PHILOSOPHICAL TRANSACTIONS A

royalsocietypublishing.org/journal/rsta

Research

Cite this article: Gregory WG, Hogg DW, Blum-Smith B, Arias MT, Wong KWK, Villar S. 2025 Equivariant geometric convolutions for dynamical systems on vector and tensor images. *Phil. Trans. R. Soc. A* **383**: 20240247. https://doi.org/10.1098/rsta.2024.0247

Received: 31 October 2024
Accepted: 26 February 2025

One contribution of 13 to a theme issue 'Partial differential equations in data science'.

Subject Areas:
applied mathematics, computational mathematics, differential equations

Keywords:
convolutional neural networks, group equivariance, emulation

Author for correspondence:
Soledad Villar
e-mail: svillar3@jhu.edu

Equivariant geometric convolutions for dynamical systems on vector and tensor images

Wilson G. Gregory[1], David W. Hogg[2,3,5], Ben Blum-Smith[1], Maria Teresa Arias[6], Kaze W. K. Wong[1] and Soledad Villar[1,4]

[1]Applied Mathematics and Statistics, Johns Hopkins University, Baltimore, MD, USA
[2]Center for Cosmology and Particle Physics, Department of Physics, New York University, New York, NY, USA
[3]CCA, and [4]CCM, Flatiron Institute, New York, NY, USA
[5]Max-Planck-Institut fur Astronomie, Heidelberg, Baden-Württemberg, Germany
[6]Mathematics, Universidad Autonoma de Madrid, Madrid, Spain

WGG, 0000-0002-5511-0683; DWH, 0000-0003-2866-9403; BBS, 0000-0001-5284-8271; MTA, 0009-0003-7304-1866; KWKW, 0000-0001-8432-7788; SV, 0000-0003-4968-3829

Machine learning methods are increasingly being employed as surrogate models in place of computationally expensive and slow numerical integrators for a bevy of applications in the natural sciences. However, while the laws of physics are relationships between scalars, vectors and tensors that hold regardless of the frame of reference or chosen coordinate system, surrogate machine learning models are not coordinate-free by default. We enforce coordinate freedom by using geometric convolutions in three model architectures: a ResNet, a Dilated ResNet and a UNet. In numerical experiments emulating two-dimensional compressible Navier–Stokes, we see better accuracy and improved stability compared with baseline surrogate models in almost all cases. The ease of enforcing coordinate freedom without making major changes to the model architecture provides an exciting recipe for any convolutional neural network-based method applied to an appropriate class of problems.

This article is part of the theme issue 'Partial differential equations in data science'.

THE ROYAL SOCIETY PUBLISHING

royalsocietypublishing.org/journal/rsta Phil. Trans. R. Soc. A 383: 20240247

1. Introduction

Contemporary natural science features many datasets that are images, lattices or grids of geometric objects. These might be observations of intensities (scalars), velocities (vectors), magnetic fields (pseudovectors) or polarizations (2-tensors) on a surface or in a volume. Any grid of vectors or tensors can be seen as a generalization of the concept of an image in which the intensity in each pixel is replaced with a geometric object—scalar, vector, tensor or their pseudo counterparts. These objects are *geometric* in the sense that they are defined in terms of their transformation properties under geometric operators such as rotation, translation and reflection. Likewise, a grid of these objects is also geometric, so we will refer to them as *geometric images*.

There are many questions that we might like to answer about a dataset of geometric images. The images could be the initial conditions of a simulation discretized to a regular grid; see figure 1 for some examples. A critical problem in astronomy, climate science and many other fields involves modelling the evolution of velocity, pressure and density fields according to the Navier–Stokes equations. Classical numerical methods to solve these equations can be reliable and accurate [5], but they are computationally expensive and rely on having the full specification of the differential equations that govern the physics. Creating surrogate models with machine learning (ML) methods has shown promise as an alternative because they allow us to make predictions based on only partially specified dynamical systems [6]. Moreover, once trained on the desired spatial and temporal scales, these surrogate models could generate an approximate solution from an initial condition much faster than a traditional solver. However, long-term stability in surrogate models remains a concern [7].

One potential culprit for unstable rollouts is that ML models are not coordinate-free by default; they operate on the *components* of the vectors rather than the vectors themselves. In typical contexts, the input channels of a convolutional neural network (CNN) are the red, green and blue channels of a colour image; these are then combined arbitrarily in the layers of the CNN. The naive, flawed approach to applying CNN methods to geometric images is to treat the components of the vector or tensor as independent channels, ignoring how these objects behave under geometric operations.

The fundamental observation inspiring this work is that when an arbitrary function is applied to the components of vectors and tensors, the geometric structure of these objects is destroyed [8]. There are strict rules, dating back to the early days of differential geometry [9], about how geometric objects can be combined to produce new geometric objects, consistent with coordinate freedom and transformation rules. These rules constitute a theme of [10], where they are combined into a *geometric principle* (see page 1 of [10]). With the tools of equivariant ML, we can make better and more efficient models by incorporating the rules of coordinate freedom. Similar ideas have had success in numerical integrators as well [11,12].

The concept of equivariance is simple. Given a group G with action on some spaces X and Y, we say a function $f : X \to Y$ is *equivariant* with respect to G if for all $x \in X$ and $g \in G$ we have $f(g \cdot x) = g \cdot f(x)$. For equivariant ML, we learn a function f over a class of equivariant functions with respect to a relevant group. Ideally, we would like our group to express all possible coordinate transformations, but this includes all possible diffeomorphisms, a computationally intractable group [13]. Thus, in practice, we will consider certain rotations, reflections and translations.

The symmetries that these rules suggest are continuous symmetries. But of course, images are usually—and for our purposes, *always*—regular, discrete grids of values. For our purposes, instead of the continuous symmetries respected by the tensor objects in the image pixels, there will be discrete symmetries for each geometric image taken as a whole. We will define these discrete symmetry groups and use them to define a useful kind of group equivariance for functions of geometric images. When we enforce this equivariance, the convolution filters that appear look very much like the differential operators that appear in discretizations of vector calculus. Even though we only implement a relatively small set of symmetries, we numerically observe that they provide a significant improvement over baselines that do not respect these symmetries. In a similar spirit, related observations have been made in the numerical analysis literature, where using

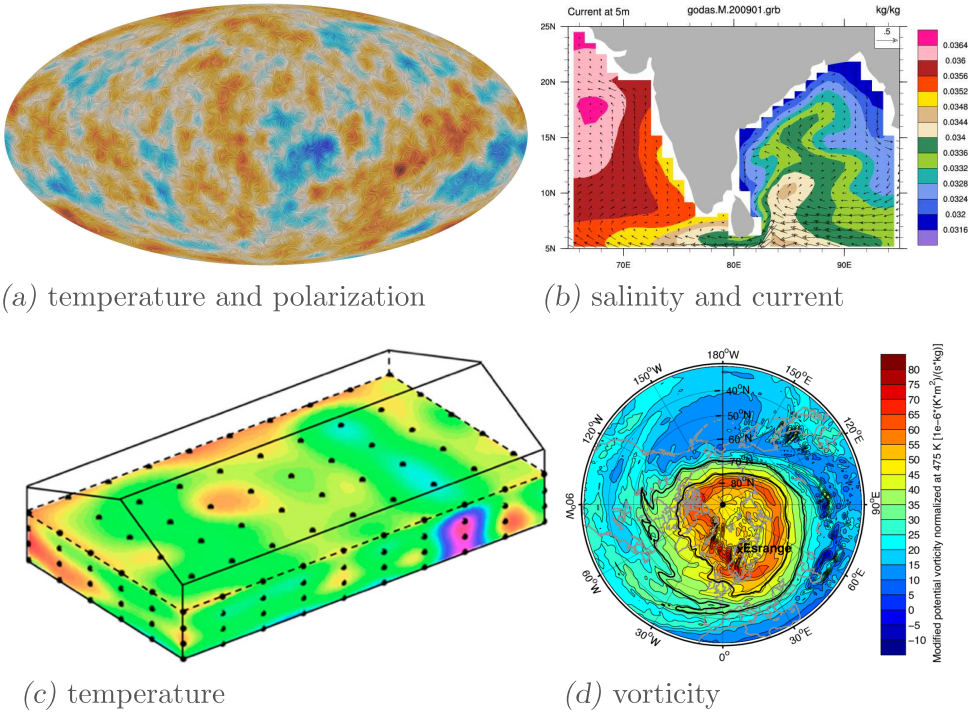

Figure 1. Examples of geometric images in the natural sciences. (*a*) A visualization of a temperature map and a polarization map from the ESA Planck Mission [1] (courtesy ESA/NASA/JPL-Caltech). The colour map shows a temperature field (a scalar or $0_{(+)}$-tensor) on the sphere, and the whiskers show the principal eigenvector direction of a $2_{(+)}$-tensor field in two dimensions. (*b*) Two-dimensional maps of ocean current (arrows; a vector or $1_{(+)}$-tensor field) and ocean salinity (colour; a scalar or $0_{(+)}$-tensor field) [2]. (*c*) A three-dimensional map of temperature (a scalar or $0_{(+)}$-tensor field) based on sensors distributed throughout the volume of a granary [3]. (*d*) A two-dimensional map of potential vorticity (a pseudoscalar or $0_{(-)}$-tensor field) in the Earth's atmosphere, measured for the purposes of predicting storms [4].

discretizations that respect the symmetries of the underlying differential operators improves the accuracy of the numerical simulations [14].

The numerical experiments in this work focus on modelling the Navier–Stokes equations, which involve scalar fields and vector fields. However, the model we develop, the *GeometricImageNet*, can be immediately applied to geometric images of any tensor order or parity.

Our contributions: The contributions of this paper are the following:

- We define the geometric convolution of tensor images using tensor products and contractions (§3b).
- We extend the results of [15] and [16] to prove translation and roto-reflection equivariance of geometric convolutions on tensor images (§4).
- We construct a novel model architecture using geometric convolutions and display its advantages over non-equivariant methods through numerical experiments on compressible Navier–Stokes simulations (§§5 and 6).

Additionally, we discuss related work in §2 and mathematical background in §3a. The proofs have been sequestered in the appendix along with a larger exploration of related work.

2. Related work

The difficulty of modelling Navier–Stokes and other partial differential equations has made the surrogate neural network approach popular in recent years. The CNN approach without regard

to coordinate freedom is common [17–20] and can be successful with sufficient data. Some approaches like the Fourier neural operator [21] are adept at handling images at any resolution. Other methods have tried to incorporate the physical laws back into ML models under the broad category of physics-informed ML [22–24].

Equivariant ML is one approach to incorporating physical laws in learned methods by explicitly enforcing the appropriate symmetry in the architecture of the network. When we expect our target function to be equivariant to that group, this strategy improves the model's generalization and accuracy (see, for instance, [25–29]) and is a powerful remedy for data scarcity (see [30]). Equivariant networks, in certain cases, can approximate any continuous equivariant function (see [31–34]).

Equivariant models have been built for many different symmetry groups, such as translations [35], gauge symmetries [36], permutations [37], rotations/reflections [16,26,38,39] or multiple symmetries [32,40]. There are many approaches to building equivariant models, such as using data augmentation [41,42], invariant theory [43], group convolutions [16], canonicalization [44,45] or irreducible representations [38,39,46]. Our paper uses tensor operations to enforce equivariance, in a similar manner to [37] and [47]. However, both works focus on single tensor inputs and outputs rather than tensor images, which significantly changes the methodology of the linear layers. Closest to our paper in both strategy and application are [26] and [48], but they implement the symmetries with irreducible representations and Clifford algebras, respectively.

Each equivariant method has some challenges. Group convolutions require convolving over the group elements in addition to the spatial dimensions, which can be expensive for larger groups. The Clifford algebras can handle vectors and pseudovectors naturally, but they cannot handle all higher order tensors because they are a quotient group of tensor algebra [49, Ch. 14, theorem 4.1]. Steerable methods require using irreducible representations and decomposing higher order tensors, which can be somewhat involved [50]. By contrast, the geometric convolutions we present in this paper operate on tensors in their natural, Cartesian form, which allows every step of the network to remain interpretable from a physics perspective. Furthermore, geometric convolutions are naturally discrete like their input images, exactly equivariant to the discrete symmetries of those images and able to handle any tensor order or parity.

See appendix B for a more in-depth description of the mathematical details of the related work.

3. Geometric objects and geometric images

We define the geometric objects and geometric images that we use to generalize classical images in scientific contexts in §3a,b. The main point is that the channels of geometric images, the components of vectors and tensors, are not independent. There is a set of allowed operations on geometric objects that respect the structure and coordinate freedom of these objects.

(a) Geometric objects

We start by fixing d, the dimension of the space, which will typically be 2 or 3. The coordinate transformations will be given by the orthogonal group $O(d)$, the space of isometries of $\mathbb{R}^d$ that fix the origin. The geometric principle from classical physics [10] states that geometric objects should be coordinate-free scalars, vectors and tensors, or their negative-parity pseudo counterparts. By coordinate-free we mean that if F is a function with geometric inputs, outputs and parameters, then $F(g \cdot v) = g \cdot F(v)$ for all objects v and all $g \in O(d)$. This is the mathematical concept of equivariance, which we will explore further in §4. This requires that the definitions of the geometric objects are inseparable from how $O(d)$ acts on them.

Definition 1 ((pseudo-)scalars)**.** Let $s \in \mathbb{R}$ have an assigned parity $p \in \{-1, +1\}$. Let $g \in O(d)$ and let $M(g)$ be the standard $d \times d$ matrix representation of g, i.e. $M(g^{-1}) = M(g)^{-1} = M(g)^\top$. Then the

action of g on s, denoted $g \cdot s$, is defined as

$$g \cdot s = \det(M(g))^{\frac{1-p}{2}}\, s. \tag{3.1}$$

When $p = +1$, s is a *scalar* and $\det(M(g))^{\frac{1-p}{2}} = 1$ so the action is just the identity. When $p = -1$, s is a *pseudoscalar*, so $\det(M(g))^{\frac{1-p}{2}} = \det(M(g)) = \pm 1$, and there is a sign flip if g involves an odd number of reflections.

Definition 2 ((pseudo-)vectors)**.** Let $v \in \mathbb{R}^d$ be a *vector*, and let v have parity $p \in \{-1, +1\}$. Let $g \in O(d)$, and let $M(g)$ be the standard matrix representation of g. Then the action of g on v, denoted $g \cdot v$, is defined as

$$g \cdot v = \det(M(g))^{\frac{1-p}{2}} M(g)\, v, \tag{3.2}$$

where parity p has the same effect as on the scalars.

We can now construct higher order tensors using the tensor (outer) product.

Definition 3 (k(p)-tensors)**.** The space $\mathbb{R}^d$ equipped with the action $O(d)$ defined by (3.2) is the space of $1_{(p)}$*-tensors*. If we have $k1_{(p_i)}$-tensors denoted v_i, then $T := v_1 \otimes ... \otimes v_k$ is a *rank–1* $k_{(p)}$*-tensor*, where $p = \prod_{i=1}^{k} p_i$ and the action of $O(d)$ is defined as

$$g \cdot (v_1 \otimes ... \otimes v_k) = (g \cdot v_1) \otimes ... \otimes (g \cdot v_k). \tag{3.3}$$

Thus, a tensor T is an element of a vector space $(\mathbb{R}^d)^{\otimes k}$, which we denote $\mathcal{T}_{d,k,p}$. To get higher rank tensors, we can add tensors of the same order k and parity p, and the action of $O(d)$ extends linearly.

Note that the parity p is not an intrinsic quality of the components of a tensor. For example, a vector and a pseudovector could be equal for a certain choice of coordinates, but they would behave differently under some coordinate transformations. Also, note the distinction between the *order* k of the $k_{(p)}$-tensor and the rank of the tensor. We could have a $2_{(p)}$-tensor of rank 1, like those we use in definition 3. We refer to the components of tensors with Einstein summation notation.

Definition 4 (Einstein summation notation)**.** In *Einstein summation notation*, the components of tensors are referred to by subscripts, e.g. $[a]_{ij}$ for the i^{th}, j^{th} component of $2_{(p)}$-tensor a where i and j are in the range $1, ..., d$. In this paper, we assume that our tensor images have a Riemannian metric of the identity matrix, so we do not need to distinguish between covariant and contravariant indices. A subscript index may appear exactly once in a term, in which case we are taking the outer product, or exactly twice, in which case we are summing over (contracting) that index.

This notation can be used to express a lot of familiar operations. For example, the dot product of vectors a, b is written as $[a]_i[b]_i$. The product of two $2_{(p)}$-tensors (represented as two $d \times d$ matrices A and B) is written as

$$[A\,B]_{i,j} = [A]_{i,k}\,[B]_{k,j} := \sum_{k=1}^{d} [A]_{i,k}\,[B]_{k,j}, \tag{3.4}$$

where the sum from 1 to d on repeated index k is implicit in the middle expression. In summation notation, the group action of (equation (3.3)) on $k_{(p)}$-tensor b is explicitly written

$$[g \cdot b]_{i_1,...,i_k} = \det(M(g))^{\frac{1-p}{2}}\, [b]_{j_1,...,j_k}\, [M(g)]_{i_1,j_1} \cdots [M(g)]_{i_k,j_k} \tag{3.5}$$

for all $g \in O(d)$. For example, a $2_{(+)}$-tensor has the transformation property $[g \cdot b]_{i,j} = [b]_{k,\ell}\,[M(g)]_{i,k}\,[M(g)]_{j,\ell}$, which, in normal matrix notation, is written as $g \cdot b = M(g)\, b\, M(g)^\top$. To make operations on general $k_{(p)}$-tensor more concise, we adopt the following two definitions.

Definition 5 (tensor product)**.** Let a be a $k_{(p)}$-tensor and let b be a $k'_{(p')}$-tensor. Then the *tensor product of a and b*, denoted $a \otimes b$, is the $(k+k')_{(pp')}$-tensor whose $i_1, \ldots, i_{k+k'}$ components are defined as

$$[a \otimes b]_{i_1,\ldots,i_{k+k'}} = [a]_{i_1,\ldots,i_k}\,[b]_{i_{k+1},\ldots,i_{k+k'}}. \tag{3.6}$$

Definition 6 (k-contraction)**.** Let a be a $(2k+k')_{(p)}$-tensor, then the *k-contraction* $\iota_k(a)$ is a $k'_{(p)}$-tensor defined as

$$[\iota_k(a)]_{j_1,\ldots,j_{k'}} = [a]_{i_1,\ldots,i_k,i_1,\ldots,i_k,j_1\ldots,j_{k'}}. \tag{3.7}$$

In other words, we are contracting over indices $(1,k)$ to $(k+1,2k)$.

It is helpful to think of the contraction as the generalization of the trace to higher order tensors, where we are summing over k pairs of axes. For a $2_{(p)}$-tensor a, the tensor contraction $\iota_1(a)$ is exactly the trace, a $0_{(p)}$-tensor. If a is a $5_{(p)}$-tensor, then the contraction $\iota_2(a)$ is the $1_{(p)}$-tensor given by

$$[\iota_2(a)]_j = [a]_{i,\ell,i,\ell,j} = \sum_{i=1}^{d}\sum_{\ell=1}^{d}[a]_{i,\ell,i,\ell,j}. \tag{3.8}$$

We use the k-contraction to define a norm for tensors, which is equivalent to the ℓ_2 norm on the vectorized tensor or the Frobenius norm for matrices extended to tensors.

Definition 7 (ℓ_2 tensor norm)**.** Let a be a $k_{(p)}$-tensor. Then the ℓ_2*tensor norm* $\|\cdot\|_2 : \mathcal{T}_{d,k,p} \to \mathcal{T}_{d,0,+}$ is defined as

$$\|a\|_2 = \sqrt{\iota_k(a \otimes a)}. \tag{3.9}$$

(b) Geometric images and operations

We will start by considering square (or cubic or hyper-cubic) images on a d-torus. We consider an image A with N equally spaced pixels in each dimension for N^d pixels total. Working on a square regular grid on the d-torus is essential for the equivariance results we develop in §4; the definitions and operations below are applicable with minor adjustments to rectangular, non-toroidal arrays as well. Each pixel contains a $k_{(p)}$-tensor, where k and p are the same for each pixel. We define the geometric images as follows.

Definition 8 (geometric image)**.** A *geometric image* is a function $A : [N]^d \to \mathcal{T}_{d,k,p}$, where $[N] = \{0, 1, \ldots, N-1\}$. The set of geometric images is denoted $\mathcal{A}_{N,d,k,p}$. We will also consider $k_{(p)}$-tensor images on the d-torus, where $[N]^d$ is given the algebraic structure of $(\mathbb{Z}/N\mathbb{Z})^d$. The pixel index of a geometric image, often $\bar{\imath}$, is naturally a $1_{(+)}$-tensor.

Just as the space of $k_{(p)}$-tensors is a vector space, the space of geometric images is also a vector space. Thus, they include vector addition and scalar multiplication. Additionally, for each tensor operation defined in §3a, we can define an analogous operation on geometric images that is performed pixel-wise.

We now turn to the first major contribution of this paper, the generalization of convolution to take geometric images as inputs and return geometric images as outputs. The idea is that a geometric image of $k_{(p)}$-tensors is convolved with a geometric filter of $k'_{(p')}$-tensors to produce a geometric image that contains $(k+k')_{(pp')}$-tensors, where each pixel is a sum of outer products.

These $(k+k')_{(p\,p')}$-tensors can then be contracted down to lower-order tensors using contractions (definition 6). Note that the side length M of the geometric filter can be any positive odd number, but typically it will be much smaller than the side length N of the geometric image.

Definition 9 (geometric convolution)**.** Given $A \in \mathcal{A}_{N,d,k,p}$ and $C \in \mathcal{A}_{M,d,k',p'}$ with $M = 2m+1$ for some positive integer m, the *geometric convolution* $A * C$ is a $(k+k')_{(p\,p')}$-tensor image such that

$$(A * C)(\bar{\imath}) = \sum_{\bar{a}\in[-m,m]^d} A(\bar{\imath}-\bar{a}) \otimes C(\bar{a}+\bar{m}), \tag{3.10}$$

where $\bar{\imath}-\bar{a}$ is the translation of $\bar{\imath}$ by $\bar{a}$ on the d-torus pixel grid $(\mathbb{Z}/N\mathbb{Z})^d$ and $\bar{m}$ is the vector of all m.

This definition is on the torus to achieve exact translation equivariance, but, in practice, we can use zero padding or any other form of padding as the situation requires. Additionally, geometric convolution can be adapted to use longer strides, filter dilation, transposed convolution or other convolution variations common in the literature. See figure 3a for examples with a scalar and vector filter. We can define max pooling using the ℓ_2 norm of a tensor as follows:

Definition 10 (max pool$_b$)**.** Let b be a positive integer, and let $A \in \mathcal{A}_{N,d,k,p}$, where b divides N. Then the function max pool$_b : \mathcal{A}_{N,d,k,p} \to \mathcal{A}_{N/b,d,k,p}$ is defined for each pixel index $\bar{\imath} \in [0,(N/b)-1]^d$

$$\text{max pool}_b(A)(\bar{\imath}) = A\left(b\,\bar{\imath} + \arg\max_{\bar{a}\in[0,b-1]^d} \|A(b\,\bar{\imath}+\bar{a})\|_2\right). \tag{3.11}$$

The convolution, contraction, index-permutation and pooling operators above effectively span a large class of linear functions from geometric images to geometric images.

4. Functions of geometric images and equivariance

We start by defining the groups of interest acting on geometric images. First is the group of discrete translations on the d-torus pixel grid, denoted $T_{N,d} \cong (\mathbb{Z}/N\mathbb{Z})^d$. If A is a $k_{(p)}$-tensor image and $\tau \in T_{N,d}$, then the action $L_\tau A$ produces the $k_{(p)}$-tensor image $(L_\tau A)(\bar{\imath}) = A(\bar{\imath}-\tau)$, where $\bar{\imath}$ is a pixel index and $\bar{\imath}-\tau$ is the translation of $\bar{\imath}$ by τ on the d-torus pixel grid.

In addition to translation symmetries, we want to consider other natural symmetries occurring in the application domains where vectors and tensors arise. Ideally, we would like to apply continuous rotations to the images, but the discretized nature of images makes this challenging. To obtain exact results on images, we focus on discrete rotations. For two-dimensional images, this is the familiar dihedral group D_4 of rotations of 90 degrees and reflections, and in the general-D case, it is the hyperoctahedral group B_d, the Euclidean symmetries of the d-dimensional hypercube. The notation B_d is standard nomenclature coming from the classification theorem for finite irreducible reflection groups [51]. Because the groups B_d are subgroups of $O(d)$, all determinants of the matrix representations of the group elements are either +1 or −1, and the matrix representation $M(g^{-1})$ of the inverse g^{-1} of group element g is the transpose of the matrix representation $M(g)$ of group element g.

Definition 11 (action of Bd on k(p)-tensors)**.** Given a $k_{(p)}$-tensor b, the action of $g \in B_d$ on b, denoted $g \cdot b$, is the restriction of the action in definition 3 to B_d, which is a subgroup of $O(d)$.

Remark. *Although we now consider B_d acting on $k_{(p)}$-tensors, we continue to use the $O(d)$-equivariant tensor operations of §3a to preserve full tensor coordinate freedom.*

Definition 12 (action of Bd on k(p)-tensor images)**.** Given $A \in \mathcal{A}_{N,d,k,p}$ on the d-torus and a group element $g \in B_d$, the action $g \cdot A$ produces a $k_{(p)}$-tensor image on the d-torus such that

$$(g \cdot A)(\bar{\imath}) = g \cdot A(g^{-1} \cdot \bar{\imath}). \tag{4.1}$$

Since $\bar{\imath}$ is a $1_{(+)}$-tensor, the action $g^{-1} \cdot \bar{\imath}$ is performed by centring $\bar{\imath}$, applying the operator, then un-centring the pixel index

$$g^{-1} \cdot \bar{\imath} = \left(M(g^{-1})(\bar{\imath} - \bar{m})\right) + \bar{m},$$

where $\bar{m}$ is the d-length $1_{(+)}$-tensor $\left[\frac{N-1}{2}, \ldots, \frac{N-1}{2}\right]^{\mathsf{T}}$. If the pixel index is already centred, such as $\bar{a} \in [-m, m]^d$, then we skip the centring and un-centring.

It might be a bit surprising that the group element g^{-1} appears in the definition of the action of the group on images. One way to think about it is that the pixels in the transformed image are 'looked up' or 'read out' from the pixels in the original untransformed image. The pixel locations in the original image are found by going back or inverting the transformation.

Definition 13 (the group $G_{N,d}$, and its action on k(p)-tensor images)**.** $G_{N,d}$ is the group generated by the elements of B_d and the discrete translations on the N^d-pixel lattice on the d-torus.

Remark. *We view the d-torus as the quotient of the d-hypercube obtained by identifying opposite faces. The torus obtains the structure of a flat (i.e. zero curvature) Riemannian manifold this way. Because the symmetries B_d of the hypercube preserve pairs of opposite faces, they act in a well-defined way on this quotient, so we can also view B_d as a group of isometries of the torus. We choose the common fixed point of the elements of B_d as the origin for the sake of identifying the N^d pixel lattice with the group $T_{N,d} \cong (\mathbb{Z}/N\mathbb{Z})^d$ of discrete translations of this lattice; then the action of B_d on the torus induces an action of B_d on $T_{N,d}$ by automorphisms. The group $G_{N,d}$ is the semidirect product $T_{N,d} \rtimes B_d$ with respect to this action. Thus, there is a canonical group homomorphism $G_{N,d} \to B_d$ with kernel $T_{N,d}$. In concrete terms, every element of $G_{N,d}$ can be written in the form $\tau \circ b$, where $b \in B_d$ and $\tau \in T_{N,d}$. Then the canonical map $G_{N,d} \to B_d$ sends $\tau \circ b$ to b.*

With our groups specified, we can define equivariance and invariance before proceeding to several theoretical results.

Definition 14 (equivariance of a geometric image function)**.** Let G be one of $T_{N,d}, B_d$ or $G_{N,d}$. Given a function on geometric images $f : \mathcal{A}_{N,d,k,p} \to \mathcal{A}_{N,d,k',p'}$, we say that f is *G-equivariant* if for all $g \in G$ and $A \in \mathcal{A}_{N,d,k,p}$ we have

$$f(g \cdot A) = g \cdot f(A). \tag{4.2}$$

Likewise, f is *invariant* to G if

$$f(g \cdot A) = f(A). \tag{4.3}$$

We also say a geometric image is G-isotropic if $g \cdot A = A$ for all $g \in G$.

The fundamental property of convolution is that it is translation equivariant and that every translation equivariant linear function can be expressed as a convolution with a fixed filter, as long as the filter can be set to be as large as the image [15]. The same property holds for geometric images.

Proposition 1. *A function $f : \mathcal{A}_{N,d,k,p} \to \mathcal{A}_{N,d,k',p'}$ is a translation equivariant linear function if and only if it can be written as $\iota_k(A * C)$ for some geometric filter $C \in \mathcal{A}_{M,d,k+k',p\,p'}$. When N is odd, $M = N$; otherwise, $M = N + 1$.*

See appendix A.1 for the proof. We can also build convolution functions that are equivariant to $G_{N,d}$. The following theorem generalizes the Cohen & Welling paper [16] for geometric convolutions.

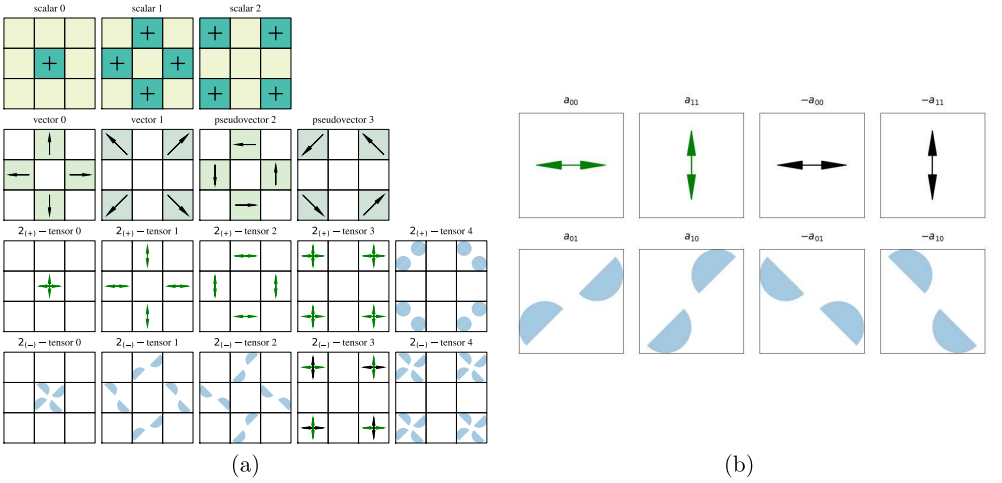

Figure 2. (a) All the filters for $d = 2, M = 3, k \in \{0, 1, 2\}$. Where there is no symbol in the box the value is zero. There are no B_d-isotropic pseudoscalar filters at $d = 2, M = 3$. Note that the vector filters look like pure divergence and the pseudovector filters look like pure curl. (b) Each signed component in the $2_{(p)}$-tensor has a particular icon, with the positive diagonal elements represented by the green double arrows, the negative diagonal elements represented by the black double arrows and the off-diagonal elements represented by the petals. Each element rotates in an obvious way, and $2_{(+)}$-tensors reflect in an obvious way as well. However, reflections on negative-parity diagonal elements flip the sign (colour) of the double arrows and have no effect on the petals other than changing their pixel location.

Theorem 1. *A function $f : \mathcal{A}_{N,d,k,p} \to \mathcal{A}_{N,d,k',p'}$ is linear and $G_{N,d}$-equivariant if and only if it can be written as $\iota_k(A * C)$ for some B_d-isotropic $C \in \mathcal{A}_{M,d,k+k',pp'}$, where $M = N$ if N is odd and $M = N + 1$ otherwise.*

The proof of this theorem is given in appendix A. Theorem 1 provides the explicit requirements for linear layers in our equivariant GeometricImageNet. All we need are the B_d-isotropic $(k + k')_{(pp')}$-tensor filters, which are straightforward to find using group averaging.

5. GeometricImageNet architectures

Per theorem 1, we construct linear $G_{N,d}$-equivariant layers using B_d-isotropic filters. A complete basis of B_d-isotropic $(k + k')_{(pp')}$-tensor filters can be found by group averaging. First, we get the standard basis of $\mathbb{R}^{M^d \times d^{(k+k')}}$ and reshape them into filters C_i with side length M and assigned parity pp'. Next, we apply the group averaging

$$\widetilde{C}_i = \frac{1}{|B_d|} \sum_{g \in B_d} g \cdot C_i, \tag{5.1}$$

where $|B_d|$ is the number of group elements. This will likely result in a linearly dependent set of filters, so we perform singular value decomposition to reduce to a single set of unique filters. The filters are then normalized so that non-zero tensors have unit norm, and the $k = 1$ filters are also re-oriented such that non-zero divergences were set to be positive, and non-zero curls were set to be counterclockwise. See figure 2 for the B_d-isotropic convolutional filters in $d = 2$ dimensions for filters of side length $M = 3$. Next, we use these B_d-isotropic filters to construct linear $G_{N,d}$-equivariant layers.

The linear layers take an input collection of geometric images $\{(k_z, p_z)\}_{z=1}^{W_{\text{in}}}$ with c_z channels and the desired output tensor orders and parities $\{(k_s, p_s)\}_{s=1}^{W_{\text{out}}}$ with c_s channels and compute all the

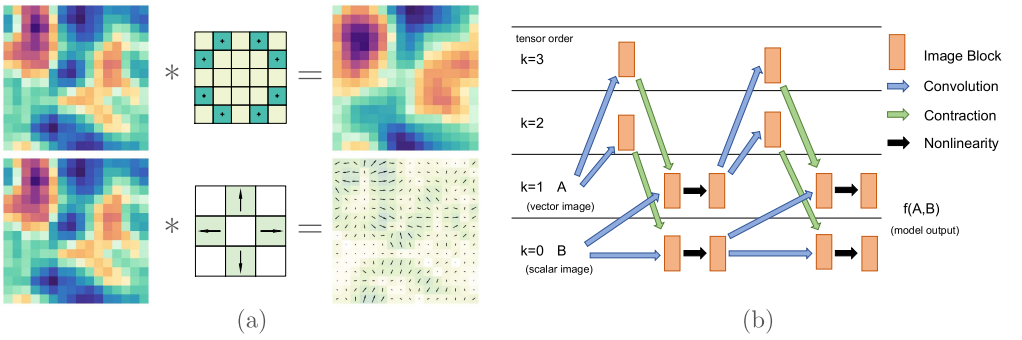

Figure 3. (a) Convolution of a scalar image with a scalar and vector filter. (b) Example architecture taking a vector image and scalar image as input and output. Linear layers are shown by the blue convolution arrows followed by green contraction arrows. The black arrows represent nonlinearities. The orange blocks represent multiple channels of images at that tensor order.

convolutions[1] and contractions to map between those two sets. Following theorem 1, there are $\ell = 1, \dots, c_s$ functions $\sum_{z=1}^{W_{\text{in}}} \sum_{i=1}^{c_z} \iota_{k_z}(A_{i,z} * C_{\ell,i,z})$ for each desired output tensor order and parity. Per the theorem, these convolution filters $C_{\ell,i,z}$ must be B_d-isotropic to guarantee that this layer is $G_{N,d}$-equivariant. Each B_d-isotropic filter is a parameterized linear combination of the B_d-isotropic basis we found by group averaging. However, using filters as large as the input image is impractical in most cases, so we use deeper networks of 3×3 or 5×5 filters, as is commonly done in CNNs [52].

Nonlinear layers present a challenge because the typical pointwise nonlinear functions such as Rectified Linear Unit (ReLU) or tanh break equivariance when applied to the individual components of a tensor. Properly building $O(d)$-equivariant nonlinear functions is a challenging and active area of research; for a larger exploration, see [53] and references therein. For this model, we extend the Vector Neuron nonlinearity [54] for any tensor order and parity. Let A_i for $i = 1, \dots, c_z$ be the input $k_{z(p_z)}$-tensor image channels as above (we drop the z from the A_i notation for simplicity), let $\alpha_i, \beta_i \in \mathbb{R}$ be learned scalar parameters and $Q = \sum_{i=1}^{c_z} \alpha_i A_i, K = \sum_{i=1}^{c_z} \beta_i A_i$. Then the nonlinearity $\sigma : \left(\mathcal{A}_{N,d,k_z,p_z}\right)^{c_z} \to \mathcal{A}_{N,d,k_z,p_z}$ is defined

$$\sigma\left((A_i)_{i=1}^{c_z}\right) = \begin{cases} Q & \text{if } \iota_k(Q \otimes K) \geq 0 \\ Q - \iota_k\left(Q \otimes \frac{K}{\|K\|_2}\right)\frac{K}{\|K\|_2} & \text{otherwise} \end{cases}, \tag{5.2}$$

where $\|\cdot\|_2$ is the tensor norm (3.9). To get c_s output channels, we can repeat this function c_s times with different learned parameters α_i, β_i. We show that this extension is $O(d)$-equivariant in appendix A.2. See figure 3b for an example of a typical architecture interlacing linear and nonlinear layers.

The final layer types we will use in our model are LayerNorm [55] and max pool. We use the original LayerNorm for scalar images, but for vector images, we follow the strategy of vector whitening used in [48], based on a similar strategy developed for neural networks with complex values [56]. This method has not yet been extended to higher order tensors. Let $(A_i)_{i=1}^{c_z}$ be a set of $1_{(p)}$-tensor images. Let $\bar{A}_i(\bar{\imath}) = A_i(\bar{\imath}) - \frac{1}{c_z N^d} \sum_{i=1}^{c_z} \sum_{\bar{\jmath} \in [N]^d} A_i(\bar{\jmath})$ for each pixel $\bar{\imath}$ be the mean-centred $1_{(p)}$-tensor image. Then the covariance is a $2_{(+)}$-tensor given by

$$\Sigma = \frac{1}{c_z N^d} \sum_{i=1}^{c_z} \sum_{\bar{\imath} \in [N]^d} (\bar{A}_i \otimes \bar{A}_i)(\bar{\imath}). \tag{5.3}$$

[1]The geometric convolution package is implemented in JAX, which in turn uses TensorFlow XLA under the hood. This means that convolution is actually cross-correlation, in line with how the term is used in ML papers. For our purposes, this results in at most a coordinate transformation in the filters.

We calculate $\Sigma^{-\frac{1}{2}}$ by performing an eigenvalue decomposition $\Sigma = U\Lambda U^\top$, where Λ is a diagonal matrix with the eigenvalues along the diagonal. We take the inverse of each eigenvalue and then its square root, then multiply $U\Lambda^{-\frac{1}{2}}U^\top$ to get $\Sigma^{-\frac{1}{2}}$. Finally, we scale the vectors by $\Sigma^{-\frac{1}{2}}$

$$[B_i(\bar{\imath})]_\ell = \left[\bar{A}_i(\bar{\imath})\right]_j \left[\Sigma^{-\frac{1}{2}}\right]_{j,\ell}, \tag{5.4}$$

and output B_i for $i = 1$ to c_z. Max pooling layers follow exactly from definition 10 for each channel of each input image. We prove that these layers are $G_{N,d}$-equivariant in appendices A.3 and A.4, respectively.

6. Numerical experiments

We will conduct numerical experiments on two-dimensional compressible Navier–Stokes simulation data from the excellent PDEBench dataset [57]. These data consist of velocity (vector) fields, density (scalar) fields and pressure (scalar) fields with periodic boundary conditions discretized into 128×128 images on the torus. The simulations are saved at 21 time points, which are a subset of the integrator timesteps. We use 128 simulation trajectories with random initial conditions as training data and another 128 trajectories as test data. The model takes the state of system at times $t = 0, 1, 2, 3$ for inferring the trajectories. We use data generated with two distinct sets of parameters: Mach number $M = 0.1$, shear viscosity $\eta = 0.01$ and bulk viscosity $\zeta = 0.01$ and $M = 1.0, \eta = 0.1, \zeta = 0.1$. The two sets of parameters are used to train entirely different models and tested separately.

The model task is to take as input the velocity, density and pressure fields at a certain time point and predict what those fields will be at the next time point. We adopt a common strategy in the surrogate model literature [20,26,48] of providing four previous time points as input to help capture temporal derivatives of the state [57]. Thus, we can turn the 128 training trajectories into 2176 training data points because each trajectory has 17 overlapping sections of four input steps and one output step. We train a Dilated ResNet [17], a ResNet [58] and a UNet [59] with and without LayerNorm [55] and large and small equivariant versions of each of those models. We train with the sum of the mean squared error loss of each field of a single step, but at test time we are also interested in the performance of autoregressively rolling out the model over 15 time steps. The baseline models and training set-up generally follow those described in [20], and additional data, model and training details are in appendix C.

The numerical results are given in table 1. In all cases of the 1-step loss and almost all cases of the 15-step rollout loss, the equivariant models outperform the non-equivariant versions. One exception is the M1.0 dataset where the equivariant ResNets have better 1-step errors, but worse rollout errors. These models appear to be overfitting in a manner that is hurting rollout stability. In figure 4, we can see with more granularity the test performance for each rollout step. In the most drastic example, the rollout error for the Dilated ResNet explodes, while the equivariant Dilated ResNet is stable and accurate over all 15 steps. In [17], the authors combat this issue by adding a small amount of Gaussian noise during training; we instead achieve stability in a physically motivated way by enforcing $O(d)$-equivariance. The equivariance also helps with parameter efficiency; the small equivariant models have a number of channels so that scalar plus vector components are comparable to the number of baseline model channels. The large equivariant models have a comparable number of parameters to baseline models (table 2). The large equivariant models generally do better than the smaller ones, but the smaller ones still outperform the non-equivariant models despite having 80% fewer parameters. Code to reproduce all these experiments and build your own GI-Net is available at https://github.com/WilsonGregory/GeometricConvolutions. The code is built in Python using JAX [60].

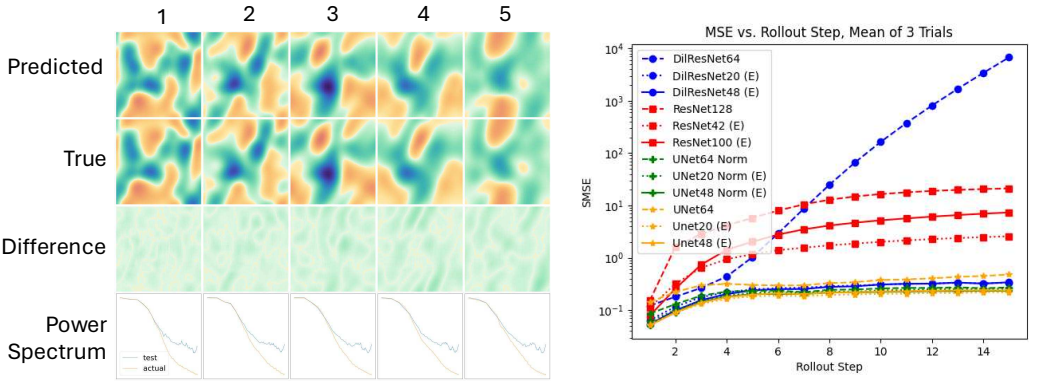

Figure 4. (a) Five steps of M0.1 rollout using the UNet48 (E). The x-component of the velocity is plotted. The power spectrum is the Fourier transform of the two-point correlation function. (b) Comparison of test performance over a 15-step rollout on the M0.1 dataset. The SMSE is shown for *each* step, rather than a cumulative loss.

Table 1. Loss values for each model, averaged over three trials. All losses are the sum of the mean squared error losses over the channels: density, pressure and velocity. The rollout loss is the sum of the error over 15 steps. The s.d. $\pm 0.xxx$ is provided if it is at least 0.001. The equivariant models are indicated by '(E)'. The number in the model name refers to the number of channels per layer, which for equivariant models refers to the number of scalar and vector channels each. The bold values indicate the best error per group of models.

model	M0.1 1-step	M0.1 rollout	M1.0 1-step	M1.0 rollout
DilResNet64	0.040	13318.773 ± 18824.855	0.005	9.574 ± 9.608
DilResNet20 (E)	0.021	3.882 ± 0.245	0.001	0.249 ± 0.012
DilResNet48 (E)	**0.018**	**3.770 ± 0.090**	**0.001**	**0.153 ± 0.023**
ResNet128	0.039	175.736 ± 17.846	0.009	**0.835 ± 0.097**
ResNet42 (E)	0.036 ± 0.004	**23.666 ± 5.507**	0.005	2.513 ± 0.450
ResNet100 (E)	**0.024 ± 0.001**	57.508 ± 9.157	**0.003**	2.943 ± 0.992
UNet64 Norm	0.027	3.414 ± 0.217	0.009 ± 0.001	1.067 ± 0.190
UNet20 Norm (E)	0.020	3.105 ± 0.130	0.001	0.140 ± 0.011
UNet48 Norm (E)	**0.017**	**2.898 ± 0.046**	**0.001**	**0.121 ± 0.007**
UNet64	0.047 ± 0.001	5.086 ± 0.105	0.012 ± 0.002	2.074 ± 0.067
Unet20 (E)	0.017	**2.672 ± 0.071**	0.001	0.172 ± 0.008
Unet48 (E)	**0.017**	2.919 ± 0.094	**0.001**	**0.109 ± 0.005**

7. Discussion

This paper presents geometric convolutions that can easily adapt any CNN architecture to be equivariant for images of vectors or tensors. This makes the model ideal for tackling many problems in the natural sciences in a principled way. We see in two-dimensional compressible Navier–Stokes simulations that we achieve better accuracy and more stable rollouts than non-equivariant baseline models.

One limitation of this work is that we use discrete symmetries instead of continuous symmetries. We expect invariance and equivariance with respect to rotations other than 90 degrees to appear in nature, but the images that we work with are always going to be d-cube grids of points.

Table 2. Comparison of various models. For equivariant models, the number of channels is for vector channels and scalar channels each. The number of channels of each larger equivariant model was chosen so that the equivariant and non-equivariant models have roughly the same number of parameters. The number of channels for each smaller equivariant model was chosen so that the total number of components across scalars and vectors is comparable to the number of channels in the baseline models.

model	params	CNN channels	norm	bias	learning rate
DilResNet64	1 043 651	64	—	Yes	2×10^{-3}
DilResNet20 (E)	171 743	20	—	Mean	1×10^{-3}
DilResNet48 (E)	979 347	48	—	Mean	1×10^{-3}
ResNet128	2 401 155	128	LayerNorm	Yes	1×10^{-3}
ResNet42 (E)	455 913	42	LayerNorm	Mean	7×10^{-4}
ResNet100 (E)	2 558 703	100	LayerNorm	Mean	7×10^{-4}
UNet64 Norm	31 053 251	64	LayerNorm	Yes	8×10^{-4}
UNet20 Norm (E)	4 704 383	20	—	Mean	6×10^{-4}
UNet48 Norm (E)	27 077 139	48	—	Mean	4×10^{-4}
UNet64	31 046 400	64	BatchNorm	No	8×10^{-4}
UNet20 (E)	4 700 100	20	—	No	7×10^{-4}
UNet48 (E)	27 066 864	48	—	No	3×10^{-4}

Thus, we use the group $G_{N,d}$ to avoid interpolating rotated images and working with approximate equivariances. This simplifies the mathematical results, and we see empirically that we still have the benefits of rotational equivariance. However, there are other possible image representations that might create continuous concepts of images. For example, if the data are on the surface of a sphere, it could be represented with tensor spherical harmonics, and it could be subject to transformations by a continuous rotation group.

Another limitation of this work is that we do not compare our method with existing state-of-the-art numerical integrator methods. Surrogate ML models for fluid dynamics simulations have generally suffered from comparisons with weak baselines that overstate the accuracy or efficiency of the surrogate model [61]. In this work, we only claim to improve upon existing vanilla CNN models, and we leave further comparisons to future work.

There are many other future directions that could be explored. Further research is required to understand how and why the equivariance helps. One interesting observation of figure 4a is that the power spectrum for the equivariant model output is still quite different from the ground truth at higher frequencies. It may be that equivariance is advantageous at certain scales and not at others.

Ethics. This work did not require ethical approval from a human subject or animal welfare committee.

Data accessibility. Data available in [62].

Declaration of AI use. We have not used AI-assisted technologies in creating this article.

Authors' contributions. W.G.: conceptualization, data curation, formal analysis, investigation, methodology, project administration, software, validation, visualization, writing—original draft, writing—review and editing; D.H.: conceptualization, formal analysis, investigation, methodology, project administration, software, supervision, writing—original draft, writing—review and editing; B.B.-S.: formal analysis, supervision; M.T.A.: formal analysis; K.W.: data curation, formal analysis, software; S.V.: conceptualization, formal analysis, funding acquisition, investigation, methodology, project administration, supervision, writing—review and editing.

All authors gave final approval for publication and agreed to be held accountable for the work performed therein.

Conflict of interest declaration. We declare we have no competing interests.

Funding. W.G. was supported by ONR N00014-22-1-2126 and an Amazon AI2AI Faculty Research Award. B.B-S. was supported by ONR N00014-22-1-2126 and NSF CAREER 2339682. M.T.A. was supported by H2020-MSCA-RISE-2017, project 777822, and from grant PID2019-105599GB-I00, Ministerio de Ciencia, Innovación y Universidades, Spain. K.W. Was supported by NSF CAREER 2339682. S.V. was partly supported by the NSF -Simons Research Collaboration on the Mathematical and Scientific Foundations of Deep Learning (MoDL; NSF DMS 2031985), NSF CISE 2212457, ONRN00014-22-1-2126, NSF CAREER 2339682 and NSF BSF 2430292.

Acknowledgements. It is a pleasure to thank Roger Blandford (Stanford), Drummond Fielding (Flatiron), Leslie Greengard (Flatiron), Ningyuan (Teresa) Huang (JHU), Kate Storey-Fisher (NYU) and the Astronomical Data Group at the Flatiron Institute for valuable discussions and input. This project made use of Python 3 [63], numpy [64], matplotlib [65] and cmastro [66]. All the code used for making the data and figures in this paper is available at https://github.com/WilsonGregory/GeometricConvolutions.

Appendix A. Proofs

A.1. Proof of theorem 1

Before proving theorem 1, we state and prove a number of helpful properties, propositions and lemmas.

Properties. *Let $A, B \in \mathcal{A}_{N,d,k,p}$, let $C, S \in \mathcal{A}_{M,d,k',p'}$, let $D, Q \in \mathcal{A}_{N,d,2k+k',p}$, let $\tau \in (\mathbb{Z}/N\mathbb{Z})^d$ be a translation on the d-torus, let $\alpha, \beta \in \mathbb{R}$ and let $g \in G_{N,d}$. Then the following properties hold.*

(1) Convolutions are translation equivariant

$$(L_\tau A) * C = L_\tau(A * C). \tag{A 1}$$

(2) Convolutions are linear in the geometric image

$$(\alpha A + \beta B) * C = \alpha(A * C) + \beta(B * C). \tag{A 2}$$

Convolutions are also linear in the filters

$$A * (\alpha C + \beta S) = \alpha(A * C) + \beta(A * S). \tag{A 3}$$

(3) The k-contraction is $G_{N,d}$-equivariant

$$g \cdot \iota_k(D) = \iota_k\big(g \cdot D\big). \tag{A 4}$$

(4) The k-contraction is a linear function

$$\iota_k(\alpha D + \beta Q) = \alpha\, \iota_k(D) + \beta\, \iota_k(Q). \tag{A 5}$$

Proof. First, we will prove (A 1). Let A, C and τ be as above, and let $\bar{\imath}$ be a pixel index of $L_\tau A * C$. Then

$$\begin{aligned}
(L_\tau A * C)(\bar{\imath}) &= \sum_{\bar{a} \in [-m,m]^d} (L_\tau A)(\bar{\imath} - \bar{a}) \otimes C(\bar{a} + \bar{m}) \\
&= \sum_{\bar{a} \in [-m,m]^d} A(\bar{\imath} - \bar{a} - \tau) \otimes C(\bar{a} + \bar{m}) \\
&= \sum_{\bar{a} \in [-m,m]^d} A((\bar{\imath} - \tau) - \bar{a}) \otimes C(\bar{a} + \bar{m}) \\
&= (A * C)(\bar{\imath} - \tau) \\
&= L_\tau(A * C)(\bar{\imath}).
\end{aligned}$$

Now we will prove (A 2). Let A, B, C, α and β be as above, and let $\bar{\imath}$ be a pixel index of $(\alpha A + \beta B) * C$. Then

$$\begin{aligned}((\alpha A + \beta B) * C)(\bar{\imath}) &= \sum_{\bar{a}\in[-m,m]^d} (\alpha A + \beta B)(\bar{\imath} - \bar{a}) \otimes C(\bar{a} + \bar{m}) \\ &= \sum_{\bar{a}\in[-m,m]^d} (\alpha A(\bar{\imath} - \bar{a}) + \beta B(\bar{\imath} - \bar{a})) \otimes C(\bar{a} + \bar{m}) \\ &= \sum_{\bar{a}\in[-m,m]^d} \alpha A(\bar{\imath} - \bar{a}) \otimes C(\bar{a} + \bar{m}) + \beta B(\bar{\imath} - \bar{a}) \otimes C(\bar{a} + \bar{m}) \\ &= \alpha \sum_{\bar{a}\in[-m,m]^d} A(\bar{\imath} - \bar{a}) \otimes C(\bar{a} + \bar{m}) + \beta \sum_{\bar{a}\in[-m,m]^d} B(\bar{\imath} - \bar{a}) \otimes C(\bar{a} + \bar{m}) \\ &= \alpha(A * C)(\bar{\imath}) + \beta(B * C)(\bar{\imath}).\end{aligned}$$

Now we will prove (A 3). Let A, C, S, α and β be as above, and let $\bar{\imath}$ be a pixel index. Then

$$\begin{aligned}(A * (\alpha C + \beta S))(\bar{\imath}) &= \sum_{\bar{a}\in[-m,m]^d} A(\bar{\imath} - \bar{a}) \otimes (\alpha C + \beta S)(\bar{a} + \bar{m}) \\ &= \sum_{\bar{a}\in[-m,m]^d} A(\bar{\imath} - \bar{a}) \otimes \alpha C(\bar{a} + \bar{m}) + A(\bar{\imath} - \bar{a}) \otimes \beta S(\bar{a} + \bar{m}) \\ &= \alpha \sum_{\bar{a}\in[-m,m]^d} A(\bar{\imath} - \bar{a}) \otimes C(\bar{a} + \bar{m}) + \beta \sum_{\bar{a}\in[-m,m]^d} A(\bar{\imath} - \bar{a}) \otimes S(\bar{a} + \bar{m}) \\ &= \alpha(A * C)(\bar{\imath}) + \beta(A * S)(\bar{\imath}).\end{aligned}$$

Next, we will prove (A 4). Let D be defined as above, and let $\bar{\imath}$ be a pixel of D. First, we will show that contractions are equivariant to translations. Let $\tau \in (\mathbb{Z}/N\mathbb{Z})^d$. Then

$$\iota_k(L_\tau D)(\bar{\imath}) = \iota_k((L_\tau D)(\bar{\imath})) = \iota_k(D(\bar{\imath} - \tau)) = \iota_k(D)(\bar{\imath} - \tau) = L_\tau \iota_k(D)(\bar{\imath}). \tag{A 6}$$

Thus contractions are equivariant to translations. Now we will show that contractions are equivariant to B_d. Let $g \in B_d$, and denote $D(g^{-1} \cdot \bar{\imath}) = a$. Then by equation (3.5), we have

$$\begin{aligned}\iota_k(g \cdot D)(\bar{\imath}) &= \iota_k((g \cdot D)(\bar{\imath})) \\ &= \iota_k\left(g \cdot D\left(g^{-1} \cdot \bar{\imath}\right)\right) \\ &= \iota_k(g \cdot a) \\ &= [g \cdot a]_{i_1,\dots,i_k,i_1,\dots,i_k,i_{2k+1},\dots,i_{2k+k'}} \\ &= [a]_{j_1,\dots,j_{2k+k'}} \prod_{q\in[k]} [M(g)]_{i_q,j_q} [M(g)]_{i_q,j_{q+k}} \prod_{q\in\left[2k+1,2k+k'\right]} [M(g)]_{i_q,j_q} \\ &\overset{(*)}{=} [a]_{j_1,\dots,j_{2k+k'}} \prod_{q\in[k]} [\delta]_{j_q,j_{q+k}} \prod_{q\in\left[2k+1,2k+k'\right]} [M(g)]_{i_q,j_q} \\ &= [a]^{q\in\left[2k+1,2k+k'\right]}_{j_1,\dots,j_k,j_1,\dots,j_k,j_{2k+1},\dots,j_{2k+k'}} \\ &= [\iota_k(a)]_{j_{2k+1},\dots,j_{2k+k'}} \prod_{q\in\left[2k+1,2k+k'\right]} [M(g)]_{i_q,j_q} \\ &= g \cdot \iota_k(a)\big]_{i_q,j_q} \\ &= g \cdot \iota_k\left(D\left(g^{-1} \cdot \bar{\imath}\right)\right) \\ &= \left(g \cdot \iota_k(D)\right)(\bar{\imath}).\end{aligned}$$

Note that $(*)$ happens because $[M(g)]_{i,j}[M(g)]_{i,k} = M(g)^\top M(g) = \delta$ because they are orthogonal matrices, and the next step follows from Kronecker delta identities. Therefore, since contractions are equivariant to the generators of $G_{N,d}$, it is equivariant to the group.

Finally, we will prove (A 5). Let D, Q, α and β be defined as above, let $\bar{\imath}$ be a pixel index of $(\alpha D + \beta Q)$ and let $a, b \in \mathcal{T}_{d,k,p}$ be the tensors of D and Q at that pixel index. Then

$$\begin{aligned}
[\iota_k(\alpha D + \beta Q)(\bar{\imath})]_{i_{2k+1},\ldots,i_{2k+k'}} &= [\iota_k(\alpha D(\bar{\imath}) + \beta Q(\bar{\imath}))]_{i_{2k+1},\ldots,i_{2k+k'}} \\
&= [\iota_k(\alpha a + \beta b)]_{i_{2k+1},\ldots,i_{2k+k'}} \\
&= [\alpha a + \beta b]_{i_1,\ldots,i_k,i_1,\ldots,i_k,i_{2k+1},\ldots,i_{2k+k'}} \\
&= \alpha[a]_{i_1,\ldots,i_k,i_1,\ldots,i_k,i_{2k+1},\ldots,i_{2k+k'}} + \beta[b]_{i_1,\ldots,i_k,i_1,\ldots,i_k,i_{2k+1},\ldots,i_{2k+k'}} \\
&= \alpha[\iota_k(a)]_{i_{2k+1},\ldots,i_{2k+k'}} + \beta[\iota_k(b)]_{i_{2k+1},\ldots,i_{2k+k'}} \\
&= \alpha[\iota_k(D(\bar{\imath}))]_{i_{2k+1},\ldots,i_{2k+k'}} + \beta[\iota_k(Q(\bar{\imath}))]_{i_{2k+1},\ldots,i_{2k+k'}} \\
&= [(\alpha\iota_k(D) + \beta\iota_k(Q))(\bar{\imath})]_{i_{2k+1},\ldots,i_{2k+k'}}.
\end{aligned}$$

Thus we have shown (A 6). ■

Lemma 1. *Given $A \in \mathcal{A}_{N,d,k,p}$ a geometric image and $C \in \mathcal{A}_{M,d,k',p'}$ a geometric filter where $M = N + 1$, there exists $C' \in \mathcal{A}_{M,d,k',p'}$ such that $A * C' = A * C$ and $C'(\bar{\imath})$ is the zero $k'_{(p')}$-tensor, for $\bar{\imath} \in [0, N]^d \setminus [0, N-1]^d$. That is, C' is totally defined by N^d pixels, and every pixel with an N in the index is equal to the zero $k'_{(p')}$-tensor.*

Proof. Let A and C be defined as above. Thus

$$N = M - 1 = 2m + 1 - 1 = 2m. \tag{A 7}$$

Consider the convolution definition (3.9) where we have $A(\bar{\imath} - \bar{a})$ where $\bar{\imath} \in [0, N-1]^d$ and $\bar{a} \in [-m, m]^d$. Since A is on the d-torus, then whenever the ℓ^{th} index of $\bar{a} = -m$ we have

$$\begin{aligned}
(\bar{\imath}_\ell - \bar{a}_\ell) \mod N &= (\bar{\imath}_\ell - (-m)) \mod N \\
&= (\bar{\imath}_\ell + m) \mod 2m \\
&= (\bar{\imath}_\ell + m - 2m) \mod 2m \\
&= (\bar{\imath}_\ell - m) \mod N.
\end{aligned}$$

Thus, any time there is an index $\bar{a}$ with a value $\pm m$, we have an equivalence class under the torus with all other indices with flipped sign of the m in any combination. If $\{\bar{a}\}$ is this equivalence class, we may group these terms in the convolution sum

$$\sum_{\bar{a}' \in \{\bar{a}\}} A(\bar{\imath} - \bar{a}') \otimes C(\bar{a}' + \bar{m}) = \sum_{\bar{a}' \in \{\bar{a}\}} A(\bar{\imath} - \bar{a}) \otimes C(\bar{a}' + \bar{m}) = A(\bar{\imath} - \bar{a}) \otimes \left(\sum_{\bar{a}' \in \{\bar{a}\}} C(\bar{a}' + \bar{m}) \right).$$

Thus, we may pick a single pixel of the convolutional filter C, set it equal to $\sum_{\bar{a}' \in \{\bar{a}\}} C(\bar{a}' + \bar{m})$ and set all other pixels of the equivalence class to the zero $k'_{(p')}$-tensor without changing the convolution. We choose the non-zero pixel to be the one whose index has all $-m$ instead of m. Thus, we can define the filter C by N^d pixels rather than $(N+1)^d$ pixels, and we have our result. ■

Lemma 2. *Let there be a space of geometric images $\mathcal{A}_{N,d,k,p}$, and let $C_1, C_2 \in \mathcal{A}_{M,d,k+k',p\,p'}$ with $M = 2m + 1$ for positive integer m. Then, $\iota_k(A * C_1) = \iota_k(A * C_2)$ for all $A \in \mathcal{A}_{N,d,k,p}$ if and only if $C_1 = C_2$.*

Here is a quick proof sketch of the forward direction. We assume for the sake of contradiction that C_1 and C_2 are different so they must have at least one differing component. Then we use the fact that $\iota_k(A * C_1) = \iota_k(A * C_2)$ holds for all possible inputs to define an input A that isolates that component to get a contradiction.

Proof. Let C_1, C_2 be defined as above. The reverse direction is immediate, so we focus our attention on the forward direction. Suppose $\iota_k(A * C_1) = \iota_k(A * C_2)$ for all $A \in \mathcal{A}_{N,d,k,p}$. Assume for the sake of contradiction that $C_1 \neq C_2$, so $C_1 - C_2 \neq \vec{0}$, where $\vec{0}$ is the zero filter. Thus, there must be at least one component of one pixel that is non-zero. Suppose this is at pixel index $\bar{b} + \bar{m}$ and $(C_1 - C_2)(\bar{b} + \bar{m}) = c$. Suppose the non-zero component is at index $j_1, \ldots, j_{k+k'}$. Let a be a $k_{(p)}$-tensor where $[a]_{i_1,\ldots,i_k}$ is non-zero and all other indices are 0. Now suppose $A \in \mathcal{A}_{N,d,k,p}$ such that for pixel index $\bar{\imath}$ of A, $A(\bar{\imath} - \bar{b}) = a$ and all other pixels are the zero tensor. Thus

$$\begin{aligned}
\vec{0} &= (\iota_k(A * C_1) - \iota_k(A * C_2))(\bar{\imath}) \\
&\overset{A\,3, A\,5}{=} \iota_k(A * (C_1 - C_2))(\bar{\imath}) \\
&= \iota_k((A * (C_1 - C_2))(\bar{\imath})) \\
&= \iota_k\left(\sum_{\bar{a} \in [-m,m]^d} A(\bar{\imath} - \bar{a}) \otimes (C_1 - C_2)(\bar{a} + \bar{m})\right) \\
&= \iota_k\left(A(\bar{\imath} - \bar{b}) \otimes (C_1 - C_2)(\bar{b} + \bar{m})\right) \\
&= \iota_k(a \otimes c).
\end{aligned}$$

Note that the penultimate step of removing the sum is because $A(\bar{\imath} - \bar{a}) = 0$ the zero tensor everywhere other than $A(\bar{\imath} - \bar{b})$. Therefore, since the only non-zero entry of a is at index $i_1, \ldots, i_k$, then at index $j_{k+1}, \ldots, j_{k+k'}$ of the resulting tensor we have

$$\vec{0} = \iota_k(a \otimes c) = [a]_{i_1 \ldots i_k}[c]_{j_1,\ldots,j_{k+k'}}.$$

Since $[a]_{i_1,\ldots,i_k}$ is non-zero and $[c]_{j_1,\ldots,j_{k+k'}}$ is non-zero, this index is non-zero. This is a contradiction, so we conclude that $C_1 = C_2$, which finishes the proof. ∎

Proposition (Restatement of 1)**.** *[A function $f : \mathcal{A}_{N,d,k,p} \to \mathcal{A}_{N,d,k',p'}$ is a translation equivariant linear function if and only if $f(A) = \iota_k(A * C)$ for some geometric filter $C \in \mathcal{A}_{M,d,k+k',pp'}$. When N is odd, $M = N$; otherwise, $M = N + 1$.*

Proof. Let $\mathcal{F} = \{f : \mathcal{A}_{N,d,k,p} \to \mathcal{A}_{N,d,k',p'}\}$, where each function f is linear and equivariant to translations. Let $\mathcal{G} = \{g : \mathcal{A}_{N,d,k,p} \to \mathcal{A}_{N,d,k',p'}\}$, where each g is defined as $g(A) = \iota_k(A * C)$ for some $C \in \mathcal{A}_{M,d,k+k',pp'}$. If N is odd, then $M = N$; otherwise, $M = N + 1$. It suffices to show that $\mathcal{F} = \mathcal{G}$.

First, we will show that $\mathcal{G} \subseteq \mathcal{F}$. Let $g \in \mathcal{G}$. By properties (A 2) and (A 5) both convolutions and contractions are linear. Additionally, by properties (A 1) and (A 4), convolutions and contractions are both equivariant to translations. Thus $g \in \mathcal{F}$, so $\mathcal{G} \subseteq \mathcal{F}$.

Now we will show that $\dim(\mathcal{F}) = \dim(\mathcal{G})$. Let $f \in \mathcal{F}$. By definition 3, $\mathcal{T}_{d,k,p} \cong \left(\mathbb{R}^d\right)^{\otimes k}$ equipped with the group action of $O(d)$. Then by definition 8, $\mathcal{A}_{N,d,k,p}$ is the space of functions $A : [N]^d \to \mathcal{T}_{d,k,p}$, where $[N]^d$ has the structure of the d-torus. Therefore, $\mathcal{A}_{N,d,k,p} \cong \left(\mathbb{R}^N\right)^{\otimes d} \times \left(\mathbb{R}^d\right)^{\otimes k}$ equipped with the group action of $G_{N,d}$. Thus, $f : \left(\mathbb{R}^N\right)^{\otimes d} \times \left(\mathbb{R}^d\right)^{\otimes k} \to \left(\mathbb{R}^N\right)^{\otimes d} \times \left(\mathbb{R}^d\right)^{\otimes k'}$. Since f is linear, the dimension of the space of functions $\mathcal{F}$ is $N^d d^{k'} N^d d^k = N^{2d} d^{k+k'}$. If this is unclear, consider the fact that the linearity of f means that it has an associated matrix F of that dimension. However, since each f is translation equivariant, the function to each of the N^d pixels in the output must be the same. Thus, we actually have that $\dim(\mathcal{F}) = \frac{N^{2d} d^{k+k'}}{N^d} = N^d d^{k+k'}$.

Now we look at $\dim(\mathcal{G})$. Each function $g \in \mathcal{G}$ is defined by the convolution filter $C \in \mathcal{A}_{M,d,k+k',pp'}$ and $\dim(\mathcal{A}_{M,d,k+k',pp'}) = \dim(\mathcal{A}_{N,d,k+k',pp'}) = N^d d^{k+k'}$, with the first equality following from lemma 1 in both the even and odd case. Clearly, $\dim(\mathcal{G})$ is upper-bounded by the dimension of the convolution filters, but does it have to be equal? In other words, is it possible that two linearly independent convolution filters result in linearly dependent functions g? We will now show that this is not possible.

Let $g_1, g_2 \in \mathcal{G}$ be defined by two linearly independent filters $C_1, C_2 \in \mathcal{A}_{M,d,k+k',p\,p'}$, and we would like to show that g_1 and g_2 are linearly independent as well. Suppose that there exists $\alpha, \beta \in \mathbb{R}$ such that $\alpha\, g_1(A) + \beta\, g_2(A) = \vec{0}$ for all $A \in \mathcal{A}_{N,d,k,p}$. It suffices to show that $\alpha = \beta = 0$. Thus

$$\begin{aligned}
\iota_k(A * \vec{0}) &= \iota_k\,(A * (0C_3)) \\
&\overset{\text{A 3,A 5}}{=} 0\iota_k\,(A * C_3) \\
&= \vec{0} \\
&= \alpha g_1(A) + \beta g_2(A) \\
&= \alpha \iota_k\,(A * C_1) + \beta \iota_k\,(A * C_2) \\
&\overset{\text{A 3,A 5}}{=} \iota_k\,(A * (\alpha C_1 + \beta C_2))\,.
\end{aligned}$$

Thus, by lemma 2, $\vec{0} = \alpha\, C_1 + \beta\, C_2$. Since C_1 and C_2 are linearly independent, this implies that $\alpha = \beta = 0$. Thus, g_1, g_2 must be linearly independent. Therefore, $\dim(\mathcal{G}) = N^d d^{k+k'}$, and since $\mathcal{G} \subseteq \mathcal{F}$, we have $\mathcal{F} = \mathcal{G}$. ■

Lemma 3. *Given $g \in B_d$, $A \in \mathcal{A}_{N,d,k,p}$ and $C \in \mathcal{A}_{M,d,k',p'}$, the action g distributes over the convolution of A with C*

$$g \cdot (A * C) = (g \cdot A) * (g \cdot C). \tag{A 8}$$

Proof. Let $A \in \mathcal{A}_{N,d,k,p}$ be a geometric image, let $C \in \mathcal{A}_{M,d,k',p'}$, let $g \in B_d$ and let $\bar{\imath}$ be any pixel index of A. By definition 12, we have

$$\begin{aligned}
(g \cdot (A * C))(\bar{\imath}) &= g \cdot \big((A * C)\big(g^{-1} \cdot \bar{\imath}\big)\big) \\
&= g \cdot \left(\sum_{\bar{a} \in [-m,m]^d} A\big(g^{-1} \cdot \bar{\imath} - \bar{a}\big) \otimes C(\bar{a} + \bar{m}) \right) \\
&= \sum_{\bar{a} \in [-m,m]^d} g \cdot \big(A\big(g^{-1} \cdot \bar{\imath} - \bar{a}\big) \otimes C(\bar{a} + \bar{m})\big) \\
&= \sum_{\bar{a} \in [-m,m]^d} g \cdot A\big(g^{-1} \cdot \bar{\imath} - \bar{a}\big) \otimes g \cdot C(\bar{a} + \bar{m}).
\end{aligned}$$

Now let $\bar{a}' = g \cdot \bar{a}$. Thus, $g^{-1} \cdot \bar{a}' = g^{-1} \cdot g \cdot \bar{a} = \bar{a}$. Then

$$\begin{aligned}
(g \cdot (A * C))(\bar{\imath}) &= \sum_{\bar{a} \in [-m,m]^d} g \cdot A\big(g^{-1} \cdot \bar{\imath} - \bar{a}\big) \otimes g \cdot C(\bar{a} + \bar{m}) \\
&= \sum_{g^{-1} \cdot \bar{a}' \in [-m,m]^d} g \cdot A\big(g^{-1} \cdot \bar{\imath} - g^{-1} \cdot \bar{a}'\big) \otimes g \cdot C\big(g^{-1} \cdot \bar{a}' + \bar{m}\big) \\
&= \sum_{g^{-1} \cdot \bar{a}' \in [-m,m]^d} g \cdot A\big(g^{-1} \cdot \bar{\imath} - g^{-1} \cdot \bar{a}'\big) \otimes g \cdot C\big(g^{-1} \cdot \bar{a}' + g^{-1} \cdot \bar{m}\big) \\
&= \sum_{g^{-1} \cdot \bar{a}' \in [-m,m]^d} g \cdot A\big(g^{-1} \cdot (\bar{\imath} - \bar{a}')\big) \otimes g \cdot C\big(g^{-1} \cdot (\bar{a}' + \bar{m})\big) \\
&= \sum_{g^{-1} \cdot \bar{a}' \in [-m,m]^d} \big(g \cdot A\big)(\bar{\imath} - \bar{a}') \otimes \big(g \cdot C\big)(\bar{a}' + \bar{m}) \\
&= \sum_{\bar{a}' \in [-m,m]^d} \big(g \cdot A\big)(\bar{\imath} - \bar{a}') \otimes \big(g \cdot C\big)(\bar{a}' + \bar{m}) \\
&= \big((g \cdot A) * (g \cdot C)\big)(\bar{\imath}).
\end{aligned}$$

For the penultimate step, we note that $g^{-1} \cdot \bar{a}' \in [-m,m]^d$ compared with $\bar{a}' \in [-m,m]^d$ is just a reordering of those indices in the sum. Thus, we have our result for pixel $\bar{\imath}$, so it holds for all pixels. ■

Now we will prove theorem 1.

Theorem (Restatement of 1). *A function $f : \mathcal{A}_{N,d,k,p} \to \mathcal{A}_{N,d,k',p'}$ is linear and $G_{N,d}$-equivariant if and only if it can be written as $\iota_k(A * C)$ for some B_d-isotropic $C \in \mathcal{A}_{M,d,k+k',pp'}$, where $M = N$ if N is odd and $M = N + 1$ otherwise.*

Proof. First, we will show the reverse direction. Let $C \in \mathcal{A}_{M,d,k+k',pp'}$ be B_d-isotropic, and let a function f be defined as $f(A) = \iota_k(A * C)$. Let $g \in B_d$, $A \in \mathcal{A}_{N,d,k,p}$. Then by the invariance of C we have

$$\begin{aligned}\iota_k((g \cdot A) * C) &= \iota_k((g \cdot A) * (g \cdot C)) \\ &\overset{3}{=} \iota_k(g \cdot (A * C)) \\ &\overset{A5}{=} g \cdot \iota_k(A * C).\end{aligned}$$

Hence, f is B_d-equivariant. By (A 1) and (A 4) f is also translation equivariant, so it is equivariant to $G_{N,d}$. Also, by the linearity of convolution (A 2) and contraction (A 5), f is linear. Thus, the reverse direction holds.

Now we will prove the forward direction. Let $f : \mathcal{A}_{N,d,k,p} \to \mathcal{A}_{N,d,k',p'}$ be a linear $G_{N,d}$-equivariant function. Thus, f must be translation equivariant, so by proposition 1 we can write f as $f(A) = \iota_k(A * C)$ for some $C \in \mathcal{A}_{M,d,k+k',pp'}$. Now it suffices to show that C is B_d-isotropic. Let $A \in \mathcal{A}_{N,d,k,p}$, let $g \in B_d$ and let $B = g^{-1} \cdot A$. Then by the equivariance of f we have

$$\iota_k(A * C) = \iota_k((g \cdot B) * C).$$

Thus, by lemma 2, we have $g \cdot C = C$. Therefore, C is B_d-isotropic, and this completes the proof. ■

A.2. Extension of Vector Neuron nonlinearities to tensors

In this section, we show that the tensor extension of the Vector Neuron nonlinearity [17] given by (5.2) is $G_{N,d}$-equivariant.

Proposition 2. *Let $A_i \in \mathcal{A}_{N,d,k_z,p_z}$, $g \in G_{N,d}$ and $\alpha_i, \beta_i \in \mathbb{R}$ for $i = 1, \dots, c_z$. Then $\sigma\left(\left(g \cdot A_i\right)_{i=1}^{c_z}\right) = g \cdot \sigma\left(\left(A_i\right)_{i=1}^{c_z}\right)$.*

Proof. Let $A_i \in \mathcal{A}_{N,d,k_z,p_z}$, and $\alpha_i, \beta_i \in \mathbb{R}$ for $i = 1, \dots, c_z$. It is clear to see that σ is translation equivariant because all the operations are pixel-wise. Thus, we will show that σ is equivariant to $g \in B_d$. First, note that applying g to all A_i results in $g \cdot Q$ and $g \cdot K$. Now

$$\iota_{k_z}(g \cdot Q \otimes g \cdot K) = \iota_{k_z}(g \cdot (Q \otimes K)) = g \cdot \iota_{k_z}(Q \otimes K) = \iota_{k_z}(Q \otimes K).$$

Note that the last step is because both Q and K are $k_{z_{(p_z)}}$-tensor images, so $\iota_{k_z}(Q \otimes K)$ is a $0_{(+)}$-tensor image. Hence, if $\iota_{k_z}(Q \otimes K) \geq 0$, then $\sigma\left(\left(g \cdot A_i\right)_{i=1}^{c_z}\right) = g \cdot Q = g \cdot \sigma\left(\left(A_i\right)_{i=1}^{c_z}\right)$ and σ is B_d-equivariant. Now suppose $\iota_{k_z}(Q \otimes K) < 0$

$$\begin{aligned}\sigma\left(\left(g \cdot A_i\right)_{i=1}^{c_z}\right) &= g \cdot Q - \iota_{k_z}\left(g \cdot Q \otimes \frac{g \cdot K}{\|g \cdot K\|_2}\right)\frac{g \cdot K}{\|g \cdot K\|_2} \\ &= g \cdot Q - \iota_{k_z}\left(g \cdot Q \otimes g \cdot \frac{K}{\|K\|_2}\right) g \cdot \frac{K}{\|K\|_2} \\ &= g \cdot Q - g \cdot \left(\iota_{k_z}\left(Q \otimes \frac{K}{\|K\|_2}\right)\frac{K}{\|K\|_2}\right)\end{aligned}$$

$$= g \cdot \left(Q - \iota_{k_z} \left(Q \otimes \frac{K}{\|K\|_2} \right) \frac{K}{\|K\|_2} \right)$$
$$= g \cdot \sigma\left((A_i)_{=1}^{c_z}\right).$$

Thus σ is B_d-equivariant. ■

A.3. LayerNorm equivariance

Proposition 3. *The LayerNorm is $G_{N,d}$-equivariant.*

Proof. Let $(A_i)_{i=1}^{c_z}$ be a set of $1_{(p_z)}$-tensor images. Clearly, this function will be translation equivariant because let $g \in B_d$. Let $\bar{A}_i$ be as defined in §5, and let $\bar{\imath}$ be a pixel index of $\bar{A}_i$. Then

$$(g \cdot \bar{A}_i)(\bar{\imath}) = g \cdot \bar{A}_i(g^{-1} \cdot \bar{\imath}) \tag{A 9}$$
$$= g \cdot \left(A_i(g^{-1} \cdot \bar{\imath}) - \sum_{j=1}^{c_z} \sum_{\bar{\jmath} \in [N]^d} A_j(\bar{\jmath}) \right) \tag{A 10}$$
$$= g \cdot A_i(g^{-1} \cdot \bar{\imath}) - \sum_{j=1}^{c_z} \sum_{\bar{\jmath} \in [N]^d} g \cdot A_j(\bar{\jmath}) \tag{A 11}$$
$$= g \cdot A_i(g^{-1} \cdot \bar{\imath}) - \sum_{j=1}^{c_z} \sum_{g^{-1} \cdot \bar{\jmath} \in [N]^d} g \cdot A_j(g^{-1} \cdot \bar{\jmath}) \tag{A 12}$$
$$= (g \cdot A_i)(\bar{\imath}) - \sum_{j=1}^{c_z} \sum_{\bar{\jmath} \in [N]^d} (g \cdot A_j)(\bar{\jmath}). \tag{A 13}$$

Note that 30 follows because $\sum_{\bar{\jmath} \in [N]^d} A_j(\bar{\jmath}) = \sum_{g^{-1} \cdot \bar{\jmath} \in [N]^d} A_j(g^{-1} \cdot \bar{\jmath})$. Thus, the mean centring is equivariant to B_d. Likewise,

$$g \cdot \Sigma = g \cdot \frac{1}{c_z N^d} \sum_{i=1}^{c_z} \sum_{\bar{\imath} \in [N]^d} (\bar{A}_i \otimes \bar{A}_i)(\bar{\imath}) \tag{A 14}$$
$$= \frac{1}{c_z N^d} \sum_{i=1}^{c_z} \sum_{\bar{\imath} \in [N]^d} g \cdot (\bar{A}_i \otimes \bar{A}_i)(\bar{\imath}) \tag{A 15}$$
$$= \frac{1}{c_z N^d} \sum_{i=1}^{c_z} \sum_{g^{-1} \cdot \bar{\imath} \in [N]^d} g \cdot (\bar{A}_i \otimes \bar{A}_i)(g^{-1} \cdot \bar{\imath}) \tag{A 16}$$
$$= \frac{1}{c_z N^d} \sum_{i=1}^{c_z} \sum_{\bar{\imath} \in [N]^d} (g \cdot \bar{A}_i \otimes g \cdot \bar{A}_i)(\bar{\imath}). \tag{A 17}$$

Finally, the inverse square root operation is B_d-equivariant. If we write it as the function f such that $f(\Sigma) = f(U \Lambda U^\top) = U \Lambda^{-\frac{1}{2}} U^\top = \Sigma^{-\frac{1}{2}}$. Then

$$g \cdot f(\Sigma) = g \cdot f(U \Lambda U^\top) \tag{A 18}$$
$$= M(g) U \Lambda^{-\frac{1}{2}} U^\top M(g)^\top \tag{A 19}$$
$$= \left(M(g)U\right) \Lambda^{-\frac{1}{2}} \left(M(g)U\right)^\top \tag{A 20}$$
$$= f\left(\left(M(g)U\right) \Lambda \left(M(g)U\right)^\top \right) \tag{A 21}$$
$$= f(g \cdot \Sigma). \tag{A 22}$$

Thus, $[g \cdot B_i(\bar{\imath})]_\ell = \left[g \cdot \bar{A}_i(\bar{\imath})\right]_\ell \left[g \cdot \Sigma^{-\frac{1}{2}}\right]_{j,\ell}$ which is the same as rotating all the input A_i, so LayerNorm is equivariant. ■

A.4. Max pool equivariance

The $O(d)$-invariance of the tensor norm allows the max pool layer to be B_d-equivariant. With a careful definition of translations for the larger and smaller images, we can also get translational equivariance, as we see in the following proposition.

Proposition 4. *Let $g \in B_d$ and let $\tau \in (\mathbb{Z}/(N/b)\mathbb{Z})^d$ be the translation on the d-torus with side lengths of N/b. For images $A \in \mathcal{A}_{N,d,k,p}$, we define the action of this translation as $(L_\tau A)(\bar{\imath}) = A(\bar{\imath} - b\,\tau)$. Then, max pool$_b$ (3.11) is equivariant to both of these groups.*

Before we prove this proposition, we need a quick lemma about the tensor norm (3.9).

Lemma 4. *The tensor norm (3.9) is $O(d)$-invariant.*

Proof. Let c be a $k_{(p)}$-tensor, and let $g \in O(d)$. Then

$$\|g \cdot c\|_2 = \sqrt{\iota_k(g \cdot c \otimes g \cdot c))} \overset{(A\,4)}{=} \sqrt{g \cdot \iota_k(c \otimes c)} \overset{(*)}{=} \sqrt{\iota_k(c \otimes c)} = \|c\|_2. \tag{A 23}$$

The $(*)$ equality is because $\iota_k(c \otimes c)$ is always a scalar. This completes the proof. ■

Now we will prove the proposition.

Proof. First, we will show equivariance to translations. Let $\tau \in (\mathbb{Z}/(N/b)\mathbb{Z})^d$ be the translation on the d-torus with side lengths of N/b as defined in the proposition. Let $A \in \mathcal{A}_{N,d,k,p}$ and let $\bar{\imath}$ be a pixel index. Then following the definitions, we have

$$\begin{aligned}
&\left(L_\tau \max \text{pool}_b(A)\right)(\bar{\imath}) \\
&\quad = \max \text{pool}_b(A)(\bar{\imath} - \tau) \\
&\quad = A\left(b\,(\bar{\imath} - \tau) + arg\,max_{\bar{a} \in [0,b-1]^d} \|A(b\,(\bar{\imath} - \tau) + \bar{a})\|_2\right) \\
&\quad = A\left(b\,\bar{\imath} - b\,\tau + arg\,max_{\bar{a} \in [0,b-1]^d} \|A(b\,\bar{\imath} - b\,\tau + \bar{a})\|_2\right) \\
&\quad = (L_\tau A)\left(b\,\bar{\imath} + arg\,max_{\bar{a} \in [0,b-1]^d} \|(L_\tau A)(b\,\bar{\imath} + \bar{a})\|_2\right) \\
&\quad = \max \text{pool}_b(L_\tau A)(\bar{\imath}).
\end{aligned}$$

Thus max pool$_b$ is equivariant to translations. Now let $g \in B_d$. Thus, by lemma 4, we have

$$\begin{aligned}
&\left(g \cdot \max \text{pool}_b(A)\right)(\bar{\imath}) \\
&\quad = g \cdot \max \text{pool}_b(A)(g^{-1} \cdot \bar{\imath}) \\
&\quad = g \cdot A\left(b\,(g^{-1} \cdot \bar{\imath}) + arg\,max_{\bar{a} \in [0,b-1]^d} \left\|A\left(b\,(g^{-1} \cdot \bar{\imath}) + \bar{a}\right)\right\|_2\right) \\
&\quad = g \cdot A\left(g^{-1} \cdot \left(b\,\bar{\imath} + g \cdot arg\,max_{\bar{a} \in [0,b-1]^d} \left\|A\left(g^{-1} \cdot (b\,\bar{\imath} + g \cdot \bar{a})\right)\right\|_2\right)\right) \\
&\quad \overset{4}{=} (g \cdot A)\left(b\,\bar{\imath} + g \cdot arg\,max_{\bar{a} \in [0,b-1]^d} \left\|g \cdot A\left(g^{-1} \cdot (b\,\bar{\imath} + g \cdot \bar{a})\right)\right\|_2\right) \\
&\quad = (g \cdot A)\left(b\,\bar{\imath} + g \cdot arg\,max_{\bar{a} \in [0,b-1]^d} \left\|(g \cdot A)(b\,\bar{\imath} + g \cdot \bar{a})\right\|_2\right) \\
&\quad \overset{(*)}{=} (g \cdot A)\left(b\,\bar{\imath} + g\,g^{-1} \cdot arg\,max_{\bar{a} \in [0,b-1]^d} \|(g \cdot A)(b\,\bar{\imath} + \bar{a})\|_2\right) \\
&\quad = (g \cdot A)\left(b\,\bar{\imath} + arg\,max_{\bar{a} \in [0,b-1]^d} \|(g \cdot A)(b\,\bar{\imath} + \bar{a})\|_2\right) \\
&\quad = \max \text{pool}_b(g \cdot A)(\bar{\imath}).
\end{aligned}$$

For the $(*)$ equality we note that $arg\,max_{\bar{a}} \|A(g \cdot \bar{a})\|_2 = g^{-1} \cdot arg\,max_{\bar{a}} \|A(\bar{a})\|_2$ because the pixel index returned by the left side would have to be transformed by g to maximize $\|A(\bar{a})\|_2$. Hence, the max pool is B_d-equivariant, and this concludes the proof. ■

Appendix B. Mathematical details of related work

The most common method to design equivariant maps is via group convolution, on the group or on the homogeneous space where the features lie. Regular convolution of a vector field $f : (\mathbb{Z}/N\mathbb{Z})^d \to \mathbb{R}^c$ and a filter $\phi : (\mathbb{Z}/N\mathbb{Z})^d \to \mathbb{R}^c$ is defined as

$$(f * \phi)(x) = \sum_{y \in (\mathbb{Z}/N\mathbb{Z})^d} \underbrace{\langle f(y), \phi(x-y) \rangle}_{\text{scalar product of vectors}} = \sum_{y \in (\mathbb{Z}/N\mathbb{Z})^d} \sum_{j=1}^{c} \underbrace{f^j(y)\phi^j(x-y)}_{\in \mathbb{R}}. \tag{B 1}$$

Our generalization of convolution replaces this scalar product of vectors with the outer product of tensors.

B.1. Clifford convolution

Probably, the most related work is by Brandstetter *et al*. [48], which replaces the scalar product in (B1) with the geometric product of multi-vector inputs and multi-vector filters of a Clifford algebra. It considers multi-vector fields, i.e.: vector fields $f : \mathbb{Z}^2 \to (Cl_{p,q}(\mathbb{R}))^c$. The real Clifford algebra $Cl_{p,q}(\mathbb{R})$ is an associative algebra generated by $p + q = d$ orthonormal basis elements: $e_1, \ldots, e_{p+q} \in \mathbb{R}^d$ with the relations

$$e_i \otimes e_i = +1 \quad (i \le p), \tag{B 2}$$

$$e_j \otimes e_j = -1 \quad (p < j \le n), \tag{B 3}$$

$$e_i \otimes e_j = -e_j \otimes e_i \quad (i \ne j). \tag{B 4}$$

For instance, $Cl_{2,0}(\mathbb{R})$ has the basis $\{1, e_1, e_2, e_1 \otimes e_2\}$ and is isomorphic to the quaternions $\mathbb{H}$.

The Clifford convolution replaces the elementwise product of scalars of the usual convolution of (B 1) by the geometric product of multi-vectors in the Clifford algebra

$$f * \phi(x) = \sum_{y \in (\mathbb{Z}/N\mathbb{Z})^d} \sum_{j=1}^{c} \underbrace{f^j(y) \otimes \phi^j(y-x)}_{\in Cl_{p,q}(\mathbb{R})}, \tag{B 5}$$

where $f : \mathbb{Z}^2 \to (Cl_{p,q}(\mathbb{R}))^c$ and $\phi : \mathbb{Z}^2 \to (Cl_{p,q}(\mathbb{R}))^c$.

The Clifford algebra $Cl_{p,q}(\mathbb{R})$ is a quotient of the tensor algebra

$$T(\mathbb{R}^d) = \bigoplus_{k \ge 0} \underbrace{\mathbb{R}^d \otimes \ldots \otimes \mathbb{R}^d}_{k \text{ times}} = \bigoplus_{k \ge 0} (\mathbb{R}^d)^{\otimes k}, \tag{B 6}$$

by the two-side ideal $\langle \{v \otimes v - Q(v) : v \in \mathbb{R}^d\} \rangle$, where the quadratic form Q is defined by $Q(e_i) = +1$,if $i \le p$, and $Q(e_j) = -1$, else $p < j \le n$. Our geometric images are functions $A : (\mathbb{Z}/N\mathbb{Z})^d \to \mathcal{T}_{d,k,p}$, where $\mathcal{T}_{d,k,p} = (\mathbb{R}^d)^{\otimes k} \subset T(\mathbb{R}^d)$. They can be related to the Clifford framework by seeing them as N-periodic functions from $\mathbb{Z}^d$ whose image is projected via the quotient map on the Clifford algebra. This projection can be seen as a contraction of tensors.

The Clifford convolution is not equivariant under multi-vector rotations or reflections. However, the authors derive a constraint on the filters for $d = 2$, which allows to build generalized Clifford convolutions that are equivariant with respect to rotations or reflections of the multi-vectors. That is, they prove equivariance of a Clifford layer under orthogonal transformations if the filters satisfy the constraint: $\phi^i(Rx) = R\phi^i(x)$.

B.2. Unified Fourier framework

Part of our work can be studied under the unified framework for group equivariant networks on homogeneous spaces derived from a Fourier perspective proposed in [63]. The idea is to consider general tensor-valued feature fields, before and after a convolution. Their fields are functions $f : G/H \to V$ over the homogeneous space G/H taking values in the vector space V and their filters are kernels $\kappa : G \to \mathrm{Hom}(V, V')$. Essentially, their convolution replaces the scalar product of vectors of traditional convolution by applying a homomorphism. In particular, if G is a finite group and $H = \{0\}$, they define convolution as

$$\kappa * f(x) = \frac{1}{|G|} \sum_{y \in G} \underbrace{\kappa(x^{-1} y) f(y)}_{\in V'}. \tag{B 7}$$

Van Rossum & Drake [63] give a complete characterization of the space of kernels for equivariant convolutions. In our framework, the group is $\mathbb{Z}/N\mathbb{Z}$ and the kernel is an outer product by a filter C: $\kappa(g)A(g) = A(g) \otimes C(g)$. Note that $\mathbb{Z}/N\mathbb{Z}$ is neither a homogeneous space of $O(d)$ nor of B^d.

We can analyse our problem from a spectral perspective; in particular, we can describe all linear equivariant using representation theory, using similar tools as in the proof of theorem 1 in [36]. This theorem states that convolutional structure is a sufficient and necessary condition for equivariance to the action of a compact group. Some useful references about group representation theory are given in [67], a classical book about the theory of abstract harmonic analysis, and in [68], about the particular applications of it.

B.3. Linear equivariant maps

In this work, we define an action over tensor images of $O(d)$ by rotation of tensors in each pixel; of B^d by rotating the grid of pixels and each tensor in the pixel; and of $(\mathbb{Z}/N\mathbb{Z})^d$ by translation of the grid of pixels. The action of each one of these groups G over $\mathcal{T}_{d,k,p}$

$$\Phi_{d,k,p} : G \to GL_{con}(\mathcal{T}_{d,k,p}), \tag{B 8}$$

can be decomposed into irreducible representations of G

$$\Phi_{d,k,p} \equiv \bigoplus_{\pi \in \hat{G}} m_{d,k,p}(\pi)\, \pi. \tag{B 9}$$

That is, there is a basis of the Hilbert space $\mathcal{T}_{d,k,p}$ in which the action of G is defined via a linear sparse map. In the case of G finite, for all $g \in G$, there is a matrix P splitting the representation in the Hilbert space into its irreducible components

$$P^{-1}\, \Phi_{d,k,p}(g)\, P = \bigoplus_{\pi \in \hat{G}} m_{d,k,p}(\pi)\, \pi(g). \tag{B 10}$$

Consider now linear maps between tensor images:

$$\mathcal{C} : \mathcal{T}_{d,k,p} \to \mathcal{T}_{d',k',p'}. \tag{B 11}$$

Linear equivariant maps satisfy that $\mathcal{C} \circ \Phi_{d,k,p} = \Phi_{d',k',p'} \circ \mathcal{C}$. That is, if $\tilde{\mathcal{C}}$ is the representation of $\mathcal{C}$ in the above basis,

$$\tilde{\mathcal{C}} \circ \bigoplus_{\pi \in G} m_{d,k,p}(\pi)\, \pi = \bigoplus_{\pi \in G} m_{d',k',p'}(\pi)\, \pi \circ \tilde{\mathcal{C}}. \tag{B 12}$$

By Schur's lemma, this implies that $\mathcal{C} \equiv \bigoplus_{\pi \in G} m_{d,k,p}(\pi)\, Id_{d_\pi}$.

The power of representation theory is not limited to compact groups. Mackey machinery allows us to study, for instance, semidirect products of compact groups and other groups, and in general to relate the representations of a normal subgroup with the ones of the whole group. This is the spirit of [15], which makes extensive use of the induced representation theory. An introduction to this topic can be found in chapter 7 in [67].

B.4. Steerable convolutional neural networks

The work in [15] deals exclusively with signals $f : \mathbb{Z}^2 \to \mathbb{R}^k$. They consider the action of $G = p4m$ on $\mathbb{Z}^2$ by translations, rotations by 90 degrees around any point and reflections. This group is a semidirect product of $\mathbb{Z}^2$ and B_2, so every $x \in p4m$ can be written as $x = t\,r$, for $t \in \mathbb{Z}^2$ and $r \in B_2$. They show that equivariant maps with respect to representations ρ and ρ' of rotations and reflections B_2 lead to equivariant maps with respect to certain representations of G, π and π'. This means that if we find a linear map $\phi : f \mapsto \phi f$ such that $\phi\,\rho(h) f = \rho'(h)\,\phi f$ for all $h \in B_2$, then for the representation of $G\pi'$ defined by

$$\pi'(t\,r) f(y) = \rho(r)\,[f((t\,r)^{-1}\,y)], \quad t\,r \in G, y \in \mathbb{Z}^2, \tag{B 13}$$

we automatically have that $\phi\,\pi(g) f = \pi'(g)\,\phi f$ for all $g \in G$. This is the representation of G induced by the representation ρ of B_2.

Note the similarity between Definition 11 of the action of B_d on tensor images and (B13). The convolution with a symmetric filter produces easily an equivariant map with respect to the action of the semidirect product of $\mathbb{Z}^d$ and B_d on the tensor images.

B.5. Approximate symmetries

The recent work [69] studies approximately equivariant networks, which are biased towards preserving symmetry but are not strictly constrained to do so. They define a relaxed group convolution which is approximately equivariant in the sense that

$$\|\rho_X(g) f *_G \Psi(x) - f *_G \Psi(\rho_Y(y)\,x)\| < \epsilon. \tag{B 14}$$

They use a classical convolution but with different kernels for different group elements.

Appendix C. Experimental details

C.1. Data

The data are the PDEBench files 2D_CFD_Rand_M0.1_Eta0.01_Zeta0.01_periodic_128_Train.hdf5 and 2D_CFD_Rand_M1.0_Eta0.1_Zeta0.1_periodic_128_Train.hdf5, which can be found at https://darus.uni-stuttgart.de/dataset.xhtml?persistentId=doi:10.18419/darus−2986 [58]. We used the first 128 trajectories as training data, the next 32 trajectories as a validation set and the next 128 trajectories as a test dataset. The density and pressure fields are mean-centred and scaled to have variance 1 based on the training and validation datasets. The velocity field is not mean-centred because the only rotationally isotropic vector is the zero vector, but it is scaled to have variance 1 in the components.

C.2. Models

Model specifics are described below. For equivariant models, we always use ReLU for scalars and the Vector Neuron activation for non-scalars. For equivariant encoder and decoder blocks, we use 3×3 filters instead of 1×1 filters because, for some order and parity pairs, there are no $1 \times 1 B_d$-isotropic filters. All convolutions use biases except for the UNet. For equivariant models, the bias is a scale of the mean tensor of that image. Additional details are in table 2.

— *Dilated ResNet* [56]: The model starts with two 'encoder' convolutions with 1×1 filters and ReLU activations. There are four blocks, each consisting of seven convolutions with dilations of $1, 2, 4, 8, 4, 2, 1$ with associated ReLU activations. There are residual connections connecting each block. The model concludes with two 'decoder' convolutions with 1×1 filters and a ReLU activation between the two.

- *ResNet* [26]: This model consists of eight blocks of two convolutions each with residual connections between each block. Each block also has LayerNorm and a GeLU activation [70]. We put the LayerNorm and activation prior to the convolution (pre-activation order [71]) following [23]. This model also uses two 'encoder' 1×1 convolutions and two 'decoder' 1×1 convolutions.
- *UNet LayerNorm* [23]: This model is referred to as 'UNetBase' in [23]. This starts with an embedding block with a convolution with a 3×3 filter followed by LayerNorm and a GeLU activation [29]. Next comes a max pool_2 followed by two convolutions with LayerNorm and GeLU activation. This process is repeated for four total downsamples, and notably the number of convolution channels is doubled for every downsample. Then the process happens in reverse, with max pooling replaced with transposed convolution to double the spatial size instead of halving it each time. See [72] for a description of transposed convolution. The number of convolution channels is also halved each time we upsample. The final kicker is that there are also residual connections from before each downsample to after each upsample for the appropriate spatial size. The model concludes with a final convolution. In the equivariant model we do not include the LayerNorm because it hurt the performance.
- *UNet* [53]: This model is the same as the one above, except it uses BatchNorm instead of LayerNorm and the convolutions are without biases.

C.3. Training

For a loss function, we use the sum of mean squared error loss or sum of mean squared error. This loss sums over the tensor components and the channels and takes the mean over the spatial components. If $\{A_i\}_{i=1}^c$ are the true $k_{i(p_i)}$-tensor images and $\left\{\hat{A}_i\right\}_{i=1}^c$ are our predicted $k_{i(p_i)}$-tensor images, then the $\mathcal{L}_{\text{smse}}$ is defined as

$$\mathcal{L}_{\text{smse}}\left(\{A_i\}_{i=1}^c, \left\{\hat{A}_i\right\}_{i=1}^c\right) = \sum_{i=1}^{c} \frac{1}{N^d} \sum_{\bar{\imath}} \left\| A_i(\bar{\imath}) - \hat{A}_i(\bar{\imath}) \right\|_2^2, \tag{C 1}$$

where $\|\cdot\|_2$ is the tensor norm. When calculating a rollout loss, we simply sum the loss of each rollout step.

We follow a similar training regime as in [23]. We train for 50 epochs using the AdamW optimizer [73] with a weight decay of 1×10^{-5} and a cosine decay schedule [74] with five epochs of warm-up. Learning rates were tuned for each model, searching for values between 1×10^{-4} and 2×10^{-3}, and are included in table 2.

We trained on 4 RTX A5000 graphics cards with a batch size of 8, for an effective batch size of 32. Experiments we averaged over three trials, using the same training data each time. It is possible that different optimizers, learning rate schedules, batch sizes or other hyperparameters may perform better on the task, but we held those fixed and only tuned the learning rate since our focus is on comparing the equivariant and non-equivariant models.

References

1. Planck Collaboration. 2016 Planck 2015 results – I. overview of products and scientific results. *A&A* **594**, A1. (doi:10.1051/0004-6361/201527101)
2. National Center for Atmospheric Research Staff (eds). 2022 The Climate Data Guide: GODAS: NCEP Global Ocean Data Assimilation System. https://climatedataguide.ucar.edu/climate-data/godas-ncep-global-ocean-data-assimilation-system (accessed 15 May 2025).
3. Wang D, Zhang X. 2019 Modeling of a 3D temperature field by integrating a physics-specific model and spatiotemporal stochastic processes. *Appl. Sci.* **9**, 2108. (doi:10.3390/app9102108)

4. Lossow S *et al*. 2009 Middle atmospheric water vapour and dynamics in the vicinity of the polar vortex during the Hygrosonde-2 campaign. *Atmos. Chem. Phys*. **9**, 07. (doi:10.5194/acp-9-4407-2009)
5. Bassi F, Rebay S. 1997 A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations. *J. Comput. Phys*. **131**, 267–279. (doi:10.1006/jcph.1996.5572)
6. Michałowska K, Goswami S, Karniadakis GE, Riemer-Sørensen S. 2024 Neural Operator Learning for Long-Time Integration in Dynamical Systems with Recurrent Neural Networks. In *2024 Int. Joint Conf. on Neural Networks (IJCNN)*, pp. 1–8. Japan: Yokohama. (doi:10.1109/IJCNN60899.2024.10650331)
7. McCabe M, Harrington P, Subramanian S, Brown J. 2023 Towards stability of autoregressive neural operators. *Trans. Mach. Learn. Res*. (doi:10.48550/arXiv.2306.10619)
8. Villar S, Hogg DW, Storey-Fisher K, Yao W, Blum-Smith B. 2021 Scalars are universal: equivariant machine learning, structured like classical physics. In *Adv. Neural Inf. Process Syst*., pp. 28848–28863 (doi:10.48550/arXiv.2308.10436)
9. Ricci MMG, Levi-Civita T. 1900 Methodes de calcul differentiel absolu et leurs applications. *Math. Ann*. **54**, 125–201. (doi:10.1007/bf01454201)
10. Thorne KS, Blandford RD. 2017 *Modern classical physics: optics, fluids, plasmas, elasticity, relativity, and statistical physics*. Princeton, NJ: Princeton University Press.
11. Canizares P, Murari D, Schönlieb CB, Sherry F, Shumaylov Z. 2024 Hamiltonian matching for symplectic neural integrators. Extended Abstract, NeurIPS 2024 Women in Machine Learning Affinity Event.
12. Hairer E, Wanner G, Lubich C. 2002 Geometric Numerical Integration. In *Springer series in computational mathematics*. Berlin, Heidelberg, Germany: Springer Berlin Heidelberg. (doi:10.1007/978-3-662-05018-7)
13. Villar S, Hogg DW, Yao W, Kevrekidis GA, Schölkopf B. 2024 Towards fully covariant machine learning. *Trans. Mach. Learn. Res*. (doi:10.48550/arXiv.2301.13724)
14. Verstappen RWCP, Veldman AEP. 2003 Symmetry-preserving discretization of turbulent flow. *J. Comput. Phys*. **187**, 343–368. (doi:10.1016/s0021-9991(03)00126-8)
15. Kondor R, Trivedi S. 2018 On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *Proc. of the 35th Int. Conf. on Machine Learning*. New York, NY: PMLR.
16. Cohen T, Welling M. 2016 Group equivariant convolutional networks. In *Int. Conf. on machine learning*. New York, NY: PMLR.
17. Stachenfeld K *et al*. 2022 Learned simulators for turbulence. In *Int. Conf. on Learning Representations*. Appleton, WI.
18. Bolton T, Zanna L. 2019 Applications of deep learning to ocean data inference and subgrid parameterization. *J. Adv. Model. Earth Syst*. **11**, 376–399. (doi:10.1029/2018ms001472)
19. Zanna L, Bolton T. 2020 Data-driven equation discovery of ocean mesoscale closures. *Geophys. Res. Lett*. **47**, L088376. (doi:10.1029/2020gl088376)
20. Gupta JK, Brandstetter J. 2022 *Towards multi-spatiotemporal-scale generalized PDE modeling*. New York, NY: Transactions on Machine Learning Research.
21. Li Z, Kovachki N, Azizzadenesheli K, Liu B, Bhattacharya K, Stuart A, Anandkumar A. 2021 *Fourier neural operator for parametric partial differential equations*. Appleton, WI: International Conference on Learning Representations.
22. Karniadakis GE, Kevrekidis IG, Lu L, Perdikaris P, Wang S, Yang L. 2021 Physics-informed machine learning. *Nat. Rev. Phys*. **3**, 422–440. (doi:10.1038/s42254-021-00314-5)
23. Pförtner M, Steinwart I, Hennig P, Wenger J. 2022 Physics-informed Gaussian process regression generalizes linear PDE solvers. *arXiv* (doi:10.48550/arXiv.2212.12474)
24. Akhound-Sadegh T, Perreault-Levasseur L, Brandstetter J, Welling M, Ravanbakhsh S. 2023 *Lie point symmetry and physics informed networks*. San Diego, CA: Advances in Neural Information Processing Systems.
25. Elesedy B, Zaidi S. 2021 Provably strict generalisation benefit for equivariant models. (doi:10.48550/arXiv.2102.10333)
26. Wang R, Walters R, Yu R. 2021 Incorporating symmetry into deep dynamics models for improved generalization. In *Int. Conf. on Learning Representations*. Appleton, WI.

27. Huang N, Levie R, Villar S. 2023 Approximately equivariant graph networks. *Adv. Neural Inf. Process. Syst*. **36**, 34627–34660. (doi:10.48550/arXiv.2308.10436)
28. Petrache M, Trivedi S. 2023 Approximation-generalization trade-offs under (approximate) group equivariance. *Adv. Neural Inf. Process. Syst*. **36**, 61936–61959. (doi:10.48550/arXiv.2305.17592)
29. Tahmasebi B, Jegelka S. 2023 The exact sample complexity gain from invariances for kernel regression. *Adv. Neural Inf. Process. Syst*. **36**. (doi:10.48550/arXiv.2303.14269)
30. Wang R, Walters R, Yu R. 2022 Data augmentation vs. equivariant networks: a theory of generalization on dynamics forecasting (doi:10.48550/arXiv.2206.09450)
31. Yarotsky D. 2018 Universal approximations of invariant maps by neural networks. (doi:10.48550/arXiv.1804.10306)
32. Dym N, Maron H. 2020 On the universality of rotation equivariant point cloud networks. (doi:10.48550/arXiv.2010.02449)
33. Bökman G, Kahl F, Flinth A. 2022 Zz-net: a universal rotation equivariant architecture for 2D point clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA*, pp. 10966–10975. (doi:10.1109/CVPR52688.2022.01070)
34. Kumagai W, Sannai A. 2020 Universal approximation theorem for equivariant maps by group CNNs. *arXiv* (doi:10.48550/arXiv.2012.13882)
35. LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD. 1989 Backpropagation applied to handwritten zip code recognition. *Neural Comput*. **1**, 541–551. (doi:10.1162/neco.1989.1.4.541)
36. Cohen T, Weiler M, Kicanaoglu B, Welling M. 2019 Gauge equivariant convolutional networks and the icosahedral CNN. In *Int. Conf. on Machine Learning*. New York, NY: PMLR.
37. Maron H, Ben-Hamu H, Shamir N, Lipman Y. 2019 Invariant and equivariant graph networks. Advances in Neural Information Processing Systems 37. San Diego, CA. (doi:10.48550/arXiv.2307.05432)
38. Cohen TS, Welling M. 2016 Group equivariant convolutional networks. In *Int. Conf. on Machine Learning*, pp. 2990–2999. New York, NY: PMLR.
39. Weiler M, Cesa G. 2021 General -equivariant steerable CNNs. Advances in Neural Information Processing Systems 32. San Diego, CA. (doi:10.48550/arXiv.1911.08251)
40. Thomas N, Smidt T, Kearnes S, Yang L, Li L, Kohlhoff K, Riley P. 2018 Tensor field networks: rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv* (doi:10.48550/arXiv.1802.08219)
41. Brandstetter J, Welling M, Worrall DE. 2022 Lie point symmetry data augmentation for neural PDE solvers. *Proc. 39th Int. Conf. on Machine Learning*. New York, NY: PMLR. (doi:10.48550/arXiv.2202.07643)
42. Mialon G, Garrido Q, Lawrence H, Rehman D, LeCun Y, Kiani BT. 2024 Self-supervised learning with Lie symmetries for partial differential equations. Advances in Neural Information Processing Systems 37. San Diego, CA. (doi:10.48550/arXiv.2307.05432)
43. Blum-Smith B, Villar S. 2022 Machine learning and invariant theory. *Notices of the American Mathematical Society* **70**, 1206–1213. (doi:10.1090/noti2760)
44. Kaba SO, Mondal AK, Zhang Y, Bengio Y, Ravanbakhsh S. 2023 Equivariance with learned canonicalization functions. In *Int. Conf. on Machine Learning*. PMLR.
45. Shumaylov Z, Zaika P, Rowbottom J, Sherry F, Weber M, Schönlieb CB. 2025 Lie algebra canonicalization: equivariant neural operators under arbitrary lie groups. International Conference on Learning Representations. Appleton, WI. (doi:10.48550/arXiv.2410.02698)
46. Jenner E, Weiler M. 2021 Steerable partial differential operators for equivariant neural networks. *arXiv* (doi:10.48550/arXiv.2106.10163)
47. Garanger K, Kraus J, Rimoli JJ. 2024 Symmetry-enforcing neural networks with applications to constitutive modeling. *Extrem. Mech. Lett*. **71**, 102188. (doi:10.1016/j.eml.2024.102188)
48. Brandstetter J, van den Berg R, Welling M, Gupta JK. 2023 Clifford neural layers for PDE modeling. *International Conference on Learning Representations*. Appleton, WI. (doi:10.48550/arXiv.2209.04934)
49. Lang S. 2002 *Algebra*. New York, NY: Springer.
50. Heilman A, Schlesinger C, Yan Q. 2024 Equivariant graph neural networks for prediction of tensor material properties of crystals. (doi:10.48550/arXiv.2406.03563)

51. Humphreys JE. 1990 *Reflection groups and coxeter groups*. Number 29 in Cambridge studies in advanced mathematics. Cambridge, UK: Cambridge University Press.
52. Simonyan K, Zisserman A. 2015 Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*. Appleton, WI. (doi:10.48550/arXiv.1409.1556)
53. Xu Y, Lei J, Dobriban E, Daniilidis K. 2022 Unified fourier-based kernel and nonlinearity design for equivariant networks on homogeneous spaces *International Conference on Machine Learning*. New York, NY: PMLR. (doi:10.48550/arXiv.2206.08362)
54. Deng C, Litany O, Duan Y, Poulenard A, Tagliasacchi A, Guibas L. 2021 Vector neurons: a general framework for so(3)-equivariant networks. *IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, Montreal, QC, Canada, pp. 12180–12189. (doi:10.1109/ICCV48922.2021.01198)
55. Ba JL, Kiros JR, Hinton GE. 2016 Layer normalization. (doi:10.48550/arXiv.1607.06450)
56. Trabelsi C *et al*. 2018 Deep complex networks. International Conference on Learning Representations. (doi:10.48550/arXiv.1705.09792)
57. Takamoto M, Praditia T, Leiteritz R, MacKinlay D, Alesiani F, Pflüger D, Niepert M. 2024 Pdebench: an extensive benchmark for scientific machine learning. Advances in Neural Information Processing Systems 36. (doi:10.48550/arXiv.2210.07182)
58. He K, Zhang X, Ren S, Sun J. 2016 Deep residual learning for image recognition. *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV*, pp. 770–778. (doi:10.1109/CVPR.2016.90)
59. Ronneberger O, Fischer P, Brox T. 2015 U-net: convolutional networks for biomedical image segmentation. (doi:10.48550/arXiv.1505.04597)
60. Bradbury J *et al*. 2018 JAX: composable transformations of Python+NumPy programs. https://github.com/jax-ml/jax.
61. McGreivy N, Hakim A. 2024 Weak baselines and reporting biases lead to overoptimism in machine learning for fluid-related partial differential equations. *Nat. Mach. Intell*. **6**, 1256–1269. (doi:10.1038/s42256-024-00897-5)
62. Gregory W, Hogg DW, Wong K. 2025. WilsonGregory/GeometricConvolutions: GI-Net Paper Version (v0.1.3). Zenodo. (doi:10.5281/zenodo.14932595)
63. Van Rossum G, Drake FL. 2009 *Python 3 reference manual*. Scotts Valley, CA: CreateSpace.
64. Harris CR *et al*. 2020 Array programming with NumPy. *Nature* **585**, 357–362. (doi:10.1038/s41586-020-2649-2)
65. Hunter JD. 2007 Matplotlib: a 2D graphics environment. *Comput. Sci. Eng*. **9**, 90–95. (doi:10.1109/mcse.2007.55)
66. Price-Whelan AM. 2021 *cmastro: colormaps for astronomers*. See https://github.com/adrn/cmastro.
67. Folland GB. 2016 *A course in abstract harmonic analysis*. Boca Raton, FL: CRC Press.
68. Chirikjian GS, Kyatkin AB. 2021 *Engineering applications of noncommutative harmonic analysis: with emphasis on rotation and motion groups*. Boca Raton, FL: CRC Press.
69. Wang R, Walters R, Yu R. 2022 Approximately equivariant networks for imperfectly symmetric dynamics. (doi:10.48550/arXiv.2201.11969)
70. Hendrycks D, Gimpel K. 2023 Gaussian error linear units (GELUs).
71. He K, Zhang X, Ren S, Sun J. 2016 Identity mappings in deep residual networks. (doi:10.48550/arXiv.1603.05027)
72. Dumoulin V, Visin F. 2018 A guide to convolution arithmetic for deep learning. (doi:10.48550/arXiv.1603.07285)
73. Loshchilov I, Hutter F. 2019 Decoupled weight decay regularization. In *Int. Conf. on Learning Representations*. (doi:10.48550/arXiv.1711.05101)
74. Loshchilov I, Hutter F. 2017 SGDR: stochastic gradient descent with warm restarts. (doi:10.48550/arXiv.1608.03983)