GAMES
Geometric Deep Learning II

Qixing Huang

Sep. 23th 2021

Slide credit: Michael Bronstein

Spectral CNN

Convolution expressed in the spectral domain

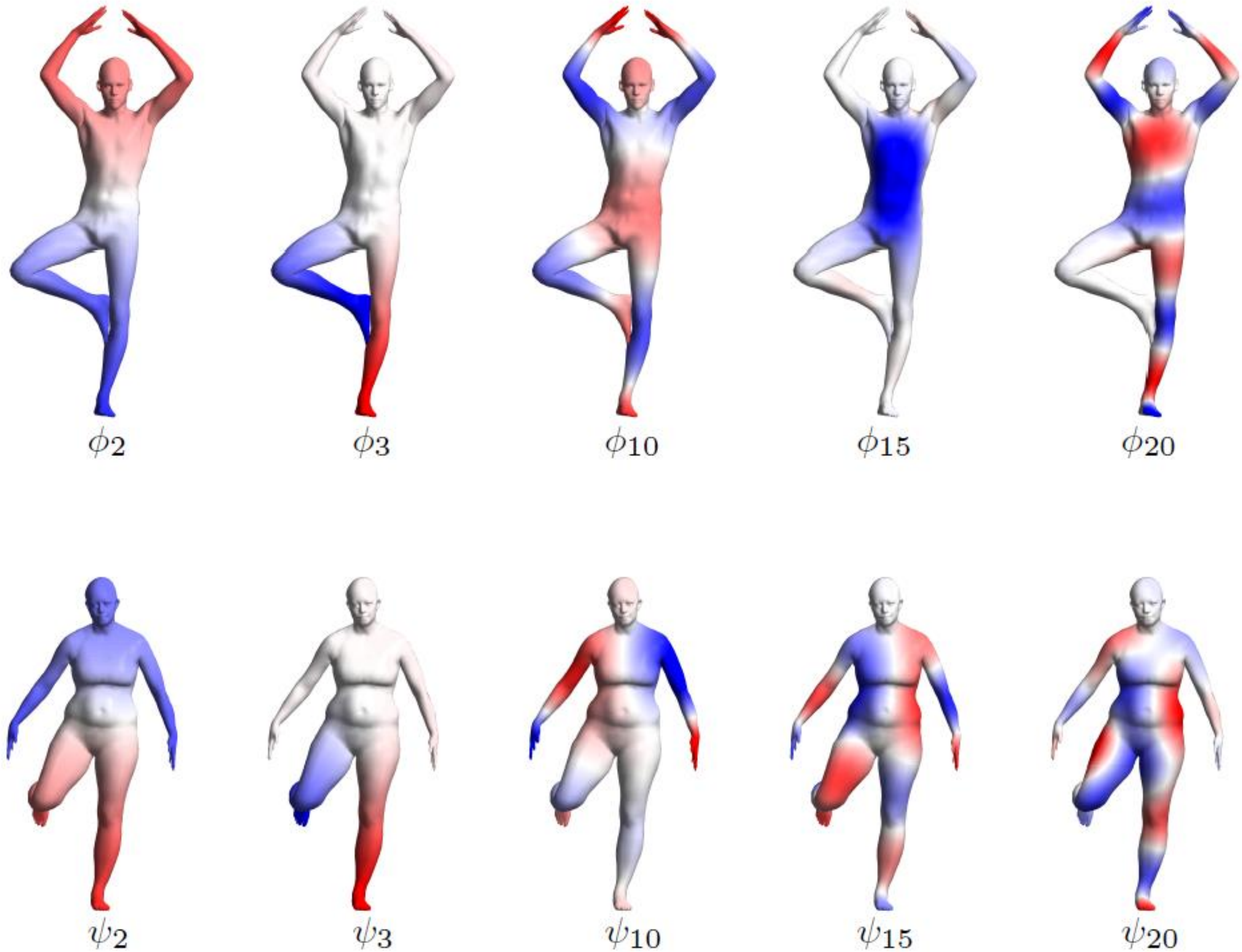
$$\mathbf{g} = \mathbf{\Phi} \mathbf{W} \mathbf{\Phi}^\top \mathbf{f}$$

where $\mathbf{W}$ is $n \times n$ diagonal matrix of learnable spectral filter coefficients

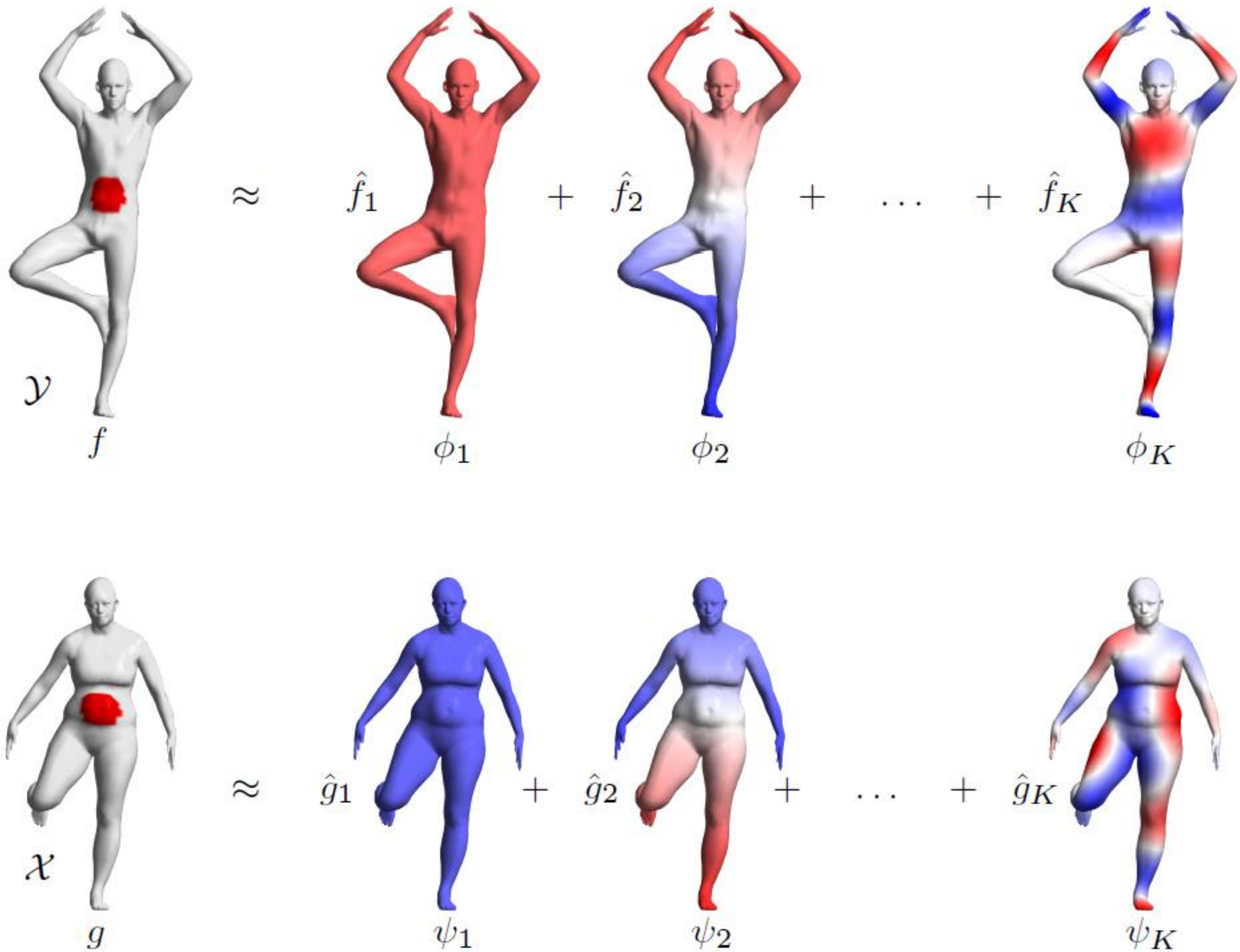
- ☹ Filters are basis-dependent $\Rightarrow$ do not generalize across domains
- ☹ $\mathcal{O}(n)$ parameters per layer
- ☹ $\mathcal{O}(n^2)$ computation of forward and inverse Fourier transforms $\mathbf{\Phi}^\top, \mathbf{\Phi}$ (no FFT on graphs)
- ☹ No guarantee of spatial localization of filters

Bruna et al. 2014

Laplacian eigenbases on non-isometric domains

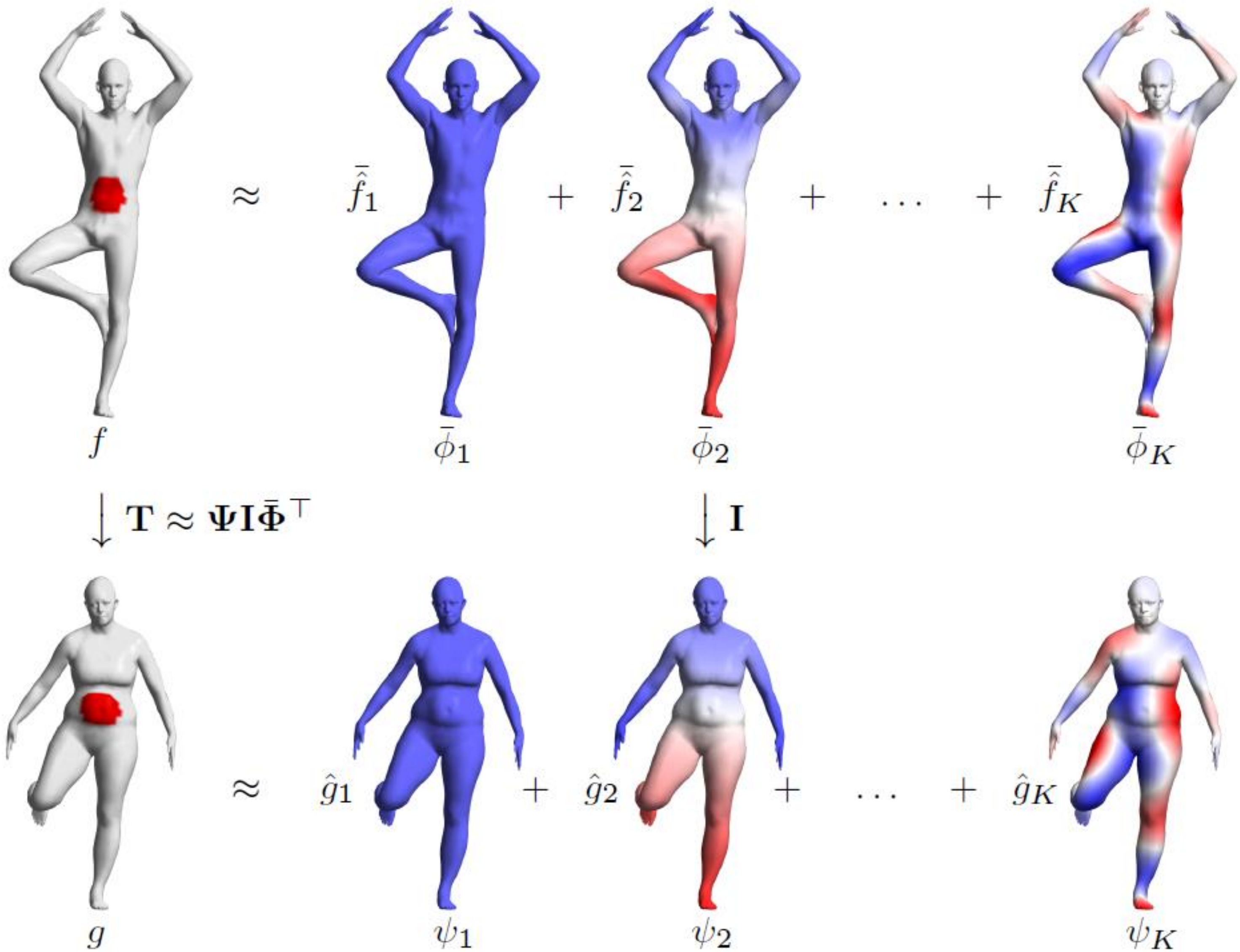


Functional maps



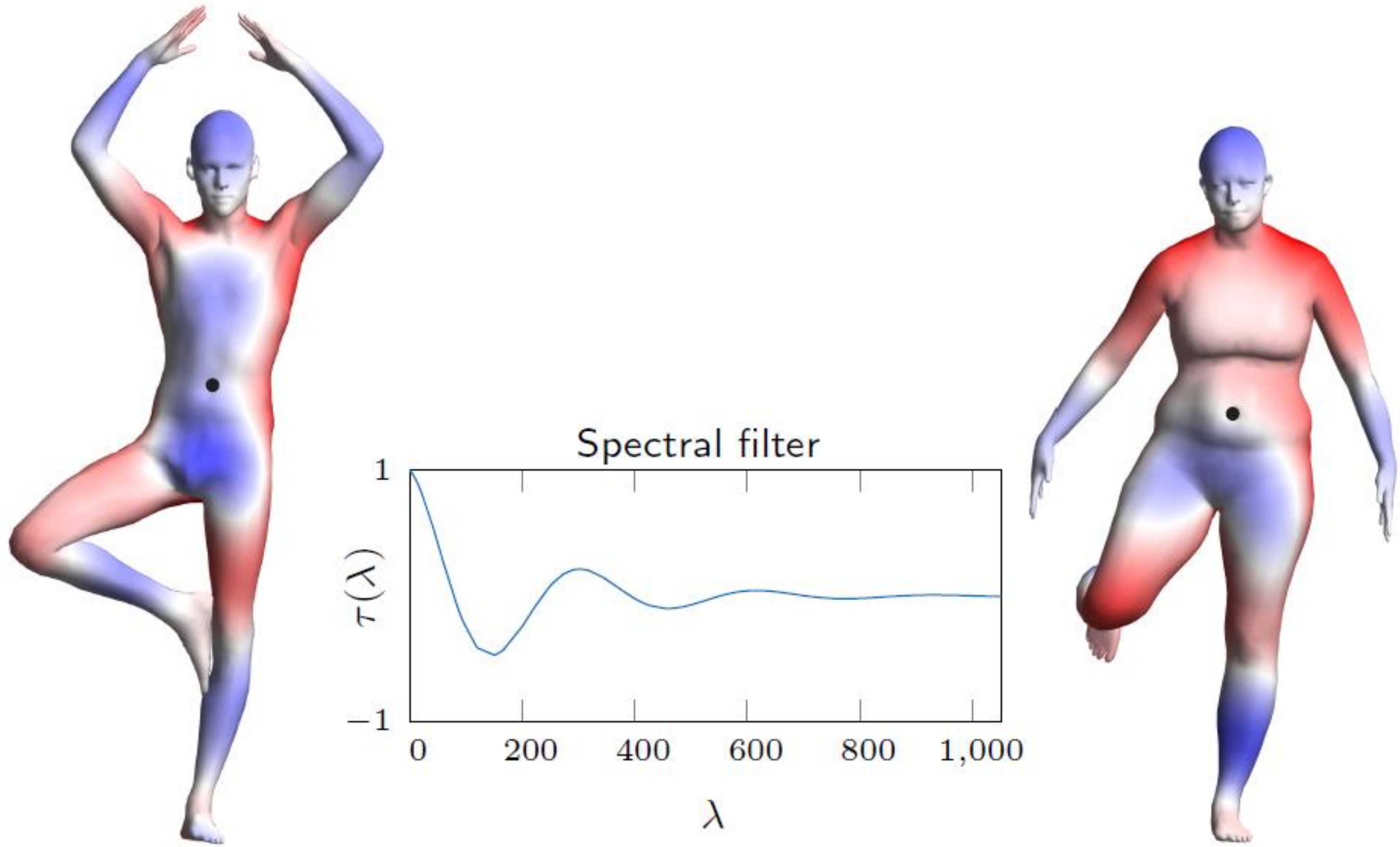
Ovsjanikov et al. 2012; Eynard et al. 2012; Kovnatsky et al. 2013

Basis synchronization with functional maps



Ovsjanikov et al. 2012; Eynard et al. 2012; Kovnatsky et al. 2013

Filtering in different bases



$\mathbf{\Phi}\tau(\mathbf{\Lambda_\Phi})\mathbf{\Phi}^\top\boldsymbol{\delta}_0$ $\mathbf{\Psi}\tau(\mathbf{\Lambda_\Psi})\mathbf{\Psi}^\top\boldsymbol{\delta}_0$

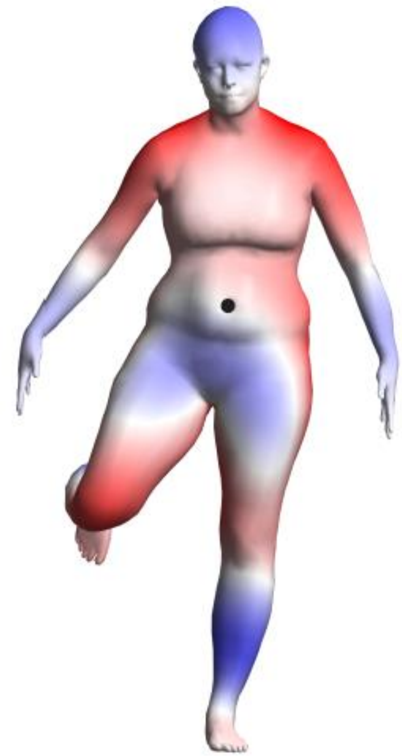
Apply spectral filter $\tau(\lambda)$ in different bases $\mathbf{\Phi}$ and $\mathbf{\Psi}$

$\Rightarrow$ **different results!**

Filtering in different bases

$\mathbf{\Phi}\tau(\mathbf{\Lambda}_{\mathbf{\Phi}})\mathbf{\Phi}^{\top}\boldsymbol{\delta}_0$

Canonical shape
with basis $\mathbf{\Sigma}, \mathbf{\Lambda}$

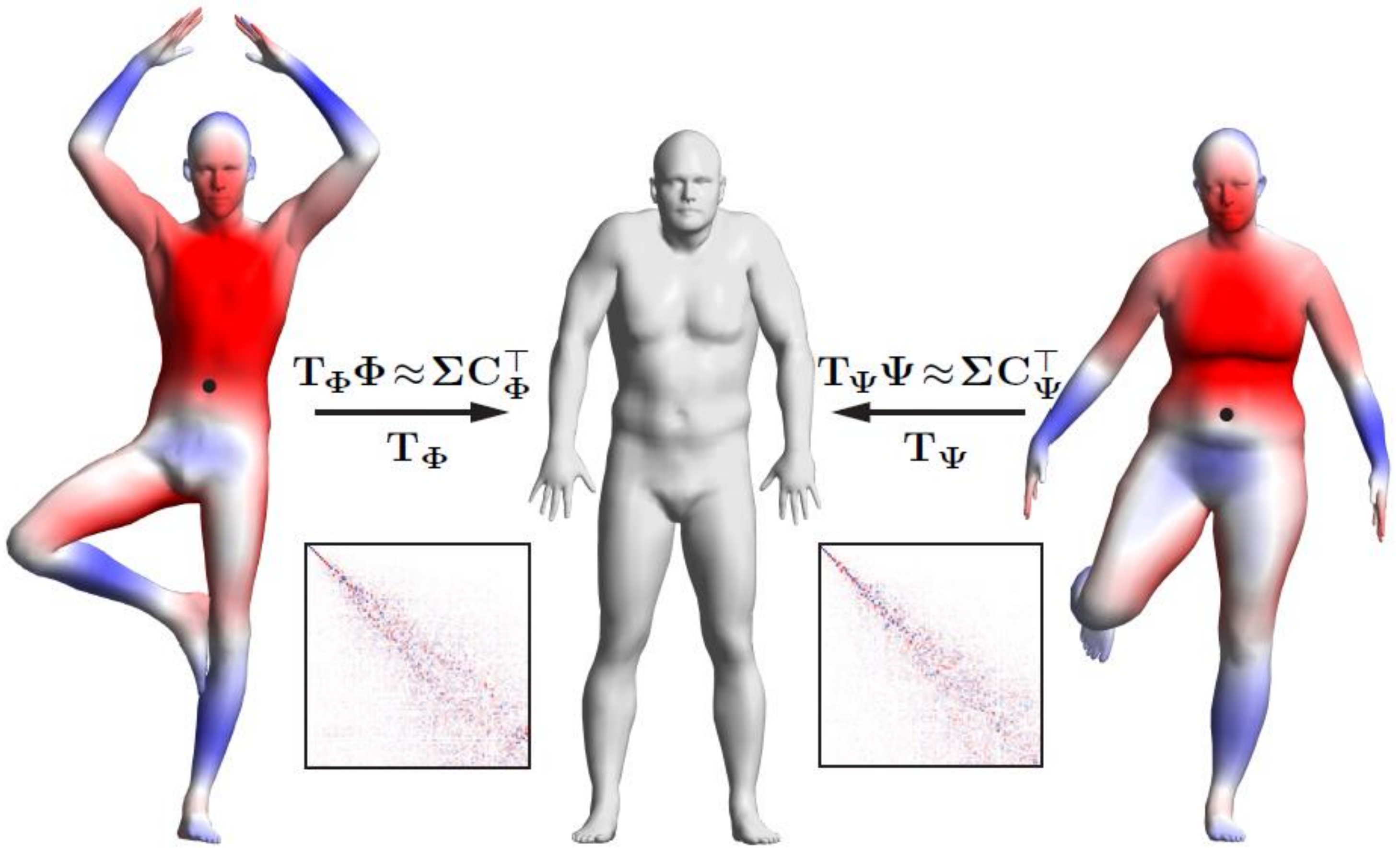


$\mathbf{\Psi}\tau(\mathbf{\Lambda}_{\mathbf{\Psi}})\mathbf{\Psi}^{\top}\boldsymbol{\delta}_0$

Apply spectral filter $\tau(\lambda)$ in different bases $\mathbf{\Phi}$ and $\mathbf{\Psi}$

$\Rightarrow$ **different results!**

Filtering in synchronized bases



$\Phi \mathbf{C}_\Phi \tau(\Lambda) \mathbf{C}_\Phi^\top \Phi^\top \delta_0$

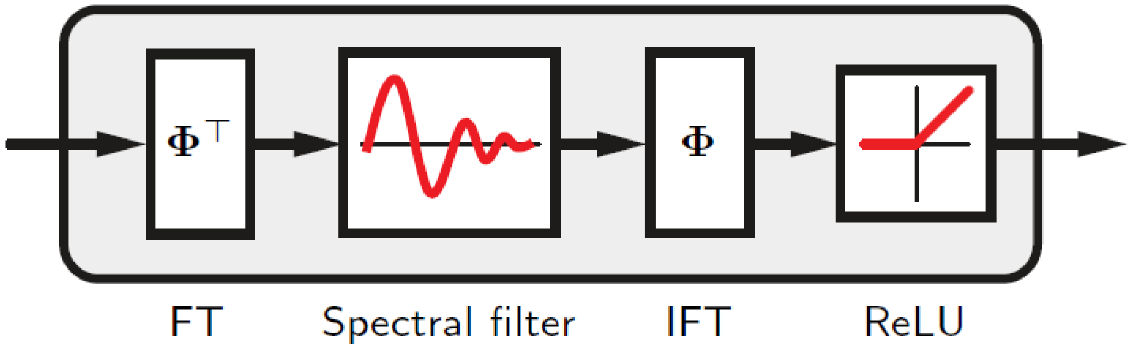
Canonical shape with basis Σ, Λ

$\Psi \mathbf{C}_\Psi \tau(\Lambda) \mathbf{C}_\Psi^\top \Psi^\top \delta_0$

Apply spectral filter $\tau(\lambda)$ in synchronized bases $\Phi \mathbf{C}_\Phi$ and $\Psi \mathbf{C}_\Psi$
$\Rightarrow$ **similar results!**

Yi et al. 2017

Spectral CNN



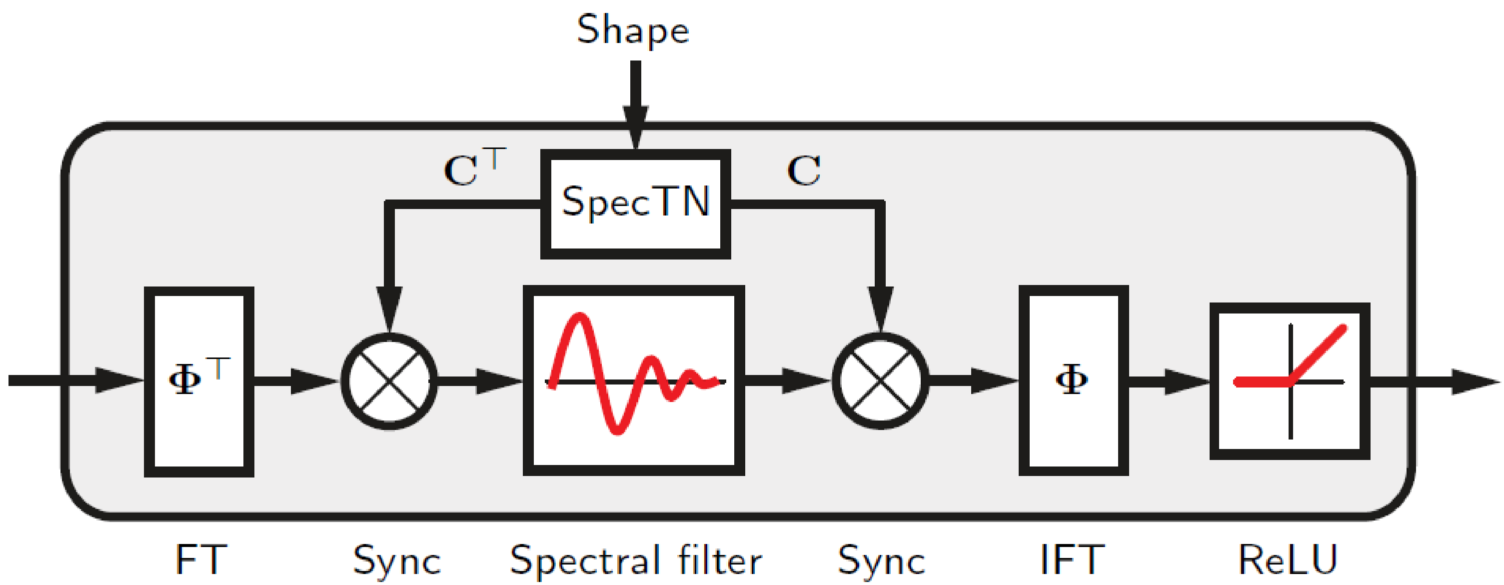
Convolutional filter of a Spectral CNN

☹ Fixed basis ⇒ Does not generalize across domains

☺ Possible $\mathcal{O}(n)$ complexity avoiding explicit FT and IFT

Bruna et al. 2014

Spectral Transformer Network



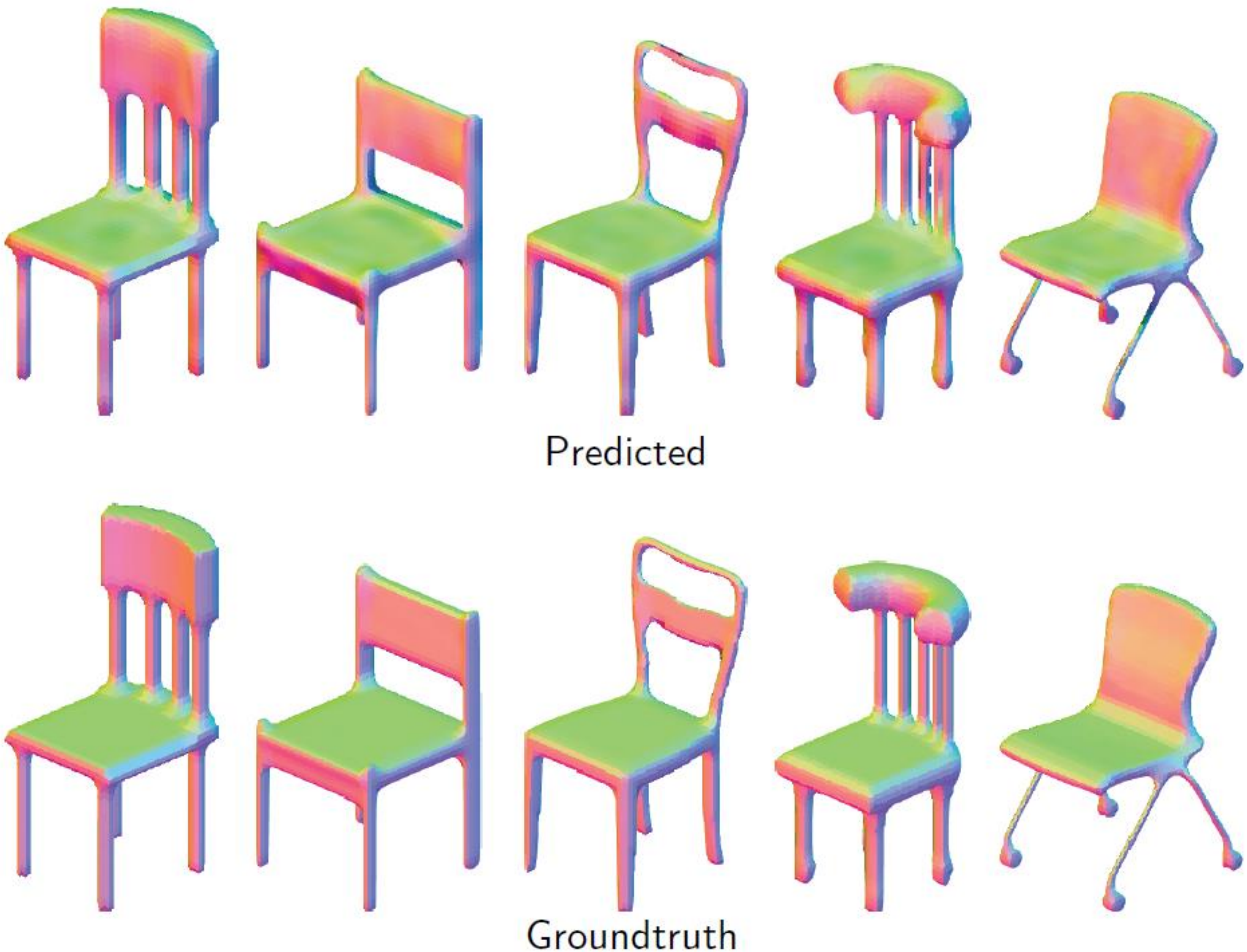
Convolutional filter of a Spectral Transformer Network

☺ Basis synchronization allows generalization across domains

☹ Explicit FT and IFT

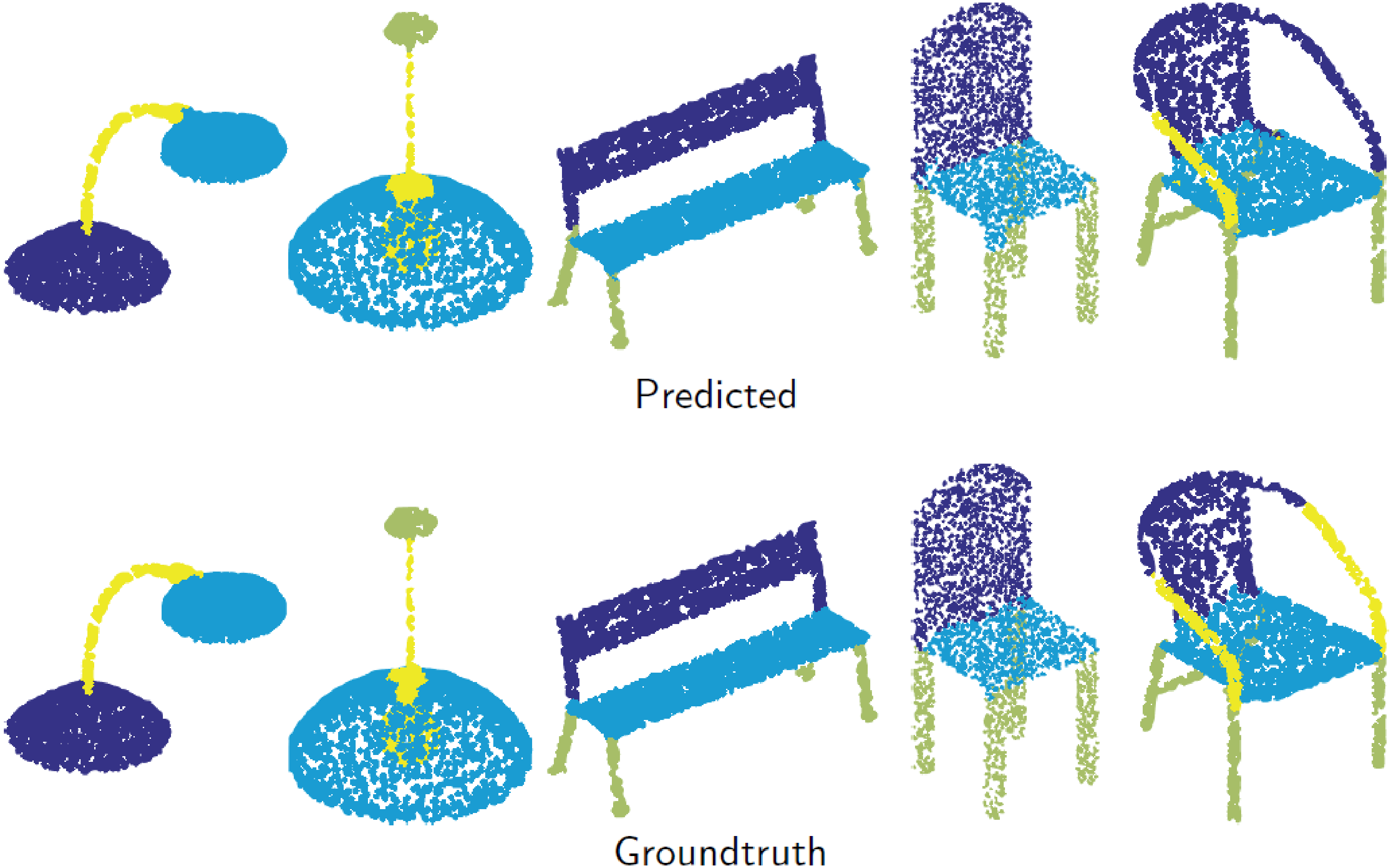
(Jaderberg et al. 2015); Yi et al. 2017

Example: normal prediction with SpecTN



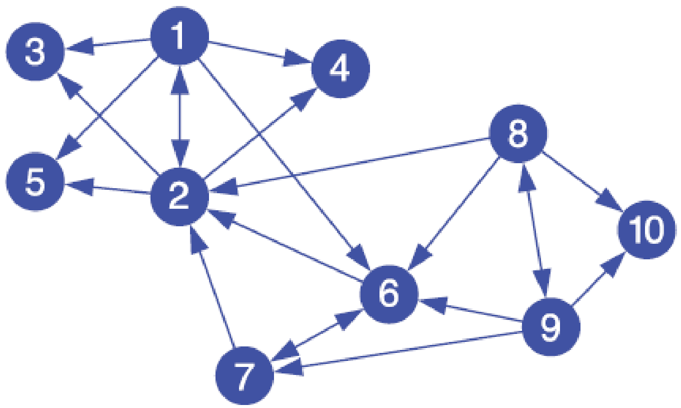
Yi et al. 2017

Example: shape segmentation with SpecTN



Yi et al. 2017

Directed graphs



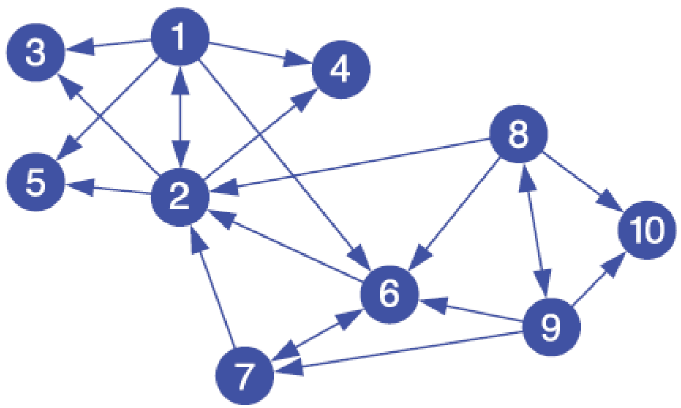
Directed graph

$$
\begin{bmatrix}
. & 1 & 1 & 1 & 1 & 1 & . & . & . & . \\
1 & . & 1 & 1 & 1 & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & 1 & . & . & . & . & 1 & . & . & . \\
. & 1 & . & . & . & 1 & . & . & . & . \\
. & 1 & . & . & . & 1 & . & . & 1 & 1 \\
. & . & . & . & . & 1 & 1 & 1 & . & 1 \\
. & . & . & . & . & . & . & . & . & .
\end{bmatrix}
$$

Asymmetric adjacency matrix

Benson et al. 2016

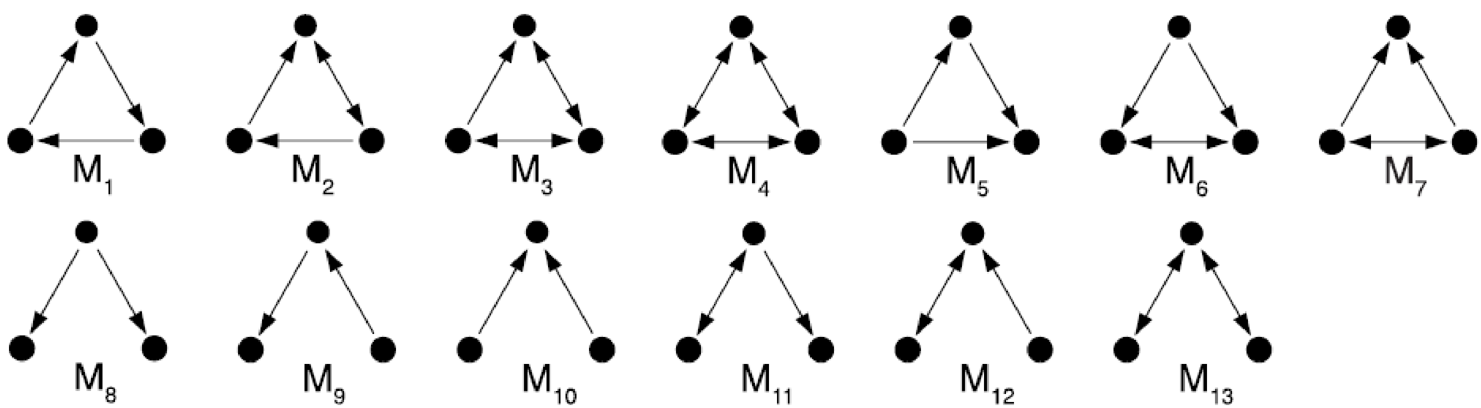
Motif-based graph analysis



Directed graph

$$
\begin{bmatrix}
. & 1 & 1 & 1 & 1 & 1 & . & . & . & . \\
1 & . & 1 & 1 & 1 & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & 1 & . & . & . & . & 1 & . & . & . \\
. & 1 & . & . & . & 1 & . & . & . & . \\
. & 1 & . & . & . & 1 & . & . & 1 & 1 \\
. & . & . & . & . & 1 & 1 & 1 & . & 1 \\
. & . & . & . & . & . & . & . & . & .
\end{bmatrix}
$$

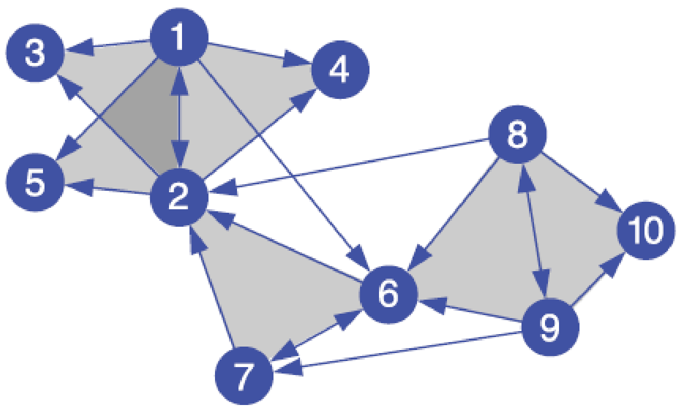
Asymmetric adjacency matrix



Graph motifs

Benson et al. 2016

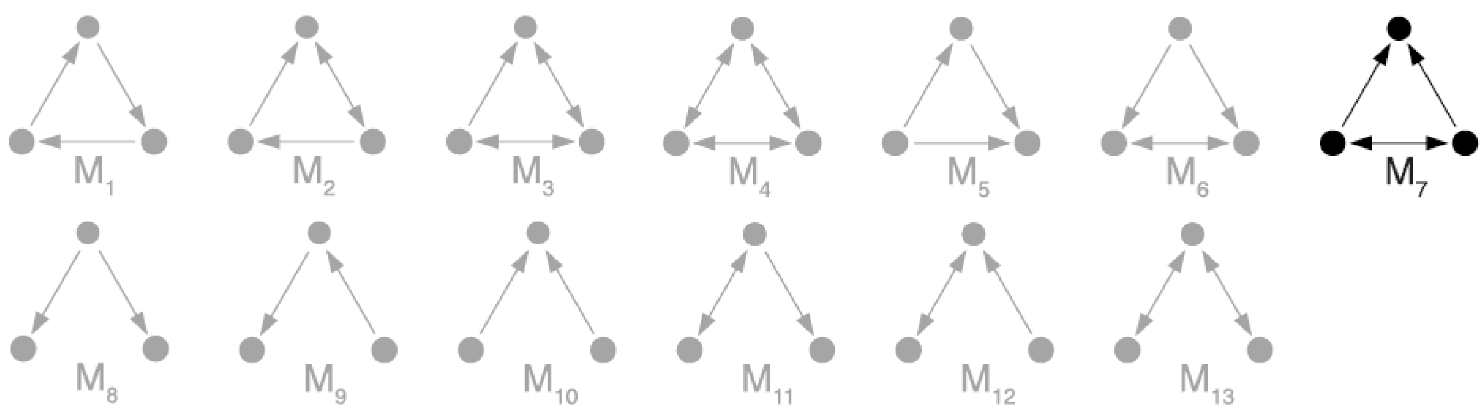
Motif-based graph analysis



Directed graph

$$\begin{bmatrix} \cdot & 3 & 1 & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ 3 & \cdot & 1 & 1 & 1 & 1 & 1 & \cdot & \cdot & \cdot \\ 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot & \cdot & \cdot & 1 & 1 & 1 & \cdot \\ \cdot & 1 & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & 2 & 1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & 2 & \cdot & 1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 1 & \cdot \end{bmatrix}$$

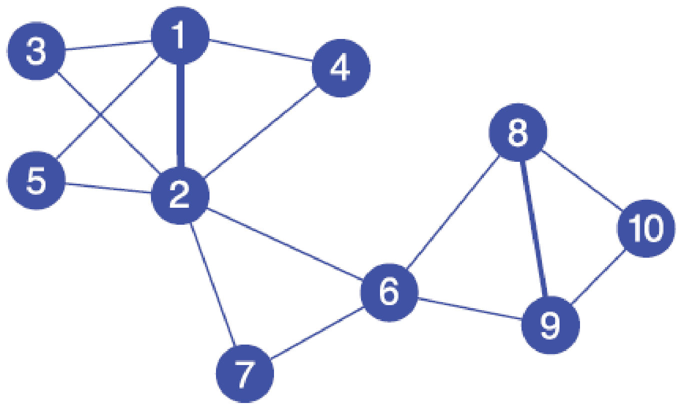
Motif adjacency matrix for M_7



Graph motifs

Benson et al. 2016

Motif Laplacians



Undirected weighted graph

$$\begin{bmatrix}
. & 3 & 1 & 1 & 1 & . & . & . & . & . \\
3 & . & 1 & 1 & 1 & 1 & 1 & . & . & . \\
1 & 1 & . & . & . & . & . & . & . & . \\
1 & 1 & . & . & . & . & . & . & . & . \\
1 & 1 & . & . & . & . & . & . & . & . \\
. & 1 & . & . & . & . & 1 & 1 & 1 & . \\
. & 1 & . & . & . & 1 & . & . & . & . \\
. & . & . & . & . & 1 & . & . & 2 & 1 \\
. & . & . & . & . & 1 & . & 2 & . & 1 \\
. & . & . & . & . & . & . & 1 & 1 & .
\end{bmatrix}$$

Motif adjacency matrix for M_7

Motif Laplacian for motif $k = 1, \ldots, K$

$$\tilde{\mathbf{\Delta}}_k = \mathbf{I} - \tilde{\mathbf{D}}_k^{-1/2} \tilde{\mathbf{W}}_k \tilde{\mathbf{D}}_k^{-1/2}$$

Benson et al. 2016

MotifNet: multivariate polynomial

Apply K-variate polynomial or order r to the motif Laplacians

$$\tau_{\boldsymbol{\alpha}}(\tilde{\boldsymbol{\Delta}}_1, \ldots, \tilde{\boldsymbol{\Delta}}_K) = \alpha_0 \mathbf{I} + \sum_{j=1}^{r} \sum_{k_1,\ldots,k_j \in \{1,\ldots,K\}} \alpha_{k_1,\ldots,k_j} \tilde{\boldsymbol{\Delta}}_{k_1} \cdots \tilde{\boldsymbol{\Delta}}_{k_j}$$

where $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots, \alpha_{K,\ldots,K})$ is the vector of filter parameters

Monti, Otness, Bronstein 2018

MotifNet: multivariate polynomial

Apply K-variate polynomial or order r to the motif Laplacians

$$\tau_{\boldsymbol{\alpha}}(\tilde{\boldsymbol{\Delta}}_1, \ldots, \tilde{\boldsymbol{\Delta}}_K) = \alpha_0 \mathbf{I} + \sum_{j=1}^{r} \sum_{k_1,\ldots,k_j \in \{1,\ldots,K\}} \alpha_{k_1,\ldots,k_j} \tilde{\boldsymbol{\Delta}}_{k_1} \cdots \tilde{\boldsymbol{\Delta}}_{k_j}$$

where $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots, \alpha_{K,\ldots,K})$ is the vector of filter parameters

- ☺ Explicitly accounts for directed graph structures
- ☺ Anisotropic kernels
- ☺ $\mathcal{O}(1)$ parameters per layer

Monti, Otness, Bronstein 2018

MotifNet: multivariate polynomial

Apply K-variate polynomial or order r to the motif Laplacians

$$\tau_{\boldsymbol{\alpha}}(\tilde{\boldsymbol{\Delta}}_1, \ldots, \tilde{\boldsymbol{\Delta}}_K) = \alpha_0 \mathbf{I} + \sum_{j=1}^{r} \sum_{k_1,\ldots,k_j \in \{1,\ldots,K\}} \alpha_{k_1,\ldots,k_j} \tilde{\boldsymbol{\Delta}}_{k_1} \cdots \tilde{\boldsymbol{\Delta}}_{k_j}$$

where $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots, \alpha_{K,\ldots,K})$ is the vector of filter parameters

- ☺ Explicitly accounts for directed graph structures
- ☺ Anisotropic kernels
- ☹ $\frac{1+K^{r+1}}{K-1}$ parameters per layer, intractable in practice

Monti, Otness, Bronstein 2018

MotifNet: multivariate polynomial

Apply K-variate polynomial or order r to the motif Laplacians

$$\tau_{\boldsymbol{\alpha}}(\tilde{\boldsymbol{\Delta}}_1, \ldots, \tilde{\boldsymbol{\Delta}}_K) = \alpha_0 \mathbf{I} + \sum_{j=1}^{r} \sum_{k_1,\ldots,k_j \in \{1,\ldots,K\}} \alpha_{k_1,\ldots,k_j} \tilde{\boldsymbol{\Delta}}_{k_1} \cdots \tilde{\boldsymbol{\Delta}}_{k_j}$$

where $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots, \alpha_{K,\ldots,K})$ is the vector of filter parameters

- ☺ Explicitly accounts for directed graph structures
- ☺ Anisotropic kernels
- ☺ $Kr + 1$ parameters per layer using recurrent multivariate polynomials with dependent coefficients

Monti, Otness, Bronstein 2018

MotifNet: multivariate polynomial

Apply K-variate polynomial or order r to the motif Laplacians

$$\tau_{\boldsymbol{\alpha}}(\tilde{\boldsymbol{\Delta}}_1, \dots, \tilde{\boldsymbol{\Delta}}_K) = \alpha_0 \mathbf{I} + \sum_{j=1}^{r} \sum_{k_1,\dots,k_j \in \{1,\dots,K\}} \alpha_{k_1,\dots,k_j} \tilde{\boldsymbol{\Delta}}_{k_1} \cdots \tilde{\boldsymbol{\Delta}}_{k_j}$$

where $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \dots, \alpha_{K,\dots,K})$ is the vector of filter parameters

- ☺ Explicitly accounts for directed graph structures
- ☺ Anisotropic kernels
- ☺ $Kr + 1$ parameters per layer using recurrent multivariate polynomials with dependent coefficients
- ☺ Filters have guaranteed r-hops support

Monti, Otness, Bronstein 2018

MotifNet: multivariate polynomial

Apply K-variate polynomial or order r to the motif Laplacians

$$\tau_{\boldsymbol{\alpha}}(\tilde{\boldsymbol{\Delta}}_1, \ldots, \tilde{\boldsymbol{\Delta}}_K) = \alpha_0 \mathbf{I} + \sum_{j=1}^{r} \sum_{k_1,\ldots,k_j \in \{1,\ldots,K\}} \alpha_{k_1,\ldots,k_j} \tilde{\boldsymbol{\Delta}}_{k_1} \cdots \tilde{\boldsymbol{\Delta}}_{k_j}$$

where $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots, \alpha_{K,\ldots,K})$ is the vector of filter parameters

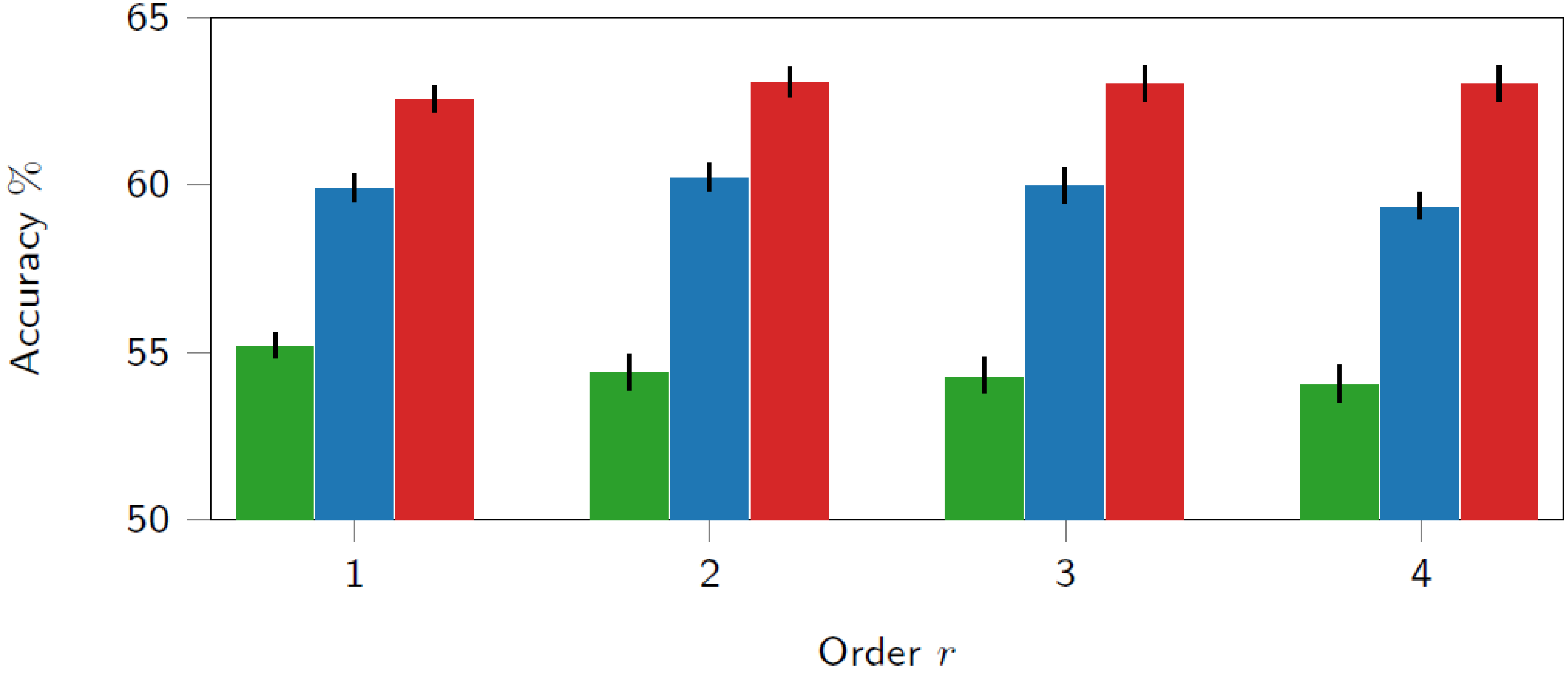
- ☺ Explicitly accounts for directed graph structures
- ☺ Anisotropic kernels
- ☺ $Kr + 1$ parameters per layer using recurrent multivariate polynomials with dependent coefficients
- ☺ Filters have guaranteed r-hops support
- ☺ $\mathcal{O}(n)$ computational complexity

Monti, Otness, Bronstein 2018

Example: directed citation networks

Figure: Monti, Otness, Bronstein 2018; data: Bojchevski, Günnemann 2017

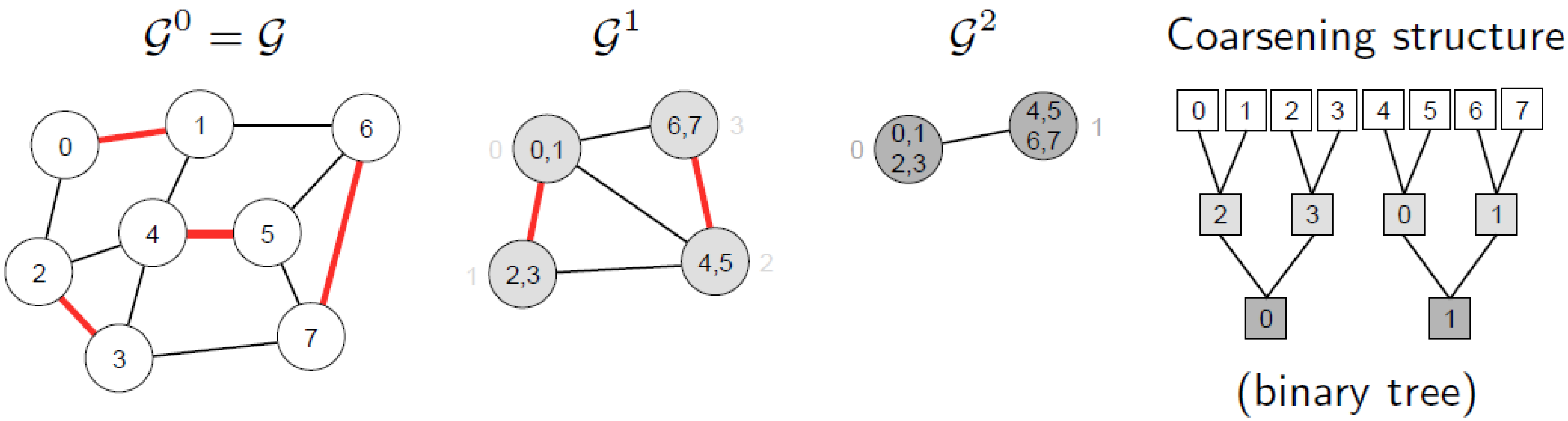
Example: directed citation networks



Classification accuracy on directed CORA obtained with ChebNet applied with directed adjacency matrix $\mathbf{W}$ (blue) and $\mathbf{W}^\top$ (green), and MotifNet-m (red)

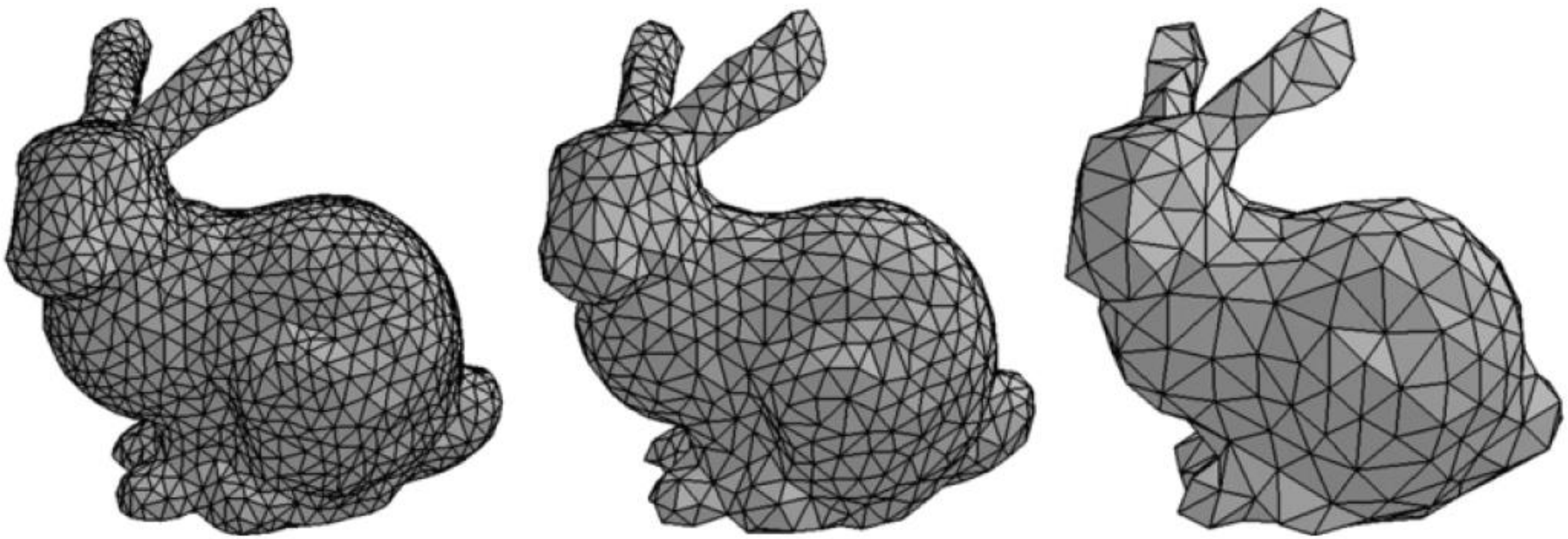
Monti, Otness, Bronstein 2018; data: Bojchevski, Günnemann, 2017

Graph pooling



- Produce a sequence of coarsened graphs
- Max or average pooling of collapsed vertices
- Binary tree arrangement of node indices

Mesh pooling



Example of progressive coarsening of a mesh

Spatial domain (charting-based) geometric deep learning methods

Convolution

Euclidean

Spatial domain

$$(f \star g)(x) = \int_{-\pi}^{\pi} f(x')g(x-x')dx'$$

Spectral domain

$$\widehat{(f \star g)}(\omega) = \hat{f}(\omega) \cdot \hat{g}(\omega)$$

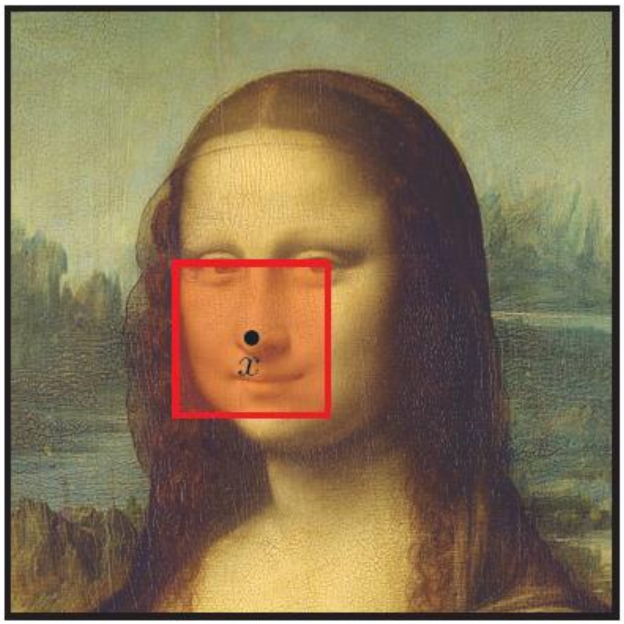
'Convolution Theorem'

Non-Euclidean

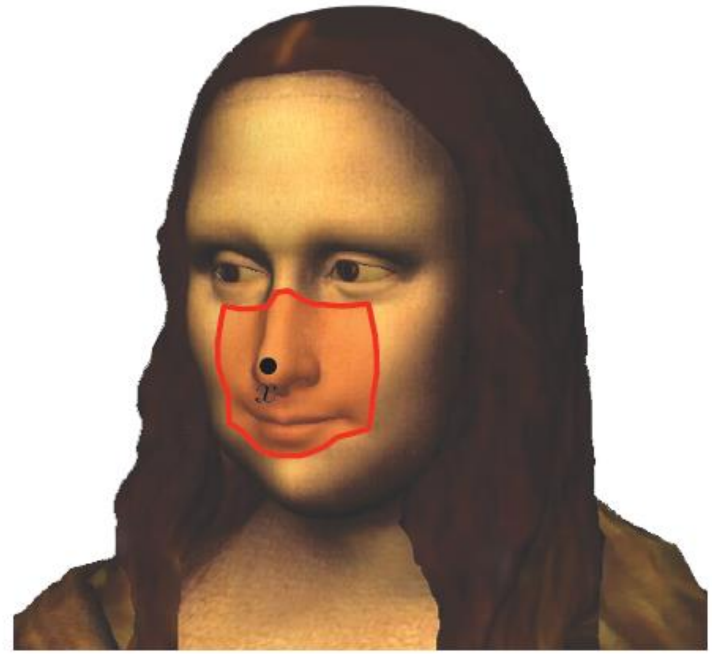
?

$$\widehat{(f \star g)}_k = \langle f, \phi_k \rangle_{L^2(\mathcal{X})} \langle g, \phi_k \rangle_{L^2(\mathcal{X})}$$

Patch operators



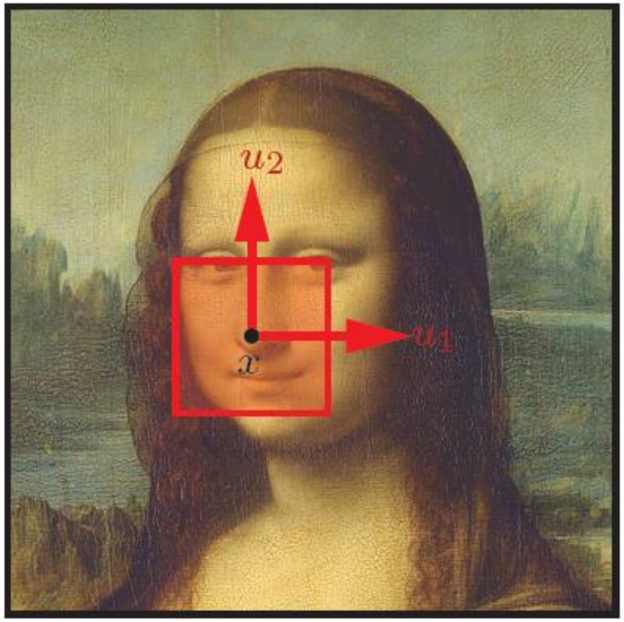
Image



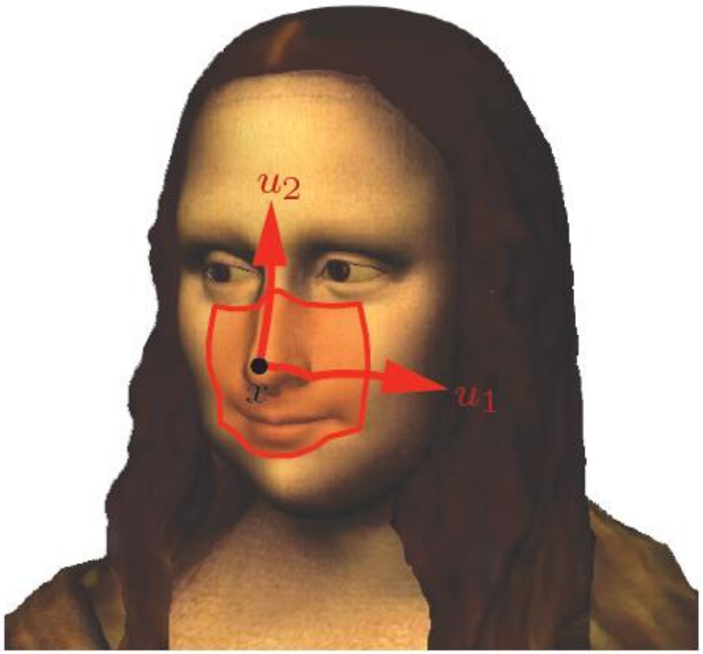
Manifold

Boscaini et al. 2015

Patch operators



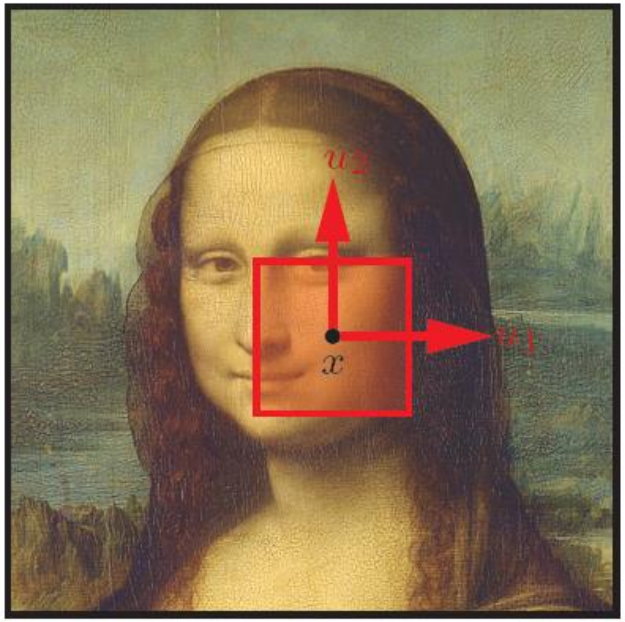
Image



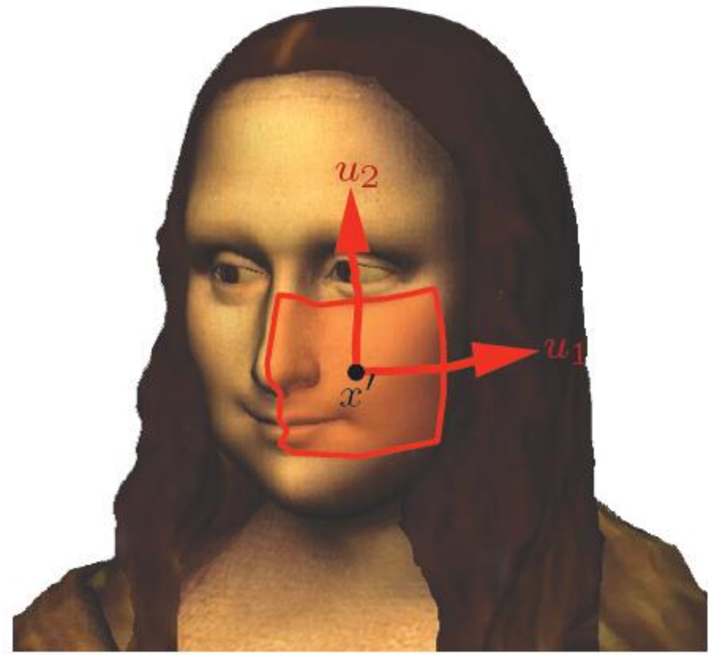
Manifold

Boscaini et al. 2015

Patch operators



Image

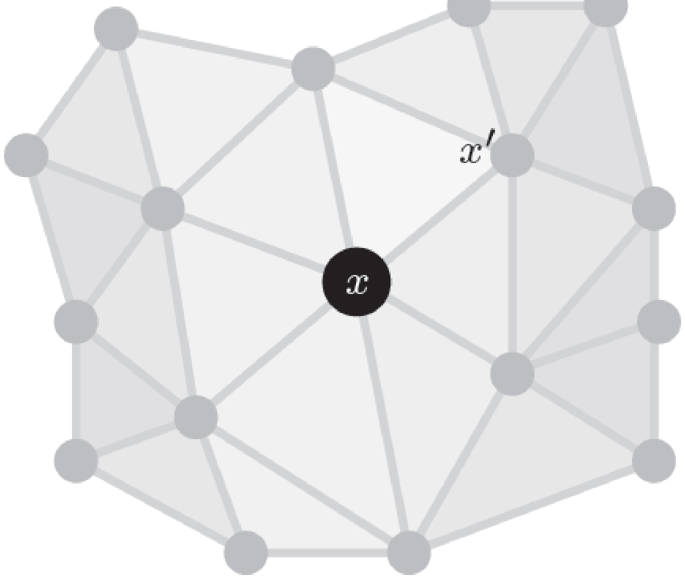


Manifold

Boscaini et al. 2015

Convolution on manifolds

- Local system of coordinates
  $\mathbf{u}(x, x')$ around x' (e.g. geodesic polar)

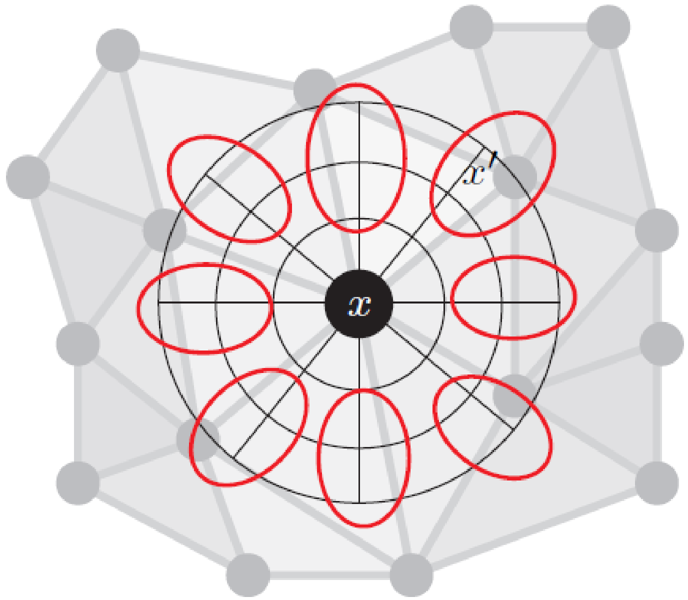


Masci et al. 2015; Boscaini et al. 2016; Monti et al. 2017

Convolution on manifolds

- Local system of coordinates $\mathbf{u}(x, x')$ around x' (e.g. geodesic polar)

- Local weights $w_1(\mathbf{u}), \ldots, w_L(\mathbf{u})$ w.r.t. $\mathbf{u}$

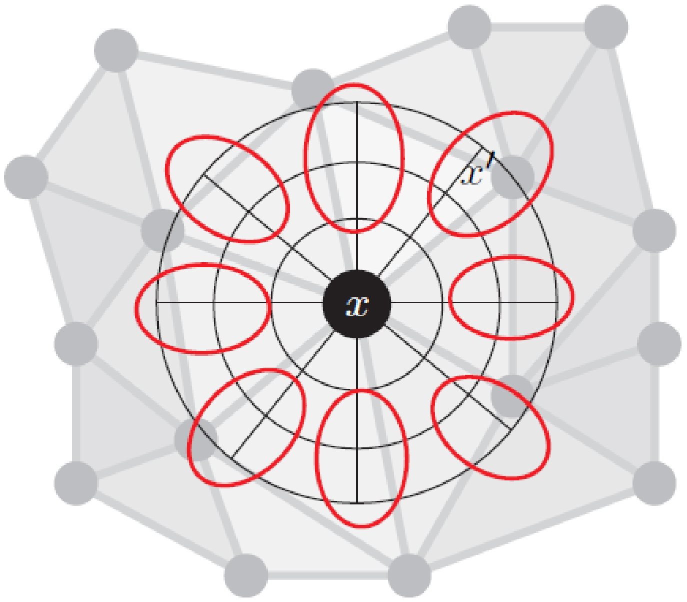


Masci et al. 2015; Boscaini et al. 2016; Monti et al. 2017

Convolution on manifolds

- Local system of coordinates $\mathbf{u}(x, x')$ around x' (e.g. geodesic polar)
- Local weights $w_1(\mathbf{u}), \ldots, w_L(\mathbf{u})$ w.r.t. $\mathbf{u}$, e.g. Gaussians

$$w_\ell = \exp\left(-(\mathbf{u} - \boldsymbol{\mu}_\ell)^\top \boldsymbol{\Sigma}_\ell^{-1} (\mathbf{u} - \boldsymbol{\mu}_\ell)\right)$$

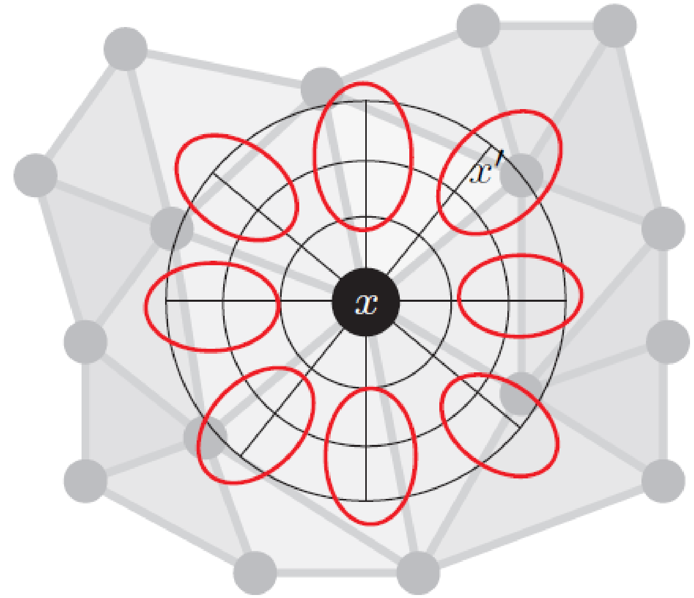


Masci et al. 2015; Boscaini et al. 2016; Monti et al. 2017

Convolution on manifolds

- Local system of coordinates $\mathbf{u}(x, x')$ around x' (e.g. geodesic polar)
- Local weights $w_1(\mathbf{u}), \ldots, w_L(\mathbf{u})$ w.r.t. $\mathbf{u}$, e.g. Gaussians

  $w_\ell = \exp\left(-(\mathbf{u} - \boldsymbol{\mu}_\ell)^\top \boldsymbol{\Sigma}_\ell^{-1} (\mathbf{u} - \boldsymbol{\mu}_\ell)\right)$



- Spatial convolution with filter g

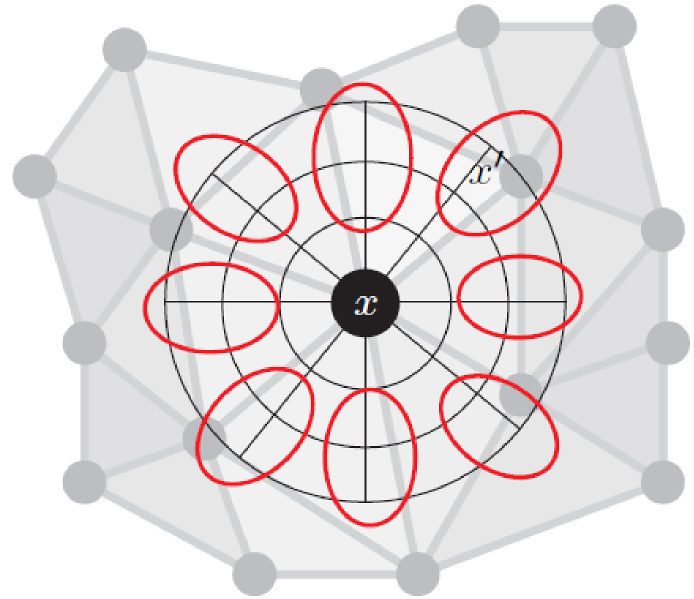
$$(f \star g)(x) \propto \sum_{\ell=1}^{L} g_\ell \int_{\mathcal{X}} w_\ell(\mathbf{u}(x, x')) f(x') \, dx'$$

Masci et al. 2015; Boscaini et al. 2016; Monti et al. 2017

Convolution on manifolds

- Local system of coordinates $\mathbf{u}(x, x')$ around x' (e.g. geodesic polar)
- Local weights $w_1(\mathbf{u}), \ldots, w_L(\mathbf{u})$ w.r.t. $\mathbf{u}$, e.g. Gaussians

  $$w_\ell = \exp\left(-(\mathbf{u} - \boldsymbol{\mu}_\ell)^\top \boldsymbol{\Sigma}_\ell^{-1}(\mathbf{u} - \boldsymbol{\mu}_\ell)\right)$$



- Spatial convolution with filter g

$$(f \star g)(x) \propto \sum_{\ell=1}^{L} g_\ell \underbrace{\int_{\mathcal{X}} w_\ell(\mathbf{u}(x, x')) f(x')\, dx'}_{\text{patch operator}}$$

Masci et al. 2015; Boscaini et al. 2016; Monti et al. 2017

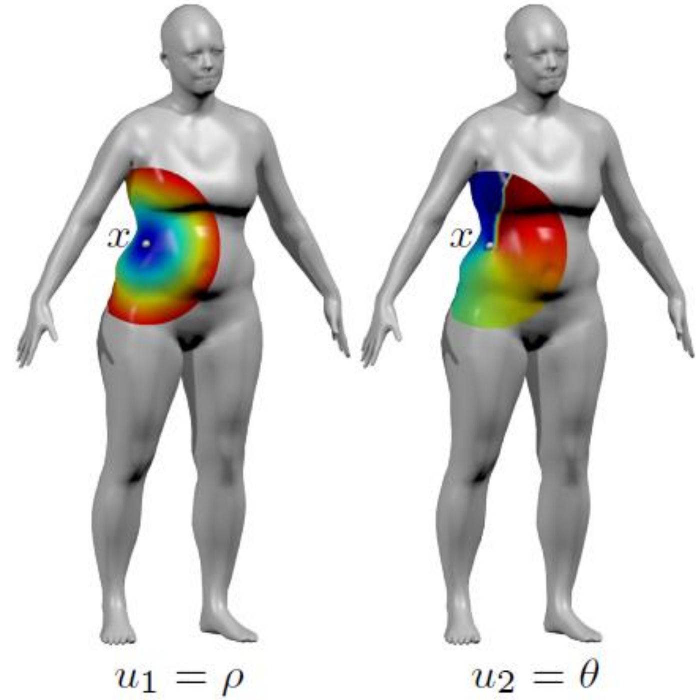
Geodesic polar coordinates

- Geodesic polar coordinates

$$\mathbf{u}(x, x') = (\rho(x, x'), \theta(x, x'))$$

$\rho(x, x')$ geodesic distance from x to x'

$\theta(x, x')$ direction of geodesic from x to x'



Masci et al. 2015; Monti et al. 2017

Geodesic polar coordinates

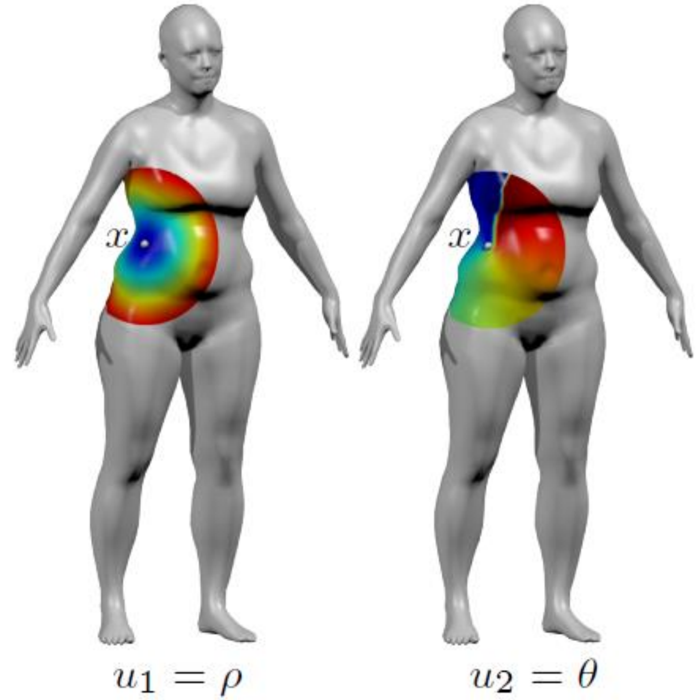
- Geodesic polar coordinates

$$\mathbf{u}(x, x') = (\rho(x, x'), \theta(x, x'))$$

$\rho(x, x')$ geodesic distance from x to x'

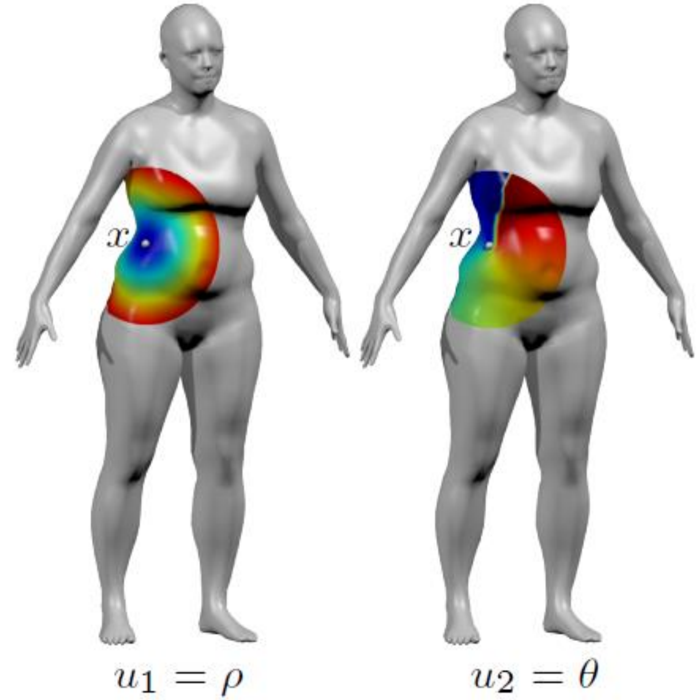
$\theta(x, x')$ direction of geodesic from x to x'

- Orientation ambiguity!



Masci et al. 2015; Monti et al. 2017

Geodesic polar coordinates



- Geodesic polar coordinates

$$\mathbf{u}(x, x') = (\rho(x, x'), \theta(x, x'))$$

$\rho(x, x')$ geodesic distance from x to x'

$\theta(x, x')$ direction of geodesic from x to x'

- Orientation ambiguity!
  - Canionical direction (e.g. intrinsic vector field, max curvature direction)
  - Angular max pooling: apply a rotating filter

$$(f \star g)(x) \propto \max_{\Delta\theta \in [0, 2\pi)} \sum_{\ell=1}^{L} g_\ell \int_{\mathcal{X}} w_\ell(\rho(x, x'), \theta(x, x') + \Delta\theta) f(x')\, dx'$$

Masci et al. 2015; Monti et al. 2017

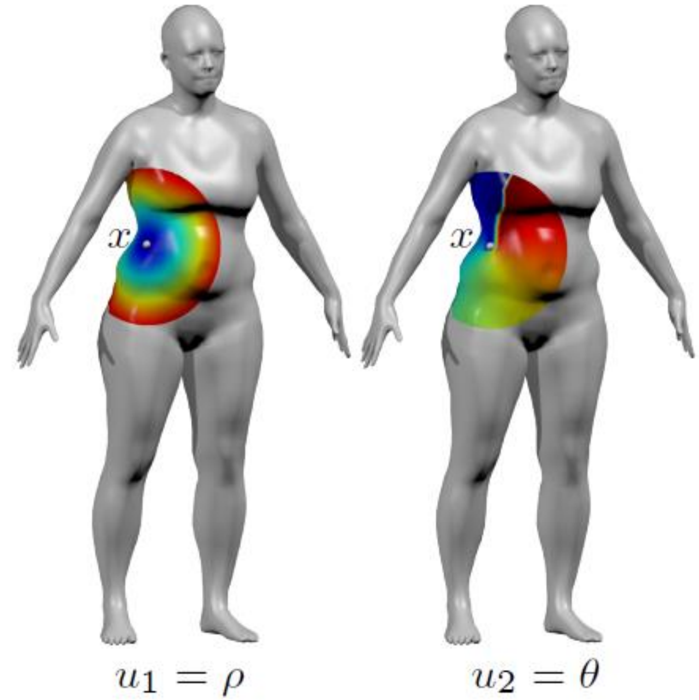
Geodesic polar coordinates

- Geodesic polar coordinates

$$\mathbf{u}(x, x') = (\rho(x, x'), \theta(x, x'))$$

$\rho(x, x')$ geodesic distance from x to x'

$\theta(x, x')$ direction of geodesic from x to x'



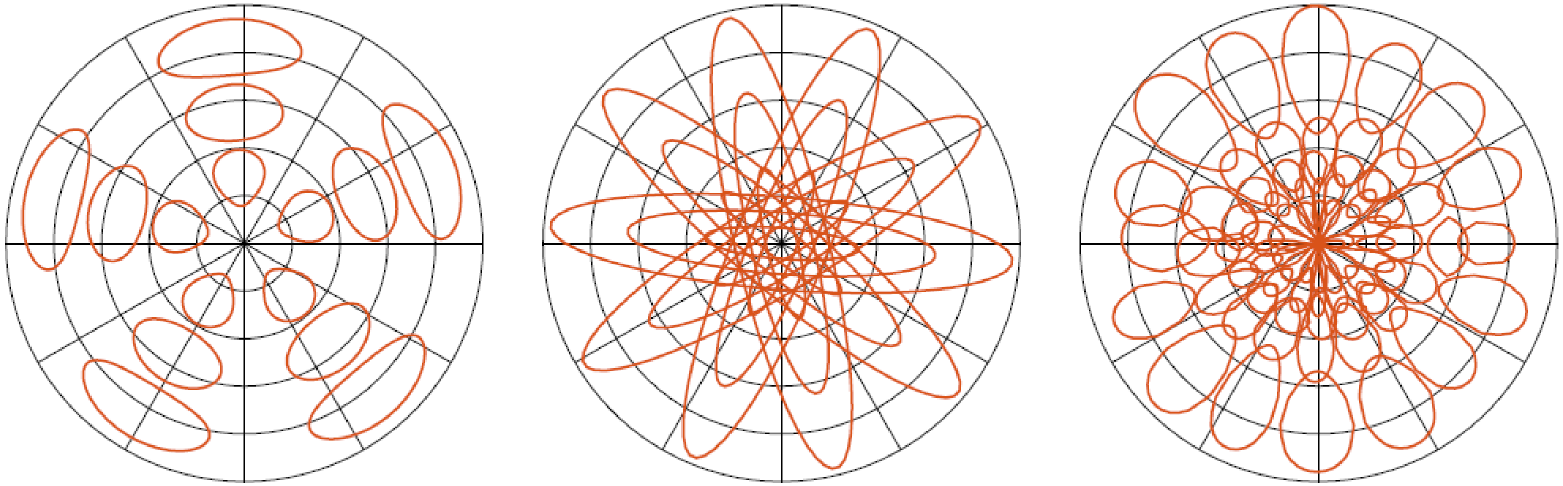
- Orientation ambiguity!
  - Canionical direction (e.g. intrinsic vector field, max curvature direction)
  - Angular max pooling: apply a rotating filter

$$(f \star g)(x) \propto \max_{\Delta\theta \in [0, 2\pi)} \sum_{\ell=1}^{L} g_\ell \int_{\mathcal{X}} w_\ell(\rho(x, x'), \theta(x, x') + \Delta\theta) f(x')\, dx'$$

  - Fourier transform magnitute w.r.t. θ

Masci et al. 2015; Monti et al. 2017

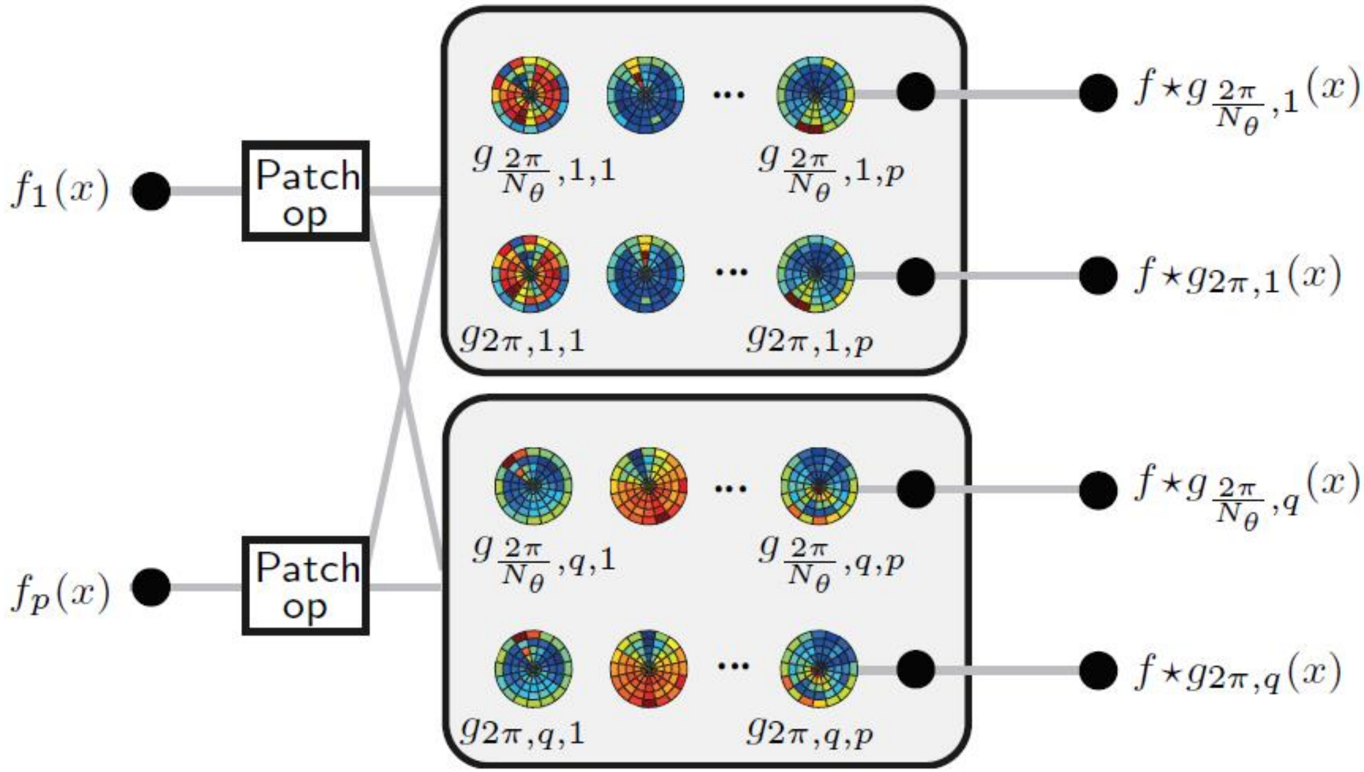
Patch operator weight functions



GCNN ACNN MoNet

Masci et al. 2015 (GCNN); Boscaini et al. 2016 (ACNN); Monti et al. 2017 (MoNet)

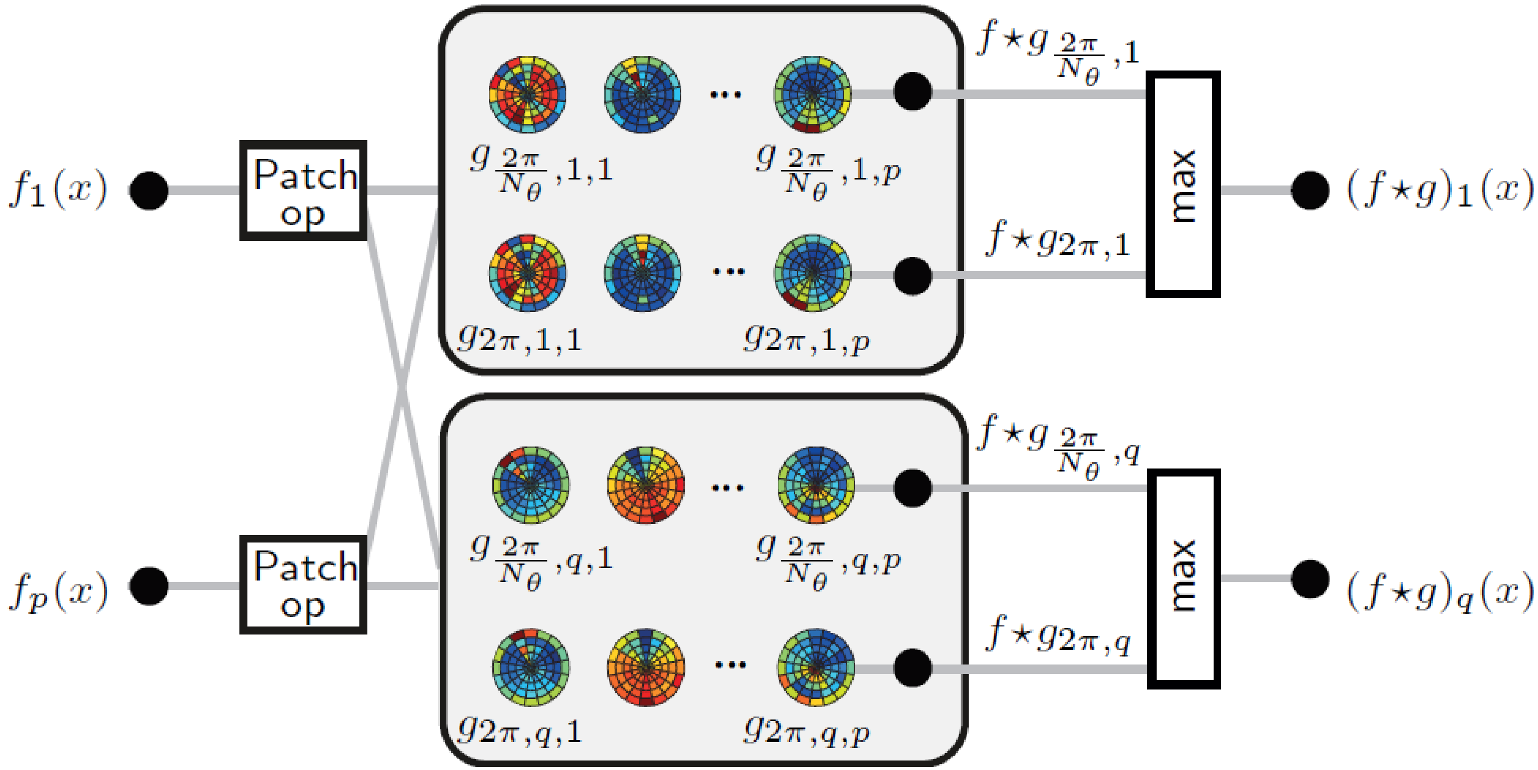
Geodesic convolution layer



Conv. layer $$(f_l \star g)_{\Delta\theta,l}(x) = \xi\left(\sum_{\ell=1}^{p}(f_\ell \star g_{\Delta\theta,l,\ell})(x)\right)$$

Masci et al. 2015

Geodesic convolution layer

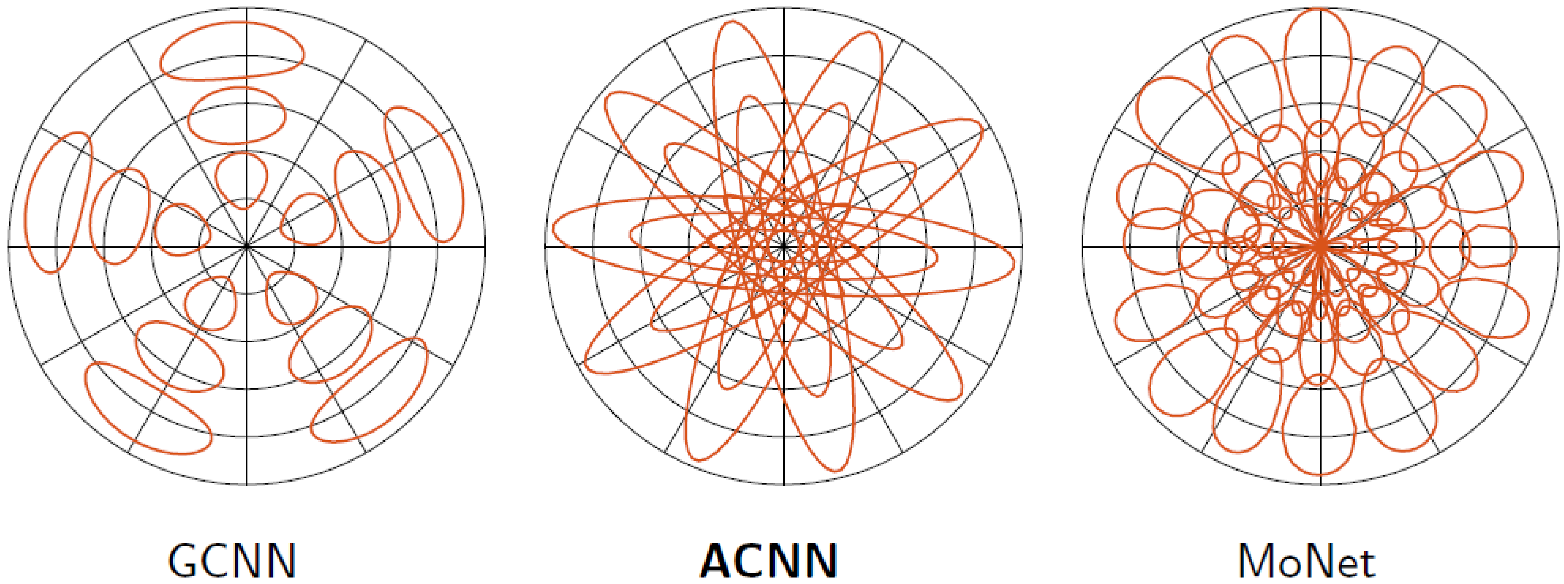


Conv. layer $$(f_l \star g)_{\Delta\theta,l}(x) = \xi\left(\sum_{\ell=1}^{p}(f_\ell \star g_{\Delta\theta,l,\ell})(x)\right)$$

Angular max pooling $$(f \star g)_l(x) = \max_{\Delta\theta}(f \star g)_{\Delta\theta,l}(x)$$

Masci et al. 2015

Patch operator weight functions



Masci et al. 2015 (GCNN); Boscaini et al. 2016 (ACNN); Monti et al. 2017 (MoNet)

Homogeneous diffusion

$$f_t(x) = -\mathrm{div}(c\nabla f(x))$$

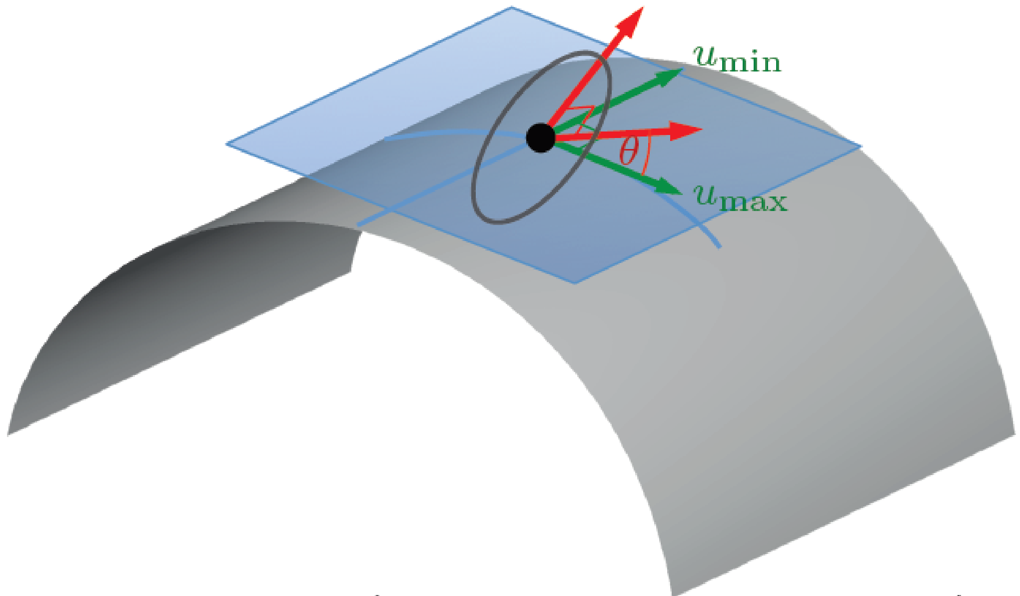
$c =$ thermal diffusivity constant describing heat conduction properties of the material (diffusion speed is equal everywhere)

Anisotropic diffusion

$$f_t(x) = -\mathrm{div}(\mathbf{A}(x)\nabla f(x))$$

$\mathbf{A}(x) =$ heat conductivity tensor describing heat conduction properties of the material (diffusion speed is position + direction dependent)

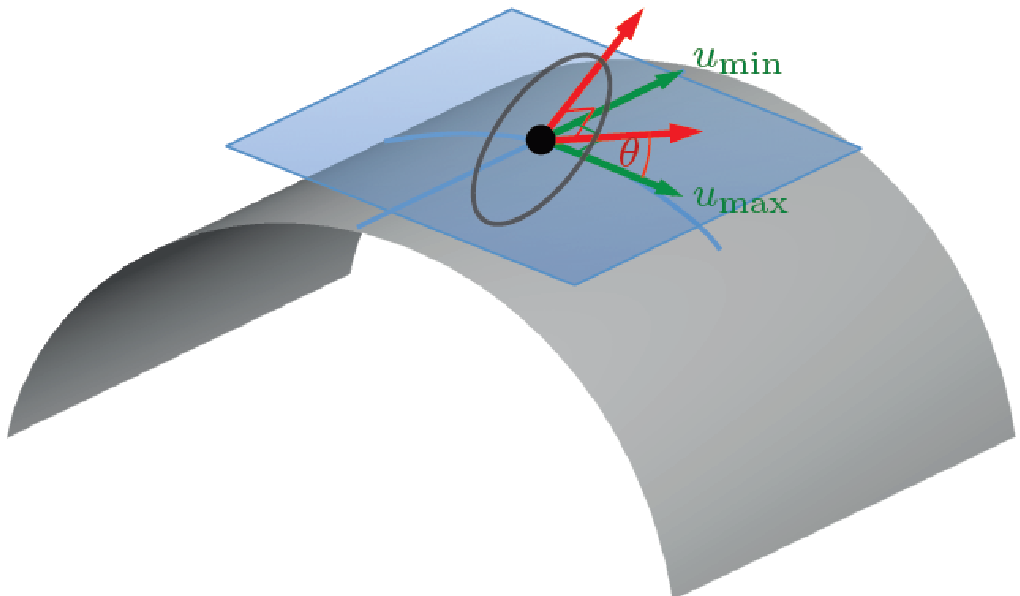
Anisotropic diffusion on manifolds



$$f_t(x) = -\mathrm{div}\left(\mathbf{R}_\theta \begin{pmatrix} \alpha & \\ & 1 \end{pmatrix} \mathbf{R}_\theta^\top \nabla f(x) \right)$$

Andreux et al. 2014; Boscaini et al. 2016

Anisotropic diffusion on manifolds



$$f_t(x) = -\text{div}\left(\underbrace{\mathbf{R}_\theta \begin{pmatrix} \alpha & \\ & 1 \end{pmatrix} \mathbf{R}_\theta^\top}_{\mathbf{D}_{\alpha\theta}(x)} \nabla f(x) \right)$$

- Anisotropic Laplacian $\Delta_{\alpha\theta} f(x) = \text{div}\left(D_{\alpha\theta}(x) \nabla f(x)\right)$
- $\theta =$ orientation w.r.t. max curvature direction
- $\alpha =$ 'elongation'

Andreux et al. 2014; Boscaini et al. 2016

Learnable patch operator

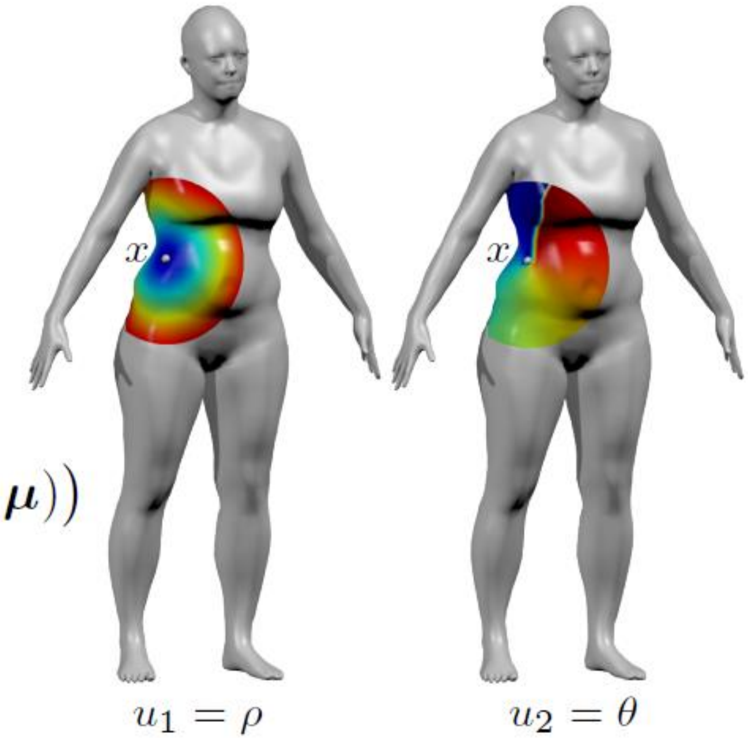
- Geodesic polar coordinates

$$\mathbf{u}(x, y) = (\rho(x, x'), \theta(x, x'))$$

- Gaussian weighting functions

$$w_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\mathbf{u}) = \exp\big(-\tfrac{1}{2}(\mathbf{u}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{u}-\boldsymbol{\mu})\big)$$

with learnable covariance $\boldsymbol{\Sigma}$ and mean $\boldsymbol{\mu}$



Monti et al. 2017

Learnable patch operator

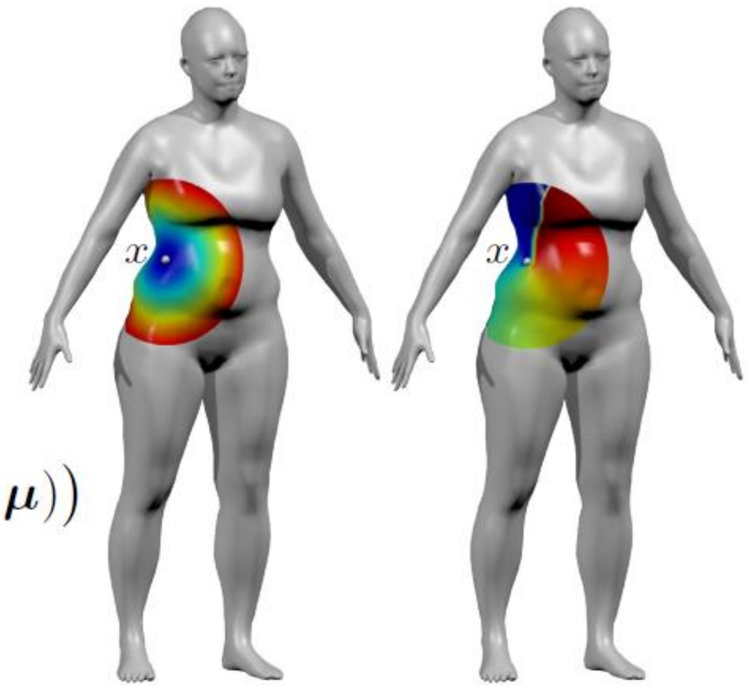
- Geodesic polar coordinates

$$\mathbf{u}(x, y) = (\rho(x, x'), \theta(x, x'))$$

- Gaussian weighting functions

$$w_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\mathbf{u}) = \exp\left(-\tfrac{1}{2}(\mathbf{u}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{u}-\boldsymbol{\mu})\right)$$

with learnable covariance $\boldsymbol{\Sigma}$ and mean $\boldsymbol{\mu}$



Spatial convolution

$$(f \star g)(x) \propto \sum_{\ell=1}^{L} g_\ell \int_{\mathcal{X}} w_{\boldsymbol{\mu}_\ell,\boldsymbol{\Sigma}_\ell}(\mathbf{u}(x, x'))\, f(x')\, dx'$$

where $g_1, \ldots, g_L$ are the spatial filter coefficients and $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_L$ and $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_L$ are patch operator parameters

Monti et al. 2017

Learnable patch operator

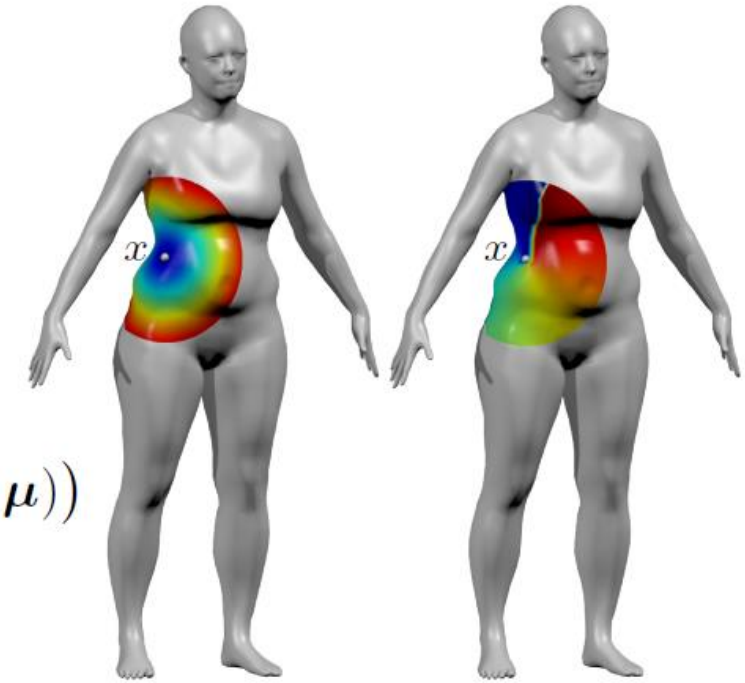
- Geodesic polar coordinates

$$\mathbf{u}(x, y) = (\rho(x, x'), \theta(x, x'))$$

- Gaussian weighting functions

$$w_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\mathbf{u}) = \exp\left(-\tfrac{1}{2}(\mathbf{u}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{u}-\boldsymbol{\mu})\right)$$

  with learnable covariance $\boldsymbol{\Sigma}$ and mean $\boldsymbol{\mu}$



$u_1 = \rho$ $u_2 = \theta$

Spatial convolution

$$(f \star g)(x) \propto \int_{\mathcal{X}} \sum_{\ell=1}^{L} g_\ell w_{\boldsymbol{\mu}_\ell,\boldsymbol{\Sigma}_\ell}(\mathbf{u}(x, x'))\, f(x')\, dx'$$

where $g_1, \ldots, g_L$ are the spatial filter coefficients and $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_L$ and $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_L$ are patch operator parameters

Monti et al. 2017

Mixture Model Network (MoNet)

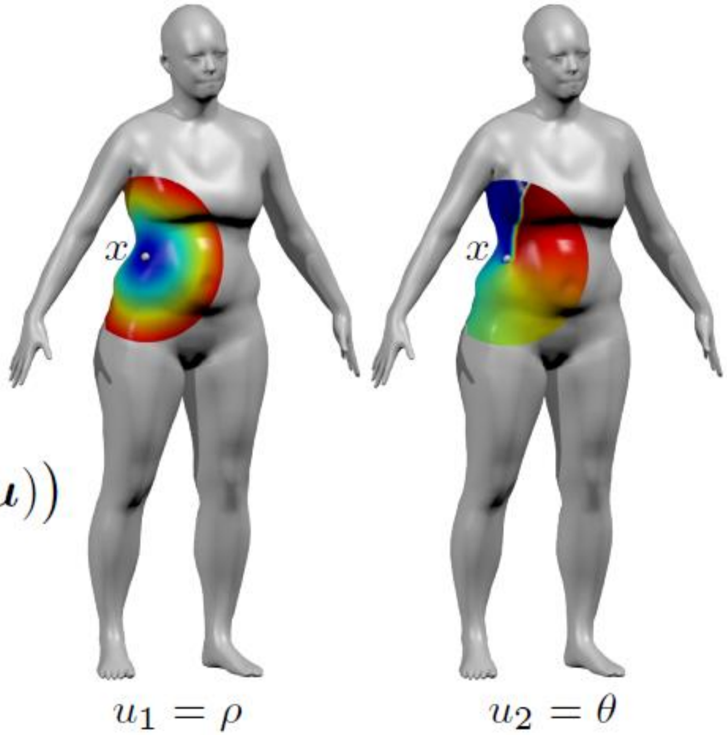
- Geodesic polar coordinates

$$\mathbf{u}(x, y) = (\rho(x, x'), \theta(x, x'))$$

- Gaussian weighting functions

$$w_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\mathbf{u}) = \exp\left(-\tfrac{1}{2}(\mathbf{u}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{u}-\boldsymbol{\mu})\right)$$

with learnable covariance $\boldsymbol{\Sigma}$ and mean $\boldsymbol{\mu}$



Spatial convolution

$$(f \star g)(x) \propto \int_{\mathcal{X}} \underbrace{\sum_{\ell=1}^{L} g_\ell w_{\boldsymbol{\mu}_\ell,\boldsymbol{\Sigma}_\ell}(\mathbf{u}(x, x'))}_{\text{Gaussian mixture}} f(x')\, dx'$$

where $g_1, \ldots, g_L$ are the spatial filter coefficients and $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_L$ and $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_L$ are patch operator parameters

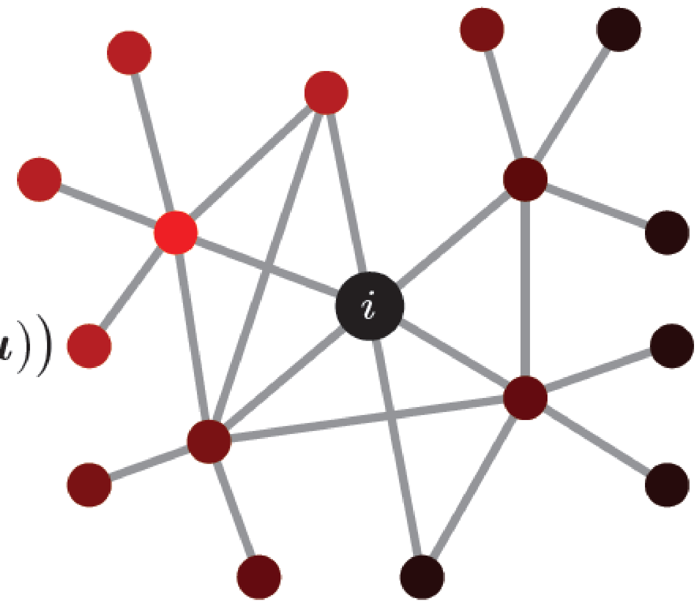
Monti et al. 2017

Mixture Model Network on graphs

- Local coordinates $\mathbf{u}_{ij}$, e.g. vertex degree, geodesic distance,...
- Gaussian weighting functions

  $$w_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\mathbf{u}) = \exp\left(-\tfrac{1}{2}(\mathbf{u}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{u}-\boldsymbol{\mu})\right)$$

  with learnable covariance $\boldsymbol{\Sigma}$ and mean $\boldsymbol{\mu}$



Local coordinates on graph

Spatial convolution

$$(f \star g)_i \propto \sum_{\ell=1}^{L} g_\ell \sum_{j=1}^{n} w_{\boldsymbol{\mu}_\ell,\boldsymbol{\Sigma}_\ell}(\mathbf{u}_{i,j}) f_j$$

where $g_1, \ldots, g_L$ are the spatial filter coefficients and $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_L$ and $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_L$ are patch operator parameters

Monti et al. 2017

MoNet as generalization of previous methods

Method	**Coordinates** $\mathbf{u}(x, x')$	**Weight function** $w_{\mathbf{\Theta}}(\mathbf{u})$
CNN[1]	$\mathbf{u}(x') - \mathbf{u}(x)$	$\delta(\mathbf{u} - \mathbf{v})$ fixed parameters $\mathbf{\Theta} = \mathbf{v}$
GCN[2]	$\deg(x), \deg(x')$	$\left(1 - \lvert 1 - \frac{1}{\sqrt{u_1}} \rvert\right) \left(1 - \lvert 1 - \frac{1}{\sqrt{u_2}} \rvert\right)$
GCNN[3]	$\rho(x, x'), \theta(x, x')$	$\exp\left(-\frac{1}{2}(\mathbf{u} - \mathbf{v})^\top \begin{pmatrix} \sigma_\rho^2 & \\ & \sigma_\theta^2 \end{pmatrix}^{-1} (\mathbf{u} - \mathbf{v})\right)$ fixed parameters $\mathbf{\Theta} = (\mathbf{v}, \sigma_\rho, \sigma_\theta)$
ACNN[4]	$\rho(x, x'), \theta(x, x')$	$\exp\left(-t\mathbf{u}^\top \mathbf{R}_\varphi \begin{pmatrix} \alpha & \\ & 1 \end{pmatrix} \mathbf{R}_\varphi^\top \mathbf{u}\right)$ fixed parameters $\mathbf{\Theta} = (\alpha, \varphi, t)$
MoNet[5]	$\rho(x, x'), \theta(x, x')$	$\exp\left(-\frac{1}{2}(\mathbf{u} - \boldsymbol{\mu})^\top \mathbf{\Sigma}^{-1} (\mathbf{u} - \boldsymbol{\mu})\right)$ learnable parameters $\mathbf{\Theta} = (\boldsymbol{\mu}, \mathbf{\Sigma})$

Some CNN models can be considered as particular settings of MoNet with weighting functions of different form

Methods: [1]LeCun et al. 1998; [2]Kipf, Welling 2016; [3]Masci et al. 2015; [4]Boscaini et al. 2016; [5]Monti et al. 2017

Spectral vs Spatial methods

ChebNet filter

$$\mathbf{h} = \tau_{\boldsymbol{\alpha}}(\boldsymbol{\Delta})\mathbf{f}$$

Spatial filter

$$\mathbf{h} = (\mathcal{D}\mathbf{f})\mathbf{g}$$

Defferrard, Bresson, Vandergheynst 2016; Masci et al. 2015; Monti et al. 2017

Spectral vs Spatial methods

ChebNet filter

$$\mathbf{h} = \sum_{\ell=0}^{r} \alpha_\ell \boldsymbol{\Delta}^\ell \mathbf{f}$$

Spatial filter

$$\mathbf{h} = (\mathcal{D}\mathbf{f})\mathbf{g}$$

Defferrard, Bresson, Vandergheynst 2016; Masci et al. 2015; Monti et al. 2017

Spectral vs Spatial methods

ChebNet filter

$$h_i = \sum_{\ell=0}^{r} \alpha_\ell (\mathbf{\Delta}^\ell \mathbf{f})_i$$

Spatial filter

$$h_i = \sum_{\ell=1}^{L} g_\ell (\mathbf{W}_\ell \mathbf{f})_i$$

ChebNet is a particular setting of spatial convolution with local weighting functions given by the powers of the Laplacian $\mathbf{W}_\ell = \mathbf{\Delta}^\ell$

Defferrard, Bresson, Vandergheynst 2016; Masci et al. 2015; Monti et al. 2017

Graph Attention Networks (GAT)

Main idea: neighborhood average

$$\mathbf{f}_i' = \sum_{j:(i,j)\in\mathcal{E}} \alpha_{ij}\mathbf{f}_j$$

weighted by attention score

$$\alpha_{ij} = \frac{e^{\xi([\mathbf{f}_i\mathbf{W},\, \mathbf{f}_j\mathbf{W}]\mathbf{a})}}{\sum_{k:(i,k)\in\mathcal{E}} e^{\xi(\,[\mathbf{f}_i\mathbf{W},\, \mathbf{f}_k\mathbf{W}]\mathbf{a})}}$$

which is a learnable transformation of the local features with learnable parameters $\mathbf{W}$, $\mathbf{a}$

Veličković et al. 2018

Graph Attention Networks (GAT)

Main idea: neighborhood average

$$\mathbf{f}'_i = \sum_{j:(i,j)\in\mathcal{E}} \alpha_{ij}\mathbf{f}_j$$

weighted by attention score

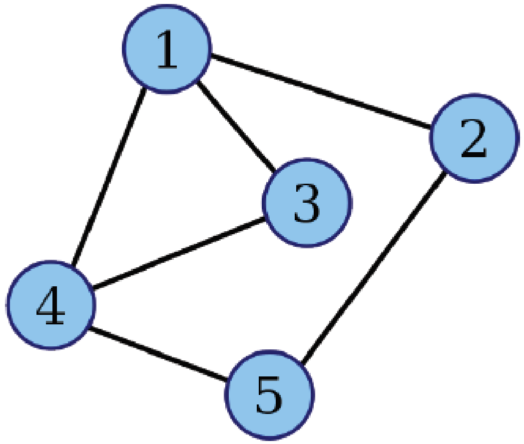
$$\alpha_{ij} = \frac{e^{\xi([\mathbf{f}_i\mathbf{W},\,\mathbf{f}_j\mathbf{W}]\mathbf{a})}}{\sum_{k:(i,k)\in\mathcal{E}} e^{\xi(\,[\mathbf{f}_i\mathbf{W},\,\mathbf{f}_k\mathbf{W}]\mathbf{a})}}$$

which is a learnable transformation of the local features with learnable parameters $\mathbf{W}$, $\mathbf{a}$

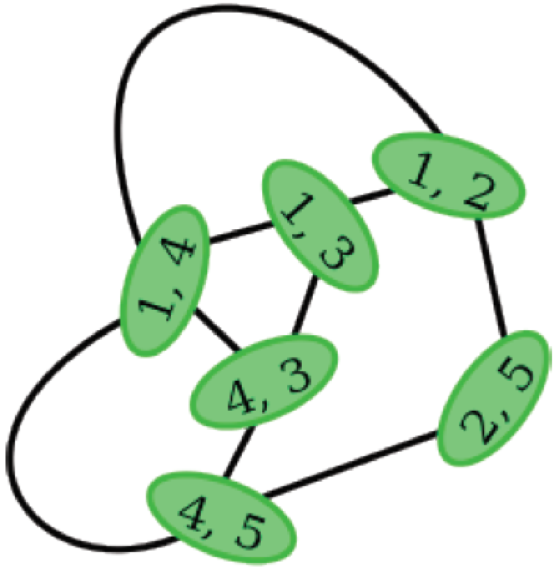
Particular case of MoNet-type architectures!

Veličković et al. 2018

Primal and Dual graphs



Primal graph
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$



Dual or line graph
$\tilde{\mathcal{G}} = (\tilde{\mathcal{V}} = \mathcal{E}, \tilde{\mathcal{E}})$

Harary, Norman 1960

Dual/Primal Graph CNN (DPGCNN)

Alternate GAT-type convolutions applied on primal and dual graphs

- Dual convolution on $\tilde{\mathcal{G}}$:

$$\tilde{\mathbf{f}}'_{ij} = \xi\left(\sum_{r\in\mathcal{N}_i}\tilde{\alpha}_{ij,ir}\tilde{\mathbf{f}}_{ir}\tilde{\mathbf{W}} + \sum_{t\in\mathcal{N}_j}\tilde{\alpha}_{ij,tj}\tilde{\mathbf{f}}_{tj}\tilde{\mathbf{W}}\right)$$

$$\tilde{\alpha}_{ij,ik} = \frac{e^{\xi([\tilde{\mathbf{f}}_{ij}\tilde{\mathbf{W}},\,\tilde{\mathbf{f}}_{ik}\tilde{\mathbf{W}}]\tilde{\mathbf{a}})}}{\sum_{r\in\mathcal{N}_i} e^{\xi([\tilde{\mathbf{f}}_{ij}\tilde{\mathbf{W}},\,\tilde{\mathbf{f}}_{ir}\tilde{\mathbf{W}}]\tilde{\mathbf{a}})} + \sum_{t\in\mathcal{N}_j} e^{\xi([\tilde{\mathbf{f}}_{ij}\tilde{\mathbf{W}},\,\tilde{\mathbf{f}}_{tj}\tilde{\mathbf{W}}]\tilde{\mathbf{a}})}}$$

- Primal convolution on $\mathcal{G}$:

$$\mathbf{f}'_i = \xi\left(\sum_{j\in\mathcal{N}_i}\alpha_{ij}\mathbf{f}_j\mathbf{W}\right) \qquad \alpha_{ij} = \frac{e^{\xi(\tilde{\mathbf{f}}'_{ij}\mathbf{a})}}{\sum_{k\in\mathcal{N}_i} e^{\xi(\tilde{\mathbf{f}}'_{ik}\mathbf{a})}}$$

Monti et al. 2018

Example: citation networks

Method	**Cora**[1]	**CiteSeer**[2]
Manifold Regularization[3]	59.5%	60.1%
Semidefinite Embedding[4]	59.0%	59.6%
Label Propagation[5]	68.0%	45.3%
DeepWalk[6]	67.2%	43.2%
Planetoid[7]	75.7%	64.7%
GCN[8]	81.6%	70.3%
MoNet[9]	81.7%	–
GAT[10]	83.0%	72.5%
DPGCN[11]	**83.3%**	**72.6%**

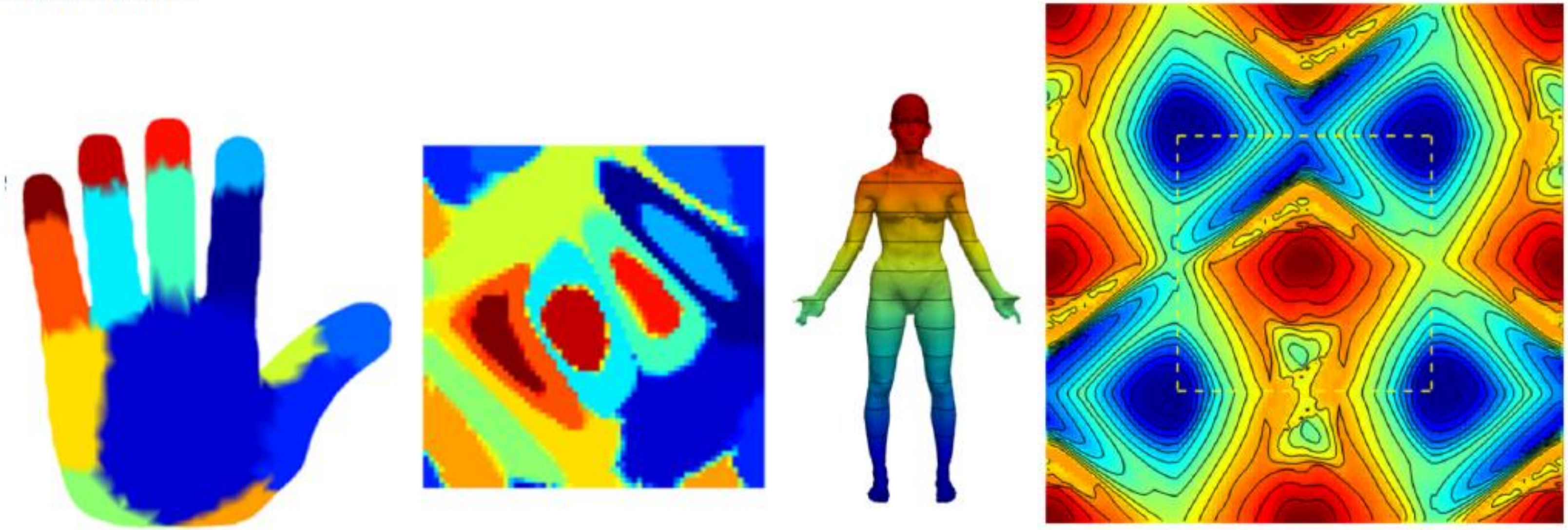
Classification accuracy of different methods on citation network datasets

Data: [1,2]Sen et al. 2008; methods: [3]Belkin et al. 2006; [4]Weston et al. 2012; [5]Zhu et al. 2003; [6]Perozzi et al. 2014; [7]Yang et al. 2016; [8]Kipf, Welling 2016 ; [9]Monti et al. 2017; [10]Veličković et al. 2018; [11]Monti et al. 2018

Parametric domain geometric deep learning methods

Global parametrization

Map the input surface to some parametric domain with shift-invariant structure



- ☺ Allows to use standard CNNs (pull back convolution from the parametric space)
- ☺ Guaranteed invariance to some classes of transformations
- ☹ Parametrization may not be unique
- ☹ Embedding may introduce distortion

Sinha et al. 2016; Maron et al. 2017; Ezuz et al. 2017