

EECE 576L

Advanced Topics in Deep Learning

Lecture 4: Group Equivariant Deep Learning

Part 3: Steerable kernels

Renjie Liao

University of British Columbia

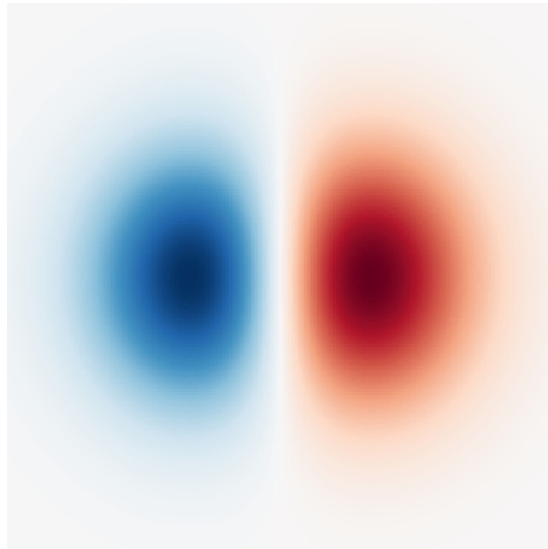
2026/27 Winter Session, Term 1 (Fall 2026)

Outline

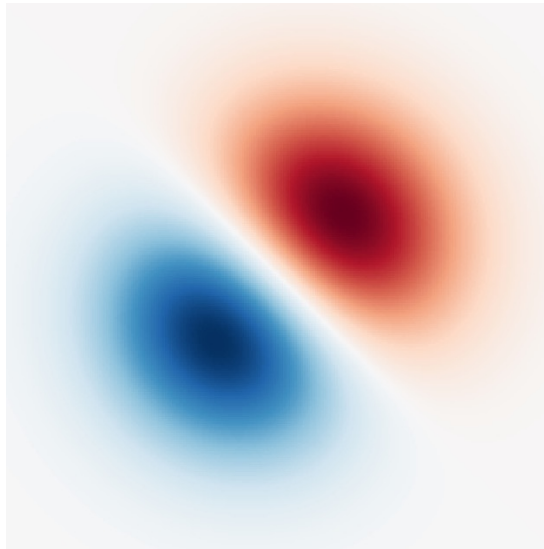
1. Groups and actions
2. Group convolutions
3. Steerable kernels
 - **Steering a filter**
 - Circular harmonics
 - Kernel synthesis and sharing
 - Scalar layers and typed features
4. Typed features and 3D networks
5. Implementation and verification (optional reading)

One Filter, Many Orientations

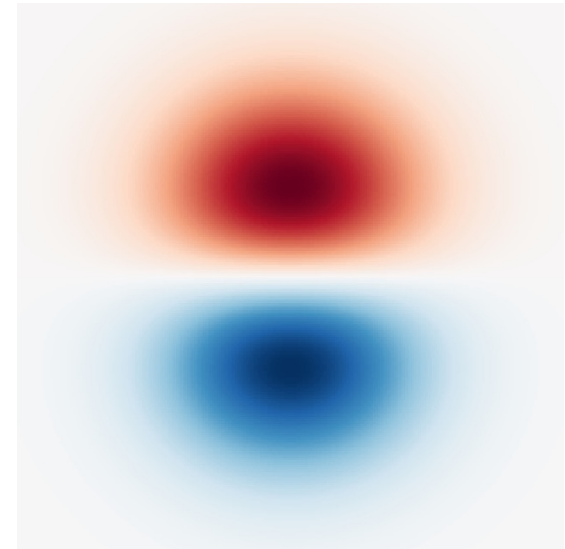
Can we rotate a learned filter without learning a new pixel array at every angle?



0°



45°



90°

Represent the filter in a fixed basis and rotate its coefficients analytically.

Rotation Convention

An active counterclockwise rotation uses an inverse coordinate lookup.

$$[L_\alpha k](x) = k(R_{-\alpha}x)$$

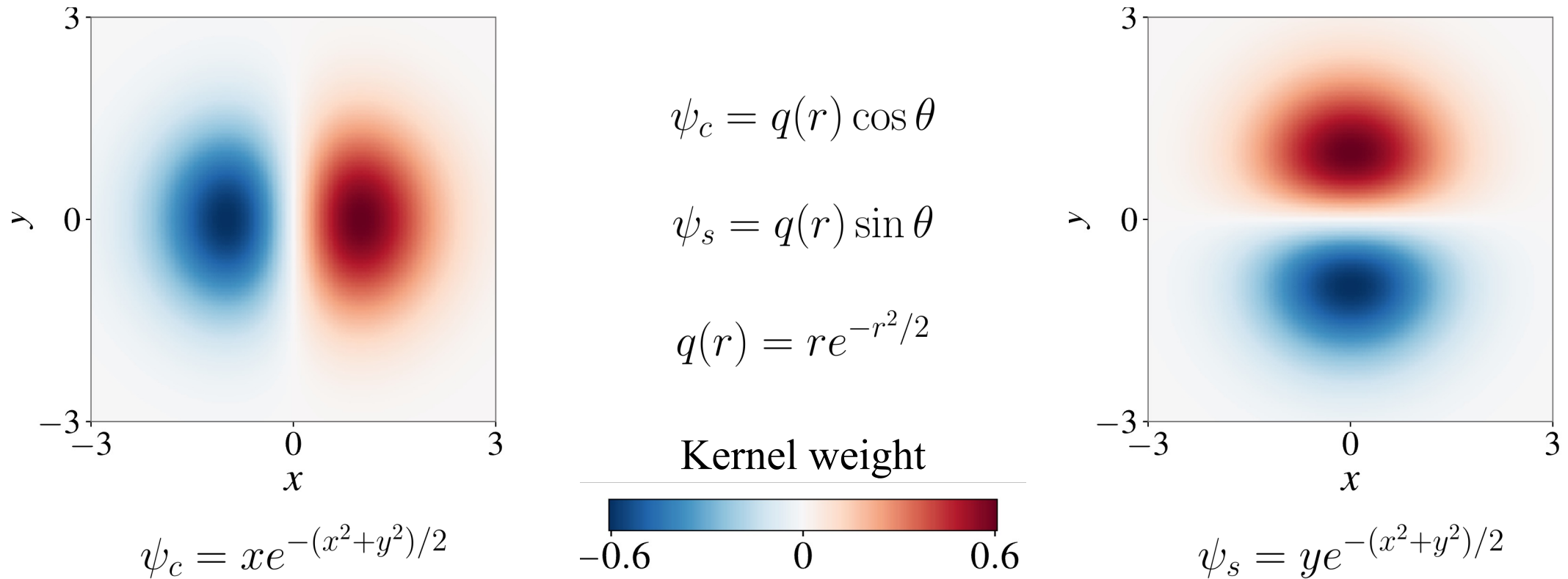
In polar coordinates, the radius is unchanged and the angle shifts.

$$[L_\alpha k](r, \theta) = k(r, \theta - \alpha)$$

The inverse lookup fixes the steering signs.

Two Perpendicular Filters

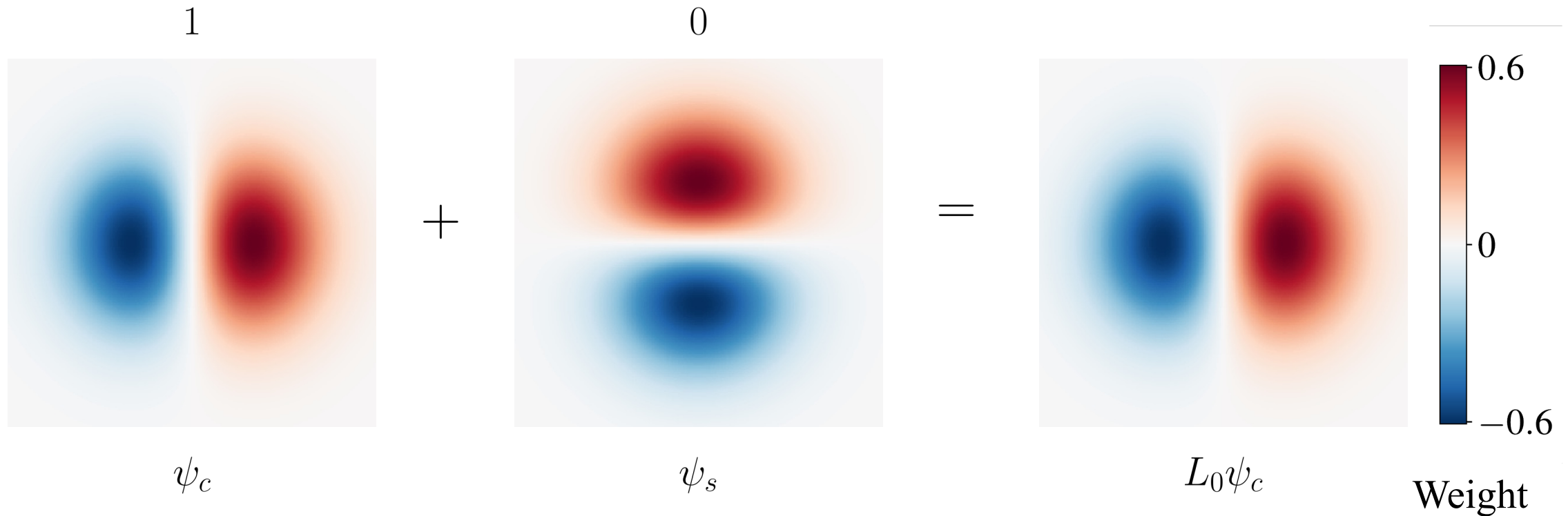
Each panel plots the filter defined below. Color shows the signed kernel weight. Coordinates are spatial offsets from the filter center.



White marks **zero weight**. The central stripes are where the filters change sign.

Steering a Filter: 0°

Start from the horizontal filter.

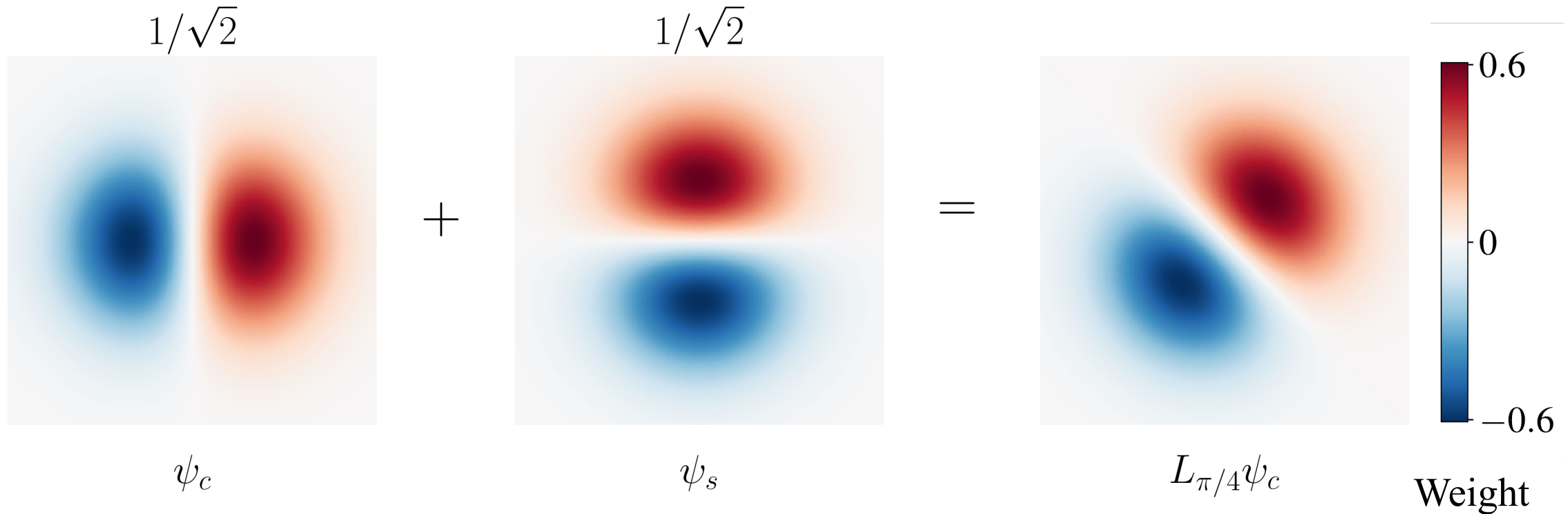


Mix the fixed basis filters.

$$L_0\psi_c = \psi_c$$

Steering a Filter: 45°

A diagonal filter is an equal mixture of the same two basis filters.

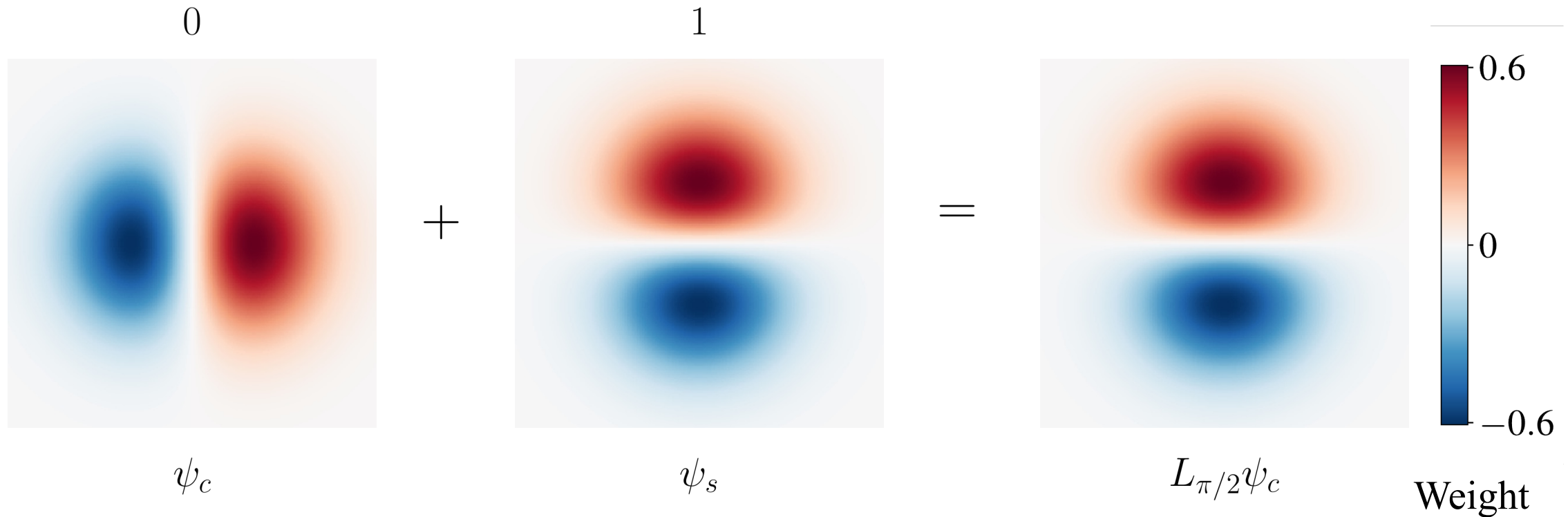


Mix the fixed basis filters.

$$L_{\pi/4}\psi_c = \frac{\psi_c + \psi_s}{\sqrt{2}}$$

Steering a Filter: 90°

The quarter-turn check fixes the sign: horizontal becomes vertical.



Mix the fixed basis filters.

$$L_{\pi/2}\psi_c = \psi_s$$

Steering within a fixed basis

A rotated basis stays in the span of these two filters.

Steering rotates a filter by changing the weights of the same fixed basis.

$$\begin{bmatrix} L_\alpha \psi_c \\ L_\alpha \psi_s \end{bmatrix} = D(\alpha) \begin{bmatrix} \psi_c \\ \psi_s \end{bmatrix}, \quad D(\alpha) = \begin{bmatrix} \cos \alpha & \sin \alpha \\ -\sin \alpha & \cos \alpha \end{bmatrix}$$

Rotation mixes the same two basis filters.

Coefficient Transformation

The learned kernel is a linear combination of those fixed basis functions.

$$k = w^\top \Psi = a\psi_c + b\psi_s$$

$$L_\alpha k = w^\top D(\alpha) \Psi = (D(\alpha)^\top w)^\top \Psi$$

$$\begin{bmatrix} a_\alpha \\ b_\alpha \end{bmatrix} = D(\alpha)^\top \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$$

The transpose comes from opposite sides of the dot product.

Outline

1. Groups and actions
2. Group convolutions
3. Steerable kernels
 - Steering a filter
 - **Circular harmonics**
 - Kernel synthesis and sharing
 - Scalar layers and typed features
4. Typed features and 3D networks
5. Implementation and verification (optional reading)

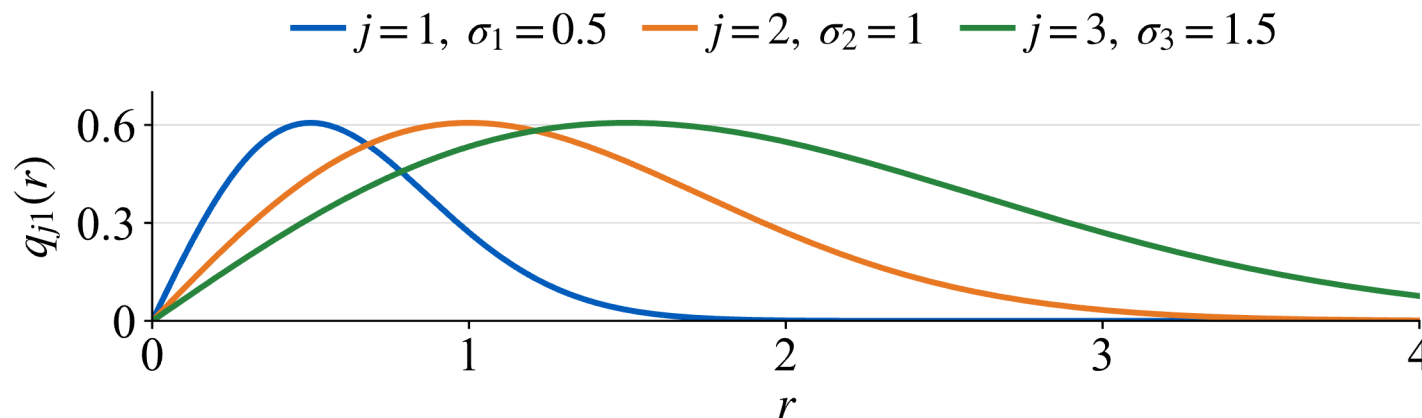
Circular Harmonics: Radial and Angular Parts

A fixed radial profile multiplies an angular cosine–sine pair.

$$\psi_{jm}^c(r, \theta) = q_{jm}(r) \cos(m\theta), \quad \psi_{jm}^s(r, \theta) = q_{jm}(r) \sin(m\theta)$$

One possible radial family:

$$q_{jm}(r) = \left(\frac{r}{\sigma_j} \right)^m \exp \left[-\frac{r^2}{2\sigma_j^2} \right], \quad \sigma_j > 0$$



Radial index

$$j = 1, \dots, J$$

Selects Gaussian width.

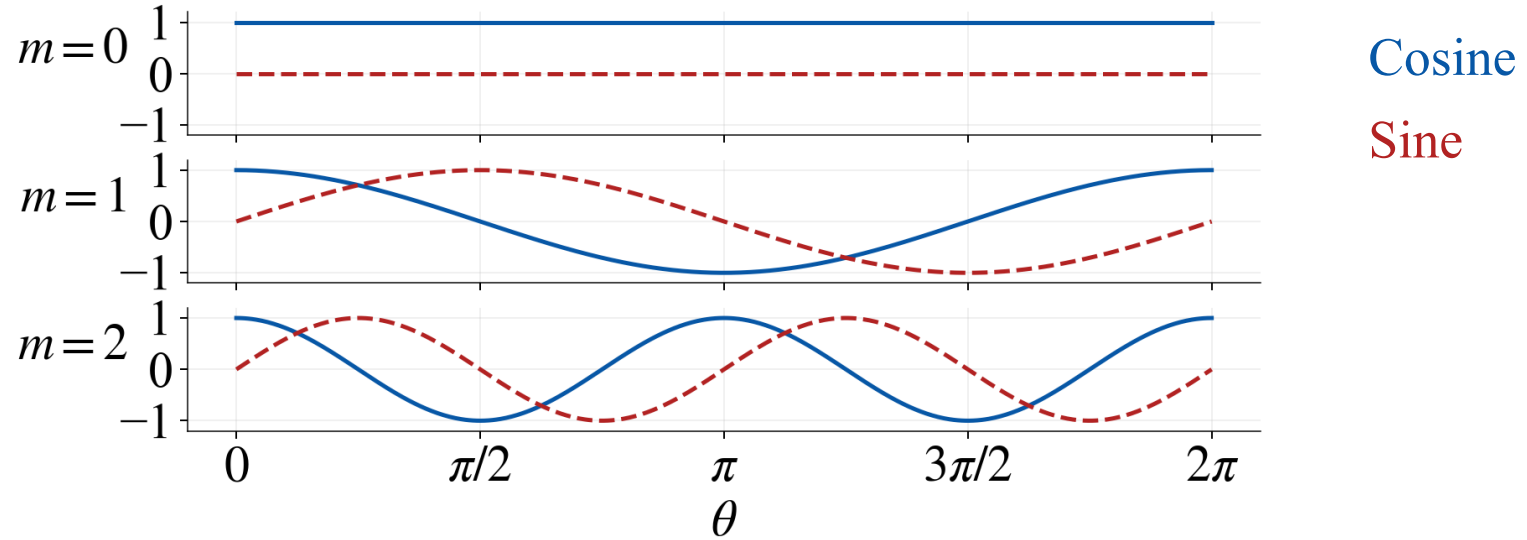
Angular cycles

$$m = 0, \dots, M$$

Frequency Controls the Rotation Rate

One spatial rotation causes a frequency-dependent mixing of each pair.

$$\begin{bmatrix} L_\alpha \psi_{jm}^c \\ L_\alpha \psi_{jm}^s \end{bmatrix} = D(m\alpha) \begin{bmatrix} \psi_{jm}^c \\ \psi_{jm}^s \end{bmatrix}$$



For the derivations, see Note 3, Section 3.3 and its Fourier appendix.

A Finite Kernel Family

Choose radial profiles and a largest angular frequency, then learn the weights.

$$k(r, \theta) = \sum_{j=1}^J a_{j0} q_{j0}(r) + \sum_{m=1}^M \sum_{j=1}^J q_{jm}(r) [a_{jm} \cos(m\theta) + b_{jm} \sin(m\theta)]$$

Zero frequency supplies the radial terms in the first sum.

Positive frequencies supply the cosine–sine pairs in the second sum.

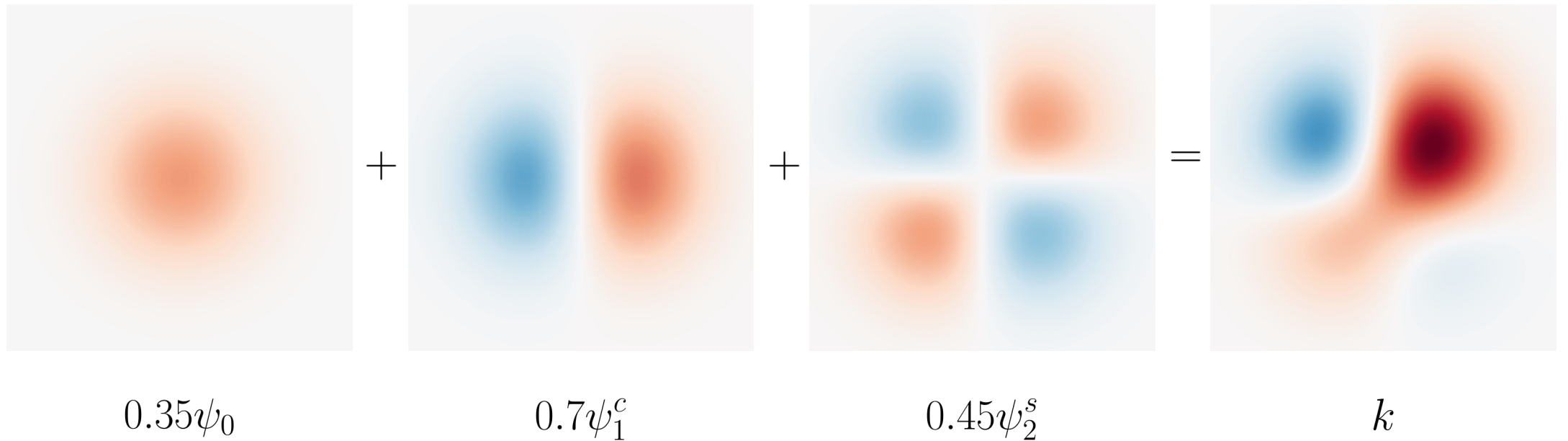
Learn the mixing weights. The basis functions stay fixed.

$$n_b = J(1 + 2M)$$

One real coefficient per radial term, two per positive-frequency pair.

Kernel Synthesis from Basis Weights

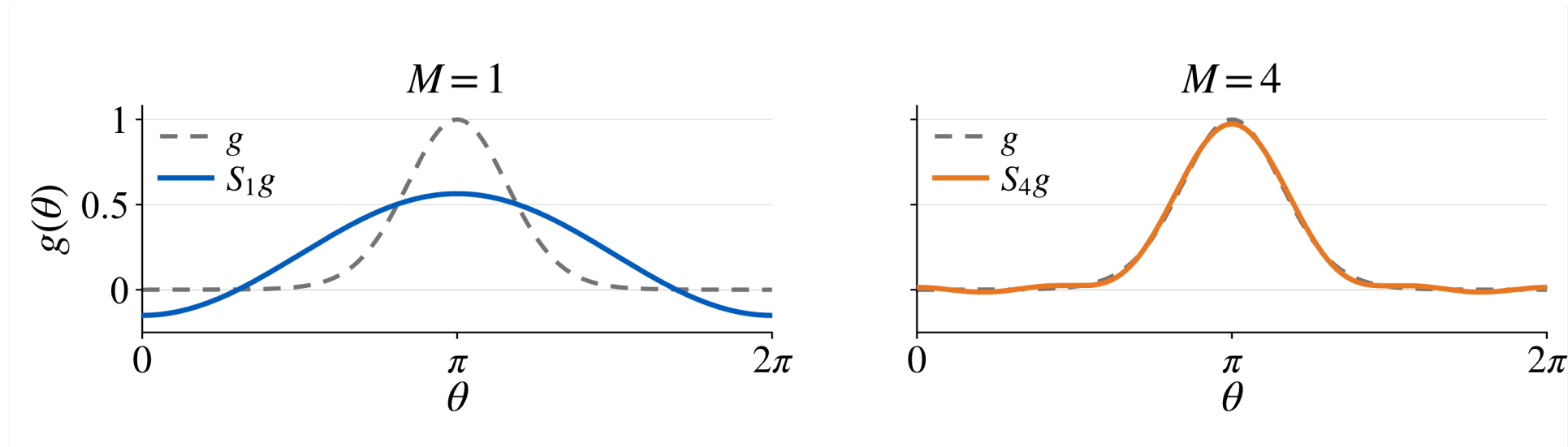
A kernel can combine radial structure and several angular patterns.



Rotate every frequency block analytically, then add the resulting components.

Angular Bandwidth and Exact Steering

More angular frequencies capture finer detail at a fixed radius.



Gray is the target. Colored curves are finite Fourier approximations.

Complete cosine–sine pairs preserve exact steering of continuous filters.

Outline

1. Groups and actions
2. Group convolutions
3. Steerable kernels
 - Steering a filter
 - Circular harmonics
 - **Kernel synthesis and sharing**
 - Scalar layers and typed features
4. Typed features and 3D networks
5. Implementation and verification (optional reading)

Two equivalent steering implementations

One base filter has one learned coefficient vector in a fixed basis.

$$k_w(u, v) = w^\top \Psi(u, v), \quad \Psi = (\psi_1, \dots, \psi_{n_b})^\top, \quad D(\alpha_\ell) \in \mathbb{R}^{n_b \times n_b}$$

1. Fixed basis, steered coefficients

$$w_\ell = D(\alpha_\ell)^\top w, \quad [L_{\alpha_\ell} k_w](u, v) = w_\ell^\top \Psi(u, v)$$

2. Rotated basis, shared coefficients

$$B_{\ell d u v} = [D(\alpha_\ell) \Psi(u, v)]_d, \quad W_{\ell u v} = \sum_{d=1}^{n_b} w_d B_{\ell d u v}$$

$$w^\top D(\alpha_\ell) \Psi = (D(\alpha_\ell)^\top w)^\top \Psi$$

The equality proves that both implementations produce the same rotated kernel.

Fixed Basis Samples

For a fixed grid and angle bank, cached geometry gives a direct synthesis step.

$$B_{\ell d u v} = [L_{\alpha_\ell} \psi_d](u, v) = \psi_d(R_{-\alpha_\ell}(u, v))$$

$$\alpha_\ell = \frac{2\pi\ell}{A}$$

Indices of the fixed sample table

$\ell = 0, \dots, A - 1$	Orientation index in the evenly spaced angle bank.
$d = 1, \dots, n_b$	Basis index: radial profile, frequency, and cosine/sine choice.
(u, v)	Centered horizontal and vertical kernel-grid offsets.

$R_{-\alpha_\ell}$ rotates the grid point clockwise by the sampled angle.

Shared Coefficients and Cached Geometry

One learned vector serves all orientations of each base filter and input channel.

Learned coefficients

$$w_{cid}$$

Same coefficients at every angle.
Learned during training.

Fixed rotated basis samples

$$B_{\ell d uv}$$

Different samples at each angle.
Fixed during training.

$$W_{cliuv} = \sum_{d=1}^{n_b} w_{cid} B_{\ell d uv}$$

For fixed angles, caching gives one direct coefficient contraction.
It stores a larger basis table than coefficient steering.
Neither route is always faster. Benchmark the actual layer.

Parameters Shared by Both Implementations

C base filters

A stored angles

M maximum angular frequency

C_{in} input channels

J radial profiles per frequency

s kernel grid side length

$$C = 16, \quad C_{\text{in}} = 3, \quad A = 8, \quad J = 3, \quad M = 2, \quad s = 7$$

$$n_b = J(1 + 2M) = 15 \text{ real basis functions}$$

Both steerable implementations

Steered coefficients or a cached basis

$$CC_{\text{in}}n_b = 720$$

Independent oriented kernels

Pixel weights learned at every angle

$$CAC_{\text{in}}s^2 = 18816$$

$CA = 128$ orientation channels still need memory and convolution work.

The finite basis and orientation sharing reduce the learned parameter count.

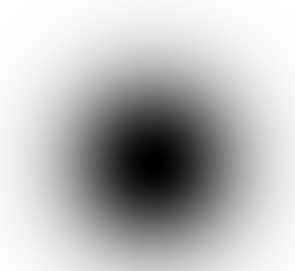
Outline

1. Groups and actions
2. Group convolutions
3. Steerable kernels
 - Steering a filter
 - Circular harmonics
 - Kernel synthesis and sharing
 - **Scalar layers and typed features**
4. Typed features and 3D networks
5. Implementation and verification (optional reading)

Ordinary 2D maps and a fixed directional filter

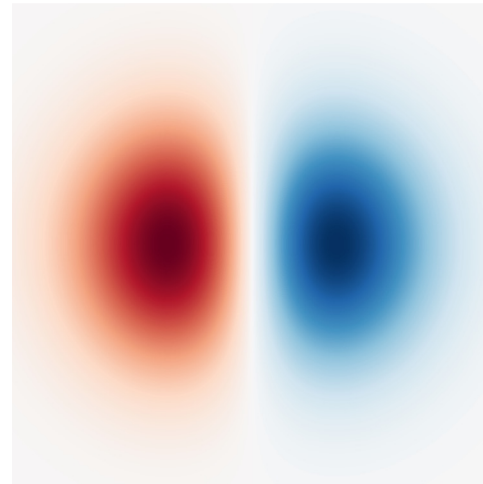
A single-channel 2D image or scalar feature map stores one number per pixel.
Keep the same filter, which measures horizontal intensity variation.

Circular input image



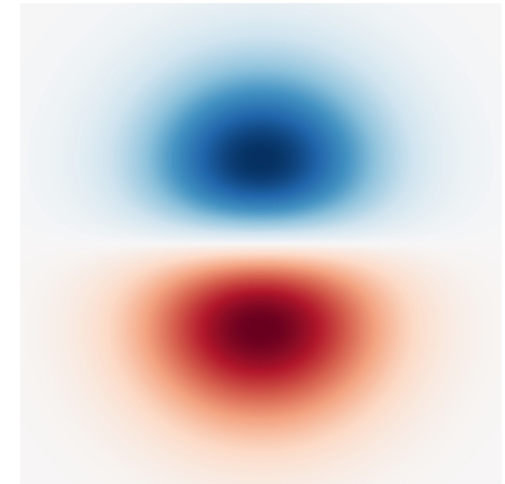
$$L_{\pi/2}f = f$$

Rotate the image,
then use the same filter



$$T_k L_{\pi/2}f = T_k f$$

Apply the filter,
then rotate its map



$$L_{\pi/2}T_k f$$

Rotating this circular input changes nothing.

Rotating its directional scalar response gives a different map.

Why Scalar Kernels Must Be Radial

Ordinary scalar maps, with a continuous integrable kernel.

Translation is built in. Require equivariance for every rotation.

$$[T_k f](x) = \int k(u) f(x + u) du, \quad [L_\beta f](x) = f(R_{-\beta} x)$$

1. Rotate the input, then convolve. Rotations preserve area.

$$[T_k L_\beta f](x) \stackrel{u=R_\beta v}{=} \int k(R_\beta v) f(R_{-\beta} x + v) dv$$

2. Convolve first, then rotate.

$$[L_\beta T_k f](x) = \int k(v) f(R_{-\beta} x + v) dv$$

Equality for every input forces the weights at rotated offsets to agree.

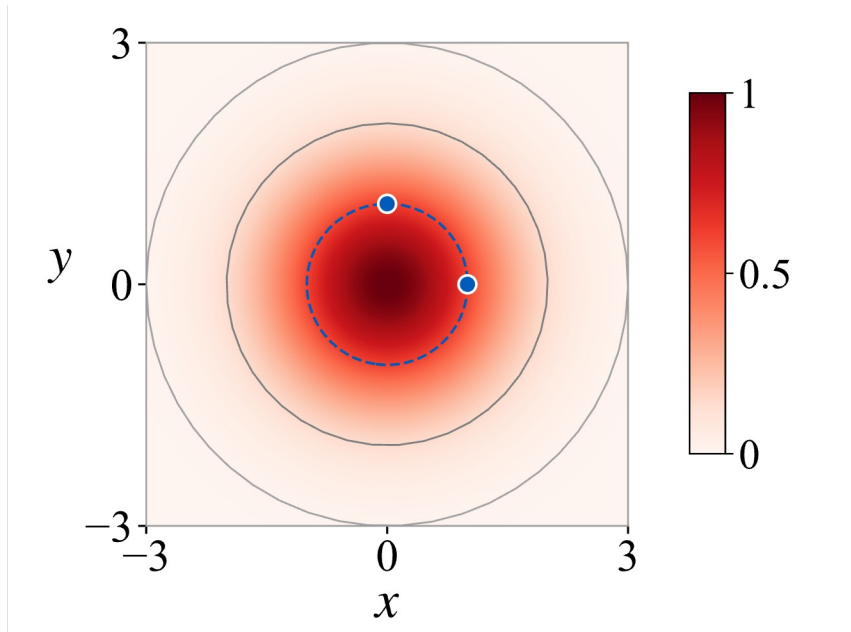
$$k(R_\beta v) = k(v) \quad (\forall \beta) \quad \implies \quad k(u) = q(\|u\|)$$

Radial Kernels in 2D Images

Rotation-equivariant scalar-to-scalar convolution requires circularly symmetric kernels.

$$T_k L_\beta = L_\beta T_k \iff k(R_{-\beta} u) = k(u)$$

For every input and rotation, with a continuous integrable kernel.



Radial means equal distance from the centre, equal weight.

$$k(x, y) = q\left(\sqrt{x^2 + y^2}\right)$$

Example: Gaussian blur

$$k(x, y) = e^{-(x^2 + y^2)/2}$$

$$k(1, 0) = k(0, 1) = e^{-1/2}$$

A radial kernel can process any image; the output need not be radial.

Why We Need Typed Features

Radial kernels cannot act as a single oriented edge detector.

Independent scalar channels keep this restriction.

1. Scalar-to-vector layer: rotate output locations and components.

$$K(u) \in \mathbb{R}^{2 \times 1}, \quad K(R_\beta u) = R_\beta K(u)$$

Directional component kernels are allowed when their outputs rotate together.

2. Orientation channels: rotate locations and shift detector angles.

$$F_{L_\beta f}(x, \alpha) = F_f(R_{-\beta} x, \alpha - \beta)$$

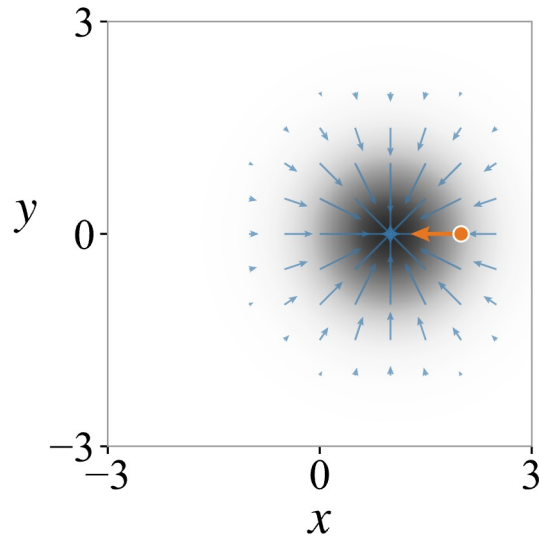
A scalar prediction can still use richer hidden features.

Gradient Features in 2D Images

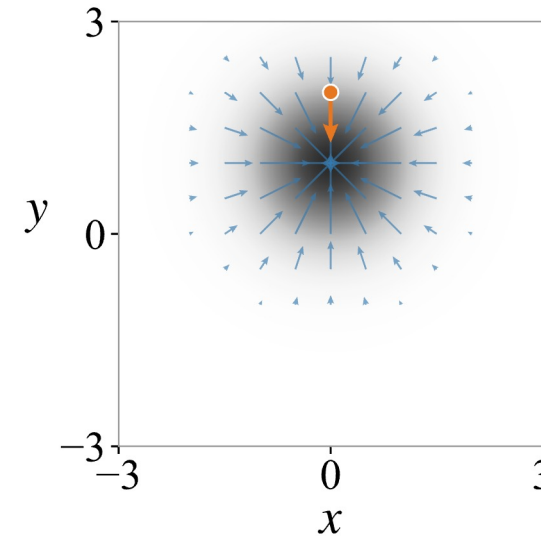
A 2D gradient stores horizontal and vertical intensity changes at each pixel.

Arrows show gradients. Orange marks corresponding pixels.

Original image and gradient



90° counterclockwise rotation



$$\nabla[L_{\beta}f](u) = R_{\beta}[\nabla f](R_{-\beta}u)$$

Part 4 shows how to build equivariant layers for vector and tensor features.

Questions?

Steerable kernels

Next: Typed features and 3D networks

References

- [1] Bekkers. Group Equivariant Deep Learning, Lecture 2.1: Steerable Kernels/Basis Functions. University of Amsterdam.
- [2] Freeman & Adelson (1991). The Design and Use of Steerable Filters. IEEE TPAMI.
- [3] Weiler et al. (2018). Learning Steerable Filters for Rotation Equivariant CNNs. CVPR.
- [4] Weiler & Cesa (2019). General $E(2)$ -Equivariant Steerable CNNs. NeurIPS.
- [5] The e2cnn authors. Neural network modules: basis expansion and R2Conv. Implementation reference.