A Practical Method for Constructing Equivariant Multilayer Perceptrons for Arbitrary Matrix Groups

Marc Finzi [1] **Max Welling** [2] **Andrew Gordon Wilson** [1]

Abstract

Symmetries and equivariance are fundamental to the generalization of neural networks on domains such as images, graphs, and point clouds. Existing work has primarily focused on a small number of groups, such as the translation, rotation, and permutation groups. In this work we provide a completely general algorithm for solving for the equivariant layers of matrix groups. In addition to recovering solutions from other works as special cases, we construct multilayer perceptrons equivariant to multiple groups that have never been tackled before, including $\mathrm{O}(1,3)$, $\mathrm{O}(5)$, $\mathrm{Sp}(n)$, and the Rubik's cube group. Our approach outperforms non-equivariant baselines, with applications including particle physics and dynamical systems. We release our software library to enable researchers to construct equivariant layers for arbitrary matrix groups.

1. Introduction

As machine learning has expanded to cover more areas, the kinds of structures and data types we must accommodate grows ever larger. While translation equivariance may have been sufficient for working with narrowly defined sequences and images, with the expanding scope to sets, graphs, point clouds, meshes, hierarchies, tables, proteins, RF signals, games, PDEs, dynamical systems, and particle jets, we require new techniques to exploit the structure and symmetries in the data.

In this work we propose a general formulation for equivariant multilayer perceptrons (EMLP). Given a set of inputs and outputs which transform according to finite dimensional representations of a symmetry group, we characterize all linear layers that map from one space to the other, and provide

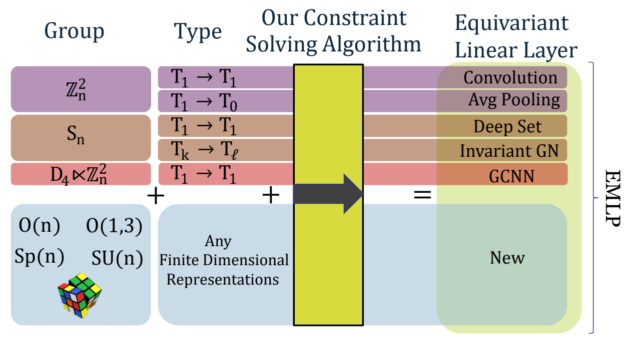


Figure 1. We provide a general and efficient method for solving equivariance constaints. For particular symmetry groups and type signatures, we recover other well known equivariant layers while also enabling application to new groups and representations.

a polynomial time algorithm for computing them. We release our library, along with documentation, and examples.

Figure 1 illustrates how the convolutional layers of a CNN (LeCun et al., 1989), the permutation equivariant deep sets (Zaheer et al., 2017), graph layers (Maron et al., 2018), and layers for networks equivariant to point clouds (Thomas et al., 2018), all arise as special cases of our more general algorithm.

We summarize our contributions as follows:

- We prove that the conditions for equivariance to matrix groups with arbitrary linear representations can be reduced to a set of $M + D$ constraints, where M is the number of discrete generators and D is the dimension of the group.
- We provide a polynomial time algorithm for solving these constraints for finite dimensional representations, and we show that the approach can be accelerated by exploiting structure and recasting it as an optimization problem.
- With the addition of a bilinear layer, we develop the Equivariant MultiLayer Perceptron (EMLP), a general equivariant architecture that can be applied to a new group by specifying the group generators.

[1]New York University [2]University of Amsterdam. Correspondence to: Marc Finzi <maf820@nyu.edu>, Andrew Gordon Wilson <andrewgw@cims.nyu.edu>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

- Demonstrating the generality of our approach, we apply our network to multiple groups that were previously infeasible, such as the orthogonal group in five dimensions $\mathrm{O}(5)$, the full Lorentz group $\mathrm{O}(1,3)$, the symplectic group $\mathrm{Sp}(n)$, the Rubik's cube group, *with the same underlying architecture*, outperforming non equivariant baselines.

2. Related Work

While translation equivariance in convolutional neural networks (LeCun et al., 1989) has been around for many years, more general group equivariant neural networks were introduced in Cohen & Welling (2016a) for discrete groups with GCNNs. There have been a number of important works generalizing the approach to make use of the irreducible group representations for the continuous rotation groups $\mathrm{SO}(2)$ (Cohen & Welling, 2016b; Esteves et al., 2017; Marcos et al., 2017), $\mathrm{O}(2)$ (Weiler & Cesa, 2019), $\mathrm{SO}(3)$ (Thomas et al., 2018; Weiler et al., 2018; Anderson et al., 2019), $\mathrm{O}(3)$ (Smidt et al., 2020) and their discrete subgroups.The requirements and complexity of working with irreducible representations has limited the scope of these methods, with only one example outside of these two rotation groups with the identity component of the Lorentz group $\mathrm{SO}^+(1,3)$ in Bogatskiy et al. (2020).

Others have used alternate approaches for equivariance through group FFTs (Cohen et al., 2018b), and regular group convolution (Worrall & Welling, 2019; Bekkers, 2019; Finzi et al., 2020b). These methods enable greater flexibility; however, achieving equivariance for continuous groups with the regular representation is fundamentally challenging, since the regular representation is infinite dimensional.

Meanwhile, the theoretical understanding and practical methods for equivariance to the permutation group S_n have advanced considerably for the application to sets (Zaheer et al., 2017), graphs (Maron et al., 2018), and related objects (Serviansky et al., 2020). Particular instances of equivariant networks have been shown to be *universal*: with a sufficient size these networks can approximate equivariant functions for the given group with arbitrary accuracy (Maron et al., 2019; Ravanbakhsh, 2020; Dym & Maron, 2020).

Despite these developments, there is still no algorithm for constructing equivariant networks that is completely general to the choice of symmetry group or representation. Furthest in this direction are the works of Lang & Weiler (2020), Ravanbakhsh et al. (2017), and van der Pol et al. (2020) with some of these ideas also appearing in Wood & Shawe-Taylor (1996). Based on the Wigner-Eckert theorem, Lang & Weiler (2020) show a general process by which equivariant convolution kernels can be derived for arbitrary compact groups. However this process still requires considerable mathematical legwork to carry out for a given group, and is not applicable beyond compact groups. Ravanbakhsh et al. (2017) show how equivariance can be achieved by sharing weights over the orbits of the group, but is limited to regular representations of finite groups. Unlike Lang & Weiler (2020) and Ravanbakhsh et al. (2017), van der Pol et al. (2020) present an explicit algorithm for computing equivariant layers. However, the complexity of this approach scales with the size of the group and quickly becomes too costly for large groups and impossible for continuous groups like $\mathrm{SO}(n)$, $\mathrm{O}(1,3)$, $\mathrm{Sp}(n)$, and $\mathrm{SU}(n)$.

3. Background

In order to present our main results, we first review some necessary background on group theory. Most importantly, symmetry groups can be broken down in terms of discrete and continuous generators, and these can act on objects through group and Lie algebra representations.

Finite Groups and Discrete Generators. A group G is finitely generated if we can write each element $g \in G$ as a sequence from a discrete set of generators $\{h_1, h_2, ...h_M\}$ and their inverses $h_{-k} = h_k^{-1}$. For example we may have an element $g = h_1 h_2 h_2 h_1^{-1} h_3$ and can be written more compactly $g = \Pi_{i=1}^N h_{k_i}$ for the integer sequence $k = [1, 2, 2, -1, 3]$.

All finite groups, like the cyclic group $\mathbb{Z}_n$, the dihedral group D_n, the permutation group S_n, and the Rubik's cube group can be produced by a finite set of generators. Even for large groups, the number of generators is *much* smaller than the size of the group: 1 for $\mathbb{Z}_n$ of size n, 2 for S_n of size $n!$, and 6 for the cube group of size 4×10^{19}.

Continuous Groups and Infinitesimal Generators. Similarly, Lie theory provides a way of analyzing continuous groups in terms of their *infinitesimal* generators. The Lie Algebra $\mathfrak{g}$ of a Lie Group G (a continuous group that forms a smooth manifold) is the tangent space at the identity $\mathfrak{g} := T_{\mathrm{id}} G \subseteq \mathbb{R}^{n \times n}$, which is a vector space of infinitesimal generators of group transformations from G. The exponential map $\exp : \mathfrak{g} \to G$ maps back to the Lie Group and can be understood through the series: $\exp(A) = \sum_{k=0}^{\infty} A^k / k!$

A classic example is the rotation group $G = \mathrm{SO}(n)$ with matrices $\mathbb{R}^{n \times n}$ satisfying $R^\top R = I$ and $\det(R) = 1$. Parametrizing a curve $R(t)$ with $R(0) = I$, $R'(0) = A$, one can find the tangent space by differentiating the constraint at the identity. The Lie Algebra consists of antisymmetric matrices: $\mathfrak{so}(n) = T_{\mathrm{id}}\mathrm{SO}(n) = \{A \in \mathbb{R}^{n \times n} : A^\top = -A\}$.

Given that $\mathfrak{g}$ is a finite dimensional vector space ($D = \dim(\mathfrak{g}) = \dim(G)$), its elements can be expanded in a basis $\{A_1, A_2, ..., A_D\}$. For some Lie Groups like $\mathrm{SO}(n)$, the orientation preserving isometries $\mathrm{SE}(n)$, the special unitary group $\mathrm{SU}(n)$, the symplectic group $\mathrm{Sp}(n)$, the exponential map is surjective meaning all elements $g \in G$ can be written in terms of this exponential $g = \exp(\sum_i \alpha_i A_i)$ with a set of real valued coefficients $\{\alpha_i\}_{i=1}^D$. But in general for other Matrix Groups like $\mathrm{O}(n)$, $\mathrm{E}(n)$, and $\mathrm{O}(1,3)$, exp is not surjective and one can instead write $g = \exp(\sum_i \alpha_i A_i)\Pi_{i=1}^N h_{k_i}$ as a product of the exponential map that traverses the identity component and an additional collection of discrete generators (see Appendix H).

Group Representations. In the machine learning context, a group element is most relevant in how it acts as a transformation on an input. A (linear finite dimensional) group *representation* $\rho : G \to \mathrm{GL}(m)$ associates each $g \in G$ to an invertible matrix $\rho(g) \in \mathbb{R}^{m\times m}$ that acts on $\mathbb{R}^m$. The representation satisfies $\forall g_1, g_2 \in G : \rho(g_1 g_2) = \rho(g_1)\rho(g_2)$, and therefore also $\rho(g^{-1}) = \rho(g)^{-1}$. The representation specifies how objects transform under the group, and can be considered a specification of the *type* of an object.

Lie Algebra Representations. Mirroring the group representations, Lie Groups have an associated representation of their Lie algebra, prescribing how infinitesimal transformations act on an input. A Lie algebra representation $d\rho : \mathfrak{g} \to \mathfrak{gl}(N)$ is a **linear** map from the Lie algebra to $m \times m$ matrices. An important result in Lie Theory relates the representation of a Lie Group to the representation of its Lie Algebra

$$\forall A \in \mathfrak{g} : \quad \rho(e^A) = e^{d\rho(A)} \tag{1}$$

Tensor Representations. Given some base group representation ρ, Lie Algebra representation $d\rho$, acting on a vector space V, representations of increasing size and complexity can be built up through the tensor operations dual ($*$), direct sum ($\oplus$), and tensor product ($\otimes$).

OP	ρ	$d\rho$	V
$*$	$\rho(g^{-1})^\top$	$-d\rho(A)^\top$	V^*
$\oplus$	$\rho_1(g) \oplus \rho_2(g)$	$d\rho_1(A) \oplus d\rho_2(A)$	$V_1 \oplus V_2$
$\otimes$	$\rho_1(g) \otimes \rho_2(g)$	$d\rho_1(A) \bar{\oplus} d\rho_2(A)$	$V_1 \otimes V_2$

Acting on matrices, $\oplus$ is the direct sum which concatenates the matrices on the diagonal $X \oplus Y = \begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix}$, and facilitates multiple representations which are acted upon separately. The $\otimes$ on matrices is the Kronecker product, and $\bar{\oplus}$ is the *Kronecker sum*: $X \bar{\oplus} Y = X \otimes I + I \otimes Y$. V^* is the dual space of V. The tensor product and dual are useful in describing linear maps from one vector space to another. Linear maps from $V_1 \to V_2$ form the vector space $V_2 \otimes V_1^*$ and have the corresponding representation $\rho_2 \otimes \rho_1^*$.

We will work with the corresponding vector spaces and representations interchangeably with the understanding that the other is defined through these composition rules. We abbreviate many copies of the same vector space $\underbrace{V \oplus V \oplus ... \oplus V}_{m}$ as mV. Similarly we will refer to the vector space formed from many tensor products $T_{(p,q)} = V^{\otimes p} \otimes (V^*)^{\otimes q}$ where $(\cdot)^{\otimes p}$ is the tensor product iterated p times. Following the table, these tensors have the group representation $\rho_{(p,q)}(g) = \rho(g)^{\otimes p} \otimes \rho^*(g)^{\otimes q}$, and the Lie algebra representation $d\rho_{(p,q)}(A) = d\rho(A)^{\bar{\oplus} p} \bar{\oplus} d\rho^*(A)^{\bar{\oplus} q}$. We will abbreviate T_{p+q} for $T_{(p,q)}$ when using orthogonal representations ($\rho = \rho^*$), as the distinction between V and V^* becomes unnecessary.

4. Equivariant Linear Maps

In building equivariant models, we need that the layers of the network are equivariant to the action of the group. Below we characterize all equivariant linear layers $W \in \mathbb{R}^{N_2 \times N_1}$ that map from one vector space V_1 with representation ρ_1 to another vector space V_2 with representation ρ_2 for a matrix group G. We prove that the infinite set of constraints can be reduced to a finite collection without loss of generality, and then provide a polynomial-time algorithm for solving the constraints.

4.1. The Equivariance Constraint

Equivariance requires that transforming the input is the same as transforming the output:

$$\forall x \in V_1, \forall g \in G : \qquad \rho_2(g) W x = W \rho_1(g) x.$$

Since true for all x, $\rho_2(g) W \rho_1(g)^{-1} = W$, or more abstractly:

$$\forall g \in G : \quad \rho_2(g) \otimes \rho_1(g^{-1})^\top \mathrm{vec}(W) = \mathrm{vec}(W) \tag{2}$$

where vec flattens the matrix into a vector. $\rho_1(g^{-1})^\top$ is the dual representation $\rho_1^*(g)$, and so the whole object $\rho_2(g) \otimes \rho_1(g^{-1})^\top = (\rho_2 \otimes \rho_1^*)(g) = \rho_{21}(g)$ is a representation of how g acts on matrices mapping from $V_1 \to V_2$.

While equation (2) is linear, the constraint must be upheld for each of the possibly combinatorially large or infinite number of group elements in the case of continuous groups. However, in the following section we show that these constraints can be reduced to a finite and small number.

4.2. General Solution for Symmetric Objects

Equation (2) above with $\rho = (\rho_2 \otimes \rho_1^*)$ is a special case of a more general equation expressing the symmetry of an object v,

$$\forall g \in G : \quad \rho(g) v = v \tag{3}$$

Writing the elements of G in terms of their generators: $g = \exp(\sum_i^D \alpha_i A_i)\Pi_{i=1}^N h_{k_i}$. For group elements with $k = \varnothing$, we have

$$\forall \alpha_i : \quad \rho\big(\exp(\sum_i \alpha_i A_i)\big)v = v$$

Using the Lie Algebra - Lie Group representation correspondence (1) and the linearity of $d\rho(\cdot)$ we have

$$\forall \alpha_i : \quad \exp\big(\sum_i \alpha_i d\rho(A_i)\big)v = v.$$

Taking the derivative with respect to α_i at $\alpha = 0$, we get a constraint for each of the infinitesimal generators

$$\boxed{\forall i = 1, ..., D: \quad d\rho(A_i)v = 0} \tag{4}$$

For group elements with all $\alpha_i = 0$ and $N = 1$, we get an additional constraint for each of the discrete generators in the group:

$$\boxed{\forall k = 1, ..., M: \quad (\rho(h_k) - I)v = 0}. \tag{5}$$

We get a total of $O(M + D)$ constraints, one for each of the discrete and infinitesimal generators. In Appendix B, we prove that these reduced constraints are not just **necessary** but also **sufficient**, and therefore characterize all solutions to the symmetry equation (3).

Solving the Constraint: We collect each of the symmetry constraints $C_1 = d\rho(A_1), C_2 = d\rho(A_2), ..., C_{D+1} = \rho(h_1) - I, ...$ into a single matrix C, which we can breakdown into its nullspace spanned by the columns of $Q \in \mathbb{R}^{m\times r}$ and orthogonal complement $P \in \mathbb{R}^{m\times(m-r)}$ using the singular value decomposition:

$$Cv = \begin{bmatrix} d\rho(A_1) \\ d\rho(A_2) \\ ... \\ \rho(h_1) - I \\ ... \end{bmatrix} v = U \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} P^\top \\ Q^\top \end{bmatrix} v = 0. \tag{6}$$

All symmetric solutions for v must lie in the nullspace of C: $v = Q\beta$ for some coefficients β, and we can then parametrize all symmetric solutions directly in this subspace. Alternatively, defining $\beta = Q^T v_0$ we can reuse any standard parametrization and initialization, but simply project onto the equivariant subspace: $v = QQ^T v_0$.

Thus given any finite dimensional linear representation, we can solve the constraints with a singular value decomposition.[1]. If $v \in \mathbb{R}^m$ the runtime of the approach is $O((M + D)m^3)$.

[1]Equations (4) and (5) apply also to infinite dimensional representations, where ρ and $d\rho$ are linear operators acting on functions v, but solving these on a computer would be more difficult.

(a) S_4 (b) $\mathbb{Z}_4$ (c) $\mathbb{Z}_2^2$ (d) $\mathbb{Z}_4 \ltimes \mathbb{Z}_2^2$

Figure 2. Equivariant basis for permutations, translation, 2d translation, and GCNN symmetries respectively, each of which are solutions to Equation 5 for different groups. The r different solutions in the basis are shown by different colors.

4.3. A Unifying Perspective on Equivariance

In order to make it more concrete and demonstrate its generality, we now show that standard convolutional layers (LeCun et al., 1989), deep sets (Zaheer et al., 2017), invariant graph networks (Maron et al., 2018), and GCNNs (Cohen & Welling, 2016a) are examples of the solutions in equation (6) when specifying a specific symmetry group and representation.

Convolutions: To start off with the 1D case with sequences of n elements and a single channel, $V = \mathbb{R}^n$ is acted upon by cyclic translations from the group $G = \mathbb{Z}_n$. The group can be generated by a single element given by the permuation matrix $\rho(h) = \mathrm{P}[n, 1, 2, ..., n-1]$. Equivariant linear maps from $V \to V$ are of type $T_{(1,1)}$. Expressing the representation and solving eq. (6) with SVD gives the $r = n$ matrices (reshaped from the rows of Q) shown by the circulant matrix in Figure 2, which is *precisely* the way to express convolution as a matrix.

In the typical case of 2D arrays with $V = \mathbb{R}^{n^2}$ elements and multiple channels $c_{\text{in}}\ c_{\text{out}}$, there are $M = 2$ generators of the group $G = \mathbb{Z}_n \times \mathbb{Z}_n = \mathbb{Z}_n^2$ that are $\rho(h_1) = \rho(h) \otimes I$ and $\rho(h_2) = I \otimes \rho(h)$ defined in terms of the generator in the 1D case. For multiple channels, the mapping is $c_{\text{in}} V \to c_{\text{out}} V$ which has type $c_{\text{in}} c_{\text{out}} T_{(1,1)}$ which yields the matrix valued 2D convolution (with $c_{\text{in}} c_{\text{out}} n^2$ independent basis elements) that we are accustomed to using for computer vision[2].

Deep Sets: We can recover the solutions in Zaheer et al. (2017) by specifying $V = \mathbb{R}^n$ and considering S_n (permutation) equivariant linear maps $V \to V$. S_n can be generated in several ways such as with the $M = 2$ generators $\rho(h_1) = \mathrm{P}[1, n-1, 2, 3, ...]$ and $\rho(h_2) = \mathrm{P}[2, 1, 3, 4, ...]$ (Conrad, 2013). Solving the constraints for $T_{(1,1)}$ yields the $r = 2$ dimensional basis $Q = [I, \mathbb{1}\mathbb{1}^\top]$ shown in Figure 2.

Equivariant Graph Networks: Equivariant graph networks in Maron et al. (2018) generalize deep sets to S_n equivariant maps from $T_k \to T_\ell$, such as maps from adja-

[2]Note that the inductive bias of *locality* restricting from $n \times n$ filters to 3×3 filters is not a consequence of equivariance.

cency matrices T_2 to themselves. They show these maps satisfy

$$\forall P \in S_n : \; P^{\otimes(k+\ell)}\mathrm{vec}(W) = \mathrm{vec}(W), \qquad (7)$$

and use analytic techniques to find a basis, showing that the size of the basis is upper bounded[3] by the Bell numbers $1, 2, 5, 15, ...$ Noting that $P = (P^{-1})^\top$, we can now recognize $P^{\otimes k+\ell} = \rho_{(k,\ell)}(P)$ acting on the maps of type $T_{(k,\ell)}$. However we need not solve the combinatorially large Equation 7; our algorithm instead solves it just for the permutation generators $\rho(h_i)$, yielding the same solutions.

GCNNs: The Group Equivariant CNNs in Cohen & Welling (2016a) can be defined abstractly through fiber bundles and base spaces, but we can also describe them in our tensor notation. The original GCNNs have the $G = \mathbb{Z}_4 \ltimes (\mathbb{Z}_n \times \mathbb{Z}_n)$ symmetry group consisting of discrete translations of the grid, as well as 90^o rotations where $\ltimes$ is the semi-direct product. [4] In total, the representation space can be written $V = \mathbb{R}^4 \otimes \mathbb{R}^{n^2}$. We can now disentangle these two parts to read off the $M = 3$ generators for x, y translation and rotation. The translation generators are $I \otimes \rho(h_1)$ and $I \otimes \rho(h_2)$ from the 2D convolution section, as well a generator for rotation $P[4, 1, 2, 3] \otimes \mathrm{Rot}_{90}$ with the Rot_{90} matrix performing 90^o rotations on the grid. Solving for the constraint on $T_{(1,1)}$ yields the G-convolutional layer embedded in a dense matrix shown in Figure 2. Note the diagonal blocks implement rotated copies of a given filter, equivalent to the orientations in the regular representation of a GCNN.

Notably, each of these solutions for convolution, deep sets, equivariant graph networks, and GCNNs are produced as solutions from Equation 6 as a direct consequence of specifying the representation and the group generators. In Appendix E we calculate the equivariant basis for tensor representations of these groups $\mathbb{Z}_n, \mathrm{S}_n, \mathrm{D}_n$, as well as unexplored territory with $\mathrm{SO}(n)$, $\mathrm{O}(n)$, $\mathrm{Sp}(n)$, $\mathrm{SO}^+(1,3)$, $\mathrm{SO}(1,3)$, $\mathrm{O}(1,3)$, $\mathrm{SU}(n)$, and the Rubiks Cube group. We visualize several of these equivariant bases in Figure 3.

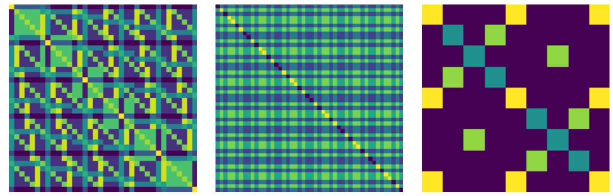
5. Efficiently Solving the Constraint

The practical application of our general approach is limited by two factors: the computational cost of computing the equivariant basis at initialization, and the computational cost of applying the equivariant maps in the forward pass of a network. In this section we address the scalability of the first factor, computing the equivariant basis.

The runtime for using SVD directly to compute the equivariant basis is too costly for all but very small representations $m = \dim(V) < 5000$. We improve upon the naive algorithm with two techniques: dividing the problem into a smaller set of independent subproblems and exploiting structure in the constraint matrices to enable an efficient iterative Krylov subspace approach for computing the nullspace. These two techniques allow us to compute the bases for high dimensional representations while not sacrificing the equivariance or completeness of the solution basis. Our resulting networks run in time similar to a standard MLP.

[3]For small n, the size of the equivariant basis for T_k can actually be less than B_k when $n^k < B_k$.

[4]Cohen & Welling (2016a) also make D_4 dihedral equivariant networks that respect reflections, which can be accomodated by in our framework with 1 additional generator.



(a) $T_4^{S_6}$ (b) T_2^{Rubiks} (c) $T_4^{\mathrm{SO}(3)}$

Figure 3. Equivariant basis for various tensor representations T_k^G where G denotes the symmetry group. The r different solutions in the basis are shown by different colors. For $\mathrm{SO}(3)$ the bases cannot be separated into disjoint set of 0 or 1 valued vectors, and so we choose overlapping colors randomly and add an additional color for 0.

5.1. Dividing into Independent Sub-problems

The feature space U in a neural network can be considered a combination of objects with different types and multiplicities. The features in standard CNN or deep set would be c copies of rank one tensors, $U = cT_1$, where c is the number of channels. Graph networks include both node features T_1 as well as edge features T_2 like the adjacency matrix. More general networks could have a mix of representations, for example 100 scalars, 30 vectors, 10 matrices and 3 higher order tensors: $U = 100T_0 \oplus 30T_1 \oplus 10T_2 \oplus 3T_3$. These composite representations with multiplicity are built from direct sums of simpler representations. $\rho_U(g) = \bigoplus_{a\in\mathcal{A}} \rho_a(g)$ for some collection of representations $\mathcal{A}$.

Since linear maps $U_1 \to U_2$ have the representation $\rho_2 \otimes \rho_1^*$, the product can be expanded as the direct sum

$$\rho_2 \otimes \rho_1^* = \bigoplus_{b\in\mathcal{A}_2} \rho_b \otimes \bigoplus_{a\in\mathcal{A}_1} \rho_a^* = \bigoplus_{(b,a)\in\mathcal{A}_2\times\mathcal{A}_1} \rho_b \otimes \rho_a^*. \quad (8)$$

Since $\oplus$ for both the group and algebra representations concatenates blocks along the diagonal, the constraints can be separated into the blocks given by each of the (b, a) pairs. Each of these constraints (4) and (5) can be solved independently for the $\rho_b \otimes \rho_a^*$ representation and then reassembled into the parts of the full matrix.

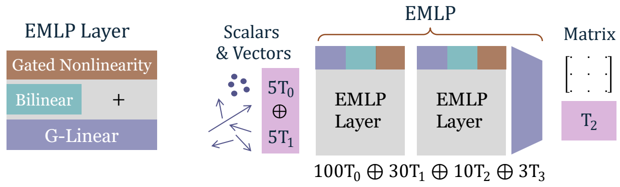


Figure 4. EMLP layers. G-equivariant linear layers, followed by the bilinear layer and a shortcut connection, and finally a gated nonlinearity. Stacking these layers together and choosing some internal representation (shown below), the EMLP maps some collection of geometric quantities to some other collection. Here we show the equivariant mappings from scalars and vectors to matrices.

Unlike Steerable CNNs which use analytic solutions of irreducible representations (Cohen & Welling, 2016b; Weiler & Cesa, 2019), we need not worry about any Clebsch-Gordon coefficients or otherwise, regardless of the representation used. [5] Note that tensor representations make things especially simple since $T_{(p,q)} \otimes T^*_{(r,s)} = T_{(p+s,q+r)}$, but are not required.

5.2. Krylov Method for Efficient Nullspaces

We can exploit structure in the matrices ρ and $d\rho$ for a more efficient solution. With this in mind, we propose to find the nullspace $Q \in \mathbb{R}^{n\times r}$ where r is the rank of the nullspace with the following optimization problem:

$$\min \ \|CQ\|_F^2 \quad s.t \quad Q^\top Q = I. \tag{9}$$

Minimizing using gradient descent, we have a very close relative of QR power iteration (Francis, 1961) and Oja's rule (Garber & Hazan, 2015; De Sa et al., 2015; Shamir, 2015), that instead finds the smallest singular vectors. As the nullspace components are preserved by the gradient updates, the orthogonalization constraint can in fact be removed during the minimization and we list the steps of the iterative method in algorithm 1. Crucially, gradients require only matrix vector multiplies (MVMs) with the constraint matrix C, we never have to form the representation matrices explicitly and can instead implement an efficient MVM for ρ and $d\rho$. Through iterative doubling of the max rank r we need not know the true rank beforehand. As we prove in Appendix C the algorithm produces an ϵ accurate solution in time $O((M+D)\mathcal{T}r\log(1/\epsilon)+r^2n)$ where $\mathcal{T}$ is the time for an MVM with ρ and $d\rho$. The method is "exact" in the sense of numerical algorithms in that we can specify a precision ϵ close to machine precision and converge in $\log(1/\epsilon)$ iterations due to the exponential convergence rate, which we verify in Figure 7.

The pairs of tensor products of representations, $\rho_b(h) \otimes \rho_a(h^{-1})^\top$ and $d\rho_b(A)\bar{\oplus}(-d\rho_a(A)^\top)$ from Equation 8 have Kronecker structure allowing efficent MVMs $(A\otimes B)\mathrm{vec}(W) = \mathrm{vec}(AWB^\top)$. Exploiting this structure alone, solving the constraints for a matrix $W \in \mathbb{R}^c \to \mathbb{R}^c$ takes time

$$O((M+D)\mathcal{T}rc + r^2c^2) \tag{10}$$

where $\mathcal{T}$ is the time for MVMs with constituent matrices $\rho_a, \rho_b, d\rho_a, d\rho_b$. For some of the groups this time $\mathcal{T}$ is in fact a *constant*, for example the permutation generators merely swap two entries, and Lie algebras can often be written in a sparse basis. For high order tensor representations, one can exploit higher order Kronecker structure. Even for discrete groups the runtime is a strict improvement over the approach by van der Pol et al. (2020) which runs in time $O(|G|\mathcal{T}rc + r^2c^2)$. For large discrete groups like S_n our approach gives an exponential speedup, $O(n!) \to O(n)$.

Algorithm 1 Fast Krylov Nullspace

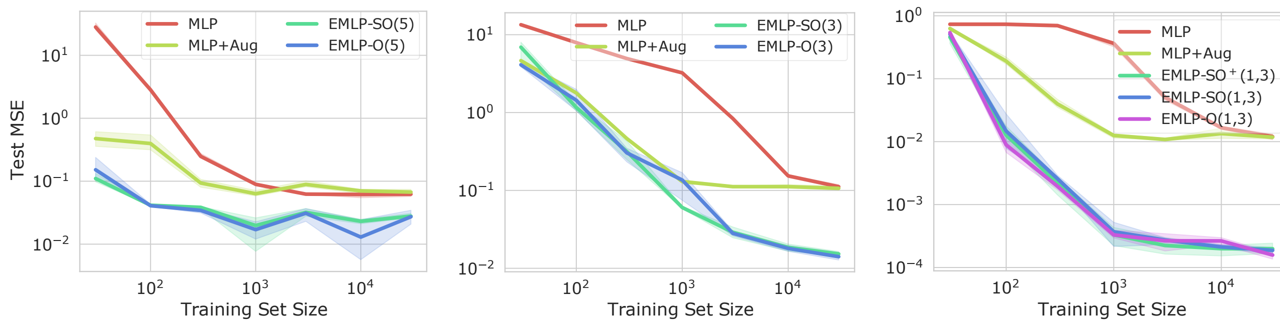
def **KrylovNullspace**(C):
$r_{\max} = r = 10$
while $r = r_{\max}$ **do**
  $r_{\max} \leftarrow 2r_{\max}$
  $Q = \mathrm{CappedKrylovNullspace}(C, r_{\max})$
  $r \leftarrow \mathrm{rank}(Q)$
end
return Q
def **CappedKrylovNullspace**($C, r_{\max}$):
$Q \sim \mathcal{N}(0,1)^{n\times r_{\max}}$
while $L(Q) > \epsilon$ **do**
  $L(Q) = \|CQ\|_F^2$
  $Q \leftarrow Q - \eta\nabla L$
end
$Q, \Sigma, V = \mathrm{SVD}(Q)$
return Q

6. Network Architecture

While the constraint solving procedure can be applied to any linear representations, we will use tensor representations to construct our network. The features in each layer are a collection of tensors of different ranks $v \in U = \bigoplus_{a\in\mathcal{A}} T_{(p_a,q_a)}$ with the individual objects $v_a \in T_{(p_a,q_a)}$. As a heuristic, we allocate the channels uniformly between tensor ranks for the intermediate layers. For example with 256 channels for an SO(3) equivariant layer with $\dim(T_{(p,q)}) = 3^{p+q}$, uniformly allocating channels produces $U = 70T_0 \oplus 23T_1 + 7T_2 + 2T_3$. The input and output layers are set by the types of the data. To build a full equivariant multilayer perceptron (EMLP) from the equivariant linear layer, we also need equivariant nonlinearities.

Gated Nonlinearities: For this purpose we use *gated* nonlinearities introduced in Weiler et al. (2018). Gated nonlin-

[5] For irreducible representations one typically decomposes $\rho_i \otimes \rho_j = Q^{-1}\big(\bigoplus_k \rho_k\big)Q$ with Clebsch-Gordan matrix Q, but we can leave the rep as $\rho_i \otimes \rho_j$ and solve numerically.



(a) O(5) invariant Synthetic (b) O(3) equivariant Inertia (c) O(1,3) invariant $(e^- + \mu^-)$-Scattering

Figure 5. Data efficiency for the synthetic equivariance experiments. Here the EMLP-G models where G are relevant symmetry groups strongly outperforms both standard MLPs and MLPs that have been trained with data augmentation to the given symmetry group, across the range of dataset sizes. The shaded regions depict 95% confidence intervals taken over 3 runs.

earities act separately for each of the different objects in the features (that are concatenated through the direct sum $z = \mathrm{Concat}(\{v_a\}_{a\in\mathcal{A}})$). The nonlinearity takes values $\mathrm{Gated}(v_a) = v_a\sigma(s_a)$ where s_a is a scalar 'gate' for each of the objects. For scalar objects $v_a \in T_0 = \mathbb{R}^1$ and regular representations (which allow pointwise nonlinearities), the gate is just the object itself and so the nonlinearity is just Swish (Ramachandran et al., 2017). For other representations the gate scalars are produced as an additional output of the previous layer.

Universality: The theorem in Maron et al. (2019) shows that tensor networks with pointwise nonlinearities and G-equivariant linear layers for $G \leq S_n$ are universal. However, this result does not extend to the gated nonlinearities required for other groups and representations. As we prove in Appendix D, gated nonlinearities are *not* sufficient for universality in this general case, and can be extremely limiting in practice. The problem relates to not being able to express any kind of contractions between elements with the different objects within a feature layer (like a dot product).

Cheap Bilinear Layers. To address this limitation we introduce an inexpensive bilinear layer which performs tensor contractions on pairs of input objects that produce a given output type. Explicitly, two input objects $v_a \in T_{(a_1,a_2)}$ and $v_b \in T_{(b_1,b_2)}$ can be contracted to give a type $T_{(c_1,c_2)}$ if and only if $(a_1,a_2) = (c_1+b_2, c_2+b_1)$ or $(b_1,b_2) = (c_1+a_2, c_2+a_1)$. In other words, if v_a can be interpreted as a linear map from $T_b \to T_c$ then we can apply $y_c = \mathrm{Reshape}(v_a)v_b$ and vice versa. We add a learnable parameter weighting each of these contractions (excluding scalars).

We can now assemble the components to build a full equivariant multilayer perceptron (EMLP) from the equivariant linear layer, the gated nonlinearities, and the additional bilinear layer. We show how these components are assembled in Figure 4.

7. Experiments

We evaluate EMLP on several synthetic datasets to test its capability on previously unexplored groups, and apply our model to the task of learning dynamical systems with symmetry.

7.1. Synthetic Experiments

O(5) Invariant Task: To start off, we evaluate our model on a synthetic O(5) invariant regression problem $2T_1 \to T_0$ in $d=5$ dimensions given by the function $f(x_1,x_2) = \sin(\|x_1\|) - \|x_2\|^3/2 + \frac{x_1^\top x_2}{\|x_1\|\|x_2\|}$. We evaluate EMLP-SO$(5)$ and EMLP-O(5) which is also equivariant to reflections. We compare against a standard MLP as well as MLP-Aug that is trained with O(5) data augmentation. We show the results in Figure 5.

O(3) **Equivariant Task**: Next we evaluate the networks on the equivariant task of predicting the moment of inertia matrix $\mathcal{I} = \sum_i m_i(x_i^\top x_i I - x_i x_i^\top)$ from $n=5$ point masses and positions. The inputs $X = \{(m_i,x_i)\}_{i=1}^5$ are of type $5T_0 + 5T_1$ (5 scalars and vectors) and outputs are of type T_2 (a matrix), both transform under the group. We apply SO(3) and O(3) equivariant models to this problem. For the baselines, we implement data augmentation for the standard MLP for this equivariant task by simultaneously transforming the input by a random matrix $R \in \mathrm{O}(3)$ and transforming the output accordingly by the inverse transformation: $\hat{y} = R^\top \mathrm{MLP}(\{(m_i, Rx_i\}_{i=1}^5)R$. This kind of equivariant data augmentation that transforms both the input and the output according to the symmetry is strong baseline.

Lorentz Equivariant Particle Scattering: Testing the ability of the model to handle Lorentz equivariance in tasks

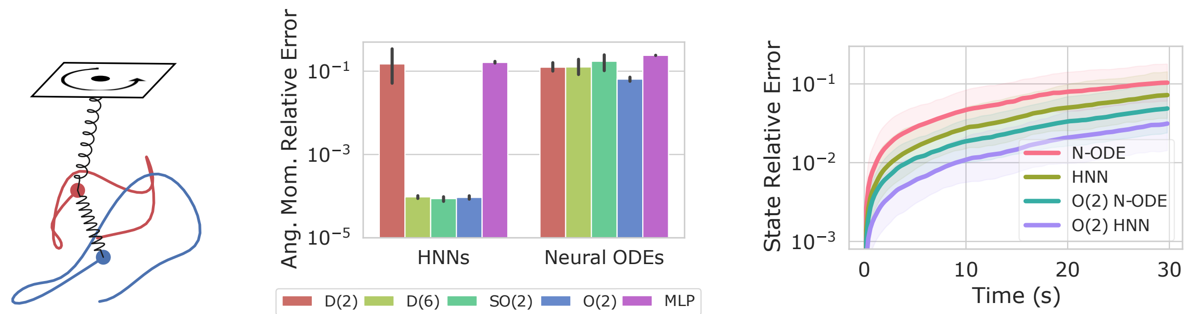


Figure 6. **Left**: A double spring pendulum (12s sample trajectory is shown). The system has an $\mathrm{O}(2)$ symmetry about the z axis. **Middle**: Conservation of angular momentum about the z-axis (the geometric mean of the relative error is computed over $30s$ rollouts and averaged across initial conditions). Errorbars are 95% confidence interval over 3 runs. **Right**: The relative error in the state as the trajectory is rolled out. Shaded regions show 1 standard deviation in log space across the different trajectories rather than models, showing the variance in the data.

relevant to particle physics, we train models to fit the matrix element in electron muon scattering $e^- + \mu^- \to e^- + \mu^-$ which is proportional to the scattering cross-section. The scattering matrix element is proportional to $|\mathcal{M}|^2 \propto$

$$[p^{(\mu}\tilde{p}^{\nu)} - (p^\alpha \tilde{p}_\alpha - p^\alpha p_\alpha)\eta^{\mu\nu}][q_{(\mu}\tilde{q}_{\nu)} - (q^\alpha \tilde{q}_\alpha - q^\alpha q_\alpha)\eta_{\mu\nu}]$$

(Martin, 2012) where q_μ and p_μ are the four momenta for the ingoing electron and muon respectively, while $\tilde{q}_\mu$ and $\tilde{p}_\mu$ are the outgoing momenta, and parentheses $(\mu\nu)$ denotes the symmetrization of indices and repeated indices are contracted. While simple enough express in closed form, the scalar output involves contractions, symmetrization, upper and lower indices, and a metric tensor. Here the inputs are $4T_{(1,0)}$ and the output is a scalar $T_{(0,0)}$. We evaluate EMLP with equivariance not just to the proper orthochronious Lorentz group $\mathrm{SO}^+(1,3)$ from Bogatskiy et al. (2020), but also the special Lorentz group $\mathrm{SO}(1,3)$, and the full Lorentz group $\mathrm{O}(1,3)$ and compare a MLP baseline that uses $\mathrm{O}(1,3)$ data augmentation.

As shown in Figure 5, our EMLP model with the given equivariance consistently outperform sthe baseline MLP trained with and without data augmentation across the different dataset sizes and tasks, often by orders of magnitude.

7.2. Modeling dynamical systems with symmetries

Finally we turn to the task of modeling dynamical systems. For dynamical systems, the equations of motion can be written in terms of the state $\mathbf{z} \in \mathbb{R}^m$ and time t as $d\,\mathbf{z}/dt = F(\mathbf{z}, t)$. Neural ODEs (Chen et al., 2018) provide a way of learning these dynamics directly from trajectory data. A neural network parametrizes the function F_θ and the learned dynamics can be rolled out using a differentiable ODE solver $(\hat{\mathbf{z}}_1, \dots, \hat{\mathbf{z}}_T) = \mathrm{ODESolve}(\mathbf{z}_0, F_\theta, (t_1, t_2, ..., t_T))$ and fit to trajectory data with the L2 loss $L(\theta) = \frac{1}{T}\sum_{t=1}^{T} ||\hat{\mathbf{z}}_t - \mathbf{z}_t||_2^2$.

Many physically occurring systems have a Hamiltonian structure, meaning that the state can be split into generalized coordinates and momenta $\mathbf{z} = (\mathbf{q}, \mathbf{p})$, and the dynamics can be written in terms of the gradients of a scalar $\mathcal{H}(z)$ known as the Hamiltonian, which often coincides with the total energy. $\frac{d\,\mathbf{z}}{dt} = J\nabla\mathcal{H}$ with $J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}$. As shown in Greydanus et al. (2019) with Hamiltonian Neural Networks (HNNs), one can exploit this Hamiltonian structure by parametrizing $\hat{\mathcal{H}}_\theta(\mathbf{z})$ with a neural network, and then taking derivatives to find the implied Hamiltonian dynamics. For problems with Hamiltonian structure HNNs often lead to improved performance, and better energy conservation.

A dynamical system can have *symmetries* such as the symmetries given by $F(\rho(g)\,\mathbf{z}, t) = \rho(g)F(\mathbf{z}, t)$ for some linear representation, which is equivariance in the first argument. Meanwhile Hamiltonian dynamics have symmetries according to invariances of the Hamiltonian $\mathcal{H}(\rho(g)\,\mathbf{z}) = \mathcal{H}(\mathbf{z})$. Continuous symmetries of the Hamiltonian are of special significance since they produce conservation laws such as conservation of linear and angular momentum or conservation of charge as part of the Noether theorem (Noether, 1971).

We apply our EMLP model to the task of learning the dynamics of a double pendulum connected by springs in 3D shown in Figure 6. The problem exhibits a $\mathrm{O}(2)$ rotational and reflectional symmetry about the z-axis as well as Hamiltonian structure. As the state space cannot be traversed by the group elements alone, it is not a homogeneous space, a setting that has been explored very little in the equivariance literature (Cohen et al., 2018a).

However, we can readily use EMLP on this problem and we show in Table 1 and Figure 6 that exploiting the $\mathrm{O}(2)$ symmetry (and subgroups $\mathrm{SO}(2)$, D_6) with EMLP leads to improved performance for both Neural ODE and HNN

	O(2)	SO(2)	D_6	MLP
N-ODEs:	**0.019(1)**	0.051(36)	0.036(25)	0.048
HNNs:	**0.012(2)**	0.015(3)	0.013(2)	0.028

Table 1. Geometric mean of rollout errors (relative error) over T=30s for the various EMLP-G symmetric HNNs and Neural ODEs (N-ODE) vs ordinary MLP HNNs and N-ODEs. Errorbars are 1 standard deviation computed over 3 trials, with notation .012(2) meaning .012 $\pm$ .002.

models. Furthermore, enforcing the continuous rotation symmetry in the EMLP-HNN models yields conservation of angular momentum about the z-axis, a useful property for learned simulations. Interestingly the dihedral group D_6 which is discrete does not satisfy Noether's theorem and yet it still yields approximate angular momentum conservation, but the coarser D_2 symmetry does not. As expected, all Neural ODE models do not conserve angular momentum as Noether's theorem only applies to the Hamiltonians and not to the more general ODEs. While conservation laws from learning invariant Hamiltonians was also explored in Finzi et al. (2020b) with LieConv, LieConv models assume permutation equivariance which is broken by the pivot in this system. Because EMLP is general, we can apply it to this non permutation symmetric and non transitively acting rotation group that is embedded in the larger state space.

8. Discussion

We presented a construction for equivariant linear layers that is completely general to the choice of representation and matrix group. Convolutions, deep sets, equivariant graph networks and GCNNs all fall out of the algorithm naturally as solutions for a given group and representation. Through an iterative MVM based approach, we can solve for the equivariant bases of very large representations. Translating these capabilities into practice, we build EMLP and apply the model to problems with symmetry including Lorentz invariant particle scattering and dynamical systems, showing consistently improved generalization.

Though EMLP is not much slower than a standard MLP, dense matrix multiplies in an MLP and our EMLP make it slow to train models the size of convnets or large graph networks which have specialized implementations. With the right techniques, this apparent generality-specialization tradeoff may be overcome. The flexibility of our approach should lower the costs of experimentation and allow researchers to more easily test out novel representations. Additionally we hope that our constraint solver can help launch a variety of new methods for learning symmetries, modeling heterogeneous data, or capturing prior knowledge.

Acknowledgements

We would like to thank Roberto Bondesan and Robert Young for useful discussions about equivariance and Lie algebra representations. This research is supported by an Amazon Research Award, NSF I-DISRE 193471, NIH R01 DA048764-01A1, NSF IIS-1910266, and NSF 1922658 NRT-HDR: FUTURE Foundations, Translation, and Responsibility for Data Science.

References

Anderson, B., Hy, T. S., and Kondor, R. Cormorant: Covariant molecular neural networks. In *Advances in Neural Information Processing Systems*, pp. 14510–14519, 2019.

Bekkers, E. J. B-spline cnns on lie groups. *arXiv preprint arXiv:1909.12057*, 2019.

Bogatskiy, A., Anderson, B., Offermann, J. T., Roussi, M., Miller, D. W., and Kondor, R. Lorentz group equivariant neural network for particle physics. *arXiv preprint arXiv:2006.04780*, 2020.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. In *Advances in neural information processing systems*, pp. 6571–6583, 2018.

Cohen, T. and Welling, M. Group equivariant convolutional networks. In *International conference on machine learning*, pp. 2990–2999, 2016a.

Cohen, T., Geiger, M., and Weiler, M. A general theory of equivariant cnns on homogeneous spaces. *arXiv preprint arXiv:1811.02017*, 2018a.

Cohen, T. S. and Welling, M. Steerable cnns. *arXiv preprint arXiv:1612.08498*, 2016b.

Cohen, T. S., Geiger, M., Köhler, J., and Welling, M. Spherical cnns. *arXiv preprint arXiv:1801.10130*, 2018b.

Conrad, K. Generating sets. *Expository, unpublished paper on the author's personal homepage*, 2013.

De Sa, C., Re, C., and Olukotun, K. Global convergence of stochastic gradient descent for some non-convex matrix problems. In *International Conference on Machine Learning*, pp. 2332–2341. PMLR, 2015.

Dym, N. and Maron, H. On the universality of rotation equivariant point cloud networks. *arXiv preprint arXiv:2010.02449*, 2020.

Eberly, W. Reliable krylov-based algorithms for matrix null space and rank. In *Proceedings of the 2004 international symposium on Symbolic and algebraic computation*, pp. 127–134, 2004.

Esteves, C., Allen-Blanchette, C., Zhou, X., and Daniilidis, K. Polar transformer networks. *arXiv preprint arXiv:1709.01889*, 2017.

Finzi, M., Bondesan, R., and Welling, M. Probabilistic numeric convolutional neural networks. *arXiv preprint arXiv:2010.10876*, 2020a.

Finzi, M., Stanton, S., Izmailov, P., and Wilson, A. G. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. In *International Conference on Machine Learning*, 2020b.

Francis, J. G. The qr transformation a unitary analogue to the lr transformation—part 1. *The Computer Journal*, 4 (3):265–271, 1961.

Fuchs, F. B., Worrall, D. E., Fischer, V., and Welling, M. Se (3)-transformers: 3d roto-translation equivariant attention networks. *arXiv preprint arXiv:2006.10503*, 2020.

Garber, D. and Hazan, E. Fast and simple pca via convex optimization. *arXiv preprint arXiv:1509.05647*, 2015.

Greydanus, S., Dzamba, M., and Yosinski, J. Hamiltonian neural networks. In *Advances in Neural Information Processing Systems*, pp. 15353–15363, 2019.

Guralnick, R. M. On the number of generators of a finite group. *Archiv der Mathematik*, 53(6):521–523, 1989.

Ipsen, I. C. Computing an eigenvector with inverse iteration. *SIAM review*, 39(2):254–291, 1997.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Lang, L. and Weiler, M. A wigner-eckart theorem for group equivariant convolution kernels. *arXiv preprint arXiv:2010.10952*, 2020.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Marcos, D., Volpi, M., Komodakis, N., and Tuia, D. Rotation equivariant vector field networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5048–5057, 2017.

Maron, H., Ben-Hamu, H., Shamir, N., and Lipman, Y. Invariant and equivariant graph networks. *arXiv preprint arXiv:1812.09902*, 2018.

Maron, H., Fetaya, E., Segol, N., and Lipman, Y. On the universality of invariant networks. *arXiv preprint arXiv:1901.09342*, 2019.

Maron, H., Litany, O., Chechik, G., and Fetaya, E. On learning sets of symmetric elements. *arXiv preprint arXiv:2002.08599*, 2020.

Martin, V. Particle physics, 2012. URL https://www2.ph.ed.ac.uk/~vjm/Lectures/SHParticlePhysics2012_files/PPLecture6.pdf.

Noether, E. Invariant variation problems. *Transport theory and statistical physics*, 1(3):186–207, 1971.

Ramachandran, P., Zoph, B., and Le, Q. V. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

Ravanbakhsh, S. Universal equivariant multilayer perceptrons. *arXiv preprint arXiv:2002.02912*, 2020.

Ravanbakhsh, S., Schneider, J., and Poczos, B. Equivariance through parameter-sharing. In *International Conference on Machine Learning*, pp. 2892–2901. PMLR, 2017.

Serviansky, H., Segol, N., Shlomi, J., Cranmer, K., Gross, E., Maron, H., and Lipman, Y. Set2graph: Learning graphs from sets. *arXiv preprint arXiv:2002.08772*, 2020.

Shamir, O. A stochastic pca and svd algorithm with an exponential convergence rate. In *International Conference on Machine Learning*, pp. 144–152. PMLR, 2015.

Smidt, T., Geiger, M., and Rackers, J. github.com/e3nn/e3nn. March 2020. doi: 10.5281/zenodo.3724982. URL https://doi.org/10.5281/zenodo.3724982.

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

Turner, W. J. A block wiedemann rank algorithm. In *Proceedings of the 2006 international symposium on Symbolic and algebraic computation*, pp. 332–339, 2006.

van der Pol, E., Worrall, D., van Hoof, H., Oliehoek, F., and Welling, M. Mdp homomorphic networks: Group symmetries in reinforcement learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Wang, R., Albooyeh, M., and Ravanbakhsh, S. Equivariant maps for hierarchical structures. *arXiv preprint arXiv:2006.03627*, 2020.

Weiler, M. and Cesa, G. General e (2)-equivariant steerable cnns. In *Advances in Neural Information Processing Systems*, pp. 14334–14345, 2019.

Weiler, M., Geiger, M., Welling, M., Boomsma, W., and Cohen, T. S. 3d steerable cnns: Learning rotationally equivariant features in volumetric data. In *Advances in Neural Information Processing Systems*, pp. 10381–10392, 2018.

Winkelmann, J. Dense random finitely generated subgroups of lie groups. *arXiv preprint math/0309129*, 2003.

Wood, J. and Shawe-Taylor, J. Representation theory and invariant neural networks. *Discrete Applied Mathematics*, 69(1):33–60, 1996. ISSN 0166-218X. doi: https://doi.org/10.1016/0166-218X(95)00075-3. URL https://www.sciencedirect.com/science/article/pii/0166218X95000753.

Worrall, D. and Welling, M. Deep scale-spaces: Equivariance over scale. In *Advances in Neural Information Processing Systems*, pp. 7366–7378, 2019.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. In *Advances in neural information processing systems*, pp. 3391–3401, 2017.