Symmetry-Based Structured Matrices for Efficient Approximately Equivariant Networks

Ashwin Samudre[*] **Mircea Petrache**[†] **Brian D. Nord**[‡] **Shubhendu Trivedi**[§]

Abstract

There has been much recent interest in designing symmetry-aware neural networks (NNs) exhibiting relaxed equivariance. Such NNs aim to interpolate between being exactly equivariant and being fully flexible, affording consistent performance benefits. In a separate line of work, certain structured parameter matrices—those with displacement structure, characterized by *low displacement rank* (LDR)—have been used to design small-footprint NNs. Displacement structure enables fast function and gradient evaluation, but permits accurate approximations via compression primarily to classical convolutional neural networks (CNNs). In this work, we propose a general framework—based on a novel construction of symmetry-based structured matrices—to build approximately equivariant NNs with significantly reduced paramter counts. Our framework integrates the two aforementioned lines of work via the use of so-called Group Matrices (GMs), a forgotten precursor to the modern notion of regular representations of finite groups. GMs allow the design of structured matrices—resembling LDR matrices—which generalize the linear operations of a classical CNN from cyclic groups to general finite groups and their homogeneous spaces. We show that GMs can be employed to extend all the elementary operations of CNNs to general discrete groups. Further, the theory of structured matrices based on GMs provides a generalization of LDR theory focussed on matrices with cyclic structure, providing a tool for implementing approximate equivariance for discrete groups. We test GM-based architectures on a variety of tasks in the presence of relaxed symmetry. We report that our framework consistently performs competitively compared to approximately equivariant NNs, and other structured matrix-based compression frameworks, sometimes with a one or two orders of magnitude lower parameter count.

1 Introduction

Over the past few years, the incorporation of symmetry in neural networks through *group equivariance* [1] has started to mature as a fruitful line of research [1–14]. Such networks involve implementing a generalized form of convolution over groups—since linear equivariant maps over general fields are necessarily convolutional in nature [7, 4, 3], but can also be designed using ideas from invariant theory [15], or by using non-linear equivariant maps via attention mechanisms [16–18]. Successful applications have ranged over a dozen areas, involving multiple data types [19–37].

[*]School of Computing Science, Simon Fraser University, Burnaby, Canada. ashwin_samudre@sfu.ca

[†]UC Chile, Fac. de Matemáticas, & Inst. de Ingeniería Matematica y Computacional, Av. Vicuña Mackenna 4860, Santiago, 6904441, Chile. mpetrache@mat.uc.cl.

[‡]Fermi National Accelerator Laboratory, Batavia, IL 60510; Department of Astronomy and Astrophysics, University of Chicago, Chicago, IL 60637; Kavli Institute for Cosmological Physics, University of Chicago Chicago, IL 60637. nord@fnal.gov

[§]Independent. shubhendu@csail.mit.edu

Despite the successes of equivariant networks, several challenges remain. For instance, it has been demonstrated empirically [38–41] that a strict equivariance constraint could harm performance—symmetry in real world tasks is rarely perfect and data measurements could be corrupted or systematically biased. Indeed, it is reasonable to expect that there could often be a mismatch between the symmetry that the model encodes and that the data possesses. A natural modeling paradigm suggested by [38–41] prescribes the design of more *flexible* models that can calibrate the level of equivariance based on the data or task. This observation has been supported by a growing body of recent empirical [42–46] and theoretical work [47, 48].

In this paper we study the principled design of approximately equivariant NNs with significantly reduced parameter counts and the attendant computational benefits. For this purpose, we examine a line of work independent of that on equivariant NNs, which is primarly concerned with designing compact deep learning models using structured matrix representations of fast transforms (see [49] and the references therein). One class of popular methods explicitly involves the design of computationally- and memory-effective structured matrices, intended to replace dense weight matrices in NNs [50–52]. Yet another class of methods, proposed in the seminal works of [53, 54], instead operate with traditional structured matrices: Vandermonde (including polynomial and Chebyshev Vandermonde), Hankel, Toeplitz, Cauchy, Pick etc. Such matrices are a key object in engineering, especially in control theory, filtering theory, and signal processing [55]. The special property of these matrices is that their compositions don't inherit their structure *exactly*; however, they do so in an *approximate sense*. This approximation is measured by the classical notion of the *displacement rank* proposed by Kailath and co-workers [56–58]—such matrices are said to possess *displacement structure*, and a low degree of error on composite operations is indicated by a *low displacement rank* (LDR). While [53, 54] demonstrated that LDR matrices could be used to design efficient and compact NNs, they were only concerned with classical MLPs and CNNs and do not obviously extend to equivariant NNs.

We start with the observation that general cyclic matrices—also known as circulant matrices, a sub-class of LDR matrices—are used to represent classical circular convolution[5]. It has already been observed by [54] that these readily yield a notion of approximate convolution (and approximate equivariance) in classical CNNs. It is this property, which is dependent on cyclic matrices being LDR, that [54] exploit to build low-resource CNNs. With motivation from the theory of regular representations, we use the somewhat forgotten notion of group matrices (GMs)[6], and obtain a new family of symmetry-based structured matrices that can be used to model convolution for general discrete groups. We first develop a generalization of classical CNNs using GMs by building analogues for each of their elementary operations. Then, simplifying older works on group matrices [60, 61], we show that our formalism permits a principled notion of approximate equivariance for general discrete groups. In fact, somewhat analogous to [54], we generalize LDR theory from cyclic matrices to their analogues for general discrete groups—thus giving a handle on quantifying error to exact equivariance. To align our exposition with that of modern equivariant NNs, we show how our framework naturally generalizes to the homogeneous spaces of discrete groups, as well as to compactly supported data on infinite discrete groups. On a variety of tasks, we show that our proposed method is able to consistently match or outperform baselines, both from approximately equivariant NNs and structured matrix-based frameworks, often with an order of magnitude or more, reduced parameter count.

This paper brings together the above mentioned lines of work: equivariant NNs, approximately equivariant NNs, and structured matrix-based compression approaches. The result is a formalism that allows the construction of symmetry-aware, approximately equivariant, and parameter-efficient NNs with competitive performance. We summarize the main contributions of our paper below:

- We develop a formalism for constructing equivariant networks for general discrete groups using the notion of group matrices (GMs). We show how all the elementary operations in classical CNNs, such as taking strides and pooling, can be generalized to discrete groups using group matrices. The resulting networks—GM-CNNs—involve the use of certain symmetry-based structured matrices which facilitate the construction of light-weight equivariant NNs. Further, the generalized pooling operation involves a form of group coarsening and is relevant to the development of group equivariant autoencoders.

[5]Or convolution over cyclic groups.

[6]Group matrices are a precursor to the modern notion of regular representations, and pre-date modern group theory. For a history, we direct the reader to [59]

- We present a simple procedure to construct GMs for larger discrete groups which are composed of direct products or semi-direct products of cyclic or permutation groups, for which GMs are easy to compute.
- We show that the GM-CNN formalism naturally permits a principled implementation of approximately equivariant group CNNs. Further, we connect GMs to classical low displacement rank (LDR) theory, generalizing the theory from the specific case of LDR matrices with cyclic structure (and thus cyclic groups) to discrete groups. We use the developed theory to quantify error from exact equivariance.
- We show how our proposed framework easily extends to homogeneous spaces of discrete groups, as well as to compactly supported data on infinite discrete groups.
- Across a variety of tasks, we demonstrate that our framework consistently returns competitive performance, often with a significantly reduced parameter count, as compared to various approximately equivariant networks, and structured matrix-based frameworks. Our code is available at: https://github.com/kiryteo/GM-CNN

2 Preliminaries and Formal Setup

2.1 Matrices with Displacement Structure and Compressed Transforms in Deep Learning

The work of [54, 53] considers traditional families of structured matrices—Hankel, Toeplitz, Vandermonde, Cauchy—to build compressed representations of MLPs and classical CNNs. In this section, we briefly describe the main idea. A matrix $M \in \mathbb{R}^{m\times n}$ could be called *structured* if it can be represented in much fewer than mn parameters. Examples include matrices where the elements have a simple formulaic relationship with other elements. The displacement operator approach, pioneered by Kailath *et al.* [56, 57], consists of representing M via matrices $A \in \mathbb{R}^{m\times m}$ and $B \in \mathbb{R}^{n\times n}$, which define a linear map known as the **displacement operator**, as follows:

Definition 2.1 (Sylvester-type Displacement Operator:)**.** *Let A and B be fixed, not necessarily square matrices with compatible dimensions. The Sylvester-type displacement operator, denoted as $\nabla_{A,B}(M)$, is defined as the linear map:*

$$\nabla_{A,B}(M) : M \mapsto AM - MB,$$

where the difference $AM - MB$ is the *residual* R. The rank of the matrix R is called the *displacement rank* (DR). Further, the recovery of M is immediate from A, B, R.

Remark. An alternative popular formulation of the displacement operator is the Stein-type displacement operator, which defines $\Delta_{A,B}(M) : M \mapsto M - AMB$. For most purposes, the two formulations can be treated interchangeably [55].

The original formulation of DR was in the context of solutions of certain least square estimation problems of systems written in state-space form [56, 57]. It was originally stated for matrices that were *Toeplitz-like*, which weren't *exactly* shift-invariant, only *approximately* so. The formulation in [56] showed that a Toeplitz-like matrix would be LDR. Such results were later shown for all the classes of structured matrices listed above [57, 55]. More importantly, many composite operations on LDR matrices—including transpose/inverse, addition, composition/multiplication, taking direct sums to construct a larger block matrix—still resulted in LDR matrices (see proposition 1 on closure in [54]). It is this property that was used by [53, 54] to construct compressed representations of NNs. A particular form of the Toeplitz matrix, the circulant matrix, could be seen as implementing the usual circular convolution (see 2.2 below). Thus, a circulant-like matrix[7] would implement an approximate convolution (and thus approximate equivariance), and stacking them together in a NN would preserve the property of approximate equivariance. This point was noted and proved in section 4 of [54] and was the basis for building approximately shift-equivariant CNNs in [54]. In the example below, we illustrate how working with circulant matrices corresponds to a convolution on the group C_n.

Example 2.2 (Circulant Matrices). Note that for $A \in \mathbb{R}^{n\times n}$ equal to the canonical cyclic permutation P on n elements and $B = P^{-1}$, we have $\nabla_{P,P^{-1}}(M) = 0$ iff $\Delta_{P,P^{-1}}(M) = 0$, or iff M is circulant. Then, we can interpret the entries of $v \in \mathbb{R}^n$ as encoding a function $f : C_n \to \mathbb{R}$ over the cyclic group

[7]A circulant-like matrix is circulant matrix to which an extremely sparse noise matrix is added. A circulant matrix implements circular convolution, but a matrix with some noise will implement approx. convolution.

$C_n = \mathbb{Z}/n\mathbb{Z}$. Thus, for circulant M with first row $(m_{11}, \ldots, m_{1n})$ encoding map $\phi : C_n \to \mathbb{R}$, the multiplication $v \mapsto Mv$ corresponds to group-convolution operation $f \mapsto \phi \star f$ on the group C_n.

The above exposition provides a tantalizing hint that building compressed approximately equivariant NNs in the spirit of [54, 53] could be a reasonable goal for general discrete groups—but we would need to identify special structured matrices for each group and ensure they obey an analogous LDR property as above. In the next section, we state group convolution and show it could be written in terms of special matrices called group matrices, which will prove key for us to achieve our goal.

2.2 Group Matrices, Group Convolutions, and CNNs

In this section, we present our formulation of GM-CNNs. We first define group matrices, classical group convolution and reformulate group convolution in terms of GMs. Following which we show GMs naturally allow to generalize all the elementary operations of classical CNNs, including strides and pooling, to any discrete group. We also provide a procedure to construct GMs of certain larger discrete groups efficiently. Finally, we also show that GMs enjoy a natural set of closure properties.

For ease of exposition, we describe the framework in this section using images as inputs, but it readily generalizes to general data types as will become clear later. In particular, we consider images of size $n \times n$, modulo the periodicity structure of each side, which is identified with the opposite side of the image. Such images could be considered as functions over copies of the group $G = C_n \times C_n$, which permits representing convolutions (as appearing in classical CNNs[8] as a group convolution on G.

Since we seek a framework that uses symmetry-based structured matrices towards specific ends, we depart from the customary treatment of group convolutions that uses *group representations*. We instead use the formalism of *group matrices* (GMs) of discrete groups; while the GM formalism has a long history, it has mostly been forgotten and has not yet been used to represent general group convolutions. We first define the requisite notions of group matrices and group diagonals:

Group matrices. If G is a group with cardinality N, then a matrix $M \in \mathbb{R}^{N\times N}$ is a *group matrix* of G if there exists a labeling of the rows and columns of M by elements of G such that whenever $gh = g'h'$ (with $g, h, g', h' \in G$) we have an identification of entries labeled by $(g, h), (g', h')$ i.e. that $M_{gh} = M_{g'h'}$.

Group diagonals. For $g \in G$ the *group diagonal* matrix B_g associated with g is a particular group matrix with entries in $\{0, 1\}$, whose entries are defined as

$$(B_g)_{h,h'} = \delta(h = gh'), \tag{1}$$

where, $\delta(h = gh')$ equals 1 if $h = gh'$, and 0 otherwise (similar to the Kronecker delta notation). We observe that **group diagonals form a basis of the space of group matrices with entries in $\mathbb{X} = \mathbb{R}$**. B_g are extremely sparse matrices with exactly one non-zero entry per row. Thus, we can store them as arrays of size $|G|$, listing only the column indices of these entries, rather than as $|G| \times |G|$ matrices.

Group convolution. For standard group convolution, if G is a finite group and $\phi, \psi : G \to \mathbb{R}$, then convolution of these functions is

$$\phi \star \psi(x) := \sum_{g\in G} \phi(g)\psi(g^{-1}x) = \sum_{(g,h)\in G\times G:\ gh=x} \phi(g)\psi(h). \tag{2}$$

In the second expression, we write the set of pairs $(g, g^{-1}x), g \in G$ implicitly as pairs $(g, h) \in G \times G$ satisfying the condition $hg = x$. Observe that GMs permit a simple formulation for group convolutions for finite groups because they encode the group operation via a linear operator h.

We next use GM formulations to produce general versions of common group convolutional network choices and operations: convolutions, small kernels, strides, and pooling.

Group convolution using group matrices. We can re-express group convolution in the language of group matrices and group diagonals as follows. Consider the functions $\phi, \psi : G \to \mathbb{R}$ as vectors in $\overrightarrow{\phi}, \overrightarrow{\psi} \in \mathbb{R}^G$, where the vector entries are indexed by G – e.g. $\overrightarrow{\phi}_g := \phi(g)$. In this case, the linear

[8]To be fully precise, in fact, classical CNNs use convolution with *infinite* group $\mathbb{Z} \times \mathbb{Z}$, which gives some extra technical difficulties as we need to keep track of image finite supports. This is solved in classical CNNs by padding. Later, in §C.2 we show how to extend our framework to general discrete infinite groups.

transformation $\overrightarrow{\psi} \mapsto \overrightarrow{\phi \star \psi}$ is expressed by a simple matrix. Moreover, it is sufficient to combine group diagonals B_g with coefficients encoded in the entries of ψ_g to define convolution as:

$$\mathsf{Conv}_\psi \overrightarrow{\phi} := \overrightarrow{\phi \star \psi} = \sum_{g \in G} \phi(g) B_g \overrightarrow{\psi}. \tag{3}$$

The verification of (3) is straightforward: using definitions (2) and (1), we have, for all $x \in G$,

$$\left[\overrightarrow{\phi \star \psi}\right]_x := \sum_{g,h: gh=x} \phi(g)\psi(h) = \sum_{g,h} \phi(g)\psi(h)\delta(gh = x) = \sum_g \phi(g) \left[B_g \overrightarrow{\psi}\right]_x = \left[\sum_g \phi(g) B_g \overrightarrow{\psi}\right]_x.$$

Choosing small kernels. In classical CNNs, kernels have support in a $k \times k$-neighborhood of the origin for small values of k: kernels are nonzero at pixels near the origin. The analogue of this choice for general finite groups G is to fix a word distance[9] and to restrict kernel coefficients $\overrightarrow{\phi}$ to be zero on elements of G at a distance larger than k from the origin. Consequently, if N_k is the number of elements in the radius-k neighborhood of the identity in G, then the sum from (3) only includes N_k non-zero summands, which is typically N_k orders of magnitude smaller than the cardinality $|G|$. Note that we are never required to compute the B_g matrices except for these N_k choices of g. This permits an efficient implementation analogous to classical CNNs.

Pooling operations: Group diagonals can be restricted to subgroups. Pooling and stride operations map input channels, written as functions $\psi : G \to \mathbb{R}$, to outputs of the form $\psi' : H \to \mathbb{R}$, indexed by a subgroup $H \subseteq G$. Convolution restricts naturally to H, a property that can be formulated in terms of group diagonals. The group diagonals B_h^H ($|H| \times |H|$ matrices with group H) acting over $\overrightarrow{\psi}'$ can be obtained from the B_h^G ($|G| \times |G|$ matrices with group G corresponding to elements $h \in H \subseteq G$) by removing rows and columns indexed by elements of $G \setminus H$. In particular, group diagonals are mapped to group diagonals under this operation, as a consequence of the following lemma:

Lemma 2.3. *If $h \in H$, then the rows of G-group diagonal matrix B_h labeled by elements of H, have entries 1 only in columns whose index labels belong to H.*

All proofs are relegated to the appendix.

Pooling operations on finite groups. As mentioned above, we define the pooling operation via a subgroup $H \subset G$, where pooling is done over subsets of size $n = |G|/|H|$. We now explicitly describe the operation. Consider right cosets $Hg_1, \ldots, Hg_n$. For each coset Hg_i, we select a coset representative at the closest word distance from the origin: $\bar{g}_i \in \mathsf{argmin}_{h \in H} \mathsf{dist}(hg_i, e)$. Note that for $h \in H$, the sets $P_h := \{h\bar{g}_i : 1 \le i \le n\}$ again form a partition of G. Then, the pooling can be defined as a coarsening operation, based on a function $\square \in \{\max, \mathsf{aver}, \ldots\}$ that fixes a way of associating to a finite set of elements of $\mathbb{R}$. Typical choices are $\square(X) = \max X$ for max-pooling and $\mathsf{aver}(X) := \frac{1}{|X|}\sum_{x \in X} x$ for average-pooling. We can then define $(H, \square)$-pooling as

$$\psi : G \to \mathbb{R} \quad \mapsto \quad \mathsf{Pool}_H^\square(\psi) : H \to \mathbb{R}, \quad \mathsf{Pool}_H^\square(\psi)(h) := \square\{\psi(g') : g' \in P_h\}. \tag{4}$$

In classical CNNs, for fixed pooling parameter k, the output value at pixel position (i, j) is set to be the maximum (or average) of pixel values at distance at $\le k$ from pixel (ki, kj) of the input.

Example 2.4. If the group is $G = C_N \times C_N$ with generators denoted as $(1, 0), (0, 1)$, and assuming that we can factorize $N = mn$ with $m, n \ge 2$, then we can focus on the subgroup $H \subseteq G$ isomorphic to $C_n \times C_n$, generated by elements $(m, 0), (0, m) \in G$. In this case, the cosets of H have the form

$$H(i, i') = \{(km + i, k'm + i') : 0 \le k < n\}, \quad \text{for} \quad 0 \le i, i' < m.$$

We can then directly take values (i, i') as above as the closest-to-identity representatives $\overline{g}_\alpha$. With this notion, the (H, aver)-pooling operation takes as input a signal of the form

$$\psi : \{0, \ldots, N-1\}^2 \to \mathbb{R}, \quad \text{with entries denoted} \quad \psi(a, a'), \quad 0 \le a, a' < N,$$

and maps it to a signal given by

$$\psi' : \{0, \ldots, n-1\}^2 \to \mathbb{R}, \quad \psi'(k, k') := \left[\mathsf{Pool}_H^{\mathsf{aver}} \psi\right](k, k') := \frac{1}{m^2} \sum_{0 \le i, i' < m} \psi(k + i, k' + i').$$

[9] For a finite group G w.r.t the generating set S, assign each edge of the Cayley graph a metric of length 1. Then the distance between $g, h \in G$ equals the shortest path length in the Cayley graph from vertex g to vertex h

Remark. In practice, successive layers to be applied to the output of a $\mathsf{Pool}_H^{\square}$-layer should use group matrices associated with group H. Such group matrices can be constructed as follows: take group diagonals B_g of $C_N \times C_N$, with g of the form $g = (km, k'm)$ only, of which we remove rows and columns whose index is not a multiple of m.

Implementing stride via subgroups. Adding stride after convolutional operations allows the user to manipulate signal sizes across the architecture. If the input is modeled by a finite group G, we consider a subgroup H, and apply the convolution kernel only at positions indexed by elements of H:

$$\mathsf{Conv}_\phi^H : \mathbb{R}^G \to \mathbb{R}^H, \quad \mathsf{Conv}_\phi^H \overrightarrow{\psi} := \sum_{g\in G} \psi(g) B_g^H \overrightarrow{\psi}, \tag{5}$$

in which B_g^H is the $|H| \times |G|$ matrix obtained by removing rows of B_g that are labeled by $G \setminus H$.

Obtaining group matrices of complex groups from simpler ones. In many applications, groups are products or semi-direct products of cyclic or permutation groups. For any finitely generated group, one can produce the group matrix using the group operation and an algorithm for bringing group elements to canonical form. However, this can be time-consuming: one would ideally use the group matrices of individual components and produce a group matrix of the larger group directly. To achieve this, we use the following procedure. Let G, H be two groups, and consider the product group $G \times H$. Then, if $g \in G, h \in H$, and B_g^G, B_h^H are the corresponding group diagonals, a group diagonal in $G \times H$ for element (g, h) is given by taking the Kronecker product:

$$B_{(g,h)}^{G\times H} = B_g^G \otimes B_h^H. \tag{6}$$

Consider a semi-direct product $G \rtimes_\phi H$. As usual, it requires defining a group homomorphism $\phi : H \to \mathrm{Aut}(G)$, and then group operation is defined by $(g, h) \cdot (g', h') := (g\ \phi_h(g'), hh')$. In this case, $g \mapsto \phi_h(g)$ induces a permutation of G, and thus we can associate to it a $|G| \times |G|$ permutation matrix P_h. Then it is direct to check that

$$B_{(g,h)}^{G\rtimes_\phi H} = (P_h B_g^G) \otimes B_h^H. \tag{7}$$

We have thus fully described generalizations of the classical CNN operations to general discrete groups via group matrices. We also showed that direct or semi-direct product groups of cyclic or permutation groups permit efficient computation of their group matrices.

Closure of group matrices under elementary operations. As reviewed in §2.1, closure properties of LDR matrices [54] were key to their use in NN compression [54]. Before we introduce error control and approximate equivariance, we note the following simple closure properties, proved in §A.2.

Proposition 2.5. *If M, M' are group matrices for group G, then M^T, M^{-1}, MM' are also group matrices for group G. If N is a group matrix for group H, then the Kronecker product $M \otimes N$ is a group matrix for the direct product group $G \times H$.*

3 Implementing Approximately Equivariant GM-CNNs

A number of researchers have recently argued, both empirically [38–41] and theoretically [47, 48], that imposing a hard equivariance constraint in NNs can be detrimental to accuracy; practitioners can benefit by relaxing equivariance requirements at minimal computational cost. Our GM-based convolutional formalism can be considered as only using a superposition of constant-value diagonals $\phi(g)B_g$ as dictated by the formula (3). We now propose a generalization that allows learnable weights beyond Section 2.2, thus permitting a principled implementation of approximately equivariant NNs.

General matrices in group-diagonal basis. We first show how general matrices can be written in a group-diagonal basis. Consider a general $|G| \times |G|$-matrix M, and let $B_g, g \in G$ be the group diagonals (1). Then we can always encode the entries from M via $|G|$-dimensional arrays $F_g, g \in G$:

$$M = \sum_{g\in G} \mathrm{diag}(F_g) B_g, \quad \text{where for } h \in G \quad (F_g)_h := M_{h,hg^{-1}}. \tag{8}$$

The validity of expression (8) follows directly from definition (1): the $(h,h')^{th}$ entry of B_g is $\delta(h=gh')$, and thus is nonzero (in fact $=1$) only if $g=h(h')^{-1}$. Using the definition (8) of F_g,

$$\left[\sum_{g\in G}\mathrm{diag}(F_g)B_g\right]_{h,h'}=\sum_{g\in G}(F_g)_h\;\delta(h=gh')=(F_{h(h')^{-1}})_h=M_{h,h(h(h')^{-1})^{-1}}=M_{h,h'},$$

and thus all the entries of the two sides of (8) coincide.

We can obtain the coordinates $F(M)$ associated with any linear operation $\overline{\psi}\mapsto M\overline{\psi}$, by shuffling the entries of M and reducing the matrix $F(M)$, with rows and columns labeled by G, and whose rows are the coefficients $F_g, g\in G$. We can interpret the columns of $F(M)$ as relative coefficients multiplied at position g of the group, and row entries as parameterizing the relative position:

$$\psi_{out}(\bar{g})=\sum_{g\in G}\psi_{in}(\bar{g}g^{-1})[F(M)]_{\bar{g},g}.$$

With the above, we get a formulation of a more general convolution (learned via M), permitting approximate equivariance, but expressed in terms of group matrices. We now connect this formulation to the theory of displacement structures and show that it corresponds to a LDR implementation.

Displacement operator for general groups. Let $\mathbf{b}$ represent the vector such that its i^{th} entry is the constant value associated with row i. Suppose if the matrix $F(M)$ has the form $F(M)=\mathbf{1}\otimes\mathbf{b}$, then it is equivalent to M being a group matrix, and the convolution described above reduces to an exact convolution as in (3). This property of $F(M)$ can be tested by taking the following difference:

$$\mathsf{D}(M)=\mathsf{D}_P(M):=F(M)-PF(M),\quad \text{where } P(x_1,\dots,x_N)=(x_2,\dots,x_N,x_1)\tag{9}$$

Then M is a group matrix (and thus encodes a group convolution) if and only if $\mathsf{D}(M)=0$.

In fact, the same property $\mathsf{D}(M)=0$ holds even if in defining D we apply *a different cyclic permutation for each row*. More explicitly, for each $g\in G$ we select a cyclic permutation $\sigma_g\in\mathsf{Perm}(G)$, and then set $\vec{P}:=(\sigma_g)_{g\in G}$ and define entrywise

$$\left[\mathsf{D}_{\vec{P}}(M)\right]_{g,g'}:=[F(M)]_{g,g'}-[F(M)]_{g,\sigma_g(g')}\,.\tag{10}$$

Displacement dimension and rank. When we increase expressivity by allowing a controlled error to equivariance, a natural metric for this control is via the dimension of the space of allowed matrices $\mathsf{D}(M)$ within a model. We define the *displacement dimension* of a subset $\mathcal{M}\subseteq\mathbb{R}^{|G|\times|G|}$ as:

$$\dim_{\mathsf{D}}(\mathcal{M}):=\dim_{\mathbb{R}}(\mathsf{Span}(\{\mathsf{D}(M):\ M\in\mathcal{M}\})).\tag{11}$$

Note that $\mathsf{dim}_{\mathsf{D}}(\mathcal{M})$ does not depend on the choice of $\vec{P}$ in (10), as the dimension considered in (11) can be computed by summing the dimensions of spans row by row, and that row spans for $\mathsf{D}_{\vec{P}}(M), M\in\mathcal{M}$ do not depend on the choice of permutations σ_g.

Another metric for measuring the discrepancy of a matrix M from being a group matrix, is the *displacement rank* of M, defined as

$$\mathsf{DR}(\mathcal{M}):=\mathrm{rank}(\mathsf{D}(M)).\tag{12}$$

For further discussion about this notion of displacement rank and connections to the classical generalization of displacement rank as introduced in [60, 61], see Appendix B.

Low displacement rank implementations. In order to introduce errors to equivariance, we use kernels, encoded as matrices M whose displacement matrices $\mathsf{D}(M)$ have low rank. A simple choice is to add a matrix with r learnable vector columns $\mathbf{a}_{g_i}, 1\le i\le r$ to the a group matrix M:

$$F(M)=\mathbf{1}\otimes\mathbf{b}+\sum_{i=1}^{d}\mathbf{a}_{g_i}\otimes\mathbf{1}.\tag{13}$$

Then it is direct to verify that $\mathrm{rank}(\mathsf{D}(M))=\dim(\mathrm{Span}(\mathbf{a}_{g_1},\dots,\mathbf{a}_{g_r}))\le r$, and that if $\mathcal{M}$ is the space of matrices of the form (13) then $\dim_{\mathsf{D}}(\mathcal{M})=|G|r$.

Quantifying equivariance error under elementary operations. As a counterpart to the closure properties of Prop. 2.5 for group matrices, it is interesting to quantify control on how a bound on the error to equivariance (or to "being a group matrix") behaves under the same operations.

A first approach is to use displacement-based structural metrics such as $\mathrm{DR}(\mathcal{M})$ or $\dim_{\mathrm{D}}(\mathcal{M})$ and ask if they behave the same way on composite operations between matrices of the same class. Such a control for the case of classical LDR is available in [54, Prop. 1]. However we found that the natural displacement operator D, in general, has a very complicated behavior under matrix products, and we were not able to extend the natural bounds for deterioration under multiplication from [54].

On the other hand, rather than demanding structural/algebraic control, a quantitative control over the error to equivariance may be more useful in practice, for equivariance error bounds. A natural quantification is to measure the distance of a matrix from the set of G-group matrices:

$$\mathrm{dist}(M, \mathcal{GM}) := \min\{\|M - M_0\| : \ M_0 \in \mathcal{GM}\}, \tag{14}$$

in which $\|\cdot\|$ is Frobenius norm for matrices. We then have the following properties, proved in §A.3:

Proposition 3.1. *Let M, M' be $|G| \times |G|$-matrices and let $\mathcal{GM} = \mathcal{GM}^G$ be the set of G-group matrices. For a second group H, let N be an $|H| \times |H|$-matrix and $\mathcal{GM}^H, \mathcal{GH}^{G\times H}$ be the set of group matrices with group $H, G \times H$ respectively. Then*

1. $\mathrm{dist}(M, \mathcal{GM}) = \mathrm{dist}(M^T, \mathcal{GM})$.
2. $\mathrm{dist}(MN, \mathcal{GM}) \leq \max\{\|M\|, \|N\|\} \left(\mathrm{dist}(M, \mathcal{GM}) + \mathrm{dist}(M', \mathcal{GM})\right)$.
3. $\mathrm{dist}(M \otimes N, \mathcal{GM}^{G\times H}) \leq \max\{\|M\|, \|N\|\}(\mathrm{dist}(M, \mathcal{GM}^G) + \mathrm{dist}(N, \mathcal{GM}^H))$.

Finally, as discussed in Section 1, our framework generalizes to not just discrete groups, but also to their homogeneous spaces. Due to space limitations, we describe the extension to homogeneous spaces in appendix C. We also show that the framework extends to finitely supported data on infinite discrete groups in C.2. The error to perfect equivariance for this general setup is quantified in C.3.

4 Experimental Results

GMConv and GMPool operations: Our architecture is built around GMConv and GMPool operations, which are designed to handle group-based interactions and symmetry-preserving pooling, respectively. The GMConv operation uses group matrices to define convolutional kernels, where the neighborhood parameter controls the extent of local interactions based on the word distance idea described in section 2.2 to implicitly set the kernel size. For cyclic groups, a neighborhood radius of 1 results in 3 group entries $[x-1, x, x+1]$ in a cyclic group, while for direct product of two such groups, it results in 9 learnable parameters. Generally, a GMConv layer with a neighborhood size k comprises $(2k + 1) \times (2k + 1) \times$ number of channels learnable parameters. In error addition experiments, we introduce an additional error matrix with same structure as the initial group matrix, doubling parameters from p to $2p$. This formulation allows learning approximate equivariance. GMPool (Section 2.2) performs pooling by leveraging predefined group structures, which are constructed by generating group elements, forming their Kronecker products, and calculating subgroup cosets. These cosets and their corresponding tensor indices are precomputed to streamline the pooling process. During the forward pass, GMPool efficiently utilizes these precomputed indices to select elements for max or mean pooling, ensuring the operation remains consistent with the group symmetry. Before presenting our results on different datasets, we would like to note that following [41], we also perform an equivariance error analysis, which can be found in appendix D.

4.1 Dynamics Prediction

We evaluate our framework on two dynamics prediction tasks: the smoke plumes task (Plumes) and the Jet Flow task (MJetFlow) as described in [41]. For Plumes, each model takes sequences of 64×64 crops of a smoke simulation generated by PhiFlow [62] as input to predict the velocity field for the next time step. The evaluation is conducted under two settings: “Future", where we evaluate on the same portion of the simulation, but predict future time steps which are not included in training. The second setting is “Domain" where the evaluation is done on the same time steps but at different spatial locations. The data are collected from simulations with different inflow positions and buoyant forces. The JetFlow task comprises 24 subregions of jet flows sized 62×23, as described in [41] following a similar evaluation protocol. Our method is not inherently steerable; but we still compare it against various steerable baselines despite putting it at a disadvantage. The baselines also include an ordinary MLP and a CNN. Others include an E2-CNN (Equiv) [63], two approximately equivariant networks e.g. RPP [64] and LIFT [65].

Experimental setup: Our architecture consists of four GMConv layers with 128 channels each, utilizing PReLU activations and residual connections. We maintain spatial resolution throughout the network by avoiding pooling operations. A final 1×1 convolution maps the features to the desired output channels. The neighborhood parameters are set to 4 for cyclic groups and 2 for dihedral groups to optimize local interactions. We initialize weights using Kaiming initialization. The model is trained using the AdamW optimizer with learning rates between 0.003 and 0.009, a weight decay of 0.0153, and a ReduceLROnPlateau scheduler (factor 0.7, patience 8) on a mean squared error loss. Training is conducted with a batch size of 256 for up to 100 epochs, employing early stopping (patience 6) based on validation accuracy to prevent overfitting. We utilize PyTorch for our experiments and the hyperparameters are fine-tuned for each dataset.

Run times: Using an Nvidia L40S GPU (48GB RAM), forward pass times are approximately 1 second for JetFlow (62×23 samples) and 1.4 seconds for Plumes (64×64 samples), reflecting the complexity of these tasks. FLOPS are detailed in Tables 1 and 2.

Results on JetFlow: Table 1 shows GM-CNN achieves top performance (tied with RGroup) on the translation task, using the fewest parameters (26,325). Standard Conv and Lift use 51,548 and 1,994,818 parameters respectively. For the rotation task, GM-CNN performs slightly better than ECNN while using only about 10% of the parameters (29,583 vs 304,128). It also outperforms Lift and RSteer. Finally, for the scaling task, GM-CNN's performance is comparable to the best performers (Rpp and RSteer) while using only a fraction of their parameters (26,325 compared to 1,421,832 for Rpp and 6,742,530 for RSteer). These results highlight the competitive performance of our method despite not being inherently steerable.

Results on Plume: As seen in Table 2, GM-CNN consistently uses the fewest parameters (19,267) across all tasks while achieving competitive results. For the translation task, GM-CNN's performance is close to the best-performing RGroup, despite using 25 times fewer parameters. In Rotation, it competes well with top performing methods like Lift and RSteer while maintaining its dramatic parameter efficiency advantage. Lastly, for scaling, GM-CNN is competitive, with RSteer achieving only slightly better results at the cost of 350 times more parameters.

Table 1: RMSE on Jet Flow dataset. See text for details [41]. FLOPS are $\times 10^{10}$.

Model	Translation				Rotation				Scaling		
	`Conv`	`Lift`	`RGroup`	`GM-CNN`	`E2CNN`	`Lift`	`RSteer`	`GM-CNN`	`Rpp`	`RSteer`	`GM-CNN`
Future	0.22 ± 0.06	0.17 ± 0.02	0.15 ± 0.00	0.15 ± 0.01	0.21 ± 0.02	0.18 ± 0.02	0.17 ± 0.01	0.16 ± 0.01	0.16 ± 0.06	0.14 ± 0.01	0.15 ± 0.01
Domain	0.23 ± 0.06	0.18 ± 0.04	0.16 ± 0.01	0.17 ± 0.01	0.27 ± 0.03	0.21 ± 0.04	0.16 ± 0.01	0.18 ± 0.02	0.16 ± 0.07	0.15 ± 0.00	0.17 ± 0.01
Params	51548	1994818	53798	26325	107136	1915872	961538	29583	1421832	6742530	26325
FLOPS	0.007	0.023	0.007	0.006	0.185	0.015	0.922	0.008	0.141	0.024	0.006

4.2 Comparison with structured matrix baselines [53, 54]

We evaluate GM-CNNs on a variety of image datasets and compare them against a set of competitive baselines—the methods reported in [54] still remain amongst the most competitive. We compare our methods on two dimensions: accuracy of the models and the total number of parameters. In addition to datasets from [54], we further consider the SmallNORB [66], Rotated MNIST, and Rectangles datasets [67]. SmallNORB is a condensed version of the NORB dataset [66], and is specifically tailored for object recognition tasks focusing on shape. The Rotated MNIST dataset comprises MNIST data samples that have been randomly rotated, offering a variation that lacks the noise typically introduced in the MNIST-bg-rot [67] dataset. The Rectangles dataset, a compact binary classification dataset, distinguishes rectangles based on whether their width or height is greater. For these image classification tasks, the input dimensions range from 24×24 to 32×32. While these are

Table 2: RMSE on the Plumes dataset. See text for details [41]. FLOPS are $\times 10^{10}$.

Model		MLP	Conv	Equiv	Rpp	Combo	Lift	RGroup	RSteer	GM-CNN
Translation	F	1.56$_{\pm 0.08}$	—	0.94$_{\pm 0.02}$	0.92$_{\pm 0.01}$	1.02$_{\pm 0.02}$	0.87$_{\pm 0.03}$	**0.71$_{\pm 0.01}$**	—	0.81$_{\pm 0.02}$
	D	1.79$_{\pm 0.13}$	—	0.68$_{\pm 0.05}$	0.93$_{\pm 0.01}$	0.98$_{\pm 0.01}$	0.70$_{\pm 0.00}$	**0.62$_{\pm 0.02}$**	—	0.74$_{\pm 0.01}$
	Params	8678240	—	1821186	7154902	3683332	8235362	1921158	—	19267
	FLOPS	0.001	—	0.745	0.002	1.492	0.145	0.756	—	0.003
Rotation	F	1.38$_{\pm 0.06}$	1.21$_{\pm 0.01}$	1.05$_{\pm 0.06}$	0.96$_{\pm 0.10}$	1.07$_{\pm 0.00}$	0.82$_{\pm 0.08}$	0.82$_{\pm 0.01}$	**0.80$_{\pm 0.00}$**	0.93$_{\pm 0.02}$
	D	1.34$_{\pm 0.03}$	1.10$_{\pm 0.05}$	0.76$_{\pm 0.02}$	0.83$_{\pm 0.01}$	0.82$_{\pm 0.02}$	0.68$_{\pm 0.09}$	0.73$_{\pm 0.02}$	**0.67$_{\pm 0.01}$**	0.79$_{\pm 0.02}$
	Params	8678240	1821186	1198080	5628298	431808	1801748	1883536	7232258	19267
	FLOPS	0.001	0.745	2.965	3.966	0.585	2.955	3.087	1.986	0.003
Scaling	F	2.40$_{\pm 0.02}$	0.83$_{\pm 0.01}$	0.75$_{\pm 0.03}$	0.81$_{\pm 0.09}$	0.78$_{\pm 0.04}$	0.85$_{\pm 0.01}$	0.76$_{\pm 0.04}$	**0.70$_{\pm 0.01}$**	0.79$_{\pm 0.02}$
	D	1.81$_{\pm 0.18}$	0.95$_{\pm 0.02}$	0.87$_{\pm 0.02}$	0.86$_{\pm 0.05}$	0.85$_{\pm 0.01}$	0.77$_{\pm 0.02}$	0.86$_{\pm 0.12}$	**0.73$_{\pm 0.01}$**	0.82$_{\pm 0.01}$
	Params	8678240	1821186	1744774	3966984	1059270	2833558	1275266	2427394	19267
	FLOPS	0.001	0.745	0.368	0.507	0.326	0.016	0.132	0.041	0.003

small datasets, they capture a wide range of variation, but more importantly, they permit comparison with existing structured matrix baselines.

Experimental setup: Our base architecture consists of two GMConv layers (120 channels each) with residual connections, LayerNorm without learnable parameters, and PReLU activations. After the GMConv layers, an adaptive max pooling layer is applied, followed directly by a single fully connected layer that performs classification with the cross-entropy loss function. We experiment with GMConv layers with a neighborhood of 3, incorporating error addition, and with GMPool. We also include experiments for different neighborhood sizes. Training is conducted with a batch size of 1024 for a maximum of 100 epochs, with early stopping to prevent overfitting. Other hyperparameters are consistent with those used in our dynamics prediction experiments. For pooling-based experiments, we employ a GM-CNN architecture with 3 GMConv layers (44, 44, and 56 output channels).

Run times: Forward pass times vary by input size: $\sim$ 0.6s for 24x24 images (smallNORB), $\sim$ 0.7s for 28x28 (MNIST variants), and $\sim$ 0.8s for 32x32 (CIFAR-10, NORB). These are on par for all the competing methods. Note that our method can be sped up significantly with custom GPU implementations for handling structured matrices.

Results. The results are presented in Table 3. GM-CNN variants demonstrate strong performance across all datasets. Our approach, which combines pooling and error addition with a neighborhood parameter set to 1, achieves the best results on the MNIST-bg-rot, MNIST-noise, and Rectangles datasets with minimal parameters. On CIFAR-10, all GM variants outperform other methods. For NORB and SmallNORB, GM-CNN with a neighborhood size of 3 and error addition shows superior performance. For Rotated MNIST, the pooling and error addition approach with a neighborhood size of 3 leads in performance. These performances are achieved with significantly fewer parameters. Note that the numbers for CIFAR-10 are much lower than state-of-the-art results, which we attribute to the significantly lower parameter counts.

5 Conclusion

In this paper, we presented the development of a novel formalism (GM-CNNs) for constructing equivariant networks for general discrete groups using group matrices, generalizing all the elementary operations of classical CNNs. GM-CNNs employ the use of a novel family of symmetry-based structured matrices, also developed via our formalism, which facilitates the construction of lightweight equivariant NNs. Further, we presented a principled implementation of approximately equivariant group CNNs using our formalism. Connecting group matrices to classical displacement structure theory, we provide a generalization of the theory for LDR matrices with cyclic structure to discrete groups. Moving beyond discrete groups, we provide an extension of GM-CNNs to homogeneous spaces and infinite discrete groups. Finally, we tested our proposed formalism on a variety of different tasks, and show that GM-CNNs can be consistently competitive, while being significantly more parameter efficient, compared to approximately equivariant NNs and structured matrix-based frameworks. For future work, it would be interesting to extend our formulation for continuous groups, and enabling our setup to be steerable. Further, exploring the group tensorization operations proposed in [68] (e.g. theorem 1.1), could help improve the scalability of our method.

Method ↑	**M-bg-rot**	**M-noise**	**CIFAR-10**	**NORB**	**SmallNORB**	**Rect**	**Rot-MNIST**
LDR-TD (r=1)	45.81 14122	78.45 14122	45.33 18442	62.75 14342	83.23 14122	98.53 14122	79.82 14122
GM (n=3)	30.07 12483	80.14 13441	58.31 13681	54.63 12967	79.01 12725	99.60 12483	83.78 48.7k
GM (n=3, E)	49.07 24734	82.55 25213	**58.29** 25213	**70.59** 24734	**85.22** 24546	99.31 24244	83.26 24734
GM (n=1, P + E)	**55.29** **5915**	**90.20** **5915**	58.07 **6003**	67.84 **5687**	80.90 5573	**99.89** **5915**	78.77 5915
GM (n=2, P + E)	53.94 **8701**	88.84 **8701**	55.10 8819	67.72 8503	78.61 8359	99.86 **8701**	79.06 8731
GM (n=3, P + E)	53.88 14747	84.92 14747	57.14 14482	66.99 14519	83.02 14482	99.67 14747	**84.50** 14747
Low-rank[67](r=4)	35.67 14122	52.25 14122	32.28 18442	43.66 14342	78.05 6916	87.48 7842	54.87 14122
Fastfood[68]	38.13 10202	63.55 10202	39.64 13322	59.02 9222	73.38 5380	89.81 10202	58.14 10202
Circulant[69]	34.46 8634	65.35 8634	34.28 11274	46.45 7174	71.23 3456	88.92 8634	52.22 8634

Table 3: Test accuracy and parameter count for GM-CNNs. In the "Method" column, **n** denotes the neighbourhood size, **P** suggests the use of pooling and **E** suggests error addition. The best accuracy is in blue, and the second best in red. If GM-CNN achieves best or second-best accuracy **with fewer parameters**, then we mark the number of parameters in **bold**. Clearly, we see that GM-CNN methods consistently provide the most accurate and usually the lowest parameter count. In cases where the best GM-CNN model has the best accuracy, but not the lowest parameter count, the second best GM-CNN model has comparable accuracy, but significantly lower parameter count.

Acknowledgments and Disclosure of Funding

This work is supported by the Deep Skies Community (deepskieslab.com), which helped in bringing together the authors. MP thanks the Centro Nacional de Inteligencia Artificial in Chile, as well as acknowledges Fondecyt Regular grant number 1210462 titled “ Rigidity, stability and uniformity for large point configurations” for supporting his research.

Notice: This work was produced by Fermi Research Alliance, LLC under contract No. DEAC02-07CH11359 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics. The United States Government retains and the publisher, by accepting the work for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this work, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

References

[1] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *ICML*, 2016.

[2] Ilyes Batatia, Mario Geiger, Jose Munoz, Tess Smidt, Lior Silberman, and Christoph Ortner. A general framework for equivariant neural networks on reductive lie groups. *Advances in Neural Information Processing Systems*, 36, 2024.

[3] Erik J. Bekkers. B-spline cnns on lie groups. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[4] Taco S Cohen, Mario Geiger, and Maurice Weiler. A general theory of equivariant CNNs on homogeneous spaces. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[5] Taco Cohen, Maurice Weiler, Berkay Kicanaoglu, and Max Welling. Gauge equivariant convolutional networks and the icosahedral cnn. In *ICML*, 2019.

[6] Marc Finzi, Max Welling, and Andrew Gordon Wilson. A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups. *ArXiv*, abs/2104.09459, 2021.

[7] Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *ICML*, 2018.

[8] Eitan Levin and Mateo Díaz. Any-dimensional equivariant neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 2773–2781. PMLR, 2024.

[9] Haggai Maron, Heli Ben-Hamu, Nadav Shamir, and Yaron Lipman. Invariant and equivariant graph networks. *ArXiv*, abs/1812.09902, 2019.

[10] Mircea Mironenco and Patrick Forré. Lie group decompositions for equivariant neural networks. *arXiv preprint arXiv:2310.11366*, 2023.

[11] Edward Pearce-Crump. Brauer's group equivariant neural networks. In *International Conference on Machine Learning*, pages 27461–27482. PMLR, 2023.

[12] Siamak Ravanbakhsh, Jeff G. Schneider, and Barnabás Póczos. Equivariance through parameter-sharing. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 2892–2901. PMLR, 2017.

[13] Maurice Weiler, Patrick Forré, Erik P. Verlinde, and Max Welling. Coordinate independent convolutional networks - isometry and gauge equivariant convolutions on riemannian manifolds. *ArXiv*, abs/2106.06020, 2021.

[14] Yinshuang Xu, Jiahui Lei, Edgar Dobriban, and Kostas Daniilidis. Unified fourier-based kernel and nonlinearity design for equivariant networks on homogeneous spaces. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 24596–24614. PMLR, 2022.

[15] Ben Blum-Smith and Soledad Villar. Machine learning and invariant theory. *Notices of the American Mathematical Society*, 70:1205–1213, 2022.

[16] Fabian Fuchs, Daniel Worrall, Volker Fischer, and Max Welling. Se (3)-transformers: 3d roto-translation equivariant attention networks. *Advances in neural information processing systems*, 33:1970–1981, 2020.

[17] David Romero, Erik Bekkers, Jakub Tomczak, and Mark Hoogendoorn. Attentive group equivariant convolutional networks. In *International Conference on Machine Learning*, pages 8188–8199. PMLR, 2020.

[18] Michael J Hutchinson, Charline Le Lan, Sheheryar Zaidi, Emilien Dupont, Yee Whye Teh, and Hyunjik Kim. Lietransformer: Equivariant self-attention for lie groups. In *International Conference on Machine Learning*, pages 4533–4543. PMLR, 2021.

[19] Raphael J. L. Townshend, Stephan Eismann, Andrew M. Watkins, Ramya Rangan, Maria Karelina, Rhiju Das, and Ron O. Dror. Geometric deep learning of rna structure. *Science*, 373:1047 – 1051, 2021.

[20] Minkyung Baek, Frank Dimaio, Ivan V. Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N. Kinch, R. Dustin Schaeffer, Claudia Millán, Hahnbeom Park, Carson Adams, Caleb R. Glassman, Andy M. DeGiovanni, Jose H. Pereira, Andria V. Rodrigues, Alberdina Aike van Dijk, Ana C Ebrecht, Diederik Johannes Opperman, Theo Sagmeister, Christoph Buhlheller, Tea Pavkov-Keller, Manoj K. Rathinaswamy, Udit Dalwadi, Calvin K. Yip, John E. Burke, K. Christopher Garcia, Nick V. Grishin, Paul D. Adams, Randy J. Read, and David Baker. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373:871 – 876, 2021.

[21] Victor Garcia Satorras, E. Hoogeboom, F. Fuchs, I. Posner, and M. Welling. E(n) equivariant normalizing flows for molecule generation in 3d. *ArXiv*, abs/2105.09016, 2021.

[22] Brandon Anderson, Truong Son Hy, and Risi Kondor. Cormorant: Covariant molecular neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[23] Johannes Klicpera, Janek Groß, and Stephan Günnemann. Directional message passing for molecular graphs. *ArXiv*, abs/2003.03123, 2020.

[24] Marysia Winkels and Taco Cohen. Pulmonary nodule detection in ct scans with equivariant cnns. *Medical image analysis*, 55:15–26, 2019.

[25] Risi Kondor, Zhen Lin, and Shubhendu Trivedi. Clebsch–gordan nets: a fully fourier space spherical convolutional neural network. *Advances in Neural Information Processing Systems*, 31, 2018.

[26] Jonathan Gordon, David Lopez-Paz, Marco Baroni, and Diane Bouchacourt. Permutation equivariant models for compositional generalization in language. In *ICLR*, 2020.

[27] I. Sosnovik, A. Moskalev, and A. Smeulders. Scale equivariance improves siamese tracking. *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2764–2773, 2021.

[28] Stephan Eismann, Raphael J. L. Townshend, Nathaniel Thomas, Milind Jagota, Bowen Jing, and Ron O. Dror. Hierarchical, rotation-equivariant neural networks to select structural models of protein complexes. *Proteins: Structure*, 89:493 – 501, 2020.

[29] Jennifer C. White and Ryan Cotterell. Equivariant transduction through invariant alignment. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4651–4663, Gyeongju, Republic of Korea, Oct. 2022. International Committee on Computational Linguistics.

[30] Xupeng Zhu, Dian Wang, Ondrej Biza, Guanang Su, Robin Walters, and Robert Platt. Sample efficient grasp learning using equivariant models. *CoRR*, abs/2202.09468, 2022.

[31] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

[32] Risi Kondor, Hy Truong Son, Horace Pan, Brandon Anderson, and Shubhendu Trivedi. Covariant compositional networks for learning graphs. *arXiv preprint arXiv:1801.02144*, 2018.

[33] Xiangyu Chen and Min Ye. Cyclically equivariant neural decoders for cyclic codes. *arXiv preprint arXiv:2105.05540*, 2021.

[34] Xiaoxun Gong, He Li, Nianlong Zou, Runzhang Xu, Wenhui Duan, and Yong Xu. General framework for e (3)-equivariant neural network representation of density functional theory hamiltonian. *Nature Communications*, 14(1):2848, 2023.

[35] Tristan Maxson and Tibor Szilvási. Transferable water potentials using equivariant neural networks. *The Journal of Physical Chemistry Letters*, 15(14):3740–3747, 2024.

[36] Koen Minartz, Yoeri Poels, Simon Koop, and Vlado Menkovski. Equivariant neural simulators for stochastic spatiotemporal dynamics. *Advances in Neural Information Processing Systems*, 36, 2024.

[37] Sophie Baker, Joshua Pagotto, Timothy T Duignan, and Alister J Page. High-throughput aqueous electrolyte structure prediction using ionsolvr and equivariant graph neural network potentials. *The Journal of Physical Chemistry Letters*, 14(42):9508–9515, 2023.

[38] Marc Finzi, Gregory Benton, and Andrew G Wilson. Residual pathway priors for soft equivariance constraints. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 30037–30049. Curran Associates, Inc., 2021.

[39] David W Romero and Suhas Lohit. Learning partial equivariances from data. *Advances in Neural Information Processing Systems*, 35:36466–36478, 2022.

[40] Tycho F. A. van der Ouderaa, David W. Romero, and Mark van der Wilk. Relaxing equivariance constraints with non-stationary continuous filters. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, Dec 2022.

[41] Rui Wang, Robin Walters, and Rose Yu. Approximately equivariant networks for imperfectly symmetric dynamics. In *International Conference on Machine Learning*, pages 23078–23091. PMLR, 2022.

[42] Daniel McNeela. Almost equivariance via lie algebra convolutions. *arXiv preprint arXiv:2310.13164*, 2023.

[43] Rui Wang, Robin Walters, and Tess E Smidt. Relaxed octahedral group convolution for learning symmetry breaking in 3d physical systems. *arXiv preprint arXiv:2310.02299*, 2023.

[44] Tycho van der Ouderaa, Alexander Immer, and Mark van der Wilk. Learning layer-wise equivariances automatically using gradients. *Advances in Neural Information Processing Systems*, 36, 2024.

[45] Kaitlin Maile, Dennis G Wilson, and Patrick Forré. Equivariance-aware architectural optimization of neural networks. *arXiv preprint arXiv:2210.05484*, 2022.

[46] Alonso Urbano and David W Romero. Self-supervised detection of perfect and partial input-dependent symmetries. *arXiv preprint arXiv:2312.12223*, 2023.

[47] Ningyuan Huang, Ron Levie, and Soledad Villar. Approximately equivariant graph networks. *ArXiv*, abs/2308.10436, 2023.

[48] Mircea Petrache and Shubhendu Trivedi. Approximation-generalization trade-offs under (approximate) group equivariance. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 61936–61959. Curran Associates, Inc., 2023.

[49] Atri Rudra. Arithmetic circuits, structured matrices and (not so) deep learning. *Theory of Computing Systems*, 67(3):592–626, 2023.

[50] Tri Dao, Beidi Chen, Nimit S Sohoni, Arjun Desai, Michael Poli, Jessica Grogan, Alexander Liu, Aniruddh Rao, Atri Rudra, and Christopher Ré. Monarch: Expressive structured matrices for efficient and accurate training. In *International Conference on Machine Learning*, pages 4690–4721. PMLR, 2022.

[51] Tri Dao, Albert Gu, Matthew Eichhorn, Atri Rudra, and Christopher Ré. Learning fast algorithms for linear transforms using butterfly factorizations. In *International conference on machine learning*, pages 1517–1527. PMLR, 2019.

[52] Tri Dao, Nimit S Sohoni, Albert Gu, Matthew Eichhorn, Amit Blonder, Megan Leszczynski, Atri Rudra, and Christopher Ré. Kaleidoscope: An efficient, learnable representation for all structured linear maps. *arXiv preprint arXiv:2012.14966*, 2020.

[53] Vikas Sindhwani, Tara Sainath, and Sanjiv Kumar. Structured transforms for small-footprint deep learning. *Advances in Neural Information Processing Systems*, 28, 2015.

[54] Anna Thomas, Albert Gu, Tri Dao, Atri Rudra, and Christopher Ré. Learning compressed transforms with low displacement rank. *Advances in neural information processing systems*, 31, 2018.

[55] Victor Y Pan. *Structured matrices and polynomials: unified superfast algorithms*. Springer Science & Business Media, 2012.

[56] Thomas Kailath, Sun-Yuan Kung, and Martin Morf. Displacement ranks of matrices and linear equations. *Journal of Mathematical Analysis and Applications*, 68(2):395–407, 1979.

[57] Thomas Kailath and Ali H Sayed. Displacement structure: theory and applications. *SIAM review*, 37(3):297–386, 1995.

[58] Thomas Kailath and Joohwan Chun. Generalized displacement structure for block-toeplitz, toeplitz-block, and toeplitz-derived matrices. *SIAM Journal on Matrix Analysis and Applications*, 15(1):114–128, 1994.

[59] Roger Chalkley. Information about group matrices. *Linear Algebra and its Applications*, 38:121–133, 1981.

[60] Paul D Gader. Displacement operator based decompositions of matrices using circulants or other group matrices. *Linear Algebra and its Applications*, 139:111–131, 1990.

[61] William C Waterhouse. Displacement operators relative to group matrices. *Linear algebra and its applications*, 169:41–47, 1992.

[62] Philipp M Holl, Kiwon Um, and Nils Thuerey. phiflow: A differentiable pde solving framework for deep learning via physical simulations. In *Workshop on Differentiable Vision, Graphics, and Physics in Machine Learning at NeurIPS*, 2020.

[63] Maurice Weiler and Gabriele Cesa. General E(2)-Equivariant Steerable CNNs. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[64] Marc Finzi, Gregory Benton, and Andrew G Wilson. Residual pathway priors for soft equivariance constraints. *Advances in Neural Information Processing Systems*, 34:30037–30049, 2021.

[65] Dian Wang, Robin Walters, Xupeng Zhu, and Robert Platt. Equivariant q learning in spatial action spaces. In *5th Annual Conference on Robot Learning*, 2021.

[66] Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–104. IEEE, 2004.

[67] Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th international conference on Machine learning*, pages 473–480, 2007.

[68] Harm Derksen. On the nuclear norm and the singular value decomposition of tensors. *Foundations of Computational Mathematics*, 16(3):779–811, 2016.

A Proofs

A.1 Proof of Lemma 2.3

Proof. Direct by definition (1). For $h, h' \in H$ and $g \in G$, the entry of B_h at position (h', g) is $\delta(g = h'h)$. Since H is closed under the group operation, $g = h'h \Rightarrow g \in H$. $\square$

A.2 Proof of Proposition 2.5

Proof. The result follows by linearity from the following explicit properties of group diagonals, which are direct to check from the definitions:

$$B_g^T = B_g^{-1} = B_{g^{-1}}, \quad B_{g_1} B_{g_2} = B_{g_1 g_2}. \tag{15}$$

Finally, the case of Kronecker product follows from (6). $\square$

A.3 Proof of Proposition 3.1

Proof. For a matrix M let M_0 be M's projection, i.e. the group matrix realizing the distance to $\mathcal{GM}$. Note that $\|M_0\| \leq \|M\|$ is a consequence of Frobenius norm coming from an inner product.

Since for all matrices X we have $\|X\| = \|X^T\|$, and since $\mathcal{GM}$ is closed under transpose, we have $\mathrm{dist}(M, \mathcal{GM}) = \|M - M_0\| = \|M^T - M_0^T\| \geq \mathrm{dist}(M^T, \mathcal{GM})$. By interchanging the roles of M, M^T we also find the other inequality, and thus prove item 1.

For item 2, note that for all matrices X, Y we have $\|XY\| \leq \|X\|\|Y\|$. Then we use the closure property of $\mathcal{GM}$ under product, and triangle inequality, in order to write $\mathrm{dist}(MN, \mathcal{GM}) \leq \|MN - M_0 N_0\| = \|M(N - N_0) + (M - M_0)N_0\| \leq \|M\|\|N - N_0\| + \|M - M_0\|\|N_0\| \leq \max\{\|M\|, \|N_0\|\}(\|M - M_0\| + \|N - N_0\|)$, which allows to conclude by noting $\|N_0\| \leq \|N\|$.

For item 3, note that for all X, Y we have $\|X \otimes Y\| \leq \|X\|\|Y\|$. Then we proceed as for item 2, with tensor product replacing matrix product, keeping in mind that, due to Prop. 2.5, if $M_0 \in \mathcal{GM}^G, N_0 \in \mathcal{GM}^H$ then $M_0 \otimes N_0 \in \mathcal{GM}^{G \times H}$. $\square$

.

B Extended discussion on notions of displacement rank for group matrices

B.1 Comparison to classical displacement dimension for group matrices

In this section, we review previous notions of displacement rank from [61, 60]. These works were motivated by aiming to generalize different aspects of LDR theory from the case of circulant matrices, formulated very similarly to our Example 2.2. In the case of group matrices, it was realized that the property of circulant matrices of being constant along diagonals can be generalized to the notion of *group diagonals* (whose name includes the term "diagonals" for this reason). Then the natural idea is that displacement needed to have the property of vanishing on group matrices as in [60, 61].

In a formula extending the Stein-type displacement operator $\Delta_{A,B} M = M - AMB$, which we defined in Section 1, [60] defined a displacement operator $\mathcal{G}(M) := M - P\mathcal{Q}(M)$, in which P is the permutation matrix corresponding to the cyclic permutation of basis vectors $e_1 \mapsto e_2 \mapsto \cdots \mapsto e_{|G|} \mapsto e_1$, and $Q(M)$ is an involved operation that cyclically permutes elements of M of each given group diagonal pattern separately. The displacement operation $\mathcal{G}(M)$ from the work of [60] was further rationalized and made more elegant in [61], which replaced the term $P\mathcal{Q}(M)$ by a more general form $\mathcal{T}(M)$ in which the permutation P can be chosen arbitrarily. In both cases, operator $\mathcal{G}(M)$ subtracts elements of M which are shifted along group diagonal patterns B_g, and as the shift is done via a cyclic permutation, we have the property that if all such differences are zero, then the entries of M are constant along group diagonals.

Our contribution is to simplify the expressions further, by introducing the intermediate reordering $F(M)$, which was not present in previous work. This allows to easily operationalize the implementation that is the focus of this work. Our definition (9), and the more complex one (10), are in direct parallel to the treatment from [60], [61] respectively. The main justification for using (10), is that it will be helpful in the proof of Proposition B.1 below.

B.2 Displacement dimension changes under elementary operations

In this section we consider displacement error $\mathsf{D}_{\vec{P}}(M)$ as defined in (10), and the derived notion of displacement dimension $\dim_{\mathsf{D}}(\mathcal{M})$ from (11). For the latter, we are interested in how it behaves under elementary operations like those from [54, 55]. Note that in previous work on classical displacement rank [54], similar results are formulated to study the closure properties of classical structured matrices, however, we were not able to adapt displacement rank (12) similarly to our case (as also mentioned in Section 3, above proposition 3.1) for the following reasons: (1) the results valid for classical LDR structured matrices are not valid in our case, (2) the neural network interpretation as number of degrees of freedom as quantified in the error to precise equivariance is more important to our setup.

We summarize our results in the following:

Proposition B.1. *Let G, H be two groups. Further, consider classes of $|G| \times |G|$-matrices $\mathcal{M}, \mathcal{M}'$ and a class of $|H| \times |H|$-matrices $\mathcal{N}$, with D-dimensions with respect to the corresponding groups denoted respectively as $d_M, d_{M'} d_N$. Then the following holds:*

1. *The class $\mathcal{M}^T = \{M^T : M \in \mathcal{M}\}$ has $\dim_{\mathsf{D}}(\mathcal{M}^T) = d_M$.*
2. *The set $\mathcal{M} + \mathcal{M}' := \{M + M' : M \in \mathcal{M}, M' \in \mathcal{M}'\}$ has $\dim_{\mathsf{D}}(\mathcal{M} + \mathcal{M}') = d_M + d_{M'}$.*
3. *Consider the set $\mathcal{M} \otimes \mathcal{N} := \{M \otimes N : M \in \mathcal{M}, N \in \mathcal{N}\}$. Then with respect to the group diagonals from a direct product group $G \times H$, we have $\dim_{\mathsf{D}}(\mathcal{M} \otimes \mathcal{N}) \le d_M + d_N$.*

Proof. In the setting of item 1, we first claim that

$$F(M^T)_{gh} = F(M)_{g^{-1}, g^{-1}h}. \tag{16}$$

Once this is proved, we note first that $F(M^T)$'s rows can be re-labeled by $g^{-1} \mapsto g$, to obtain the matrix $\widetilde{F}(M)$ with entries $F(M)_{g,gh}$. Then, if $\tau_g(h) := gh$, and σ_g was the permutation used in the definition of $\mathsf{D}_{\vec{P}}(M)$ in (10), then for the rows of $\widetilde{F}(M)$ we can use cyclic permutation $\tau_g^{-1}\sigma_g\tau_g$. After this change coordinates $\widetilde{F}(M) \mapsto F(M^T)$. Since all the applied transformations do not depend on the choice of $M \in \mathcal{M}$, and do not change the dimension counts (since, as observed before, $\dim_{\mathsf{D}}$ does not depend on the choices of σ_g's), this means that $\mathcal{M}, \mathcal{M}^T$ have the same D-dimension.

It now suffices to prove (16). For this, we first note the following commutation relation of vectors with group diagonal matrices B_g:

$$B_g \mathrm{diag}(v) = \mathrm{diag}(w) B_g, \quad \text{where} \quad w_h = v_{g^{-1}h}, h \in G. \tag{17}$$

The above follows directly using the definition $(B_g)_{h,h'} = \delta(h = gh')$.

Recall that B_g is a permutation matrix, thus $B_g^T = B_g^{-1}$ and we can directly verify $B_g B_h = B_{gh}$, $B_{id} = Id$, thus

$$B_g^T = B_g^{-1} = B_{g^{-1}}. \tag{18}$$

Using (17), (18) and a change of variable $g \mapsto g^{-1}$ in the sums, we now get, denoting $F = F(M)$,

$$M^T = \sum_g B_g^T \mathrm{diag}((F_{gh})_h) = \sum_g \mathrm{diag}((F_{g,gh})_h B_g^{-1} = \sum_g \mathrm{diag}((F_{g^{-1},g^{-1}h})_h) B_g.$$

This directly implies (16), as desired.

Item 2 follows directly from linearity and using the property that $\dim(V + W) \le \dim V + \dim W$ for any vector subspaces $V, W \subseteq \mathbb{R}^{|G| \times |G|}$.

Item 3 follows from the definition of Kronecker product, observing that $F(M \otimes N) = F(M) \otimes F(N)$. □

C Generalization of Framework to Homogeneous Spaces and Infinite Groups

C.1 Actions on homogeneous spaces.

In most machine learning scenarios, the input is more frequently defined not as a function over a group, but rather on a space on which the group acts [7]. That is, the space X that encodes the inputs, is not identifiable with the symmetry group G, but rather is a *homogeneous space of* G, i.e. it can be identified with a quotient by a subgroup $H \subseteq G$, denoted $X = G/H$. Elements $x \in X$ are then identified with subsets of the partition into right cosets $[x] := xH = \{\bar{x} \in G : \exists h \in H, \bar{x} = xh\}$. In general, G/H still has a natural action of G, given by $g \cdot [x] := [gx]$.

Convolution over G/H. The convolution operations can be encoded in this case as follows. Consider a kernel $\phi : G \to \mathbb{R}$ and a channel $f : G/H \to \mathbb{R}$, we define

$$\phi \star f([x]) := \sum_{g \in G} \phi(g) f(g^{-1}[x]) = \sum_{g \in G, [y] \in G/H} \phi(g) f([y]) \delta([y] = [g^{-1}x]). \tag{19}$$

The above operation can be expressed using group matrices by observing the following

$$\begin{aligned} \delta\left([y] = [g^{-1}x]\right) &= \delta(\exists h \in H : yh = g^{-1}x) = \sum_{h \in H} \delta(yh = g^{-1}x) = \sum_{h \in H} \delta(x = gyh) \\ &= \sum_{h \in H} (B_{gy})_{xh} = \left[B_{gy} \mathbf{1}_H\right]_x , \end{aligned} \tag{20}$$

in which $\mathbf{1}_H$ is the characteristic vector of H. One can verify that $(B_{gy}\mathbf{1}_H)(x) = (B_{gy'}\mathbf{1}_H)(x')$ whenever $[x] = [x']$ and $[y] = [y']$, thus expression (20) does not depend on the choices of representatives x, y. Using this observation, we can fix a choice of representatives, namely

$$X \subset G, \quad \text{such that} \quad \forall gH \in G/H, \ |gH \cap X| = 1.$$

Then we can equivalently work with $f, \phi \star f : X \to \mathbb{R}$. We then write (19) in terms of group diagonal matrices and representatives from X as follows:

$$\phi \star f(x) = \sum_{g \in G, y \in X} \phi(g) f([y]) \left(B_{gy}\mathbf{1}_H\right)(x). \tag{21}$$

C.2 Infinite discrete groups and padding.

Although for ease of exposition in section 2 we used periodic translations, more traditionally CNNs use the group $G = \mathbb{Z}^2$, which is infinite. Then, operation (2) involves an infinite sum, and is thus not computable. This is naturally taken care of by the following adjustments, which we state directly for general groups G:

1. We work only with inputs ψ of support contained in a fixed finite set X_{in} (for CNNs, X_{in} is a square $\{0, \ldots, n-1\}^2 \subset \mathbb{Z}^2$).
2. We use kernels ϕ of support constrained to a finite set $\mathcal{N}$, typically a radius-k (in the word metric induced by a set of generators) neighborhood of the identity. We further assume $\mathcal{N}_k$ to be symmetric (we use a symmetric set of generators), in which "symmetric" means closed under taking inverses: $g \in \mathcal{N} \Leftrightarrow g^{-1} \in \mathcal{N}$.
3. While applying $\phi \star \psi(x), x \in X_{in}$, the computation (2) involves terms $\psi(g^{-1}x)$, in which sometimes $g^{-1}x \notin X_{in}$. We then have to extend ϕ by zero on the set $(X_{in})_{\mathcal{N}}$, where we use notation
$$(A)_B := \{b^{-1}a : a \in A, b \in B\} = \mathsf{supp}(1_B \star 1_A).$$
Note that if $\mathcal{N}$ is the radius-k word-distance ball around the identity, then $(A)_{\mathcal{N}}$ also can be described as the set of all elements at word-distance $\leq k$ from A.

The above operations can be summarized as follows:

$$\begin{aligned} \mathsf{Pad}(\psi)(x) &:= \begin{cases} \psi(x) & \text{if } x \in X_{in}, \\ 0 & \text{if } x \in \partial_{\mathcal{N}} X_{in} := (X_{in})_{\mathcal{N}} \setminus X_{in}, \end{cases} \\ \left[\phi \star \mathsf{Pad}(\psi)\right](x) &:= \sum_{g \in \mathcal{N}} \phi(g) \mathsf{Pad}(\psi(g^{-1}x)) \quad \text{for} \quad x \in X_{in}. \end{aligned} \tag{22}$$

Formally, extension by zero corresponds to a subspace immersion given by a matrix multiplication $E : \mathbb{R}^{X_{in}} \to \mathbb{R}^{(X_{in})_{\mathcal{N}}}$ with entries $E_{x,y} = \delta(x = y), x \in X_{in}, y \in (X_{in})_{\mathcal{N}}$, and then we can define analogues of group matrices $B_g^{(X_{in})_{\mathcal{N}}}, g \in \mathcal{N}$ by the same formula (1) with the restriction of $h, h' \in (X_{in})_{\mathcal{N}}$ only. After this, the formula for (22) (analogous to (3) for this case) reads

$$\widetilde{\mathsf{Conv}}_\phi \overrightarrow{\psi} := E^T \left(\sum_{g \in \mathcal{N}} \phi(g) B_g^{(X_{in})_{\mathcal{N}}} \right) E \, \overrightarrow{\psi}. \tag{23}$$

C.3 Error to equivariance.

Note that operation $\widetilde{\mathsf{Conv}}_\phi$ is not equivariant under G-action, simply because $\overrightarrow{\psi}$ has entries in X_{in}, which in general is not a union of G-action orbits, and thus is not invariant under the G-action. In other words, the restriction map encoded in I^T in (23) is "the culprit" responsible for the loss of equivariance, whereas the term in parenthesis in (23) is actually equivariant when applied to elements of the image of E (corresponding to functions over $(X_{in})_{\mathcal{N}}$ that are zero outside X_{in}). A benefit of (23) is that it makes it straightforward for us to measure the equivariance error of $\widetilde{\mathsf{Conv}}$, by extending the theory from Section 3. Working on the image of the extension map I, we define a version of the displacement operator, denoted $\widetilde{\mathsf{D}}$, as follows. Consider $\widetilde{M} := EE^TM$ where $M := \left(\sum_{g\in\mathcal{N}} \phi(g) B_g^{(X_{in})_{\mathcal{N}}}\right)$: following (8), we can encode this matrix as

$$\widetilde{M} = \sum_{g\in\mathcal{N}} \mathrm{diag}(\widetilde{F}_g) B_g^{(X_{in})_{\mathcal{N}}}, \quad \text{where} \quad (\widetilde{F}_g)_x := \widetilde{M}_{x,xg^{-1}}. \tag{24}$$

Then we form a matrix $F(\widetilde{M})$ with rows $\widetilde{F}_g, g \in \mathcal{N}$ and define $\mathsf{D}(\widetilde{M})$ as in (9). We can bound the displacement dimension of matrices of the form (24) by noting that the M defined earlier has $\mathsf{D}(M) = 0$ and by linearity of D, only columns of $\widetilde{F}$ corresponding to elements $x \in X_{in}$ such that there exists $g \in \mathcal{N}$ such that $g^{-1}x' \in \partial_{\mathcal{N}} X_{in}$ can contribute to degrees of freedom of $\mathsf{D}(\widetilde{M})$. We thus find that for $\mathcal{M} := \{\widetilde{M}$ as in (24)$\}$ it holds that:

$$\dim_{\mathsf{D}}(\mathcal{M}) \le |(X_{in})_{\mathcal{N}} \setminus X_{in}|\, |\mathcal{N}|, \qquad \mathsf{DR}(\mathcal{M}) \le |(X_{in})_{\mathcal{N}} \setminus X_{in}|. \tag{25}$$

The intuitive explanation for (25) in the case of padding for classical CNNs is that **the number of padding pixels provides rank control for the error to precise equivariance in each layer**.

D Equivariance Error Analysis:

In [41], the authors use the equivariance error as a metric to quantify the degree to which a model deviates from perfect symmetry under group transformations. This measure is relevant for approximately equivariant networks, which aim to balance the symmetry constraints and model flexibility. The authors argue that approximately equivariant networks can better capture the imperfect symmetries in real-world scenarios via adding soft equivariance regularization during training. However, equivariance error analysis for GM-CNN suggests that while it does not achieve the optimal balance between data and model equivariance error, it consistently outperforms most other methods, including ConvNet, RPP, Lift as seen in Figure 1. Our analysis raises the possibility that the balance between model and data equivariance error and its relation to overall performance needs further analysis to accurately quantify. GM-CNN's ability to capture relevant symmetries and offer strong predictive capabilities indicate that certain applications may benefit from a more flexible approach to equivariance. Despite RSteer showing equivariance error close to optimal level, GM-CNN offers a competitive balance and a promising framework for real-world applications where both symmetry and model flexibility are crucial.

E Broader Impact

This paper presents a novel formulation for the design of light-weight approximately equivariant CNNs. The development combines two different threads of research into one framework. It also presents a theory generalizing classical LDR theory from cyclinc groups to general discrete groups. As such, while the contribution has a methodological orientation, it is primarily intended to be a conceptual contribution. We do not foresee immediate societal broader impact of this work. However, if the formulation presented is scaled up, it could provide very efficient equivariant networks that could possibly run on mobile devices.

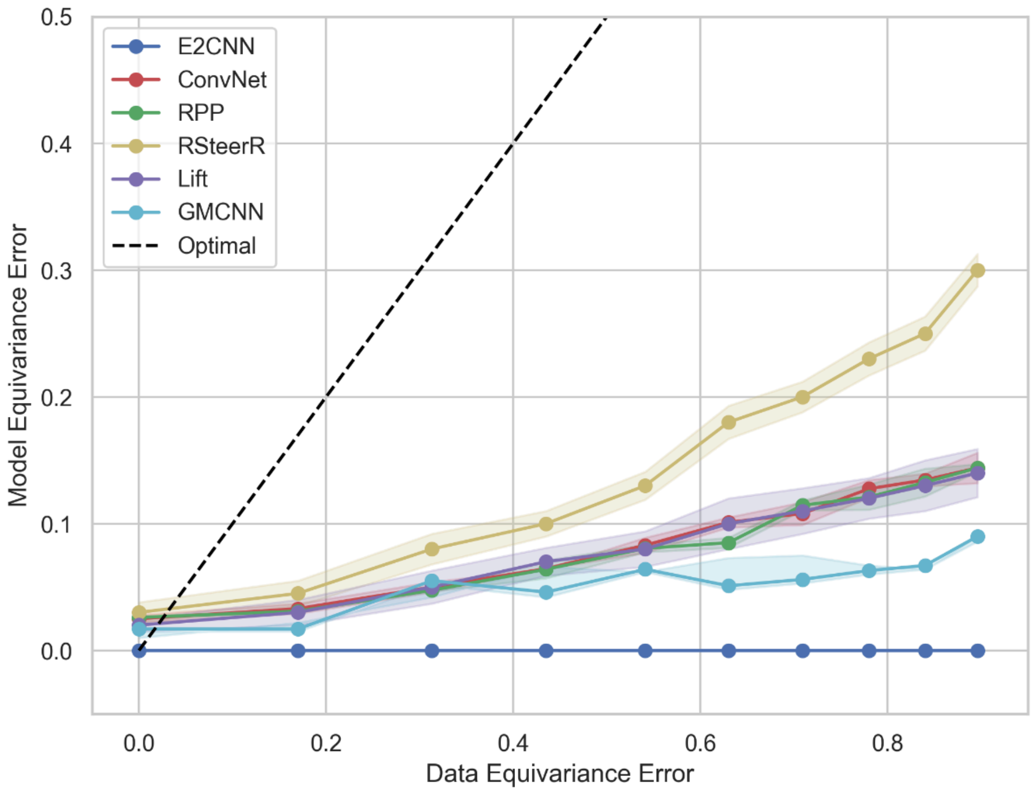

Figure 1: Equivariance error analysis on synthetic smoke plume with different levels of rotational equivariance as described in [41].