preserving SE(3) equivariance (**?**). Conceptually, it replaces dot-product attention with an attention mechanism built from equivariant kernels, again defined in the TFN framework. This yields a model that can flexibly aggregate information across many neighbors (a strength of attention) without sacrificing predictable behavior under roto-translations.

Computational tradeoffs. Equivariant attention is more expensive than EGNN-style updates because it must carry and transform higher-order features. This has motivated a spectrum of architectures trading expressiveness for efficiency, from EGNN to TFN and SE(3)-Transformers, and newer interatomic potential models such as NequIP that show substantial data efficiency gains from $E(3)$-equivariance (**?**).

7 Where these architectures appear in constrained generation and control

Equivariance has become a default design choice in several constrained-aware generative pipelines, because it aligns the learned prior with physical symmetries of the task.

Protein structure and design. Modern protein structure prediction includes geometry-aware attention and equivariant coordinate updates. AlphaFold introduces "invariant point attention" and an equivariant structure module, reflecting the centrality of SE(3) structure in proteins (**?**). In generative design, diffusion models over 3D backbones or frames typically rely on SE(3)-equivariant score networks, because denoising steps must respect global roto-translation symmetry (**??**).

Molecular docking and 3D generative chemistry. Docking is naturally defined on relative rigid-body transformations between a ligand and a receptor. Diffusion models such as DiffDock explicitly predict score components in translation and rotation spaces in a way that is consistent with SE(3) geometry, which can be read as an equivariance requirement on the learned vector fields (**?**).

Control on manifolds and Lie groups. Many control states live on manifolds: orientations on SO(3), poses on SE(3), and configurations with constraints. Encoding the correct transformation laws directly into policy and dynamics networks is often more robust than attempting to learn these properties from data. This viewpoint will recur when we study optimization-based control mechanisms (Lecture 6 and beyond), because respecting geometry can be interpreted as working on the right constraint set or quotient space rather than on an unconstrained Euclidean relaxation.

8 Summary: equivariance as a reusable architectural primitive

Equivariance is a hard architectural constraint: it restricts the model class so that symmetries hold for every input, not just on average. This lecture framed CNNs, GNNs, EGNNs, and SE(3)-equivariant attention as a single design pattern: choose a group $\mathcal{G}$ that captures domain symmetries, define its actions on inputs and outputs, and build layers satisfying $f(g \cdot x) = g \cdot f(x)$.

In the broader "constrained-aware" view of the course, equivariance complements optimization and inference-time mechanisms. It moves a subset of constraints from the verifier or solver into the parametric form of the generator itself. Next, we return to optimization primitives that enforce general constraints beyond symmetry, including projections, penalties, and proximal operators.

References

M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

T. S. Cohen and M. Welling. Group equivariant convolutional networks. In *International Conference on Machine Learning (ICML)*, 2016.

M. Weiler and G. Cesa. General $E(2)$-equivariant steerable CNNs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

M. Zaheer, S. Kottur, S. Ravanbhakhsh, B. Póczos, R. Salakhutdinov, and A. Smola. Deep sets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

V. G. Satorras, E. Hoogeboom, and M. Welling. $E(n)$ equivariant graph neural networks. In *International Conference on Machine Learning (ICML)*, 2021.

N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley. Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

F. B. Fuchs, D. E. Worrall, V. Fischer, and M. Welling. SE(3)-transformers: 3D roto-translation equivariant attention networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

S. Batzner, A. Musaelian, L. Sun, M. Geiger, J. P. Mailoa, M. Kornbluth, N. Molinari, T. E. Smidt, and B. Kozinsky. $E(3)$-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *arXiv preprint arXiv:2101.03164*, 2021.

J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Zidek, A. Potapenko, and *others*. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596:583–589, 2021.

J. L. Watson, D. Juergens, N. B. Bennett, B. L. Trippe, J. Yim, H. E. Eisenach, J. Ahern, A. Borromaeo, A. Carter, C. Dovas, and *others*. De novo design of protein structure and function with RFdiffusion. *Nature*, 620:1089–1100, 2023.

J. Yim, B. L. Trippe, V. De Bortoli, E. Mathieu, A. Doucet, R. Barzilay, and T. Jaakkola. SE(3) diffusion model with application to protein backbone generation. *arXiv preprint arXiv:2302.02277*, 2023.

G. Corso, H. Stärk, C. J. Ying, R. Barzilay, and T. Jaakkola. DiffDock: Diffusion steps, twists, and turns for molecular docking. In *International Conference on Learning Representations (ICLR)*, 2022.