Geometric Deep Learning

Erik Schultheis, Kate Haitsiukevich, Çağlar Hızlı, Alison Pouplin, Vikas Garg

7.11.2024

Part VII: Convolutions

A note on terminology

Caveat: Convolution operation

$$(f \star g)(t) = \int_0^T f(\tau) g(T - \tau)\,,$$

but in ML we call *cross-correlation* convolution

$$(f \star g)(t) = \int_0^T f(\tau) g(\tau)\,.$$

Grids vs Graphs

What makes the geometric domain of a grid different from that of a graph?

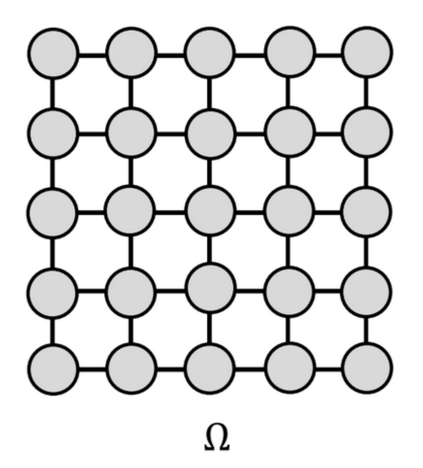


Image: Bronstein et al. (2021)

Grids vs Graphs

What makes the geometric domain of a grid different from that of a graph?

The symmetry group. Each node has well-defined top/bottom/left/right neighbour $\implies$ not *permutation-invariant*

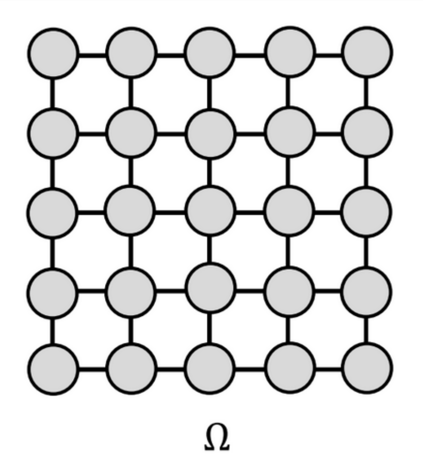


Image: Bronstein et al. (2021)

Grids vs Graphs

What makes the geometric domain of a grid different from that of a graph?

The symmetry group. Each node has well-defined top/bottom/left/right neighbour $\implies$ not *permutation-invariant*

But *shift-invariant* (periodic boundary)

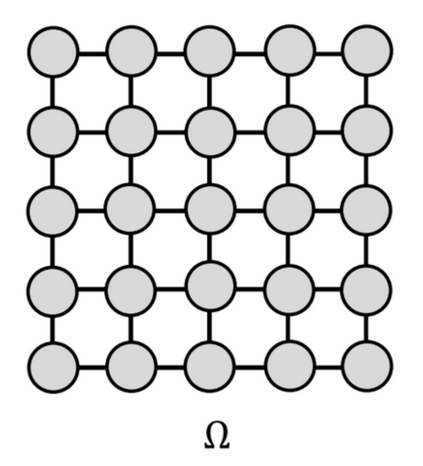


Image: Bronstein et al. (2021)

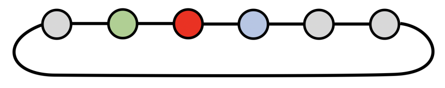
Grids vs Graphs

What makes the geometric domain of a grid different from that of a graph?

The symmetry group. Each node has well-defined top/bottom/left/right neighbour $\implies$ not *permutation-invariant*

But *shift-invariant* (periodic boundary)

For simplicity: limit to one dimension



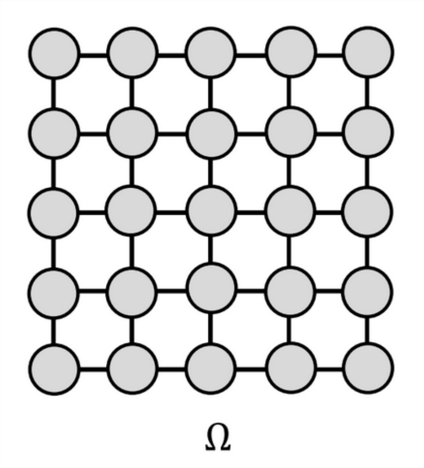


Image: Bronstein et al. (2021)

Formal setup

Let $G = C(n)$ be the cyclic group of order n, and $\Omega = 1, \ldots, n$ be the domain of sequences of length n with periodic boundary conditions

Formal setup

Let $G = C(n)$ be the cyclic group of order n, and $\Omega = 1, \ldots, n$ be the domain of sequences of length n with periodic boundary conditions

What are linear equivariant mappings?

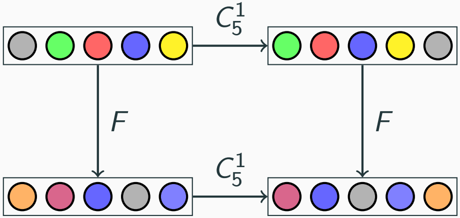
Formal setup

Let $G = C(n)$ be the cyclic group of order n, and $\Omega = 1, \ldots, n$ be the domain of sequences of length n with periodic boundary conditions

What are linear equivariant mappings?

We want $CF\boldsymbol{a} = FC\boldsymbol{a}$,

C shift-matrix, $\boldsymbol{a}$ activations/inputs, F equivariant mapping



Eigendecomposition of shift-matrices

The regular representation of $C(n)$ is given by circulant matrices

$$C_n^k := \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}^k$$

Eigendecomposition of shift-matrices

The regular representation of $C(n)$ is given by circulant matrices

$$C_n^k := \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}^k$$

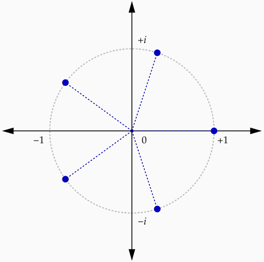
We know:

- Over $\mathbb{C}$, the representation is decomposible

Eigendecomposition of shift-matrices

The regular representation of $C(n)$ is given by circulant matrices

$$C_n^k := \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}^k$$



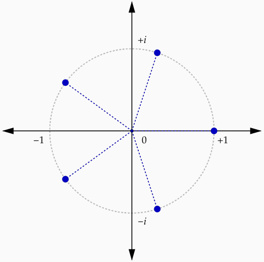
We know:

- Over $\mathbb{C}$, the representation is decomposible
- The irreps of $C(n)$ are the roots of unity $\psi_n^k = \exp(2\pi ik/n)$

Eigendecomposition of shift-matrices

The regular representation of $C(n)$ is given by circulant matrices

$$C_n^k := \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}^k$$



We know:

- Over $\mathbb{C}$, the representation is decomposible
- The irreps of $C(n)$ are the roots of unity $\psi_n^k = \exp(2\pi ik/n)$
- $\implies C_n^k = J^{-1}DJ$ for *diagonal* D with entries $\psi_\cdot^n$

Gaussian elimination

To find J, we can solve

$$(C_n^1 - \exp(2\pi ik/n)I)\mathbf{v} = 0$$

using Gaussian elimination.

Gaussian elimination

To find J, we can solve

$$(C_n^1 - \exp(2\pi i k/n) I)\mathbf{v} = 0$$

using Gaussian elimination.

$$\begin{pmatrix} -\psi_n^k & 1 & 0 & \dots & 0 \\ 0 & -\psi_n^k & 1 & \dots & 0 \\ 0 & 0 & -\psi_n^k & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 1 & 0 & 0 & 0 & -\psi_n^k \end{pmatrix}$$

Gaussian elimination

To find J, we can solve

$$(C_n^1 - \exp(2\pi ik/n)I)\mathbf{v} = 0$$

using Gaussian elimination.

$$\begin{array}{cl}
\begin{pmatrix}
\boldsymbol{-\psi_n^k} & \mathbf{1} & \mathbf{0} & \dots & \mathbf{0} \\
0 & -\psi_n^k & 1 & \dots & 0 \\
0 & 0 & -\psi_n^k & \ddots & 0 \\
\vdots & \vdots & \vdots & \ddots & 1 \\
1 & 0 & 0 & 0 & -\psi_n^k
\end{pmatrix}
&
\left|\begin{array}{l}
\cdot\, \psi_n^{n-k} \\
\\
\\
\\
\text{-row } 1
\end{array}\right.
\end{array}$$

Gaussian elimination

To find J, we can solve

$$(C_n^1 - \exp(2\pi ik/n)I)\mathbf{v} = 0$$

using Gaussian elimination.

$$\begin{pmatrix} -\psi_n^k & 1 & 0 & \dots & 0 \\ 0 & -\psi_n^k & 1 & \dots & 0 \\ 0 & 0 & -\psi_n^k & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & \psi_n^{n-k} & 0 & 0 & -\psi_n^k \end{pmatrix}$$

Gaussian elimination

To find J, we can solve

$$(C_n^1 - \exp(2\pi ik/n)I)\mathbf{v} = 0$$

using Gaussian elimination.

$$\begin{pmatrix} -\psi_n^k & 1 & 0 & \dots & 0 \\ 0 & -\psi_n^k & 1 & \dots & 0 \\ 0 & 0 & -\psi_n^k & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & 0 & \color{red}{\psi_n^{n-2k}} & 0 & -\psi_n^k \end{pmatrix}$$

Gaussian elimination

To find J, we can solve

$$(C_n^1 - \exp(2\pi ik/n)I)\mathbf{v} = 0$$

using Gaussian elimination.

$$\begin{pmatrix} -\psi_n^k & 1 & 0 & \dots & 0 \\ 0 & -\psi_n^k & 1 & \dots & 0 \\ 0 & 0 & -\psi_n^k & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & 0 & 0 & \boldsymbol{\psi_n^{n-(n-2)k}} & \boldsymbol{-\psi_n^k} \end{pmatrix}$$

Gaussian elimination

To find J, we can solve

$$(C_n^1 - \exp(2\pi ik/n)I)\mathbf{v} = 0$$

using Gaussian elimination.

$$\begin{pmatrix} -\psi_n^k & 1 & 0 & \dots & 0 \\ 0 & -\psi_n^k & 1 & \dots & 0 \\ 0 & 0 & -\psi_n^k & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & 0 & 0 & \boldsymbol{\psi_n^{2k}} & \boldsymbol{-\psi_n^k} \end{pmatrix}$$

Gaussian elimination

To find J, we can solve

$$(C_n^1 - \exp(2\pi i k/n) I)\mathbf{v} = 0$$

using Gaussian elimination.

$$\begin{pmatrix} -\psi_n^k & 1 & 0 & \dots & 0 \\ 0 & -\psi_n^k & 1 & \dots & 0 \\ 0 & 0 & -\psi_n^k & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Resolving the Eigenvectors

$$\begin{pmatrix} -\psi_n^k & 1 & 0 & \dots & 0 & 0 \\ 0 & -\psi_n^k & 1 & \dots & 0 & 0 \\ 0 & 0 & -\psi_n^k & \ddots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & -\psi_n^k & 1 \\ 0 & 0 & 0 & \dots & 0 & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \dots \\ v_{n-1} \\ v_n \end{pmatrix}$$

Set $v_n = 1$, we can start solving:

$$-\psi_n^k v_{n-1} + 1 = 0 \qquad \Longrightarrow \qquad \psi_n^{-k} = v_{n-1}$$

Resolving the Eigenvectors

$$\begin{pmatrix} -\psi_n^k & 1 & 0 & \dots & 0 & 0 \\ 0 & -\psi_n^k & 1 & \dots & 0 & 0 \\ 0 & 0 & -\psi_n^k & \ddots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & -\psi_n^k & 1 \\ 0 & 0 & 0 & \dots & 0 & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \dots \\ v_{n-1} \\ v_n \end{pmatrix}$$

Set $v_n = 1$, we can start solving:

$$\begin{aligned} -\psi_n^k v_{n-1} + 1 = 0 \qquad & \implies \qquad \psi_n^{-k} = v_{n-1} \\ -\psi_n^k v_{n-2} + v_{n-1} = 0 \qquad & \implies \qquad \psi_n^{-2k} = v_{n-2}\,. \end{aligned}$$

The Fourier Basis

Therefore, the kth eigenvector is given by:

$$\boldsymbol{v}_k = (\psi_n^{n-k}, \psi_n^{n-2k}, \ldots, \psi_n^{k}, 1)^\mathsf{T} \,.$$

The Fourier Basis

Therefore, the kth eigenvector is given by:

$$\boldsymbol{v}_k = (\psi_n^{n-k}, \psi_n^{n-2k}, \ldots, \psi_n^{k}, 1)^{\mathsf{T}}.$$

Let $J = [\boldsymbol{v}_0, \ldots, \boldsymbol{v}_{n-1}]^{\mathsf{T}}$, u an arbitrary vector. Change-of-basis results in **discrete Fourier transform**

$$\begin{aligned}(Ju)_k = \langle v_k, u \rangle &= \sum_t \exp(2\pi i \tfrac{k}{n} \cdot t) u_t \\ &= \sum_t \cos(2\pi \tfrac{k}{n} \cdot t) u_t + i \sin(2\pi \tfrac{k}{n} \cdot t) u_t\end{aligned}$$

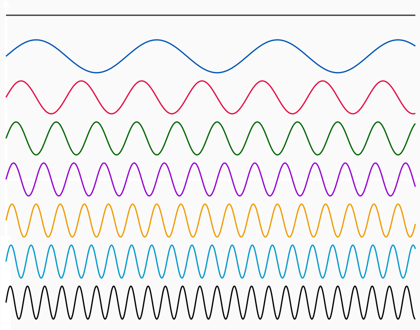


Image: Bronstein et al. (2021)

Convolutions are shift-equivariant

Fourier-transform diagonalizes *all* shifts simultaneously:

$$C_n^1 = J^{-1} D J \implies C_n^k = J^{-1} D^k J$$

Convolutions are shift-equivariant

Fourier-transform diagonalizes *all* shifts simultaneously:

$$C_n^1 = J^{-1} D J \implies C_n^k = J^{-1} D^k J$$

Thus, we can use arbitrary combinations

$$F = \sum \alpha_k C_n^k = J^{-1} (\sum \alpha_k D^k) J$$

$$\begin{pmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \cdots \\ \alpha_{n-1} & \alpha_0 & \alpha_1 & \cdots \\ \alpha_{n-2} & \alpha_{n-1} & \alpha_0 & \cdots \\ \vdots & \vdots & \vdots & \ddots \\ \alpha_1 & \alpha_2 & \alpha_3 & \cdots \end{pmatrix}$$

General convolution.

Convolutions are shift-equivariant

Fourier-transform diagonalizes *all* shifts simultaneously:

$$C_n^1 = J^{-1} D J \implies C_n^k = J^{-1} D^k J$$

Thus, we can use arbitrary combinations

$$F = \sum \alpha_k C_n^k = J^{-1} (\sum \alpha_k D^k) J$$

Then

$$F C_n^1 \boldsymbol{x} = C_n^1 F \boldsymbol{x} \,,$$

so *any* convolution commutes with the shift operator.

$$\begin{pmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \dots \\ \alpha_{n-1} & \alpha_0 & \alpha_1 & \dots \\ \alpha_{n-2} & \alpha_{n-1} & \alpha_0 & \dots \\ \vdots & \vdots & \vdots & \ddots \\ \alpha_1 & \alpha_2 & \alpha_3 & \dots \end{pmatrix}$$

General convolution.

Shift-equivariant functions are convolutions

Assume the converse; for matrix F:

$$\forall \boldsymbol{x} : FC_n^1\boldsymbol{x} = C_n^1F\boldsymbol{x} \Leftrightarrow FJ^{-1}DJ\boldsymbol{x} = J^{-1}DJF\boldsymbol{x}$$

Shift-equivariant functions are convolutions

Assume the converse; for matrix F:

$$\forall \boldsymbol{x} : FC_n^1\boldsymbol{x} = C_n^1F\boldsymbol{x} \Leftrightarrow FJ^{-1}DJ\boldsymbol{x} = J^{-1}DJF\boldsymbol{x}$$

Set $F = J^{-1}F'J$, and $\boldsymbol{x} = J^{-1}\boldsymbol{y}$, then

$$\forall \boldsymbol{x} : J^{-1}F'DJ\boldsymbol{x} = J^{-1}DFJ\boldsymbol{x} \Leftrightarrow \forall \boldsymbol{y} : F'D\boldsymbol{y} = DF\boldsymbol{y}$$

Shift-equivariant functions are convolutions

Assume the converse; for matrix F:

$$\forall \boldsymbol{x} : FC_n^1 \boldsymbol{x} = C_n^1 F \boldsymbol{x} \Leftrightarrow FJ^{-1}DJ\boldsymbol{x} = J^{-1}DJF\boldsymbol{x}$$

Set $F = J^{-1}F'J$, and $\boldsymbol{x} = J^{-1}\boldsymbol{y}$, then

$$\forall \boldsymbol{x} : J^{-1}F'DJ\boldsymbol{x} = J^{-1}DFJ\boldsymbol{x} \Leftrightarrow \forall \boldsymbol{y} : F'D\boldsymbol{y} = DF\boldsymbol{y}$$

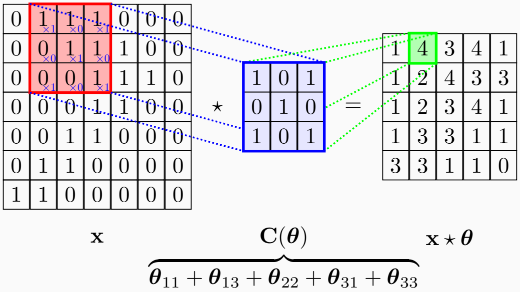
Pick $\boldsymbol{x} = \hat{e}_1$:

$$F'(d_{11}, 0, \ldots)^{\mathsf{T}} = D(f_{11}, f_{21}, \ldots)^{\mathsf{T}} \Leftrightarrow (f_{11}d_{11}, 0 \ldots)^{\mathsf{T}} = (d_{11}f_{11}, d_{22}f_{21}, \ldots)^{\mathsf{T}} \tag{1}$$

Thus, equality requires $f_{21}, f_{32} \ldots = 0$

Every convolution is shift-equivariant, and any shift-equivariant linear function is a convolution

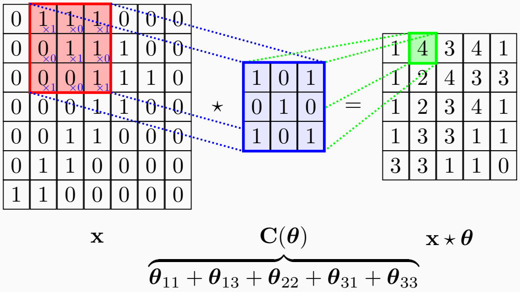
Recap



Interpretation: Given some filter F, we apply *all* possible group-elements (shift) and record all corresponding responses.

Image: Bronstein et al. (2021)

Recap



Interpretation: Given some filter F, we apply *all* possible group-elements (shift) and record all corresponding responses.

Can we generalize this to arbitrary groups?

Image: Bronstein et al. (2021)

Part VIII: Steerable Kernels

A first construction for an equivariant linear layer

Standard linear layer

Select a filter $f \in \mathcal{X}(\Omega, \mathcal{C})$, output is given by inner product

$$f(x) = \int_{\Omega} \langle f(\omega), x(\omega) \rangle \mathrm{d}\omega \, .$$

Stacking c filters defines the layer with c output channels.

A first construction for an equivariant linear layer

Standard linear layer

Select a filter $f \in \mathcal{X}(\Omega, \mathcal{C})$, output is given by inner product

$$f(x) = \int_{\Omega} \langle f(\omega), x(\omega) \rangle \mathrm{d}\omega \,.$$

Stacking c filters defines the layer with c output channels.

Equivariant linear layer

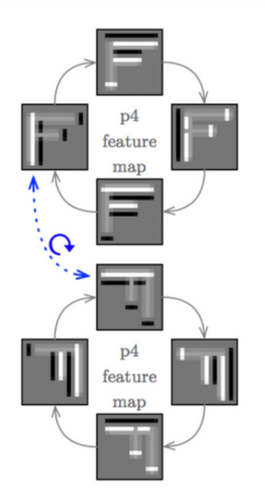
Let R be a representation of the action of G on $\mathcal{X}$. Then, for any filter f and group element g, define

$$f_g(x) = \int_{\Omega} \langle g.f(\omega), x(\omega) \rangle \mathrm{d}\omega \,.$$

Stack all such activations, resulting in $|G|c$ output channels.

Example: P4 symmetry

- 90° rotations, translations
- rotated input rotated in rotated output, and **cyclic shift** in feature maps
- translated input results in translated output



The resulting operation is equivariant

Let $h \in G$ be a group element. Then, for a single filter f we have

$$f_g(h.x) = \int_\Omega \langle g.f(\omega), h.x(\omega) \rangle \mathrm{d}\omega$$

The resulting operation is equivariant

Let $h \in G$ be a group element. Then, for a single filter f we have

$$\begin{aligned} f_g(h.x) &= \int_\Omega \langle g.f(\omega), h.x(\omega)\rangle \mathrm{d}\omega \\ &= \int_\Omega \langle f(g^{-1}\omega), x(h^{-1}\omega)\rangle \mathrm{d}\omega \end{aligned}$$

The resulting operation is equivariant

Let $h \in G$ be a group element. Then, for a single filter f we have

$$
\begin{aligned}
f_g(h.x) &= \int_\Omega \langle g.f(\omega), h.x(\omega)\rangle \mathrm{d}\omega \\
&= \int_\Omega \langle f(g^{-1}\omega), x(h^{-1}\omega)\rangle \mathrm{d}\omega \\
&= \int_\Omega \langle f(g^{-1}h\omega'), x(\omega')\rangle \mathrm{d}\omega'
\end{aligned}
$$

- Change of variables $\omega \to h^{-1}\omega =: \omega'$ assumes $\mathrm{d}\omega = \mathrm{d}\omega'$ (otherwise, we shouldn't call G a symmetry of Ω).

The resulting operation is equivariant

Let $h \in G$ be a group element. Then, for a single filter f we have

$$\begin{aligned} f_g(h.x) &= \int_\Omega \langle g.f(\omega), h.x(\omega)\rangle \mathrm{d}\omega \\ &= \int_\Omega \langle f(g^{-1}\omega), x(h^{-1}\omega)\rangle \mathrm{d}\omega \\ &= \int_\Omega \langle f(g^{-1}h\omega'), x(\omega')\rangle \mathrm{d}\omega' \\ &= \int_\Omega \langle (g^{-1}h)^{-1}.f(\omega'), x(\omega')\rangle \mathrm{d}\omega' \end{aligned}$$

- Change of variables $\omega \to h^{-1}\omega =: \omega'$ assumes $\mathrm{d}\omega = \mathrm{d}\omega'$ (otherwise, we shouldn't call G a symmetry of Ω).

The resulting operation is equivariant

Let $h \in G$ be a group element. Then, for a single filter f we have

$$\begin{aligned} f_g(h.x) &= \int_\Omega \langle g.f(\omega), h.x(\omega)\rangle \mathrm{d}\omega \\ &= \int_\Omega \langle f(g^{-1}\omega), x(h^{-1}\omega)\rangle \mathrm{d}\omega \\ &= \int_\Omega \langle f(g^{-1}h\omega'), x(\omega')\rangle \mathrm{d}\omega' \\ &= \int_\Omega \langle (g^{-1}h)^{-1}.f(\omega'), x(\omega')\rangle \mathrm{d}\omega' \\ &= f_{h^{-1}g}(x) \end{aligned}$$

- Change of variables $\omega \to h^{-1}\omega =: \omega'$ assumes $\mathrm{d}\omega = \mathrm{d}\omega'$ (otherwise, we shouldn't call G a symmetry of Ω).

The resulting operation is equivariant

Let $h \in G$ be a group element. Then, for a single filter f we have

$$
\begin{aligned}
f_g(h.x) &= \int_\Omega \langle g.f(\omega), h.x(\omega) \rangle \mathrm{d}\omega \\
&= \int_\Omega \langle f(g^{-1}\omega), x(h^{-1}\omega) \rangle \mathrm{d}\omega \\
&= \int_\Omega \langle f(g^{-1}h\omega'), x(\omega') \rangle \mathrm{d}\omega' \\
&= \int_\Omega \langle (g^{-1}h)^{-1}.f(\omega'), x(\omega') \rangle \mathrm{d}\omega' \\
&= f_{h^{-1}g}(x)
\end{aligned}
$$

- Change of variables $\omega \to h^{-1}\omega =: \omega'$ assumes $\mathrm{d}\omega = \mathrm{d}\omega'$ (otherwise, we shouldn't call G a symmetry of Ω).
- The mapping $h : G \longrightarrow G$, $g \mapsto hg$ is bijective. Represented by permutation matrix P_h.

The resulting operation is equivariant

Let $h \in G$ be a group element. Then, for a single filter f we have

$$
\begin{aligned}
f_g(h.x) &= \int_\Omega \langle g.f(\omega), h.x(\omega)\rangle \mathrm{d}\omega \\
&= \int_\Omega \langle f(g^{-1}\omega), x(h^{-1}\omega)\rangle \mathrm{d}\omega \\
&= \int_\Omega \langle f(g^{-1}h\omega'), x(\omega')\rangle \mathrm{d}\omega' \\
&= \int_\Omega \langle (g^{-1}h)^{-1}.f(\omega'), x(\omega')\rangle \mathrm{d}\omega' \\
&= f_{h^{-1}g}(x)
\end{aligned}
$$

- Change of variables $\omega \to h^{-1}\omega =: \omega'$ assumes $\mathrm{d}\omega = \mathrm{d}\omega'$ (otherwise, we shouldn't call G a symmetry of Ω).
- The mapping $h : G \longrightarrow G$, $g \mapsto hg$ is bijective. Represented by permutation matrix P_h.

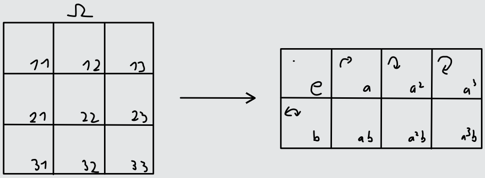
$\implies$ Collect $\mathbf{f}_G(x) = (f_g(x))_{g \in G} \in \mathbb{R}^{|G|}$, then $\mathbf{f}_G(h.x) = P_h^{-1}\mathbf{f}_G(x)$.

$\implies$ $\mathbf{f}_G$ transforms according to the regular representation.

Group-convolution maps signals to signals over the group itself

$$\mathbf{f}_G : \mathcal{X}(\Omega, \mathcal{C}) \longrightarrow \mathcal{X}(G, \mathcal{C})$$

D4-Convolution



T2-Convolution

Original (image) domain: $\Omega \cong (\mathbb{Z}/n\mathbb{Z})^2$

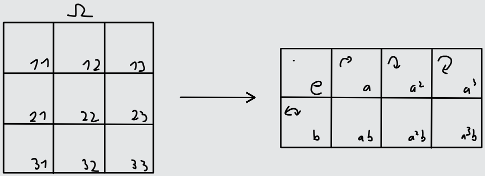
Translation group:

$T_2(n) \cong C(n)^2 \cong (\mathbb{Z}/n\mathbb{Z})^2$

Group-convolution maps signals to signals over the group itself

$$\mathbf{f}_G : \mathcal{X}(\Omega, \mathcal{C}) \longrightarrow \mathcal{X}(G, \mathcal{C})$$

D4-Convolution



T2-Convolution

Original (image) domain: $\Omega \cong (\mathbb{Z}/n\mathbb{Z})^2$
Translation group:
$T_2(n) \cong C(n)^2 \cong (\mathbb{Z}/n\mathbb{Z})^2$

This construction requires the output to have $c|G|$ channels, which becomes infeasible for large groups.

Steerable Kernels

Steerable Mapping (Intertwiner)

Let G be a group, Ω, Ψ two geometric domains. A linear mapping L from $\mathcal{X}(\Omega, \mathcal{C})$ to $\mathcal{X}(\Omega', \mathcal{C}')$ is called (linearly) **steerable** if there exist group actions (representations) μ, ν such that

$$\forall x \in \mathcal{X}(\Omega, \mathcal{C}), g \in G : \; \nu(g).L(x) = L(\mu(g).x)$$

Steerable Kernels

Steerable Mapping (Intertwiner)

Let G be a group, Ω, Ψ two geometric domains. A linear mapping L from $\mathcal{X}(\Omega, \mathcal{C})$ to $\mathcal{X}(\Omega', \mathcal{C}')$ is called (linearly) **steerable** if there exist group actions (representations) μ, ν such that

$$\forall x \in \mathcal{X}(\Omega, \mathcal{C}), g \in G: \; \nu(g).L(x) = L(\mu(g).x)$$

$$\begin{tikzcd}
\boxed{\mathcal{X}(\Omega, \mathcal{C})} \arrow[r, "\mu(g)"] \arrow[d, "L"] & \boxed{\mathcal{X}(\Omega, \mathcal{C})} \arrow[d, "L"] \\
\boxed{\mathcal{X}(\Omega', \mathcal{C}')} \arrow[r, "\nu(g)"] & \boxed{\mathcal{X}(\Omega', \mathcal{C}')}
\end{tikzcd}$$

Steerable Kernels

Steerable Mapping (Intertwiner)

Let G be a group, Ω, Ψ two geometric domains. A linear mapping L from $\mathcal{X}(\Omega, \mathcal{C})$ to $\mathcal{X}(\Omega', \mathcal{C}')$ is called (linearly) **steerable** if there exist group actions (representations) μ, ν such that

$$\forall x \in \mathcal{X}(\Omega, \mathcal{C}), g \in G: \ \nu(g).L(x) = L(\mu(g).x)$$

$$\begin{tikzcd}
\boxed{\mathcal{X}(\Omega, \mathcal{C})} \arrow[r, "\mu(g)"] \arrow[d, "L"] & \boxed{\mathcal{X}(\Omega, \mathcal{C})} \arrow[d, "L"] \\
\boxed{\mathcal{X}(\Omega', \mathcal{C}')} \arrow[r, "\nu(g)"] & \boxed{\mathcal{X}(\Omega', \mathcal{C}')}
\end{tikzcd}$$

Note: If ν is the trivial representation, then L is *invariant* ($\nu(g) = \mathrm{id}$).

Steerable Kernels

Steerable Mapping (Intertwiner)

Let G be a group, Ω, Ψ two geometric domains. A linear mapping L from $\mathcal{X}(\Omega, \mathcal{C})$ to $\mathcal{X}(\Omega', \mathcal{C}')$ is called (linearly) **steerable** if there exist group actions (representations) μ, ν such that

$$\forall x \in \mathcal{X}(\Omega, \mathcal{C}), g \in G : \; \nu(g).L(x) = L(\mu(g).x)$$

$$\begin{tikzcd}
\boxed{\mathcal{X}(\Omega, \mathcal{C})} \arrow[r, "\mu(g)"] \arrow[dr, "L"'] & \boxed{\mathcal{X}(\Omega, \mathcal{C})} \arrow[d, "L"] \\
 & \boxed{\mathcal{X}(\Omega', \mathcal{C}')}
\end{tikzcd}$$

Note: If ν is the trivial representation, then L is *invariant* ($\nu(g) = \mathrm{id}$).

Equivariance implies invariant mapping

Invariance of weights

For $\nu(g) \in \mathrm{GL}(m)$, $\mu(g) \in \mathrm{GL}(n)$

$$
\begin{aligned}
& \forall g, x : \; \nu(g).L(x) = L(\mu(g).x) \\
\Leftrightarrow\; & \forall g : \; \nu(g) \circ L = L \circ \mu(g)
\end{aligned}
$$

Equivariance implies invariant mapping

Invariance of weights

For $\nu(g) \in \mathrm{GL}(m)$, $\mu(g) \in \mathrm{GL}(n)$

$$\begin{aligned}
&\forall g, x : \nu(g).L(x) = L(\mu(g).x) \\
\Leftrightarrow\ &\forall g : \nu(g) \circ L = L \circ \mu(g) \\
\Leftrightarrow\ &\forall g : L = \nu(g)^{-1} \circ L \circ \mu(g)
\end{aligned}$$

Equivariance implies invariant mapping

Invariance of weights

For $\nu(g) \in \mathrm{GL}(m)$, $\mu(g) \in \mathrm{GL}(n)$

$$\begin{aligned} &\forall g, x : \ \nu(g).L(x) = L(\mu(g).x) \\ \Leftrightarrow &\forall g : \ \nu(g) \circ L = L \circ \mu(g) \\ \Leftrightarrow &\forall g : \ L = \nu(g)^{-1} \circ L \circ \mu(g) \end{aligned}$$

The weight matrix L is invariant under this particular group action.

Equivariance implies invariant mapping

Invariance of weights

For $\nu(g) \in \mathrm{GL}(m)$, $\mu(g) \in \mathrm{GL}(n)$

$$\begin{aligned} &\forall g, x : \ \nu(g).L(x) = L(\mu(g).x) \\ \Leftrightarrow &\forall g : \ \nu(g) \circ L = L \circ \mu(g) \\ \Leftrightarrow &\forall g : \ L = \nu(g)^{-1} \circ L \circ \mu(g) \end{aligned}$$

Group averaging

Can reuse our group-averaging to construct equivariant weights from arbitrary weights $\tilde{L}$:

$$L = \sum_{g \in G} \nu(g)^{-1} \, \tilde{L} \, \mu(g)$$

Convolutions through weight-sharing

Let $\mu(k) = C_3^k, \nu(k) = C_3^k$, select $\tilde{L} = \delta_{11}$.
Group action:

$$(C_3^1)^{-1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} C_3^1$$

Convolutions through weight-sharing

Let $\mu(k) = C_3^k, \nu(k) = C_3^k$, select $\tilde{L} = \delta_{11}$.

Group action:

$$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

Convolutions through weight-sharing

Let $\mu(k) = C_3^k, \nu(k) = C_3^k$, select $\tilde{L} = \delta_{11}$.

Group action:

$$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \color{red}{1} & \color{red}{0} & \color{red}{0} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & \color{red}{1} & 0 \\ 0 & \color{red}{0} & 1 \\ 1 & \color{red}{0} & 0 \end{pmatrix}$$

Convolutions through weight-sharing

Let $\mu(k) = C_3^k, \nu(k) = C_3^k$, select $\tilde{L} = \delta_{11}$.

Group action:

$$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & {\color{red}1} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Convolutions through weight-sharing

Let $\mu(k) = C_3^k, \nu(k) = C_3^k$, select $\tilde{L} = \delta_{11}$.

Group action:

$$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Convolutions through weight-sharing

Let $\mu(k) = C_3^k, \nu(k) = C_3^k$, select $\tilde{L} = \delta_{11}$.
Group action:

$$(C_3^1)^{-1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} C_3^1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

In general: Move to the right by k places, then move down k places.

Convolutions through weight-sharing

In total, we get the following nine group-averages (color indicates original matrix):

$$\begin{pmatrix} \color{blue}{1} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} 0 & \color{orange}{1} & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \qquad \begin{pmatrix} 0 & 0 & \color{red}{1} \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 0 & 1 \\ \color{red}{1} & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 & 0 \\ 0 & \color{blue}{1} & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & \color{orange}{1} \\ 1 & 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ \color{orange}{1} & 0 & 0 \end{pmatrix} \qquad \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & \color{red}{1} & 0 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \color{blue}{1} \end{pmatrix}$$

Convolutions through weight-sharing

In total, we get the following nine group-averages (color indicates original matrix):

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Three distinct matrices $\rightarrow$ three orbits that serve as basis vectors

Steerable Filter Banks

- If L and L' are linearly steerable (w.r.t. μ, ν), then so is $\alpha L + L'$ for $\alpha \in \mathbb{R}$.

Steerable Filter Banks

- If L and L' are linearly steerable (w.r.t. μ, ν), then so is $\alpha L + L'$ for $\alpha \in \mathbb{R}$.

$$
\begin{aligned}
&(\alpha L + L')(\mu(g).x) \\
&\qquad = \alpha L(\mu(g).x) + L'(\mu(g).x)
\end{aligned}
$$

Steerable Filter Banks

- If L and L' are linearly steerable (w.r.t. μ, ν), then so is $\alpha L + L'$ for $\alpha \in \mathbb{R}$.

$$\begin{aligned}(\alpha L + L')(\mu(g).x) \\ &= \alpha L(\mu(g).x) + L'(\mu(g).x) \\ &= \alpha \nu(g).L(x) + \nu(g).L'(x)\end{aligned}$$

Steerable Filter Banks

- If L and L' are linearly steerable (w.r.t. μ, ν), then so is $\alpha L + L'$ for $\alpha \in \mathbb{R}$.

$$
\begin{aligned}
(\alpha L + L')(\mu(g).x) \\
&= \alpha L(\mu(g).x) + L'(\mu(g).x) \\
&= \alpha \nu(g).L(x) + \nu(g).L'(x) \\
&= \nu(g)(\alpha L(x) + L'(x))
\end{aligned}
$$

Steerable Filter Banks

- If L and L' are linearly steerable (w.r.t. μ, ν), then so is $\alpha L + L'$ for $\alpha \in \mathbb{R}$.

$$
\begin{aligned}
(\alpha L + L')(\mu(g).x) \\
&= \alpha L(\mu(g).x) + L'(\mu(g).x) \\
&= \alpha \nu(g).L(x) + \nu(g).L'(x) \\
&= \nu(g)(\alpha L(x) + L'(x)) \\
&= \nu(g)(\alpha L + L')(x)
\end{aligned}
$$

Steerable Filter Banks

- If L and L' are linearly steerable (w.r.t. μ, ν), then so is $\alpha L + L'$ for $\alpha \in \mathbb{R}$.
- $\implies \mathcal{L} := \{L : L \text{ is } \mu, \nu \text{ steerable}\}$ is a vector space.

$$
\begin{aligned}
&(\alpha L + L')(\mu(g).x) \\
&\qquad = \alpha L(\mu(g).x) + L'(\mu(g).x) \\
&\qquad = \alpha \nu(g).L(x) + \nu(g).L'(x) \\
&\qquad = \nu(g)(\alpha L(x) + L'(x)) \\
&\qquad = \nu(g)(\alpha L + L')(x)
\end{aligned}
$$

Steerable Filter Banks

- If L and L' are linearly steerable (w.r.t. μ, ν), then so is $\alpha L + L'$ for $\alpha \in \mathbb{R}$.
- $\implies \mathcal{L} := \{L : L \text{ is } \mu, \nu \text{ steerable}\}$ is a vector space.
- There exists a *basis*, in which we can expand the filters.

$$
\begin{aligned}
&(\alpha L + L')(\mu(g).x) \\
&\qquad = \alpha L(\mu(g).x) + L'(\mu(g).x) \\
&\qquad = \alpha \nu(g).L(x) + \nu(g).L'(x) \\
&\qquad = \nu(g)(\alpha L(x) + L'(x)) \\
&\qquad = \nu(g)(\alpha L + L')(x)
\end{aligned}
$$

Steerable Filter Banks

- If L and L' are linearly steerable (w.r.t. μ, ν), then so is $\alpha L + L'$ for $\alpha \in \mathbb{R}$.
- $\implies \mathcal{L} := \{L : L \text{ is } \mu, \nu \text{ steerable}\}$ is a vector space.
- There exists a *basis*, in which we can expand the filters.

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

$$\begin{aligned}
&(\alpha L + L')(\mu(g).x) \\
&\qquad = \alpha L(\mu(g).x) + L'(\mu(g).x) \\
&\qquad = \alpha \nu(g).L(x) + \nu(g).L'(x) \\
&\qquad = \nu(g)(\alpha L(x) + L'(x)) \\
&\qquad = \nu(g)(\alpha L + L')(x)
\end{aligned}$$

Steerable Filter Banks

- If L and L' are linearly steerable (w.r.t. μ, ν), then so is $\alpha L + L'$ for $\alpha \in \mathbb{R}$.
- $\implies \mathcal{L} := \{L : L \text{ is } \mu, \nu \text{ steerable}\}$ is a vector space.
- There exists a *basis*, in which we can expand the filters.

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

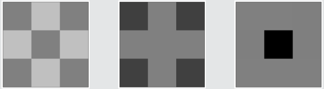
$$\begin{aligned} &(\alpha L + L')(\mu(g).x) \\ &\quad = \alpha L(\mu(g).x) + L'(\mu(g).x) \\ &\quad = \alpha \nu(g).L(x) + \nu(g).L'(x) \\ &\quad = \nu(g)(\alpha L(x) + L'(x)) \\ &\quad = \nu(g)(\alpha L + L')(x) \end{aligned}$$

In general, generate all orbits and orthogonalize (e.g., Gram-Schmidt)

The same strategy also works for trivial (invariant) representation of D4

Trivial Representation A1

We have already seen this: rotation-invariant linear functions

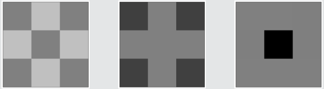


What about non-trivial representations?

The same strategy also works for trivial (invariant) representation of D4

Trivial Representation A1

We have already seen this: rotation-invariant linear functions



What about non-trivial representations?

Constructing a basis for D4's B1 representation

B1

a: rotate 90°, b: flip along y-axis.

B1 is a 1-dimensional representation with

$$\nu(a) = -1, \quad \nu(b) = 1$$

Smoothing

Constructing a basis for D4's B1 representation

B1

a: rotate 90°, b: flip along y-axis.

B1 is a 1-dimensional representation with

$$\nu(a) = -1, \quad \nu(b) = 1$$

Smoothing

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}}_{E_{11}} + \underbrace{(-1)}_{\nu(a)} \cdot \underbrace{\begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}}_{R(a)E_{11}}$$

$$+ \underbrace{(1)}_{\nu(a^2)} \cdot \underbrace{\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{R(a^2)E_{11}} + \dots$$

Constructing a basis for D4's B1 representation

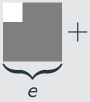
B1

a: rotate 90°, b: flip along y-axis.

B1 is a 1-dimensional representation with

$$\nu(a) = -1, \quad \nu(b) = 1$$

Smoothing



Constructing a basis for D4's B1 representation

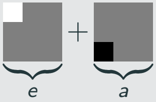
B1

a: rotate 90°, *b*: flip along y-axis.

B1 is a 1-dimensional representation with

$$\nu(a) = -1, \quad \nu(b) = 1$$

Smoothing



Constructing a basis for D4's B1 representation

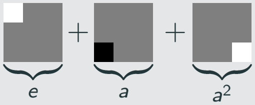
B1

a: rotate 90°, b: flip along y-axis.

B1 is a 1-dimensional representation with

$$\nu(a) = -1, \quad \nu(b) = 1$$

Smoothing



Constructing a basis for D4's B1 representation

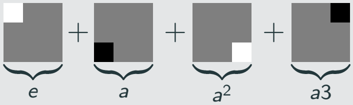
B1

a: rotate 90°, b: flip along y-axis.

B1 is a 1-dimensional representation with

$$\nu(a) = -1, \quad \nu(b) = 1$$

Smoothing



Constructing a basis for D4's B1 representation

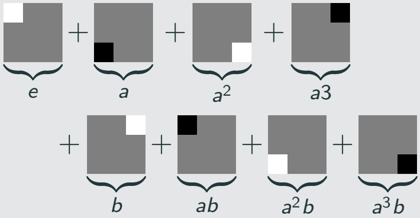
B1

a: rotate 90°, b: flip along y-axis.

B1 is a 1-dimensional representation with

$$\nu(a) = -1, \quad \nu(b) = 1$$

Smoothing



Constructing a basis for D4’s B1 representation

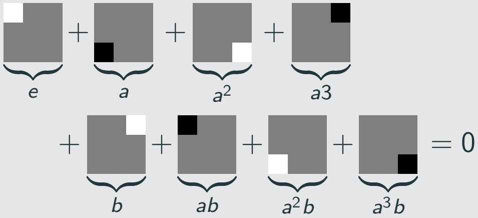
B1

a: rotate 90°, b: flip along y-axis.

B1 is a 1-dimensional representation with

$$\nu(a) = -1, \quad \nu(b) = 1$$

Smoothing



Constructing a basis for D4’s B1 representation

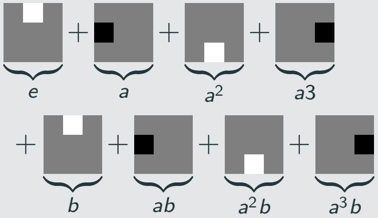
B1

a: rotate 90°, b: flip along y-axis.

B1 is a 1-dimensional representation with

$$\nu(a) = -1, \quad \nu(b) = 1$$

Smoothing



Constructing a basis for D4's B1 representation

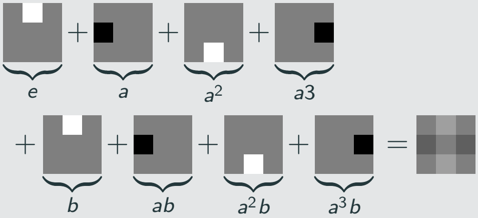
B1

a: rotate 90°, b: flip along y-axis.

B1 is a 1-dimensional representation with

$$\nu(a) = -1, \quad \nu(b) = 1$$

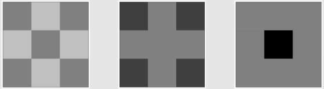
Smoothing



One-dimensional representations of D4

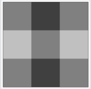
Trivial Representation A1

We have already seen this: rotation-invariant linear functions



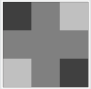
B1

Rotation sign, reflection inv:



B2

Rotation, reflection sign:

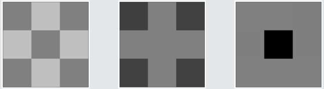


T. S. Cohen and Welling. (2016). *Steerable CNNs*

One-dimensional representations of D4

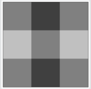
Trivial Representation A1

We have already seen this: rotation-invariant linear functions



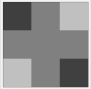
B1

Rotation sign, reflection inv:



B2

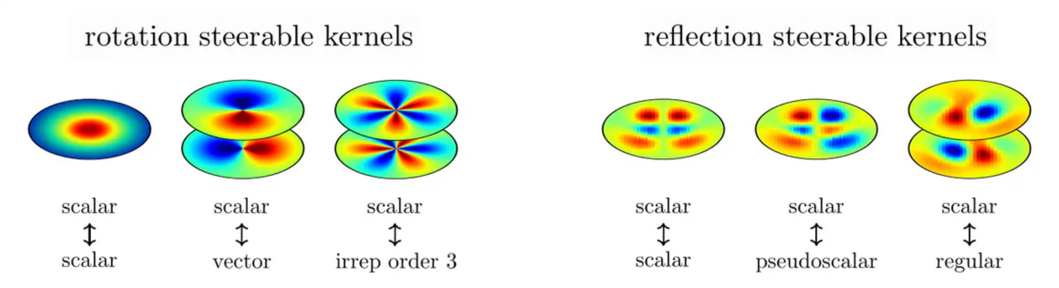
Rotation, reflection sign:



What about A2? (rotation invariant, reflection sign-change)

T. S. Cohen and Welling. (2016). *Steerable CNNs*

Steerable kernels for SO(2)



Homogenous spaces

Definition (Transitive Action)

A group action is called **transitive** iff

$$\forall x, y \in \Omega : \exists g \in G \text{ s. t. } x = gy\,.$$

T. Cohen, Geiger, and Weiler. (2019). "A general theory of equivariant cnns on homogeneous spaces"

Homogenous spaces

Definition (Transitive Action)

A group action is called **transitive** iff

$$\forall x, y \in \Omega : \exists g \in G \text{ s.t. } x = gy \,.$$

Definition (Homogenous space)

A geometric domain Ω with a transitive group action is called a **homogenous space**.

T. Cohen, Geiger, and Weiler. (2019). "A general theory of equivariant cnns on homogeneous spaces"

Homogenous spaces

Definition (Transitive Action)

A group action is called **transitive** iff

$$\forall x, y \in \Omega : \exists g \in G \text{ s.t. } x = gy \,.$$

Definition (Homogenous space)

A geometric domain Ω with a transitive group action is called a **homogenous space**.

Theorem (Convolutions are all you need)

An equivariant map can always be written as a convolution-like integral

T. Cohen, Geiger, and Weiler. (2019). "A general theory of equivariant cnns on homogeneous spaces"

Part IX: Scale Separation

Can we just use any convolution operation?

Parameter Cost

A generic convolution on an $n \times n$ image with c input and d output channels requires

$$c \times n \times n \times d \text{ parameters}.$$

Can we just use any convolution operation?

Parameter Cost

A generic convolution on an $n \times n$ image with c input and d output channels requires

$$c \times n \times n \times d \text{ parameters}\,.$$

Scales quadratically with input size.

Group-smoothing

For a single linear layer, a "global" convolution is just *group smoothing*.

Downsampling

Rescale the input to be small enough that we can easily handle it.

Loses detail information

Rigid

Local Patches

Use very sparse filter matrices so that only local neighbourhoods interact.

Can we just use any convolution operation?

Parameter Cost

A generic convolution on an $n \times n$ image with c input and d output channels requires

$$c \times n \times n \times d \text{ parameters}.$$

Scales quadratically with input size.

Group-smoothing

For a single linear layer, a "global" convolution is just *group smoothing*.

Downsampling

Rescale the input to be small enough that we can easily handle it.

Loses detail information

Rigid

Local Patches

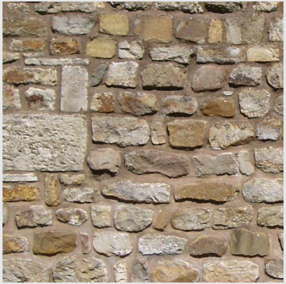
Use very sparse filter matrices so that only local neighbourhoods interact.

Cannot capture global structure

Scale separation prior

Sometimes, coarse scales are sufficient


Sometimes, local patches are sufficient



Deformation Stability

- Let G be translations, a subgroup of the diffeomorphisms $\tau : \Omega \longrightarrow \Omega$.
- Define $c(\tau) := \sup \|\nabla \cdot \tau(u)\|$. For $g \in G$ we have $c(g) = 0$; c measures the distance to the group
- We want **deformation stability**: $\|f(x) - f(\tau.x)\| \leq c(\tau)$.

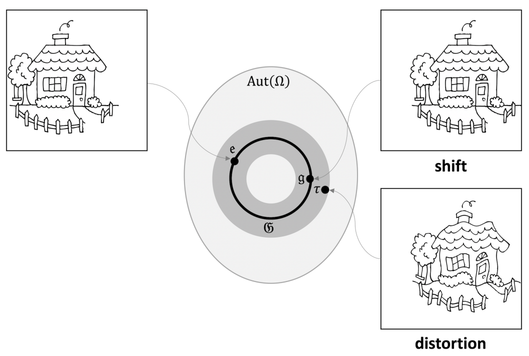
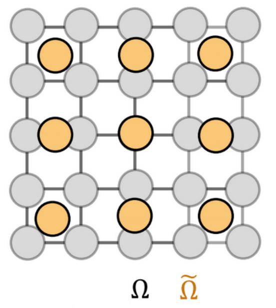


Image: Bronstein et al. (2021)

Multiresolution analysis

- Decompose signal $x \in \mathcal{X}(\Omega)$ into a lower resolution version $\tilde{x} \in \mathcal{X}(\tilde{\Omega})$ and residuals.



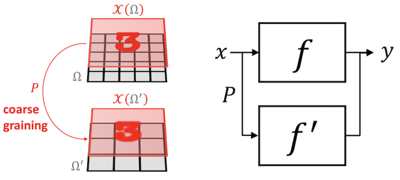
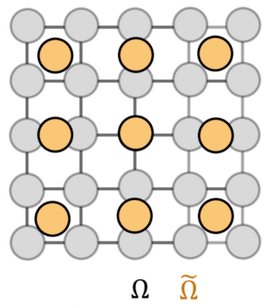
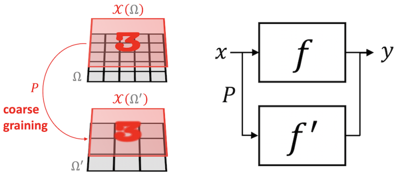


Image: Bronstein et al. (2021),

Multiresolution analysis

- Decompose signal $x \in \mathcal{X}(\Omega)$ into a lower resolution version $\tilde{x} \in \mathcal{X}(\tilde{\Omega})$ and residuals.
- (Wavelet) filters localized in space





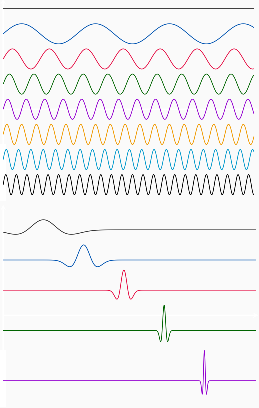
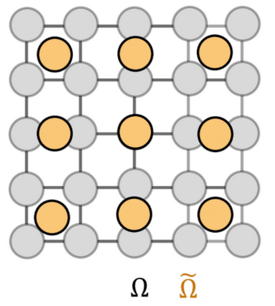
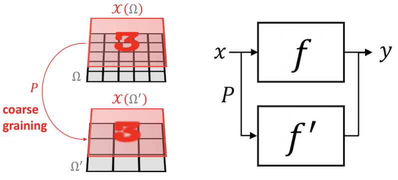


Image: Bronstein et al. (2021),

Multiresolution analysis

- Decompose signal $x \in \mathcal{X}(\Omega)$ into a lower resolution version $\tilde{x} \in \mathcal{X}(\tilde{\Omega})$ and residuals.
- (Wavelet) filters localized in space
- Exhibit deformation stability[1]





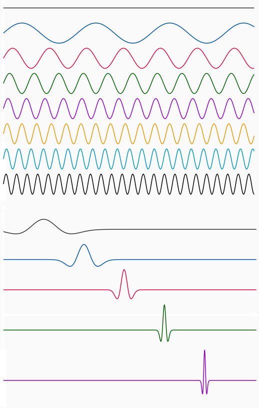


Image: Bronstein et al. (2021), [1]Mallat. (2010)

Compositionality

- The *deep* in deep learning
- Multiple layers of *local* connectivity can accumulate a *global* receptive field
- $\Rightarrow$ **The GDL Blueprint**

Image: Zeiler. (2014)

Part X: The GDL Blueprint

The GDL Blueprint

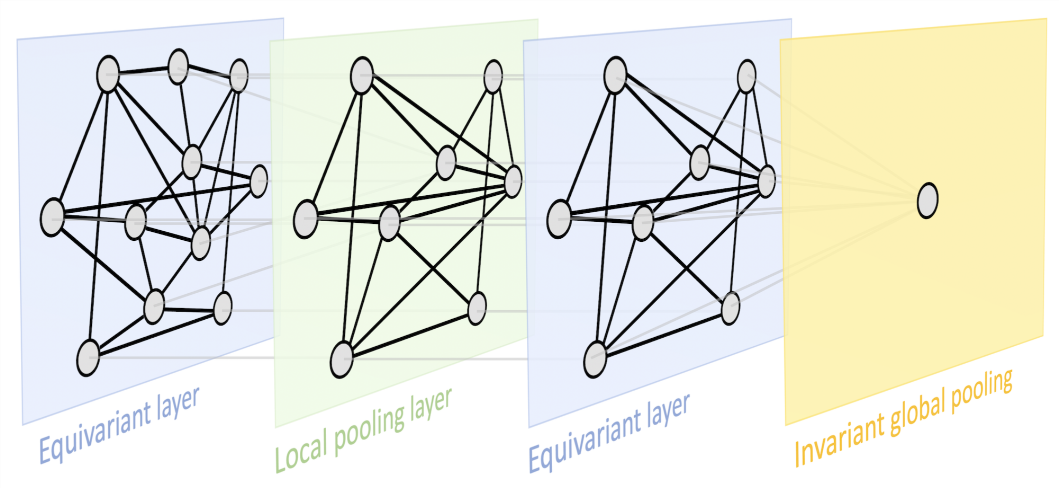


Image: Bronstein et al. (2021)

The GDL Blueprint

- Linear G-equivariant layer
  $B : \mathcal{X}(\Omega, \mathcal{C}) \longrightarrow \mathcal{X}(\Omega', \mathcal{C}')$ s.t.
  $g.B(x) = B(g.x)$
- Pointwise non-linearity
- Local pooling layer (coarsening)
  $P : \mathcal{X}(\Omega, \mathcal{C}) \longrightarrow \mathcal{X}(\Omega', \mathcal{C})$ s.t. $\Omega' \subset \Omega$
- G-invariant layer (global pooling)
  $A : \mathcal{X}(\Omega, \mathcal{C}) \longrightarrow \mathcal{Y}$ s.t. $A(x) = A(g.x)$

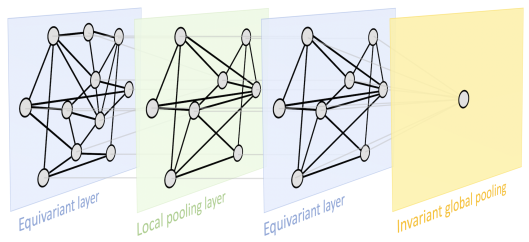


Image: Bronstein et al. (2021)

Many famous architectures follow this blueprint

Architecture	**Domain** Ω	**Group** G
CNN	Grid	Translation
LSTM	1D-Grid	Time translation
Spherical CNN	Sphere	Rotation SO(3)
Mesh CNN	Manifold	Gauge SO(2)
GNN	Graph	Permutation $S(n)$
Deep Sets	Set	Permutation $S(n)$
Transformer	Complete Graph	Permutation $S(n)$

References

[1] Michael M. Bronstein et al. **Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges.** 2021. arXiv: 2104.13478 [cs.LG]. URL: https://arxiv.org/abs/2104.13478.

[2] Taco Cohen, Mario Geiger, and Maurice Weiler. **"A general theory of equivariant cnns on homogeneous spaces".** In: *Advances in neural information processing systems* 32 (2019).

[3] Taco S. Cohen and Max Welling. **Steerable CNNs.** 2016. arXiv: 1612.08498 [cs.LG]. URL: https://arxiv.org/abs/1612.08498.

[4] Stéphane Mallat. **"Recursive interferometric representation".** In: *Proc. of EUSICO conference, Danemark*. Vol. 3. 2010.

[5] MD Zeiler. **"Visualizing and Understanding Convolutional Networks".** In: