Visual Computing Seminar

EPFL, Switzerland, 2019-11-15

DeepSphere

a graph-based spherical CNN

Michaël Defferrard

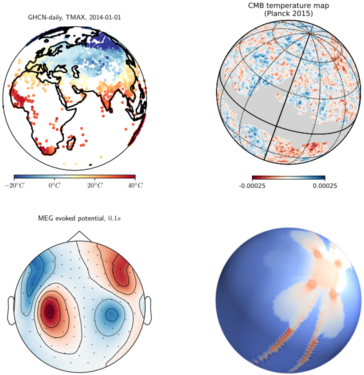
Joint work with Martino Milani,
Frédérick Gusset, Nathanaël Perraudin.

EPFL

Problem: learning from spherical data

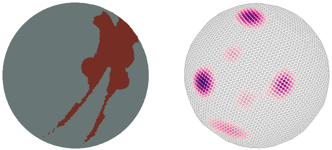
$x : S^2 \to \mathbb{R}^d$

intrinsic	projection



$f(x)$

classification	regression
$\begin{cases} \text{“sleepy”} \\ \text{“alert”} \end{cases}$	$\sigma_8, \ \Omega_m$

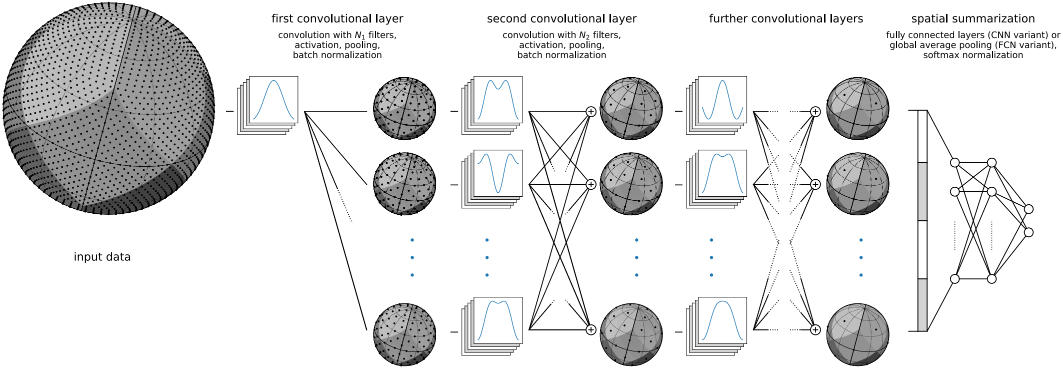


More:

- learn a representation of maps
- learn a metric between maps

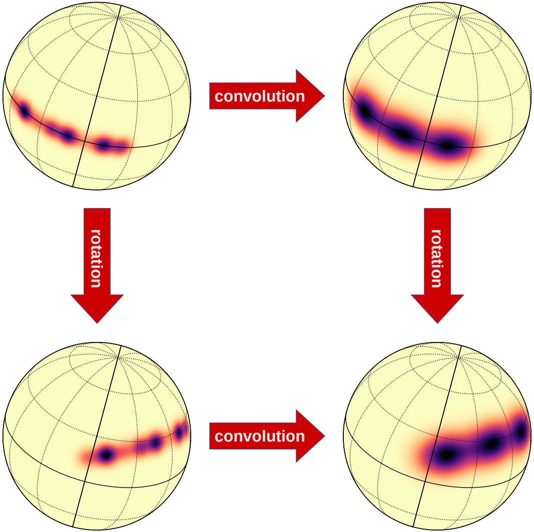
Acoustic field from Simeoni et al. 2019. 3D shape from Esteves et al. 2018.

Solution: spherical neural networks



Why a NN? We don't know the statistics we should be looking for.

Desideratum 1: equivariant to rotations



- *Equivariance* for dense tasks:
  $f(Rx) = Rf(x)\ \forall R \in SO(3)$
- *Invariance* for global tasks:
  $f(Rx) = f(x)\ \forall R \in SO(3)$

Why exploit symmetries?

- reduced sample complexity
- generalization guarantee.

Desideratum 2: scalable

- Many inferences needed for training.
- Increasingly larger maps.

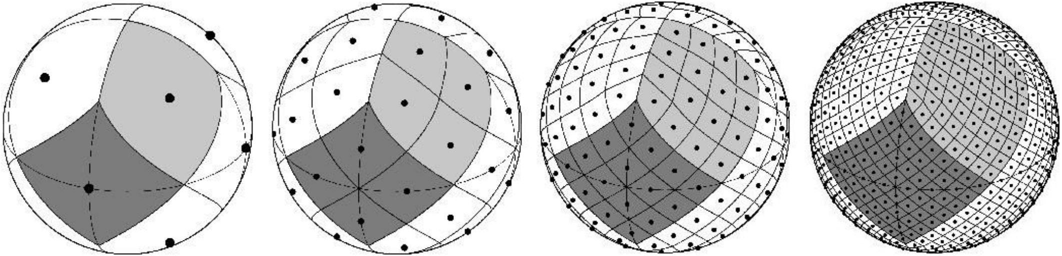
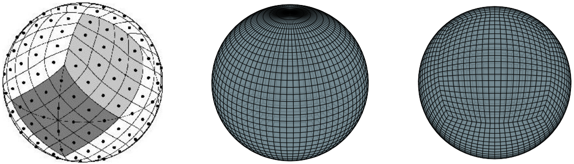
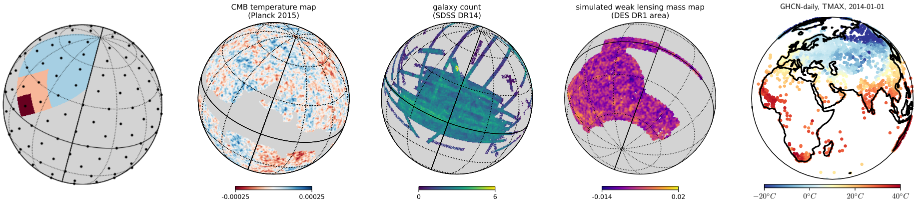


Figure from `https://healpix.sourceforge.io`.

Desideratum 3: flexible sampling



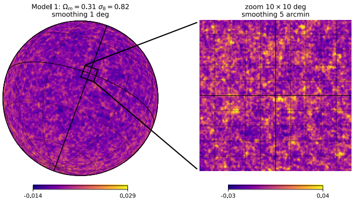
Sampling schemes: HEALPix, equiangular, icosahedral, cubed-sphere, etc.

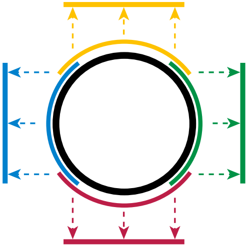


Partial and irregular sampling.

Equiangular and cubed-sphere figures from Boomsma and Frellsen 2017.

Method 1: 2D projections





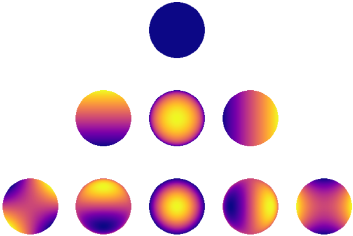
Manifold is locally Euclidean!
Project on 2D tangent planes.

Desiderata

- ⊖ Rotation equivariance: hard to properly glue planes together.
- ⊕ Scalability: well developed NN architectures and implementations. Some wastes at boundaries.
- ⊖ Flexibility: only handle compact subspaces.

Charting figure from `https://en.wikipedia.org/wiki/manifold`.

Method 2: discretization of continuous domain



Spectral decomposition.

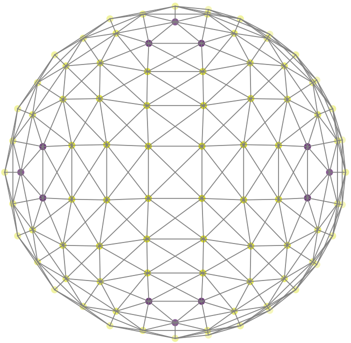
Discretize the continuous problem!
Compute the spherical harmonic transform (SHT), filter in the spectrum.

Desiderata

- ⊕ Rotation equivariance: well understood theory.
- ⊖ SHT is expensive. Fast transforms exist for some samplings.
- ⊖ Flexibility: unused pixels are mostly wasted.

Figure from `https://rodluger.github.io/starry`.

Our proposition: discrete domain



graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$ with
$A_{ij} = \exp(-d(z_i, z_j)/\sigma)$

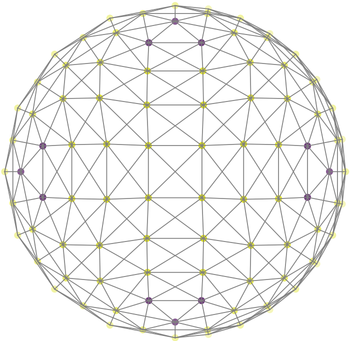
Domain set of pixels $\mathcal{V}$
topology given by geodesic distances

Data function $x : \mathcal{V} \to \mathbb{R}$
seen as $x \in \mathbb{R}^{N_{pix}}$

Method in a nutshell

1. Model the topology by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$.
2. From it stems a Laplacian, e.g. $L = D - A$.
3. The Fourier basis diagonalizes the Laplacian.
4. Convolution is a multiplication in Fourier.
5. Spatial implementation for speed,
   e.g. $g_\alpha(L)x = \sum_k \alpha_k L^k x$.

Graph construction



- undirected weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$

- set $\mathcal{V}$ of $n = |\mathcal{V}|$ vertices
- set $\mathcal{E}$ of $m = |\mathcal{E}|$ edges

- weighted adjacency matrix
  $A_{ij} = \exp(-\|z_i - z_j\|_2^2/\sigma^2)$
- diagonal degree matrix $D_{ii} = \sum_j A_{ij}$

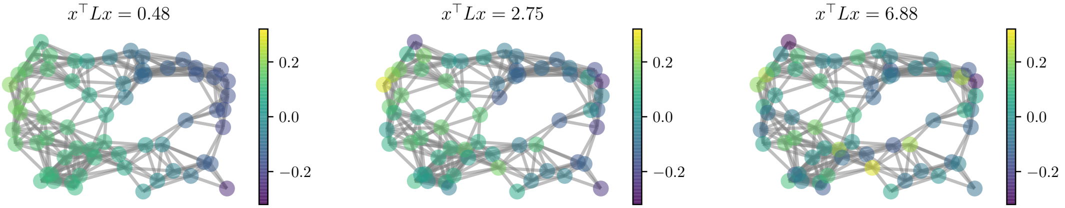
- combinatorial Laplacian $L = D - A$

Graph Laplacian

Shuman et al. 2013

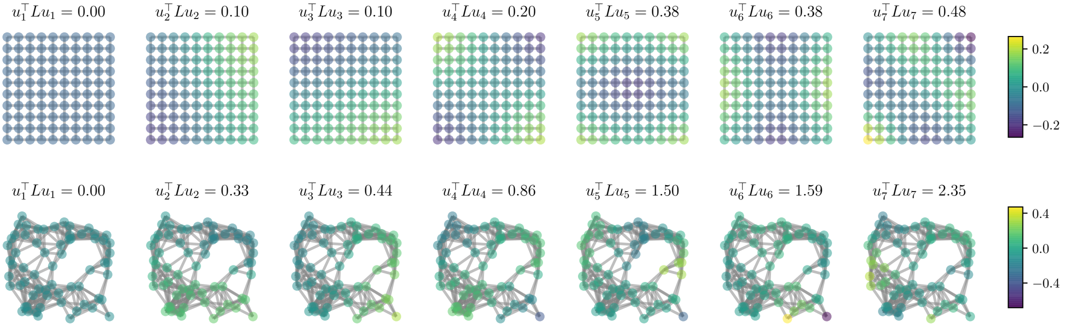
$$L = S^\top S \text{ with } S \in \mathbb{R}^{m\times n} \text{ the incidence matrix}$$

- Gradient: $z = \nabla_\mathcal{G} x = Sx \in \mathbb{R}^m$ $\qquad (Sx)_{(i,j)} = \sqrt{A_{ij}}(x_i - x_j)$
- Divergence: $\operatorname{div}_\mathcal{G} z = S^\top z \in \mathbb{R}^n$ $\qquad (Sz)_i = \sum_j \sqrt{A_{ij}} z_{(i,j)}$
- Dirichlet energy: $\|\nabla_\mathcal{G} x\|_2 = x^\top L x = \sum_{i,j} A_{ij}(x_i - x_j)^2$



Definition: the Fourier basis diagonalizes the Laplacian operator $\rightarrow L = U \Lambda U^\top$

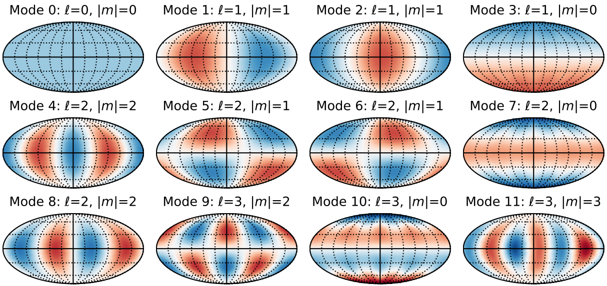
- Graph Fourier basis $U = [u_1, \dots, u_n] \in \mathbb{R}^{n \times n}$
- Graph "frequencies" $\Lambda = \mathrm{diag}(\lambda_1, \dots, \lambda_n)$

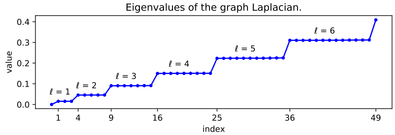


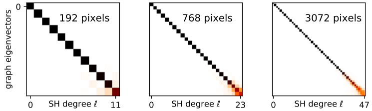
Graph Fourier basis on the sphere

Perraudin et al. 2018

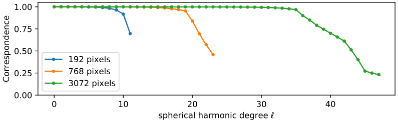
- Fourier modes resemble spherical harmonics.
- Graphs approximate manifolds.





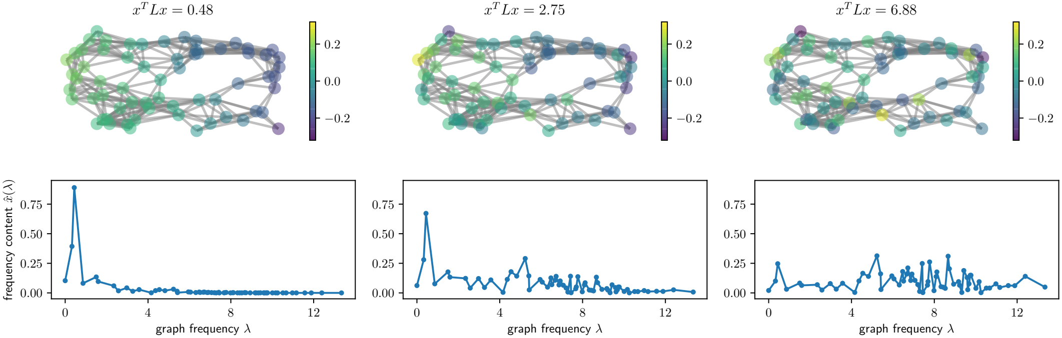


Spherical harmonic transform of the graph Laplacian's eigenvectors.



Graph Fourier Transform
Shuman et al. 2013

- Graph signal $x : \mathcal{V} \to \mathbb{R}$ seen as $x \in \mathbb{R}^n$
- Transform: $\hat{x} = \mathcal{F}_{\mathcal{G}}(x) = U^\top x \in \mathbb{R}^n$
- Inverse: $x = \mathcal{F}_{\mathcal{G}}^{-1}(x) = U\hat{x} = UU^\top x = x$



Filtering

kernel a function $g : \mathbb{R}^+ \to \mathbb{R}$ that defines the action of the filter

filter the operator $g(L)$ acting on signals

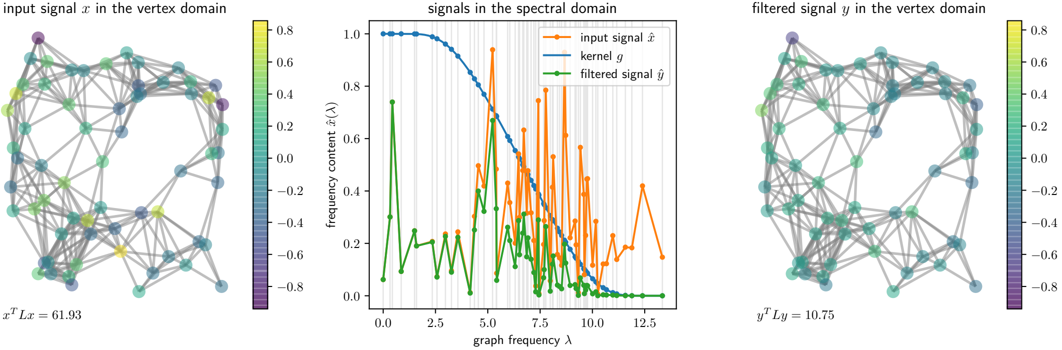
A signal $x \in \mathbb{R}^n$ is filtered by the kernel g as:

$$y = g(L)x = Ug(\Lambda)U^\top x$$

Step by step

1. take the Fourier transform (a change of coordinate): $\hat{x} = U^\top x$
2. take an element-wise product in the spectrum: $\hat{y} = (g(\lambda_1), \ldots, g(\lambda_n)) \odot \hat{x}$
3. take the inverse Fourier transform: $y = U\hat{y}$

Filtering example



Observation: the *low-pass filtered* signal y is much smoother[1] than x!

[1] lower Dirichlet energy

Fast filtering

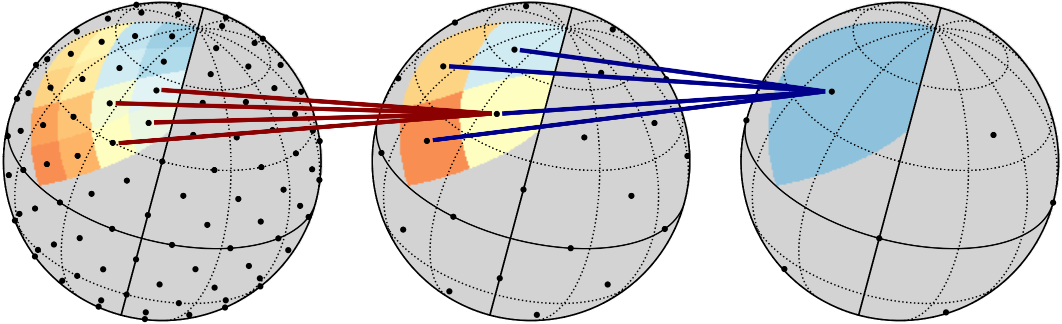
Defferrard et al. 2016

$$y = g(L)x = \left(\sum_{k=0}^{K} \alpha_k L^k\right) x = \sum_{k=0}^{K} \bar{x}_k$$

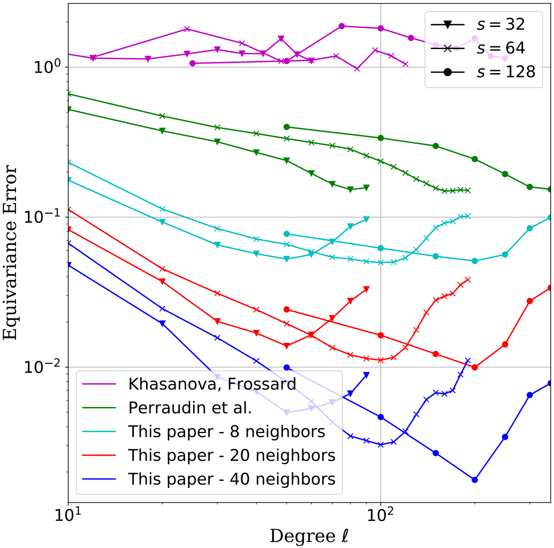
Recursive computation of $\bar{x}_0 = x, \bar{x}_k = L\bar{x}_{k-1}$.

- K-localized
- Learning complexity is $\mathcal{O}(K)$
- Computational complexity is $\mathcal{O}(K|\mathcal{E}|)$

Pooling



Desideratum 1: equivariant to rotations



Equivariance error:

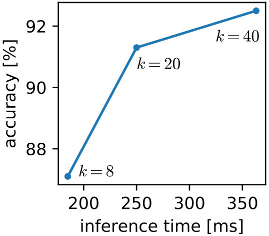
$$\mathbb{E}_{x,R}\left(\frac{\|RLx - LRx\|}{\|Lx\|}\right)^2$$

Clear tradeoff between cost (number of neighbors k and vertices $n \propto s^2$) and equivariance error!

Desideratum 1: it matters!

	accuracy	time
Perraudin et al. 2018, 2D CNN baseline	54.2	104 ms
Perraudin et al. 2018, CNN variant, $k = 8$	62.1	185 ms
Perraudin et al. 2018, FCN variant, $k = 8$	83.8	185 ms
$k = 8$ neighbors, optimal t	87.1	185 ms
$k = 20$ neighbors, optimal t	91.3	250 ms
$k = 40$ neighbors, optimal t	92.5	363 ms

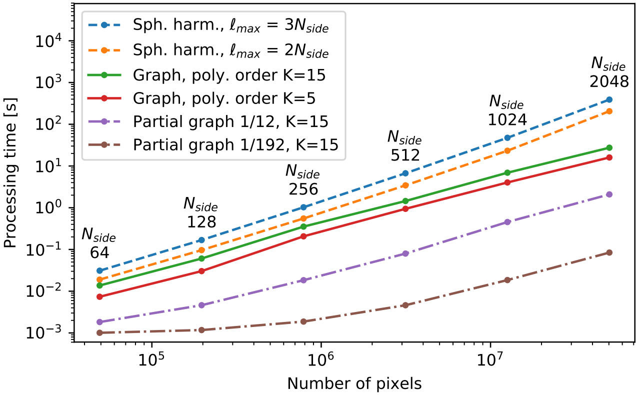
Lower equivariance error translates to higher performance.



Tradeoff between cost and accuracy.

Desideratum 2: scalable

- Graph convolutions[2] cost $\mathcal{O}(N_{pix})$.
- Spherical convolutions cost $\mathcal{O}(N_{pix}^2)$ in general, $\mathcal{O}(N_{pix}^{3/2})$ for some samplings.

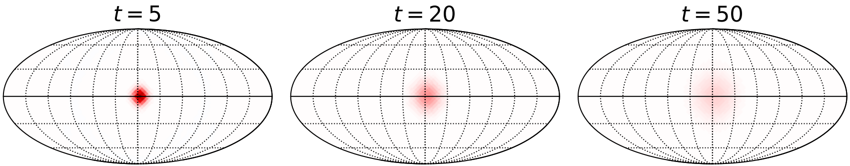


[2]They only involve the multiplications of vectors by a sparse matrix with $O(N_{pix})$ non-zeros.

Desideratum 2: it matters!

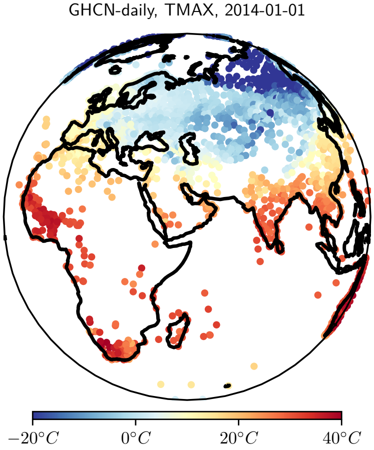
	performance		size	speed	
	F1	mAP	params	inference	training
$SO(3)$ [Cohen et al.]	-	0.676	1400 k	19.0 ms	50 h
S^2 [Esteves et al.]	79.36	0.685	500 k	9.8 ms	3 h
graph [DeepSphere]	80.65	0.686	190 k	1.6 ms	40 m

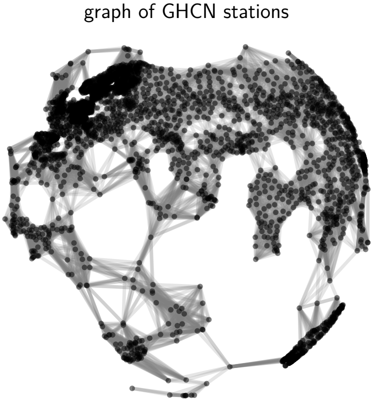
Classification of 3D shapes (SHREC'17): anisotropy is an unnecessary price to pay.



Example graph filters (heat kernel).

Desideratum 3: flexible sampling



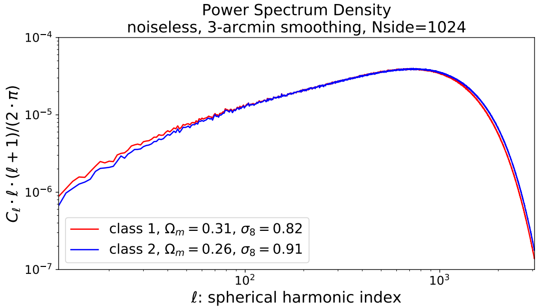


This extreme flexibility probably breaks rotation equivariance.

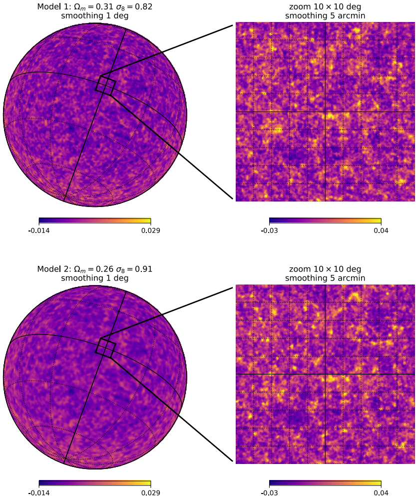
Experiment: discrimination of cosmological models

Classification of convergence maps created from two sets of cosmological parameters.

$$(\Omega_m, \sigma_8) = (0.31, 0.82) \text{ or } (0.26, 0.91)$$

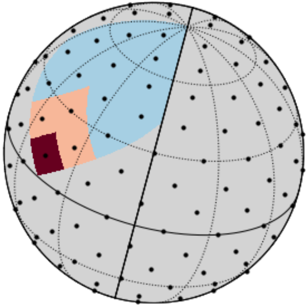


Ω_m, σ_8, smoothing chosen to get identical PS.



Maps with identical initial conditions.

Experiment: discrimination of cosmological models (data)



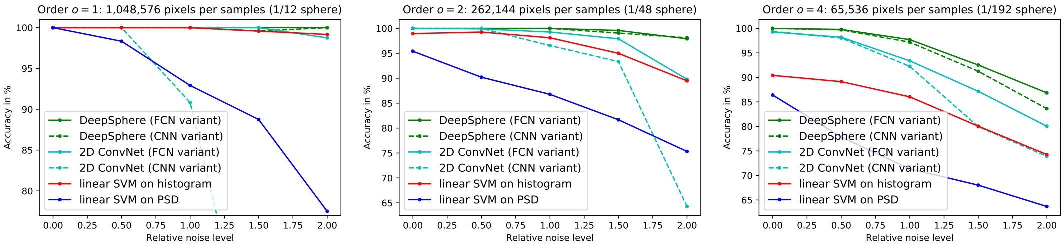
- 30 N-body simulations per class
  $\Rightarrow$ 60 full-sky maps (32 train, 8 val, 20 test)

- resolution of $N_{side} = 1024$
  $\Rightarrow 12 \cdot 10^6$ pixels per map

- How many samples do we have?
  Amount of supervision is $\mathcal{O}(N_{pix})$.

#samples	#pixels per sample	
720	$1 \cdot 10^6$	$(^1/_{12} \approx 8\%)$
2,900	$260 \cdot 10^3$	$(^1/_{48} \approx 2\%)$
12,000	$65 \cdot 10^3$	$(^1/_{192} \approx 0.5\%)$

Experiment: discrimination of cosmological models (results)

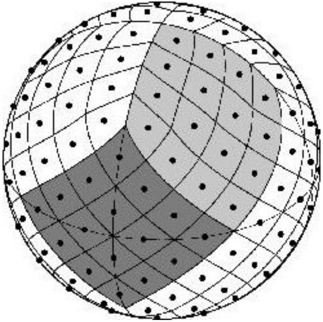


- Difficulty controlled by #pixels per sample and amount of noise.
- Better performance than SVM on PSDs and histograms.
  Those statistics destroy too much information.
- Better performance than ConvNet on 2D projections.
  Curvature plays a role even on 0.5% of the sphere.
- Global pooling better than fully connected layer. Why?

Spatial summarization

Goal: get rid of the spatial dimension. How?

x



fully connected $f(x) = Ax$

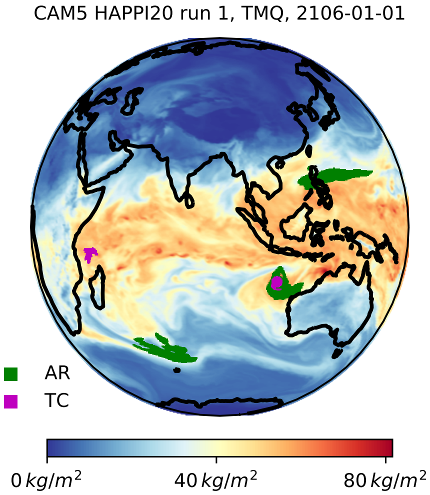
global pooling $f(x) = \sum_i x_i$

$\Rightarrow$ invariant to rotations!

Experiment: climate event segmentation

Segmentation of extreme climate events: tropical cyclones (TC) and atmospheric rivers (AR).

- $>$1M spherical maps
- down-sampled to 10k pixels (original 900k)
- 0.1% TC, 2.2% AR, 97.7% background



Experiment: climate event segmentation (results)

	accuracy	mAP
Jiang et al. 2019 (rerun)	94.95	38.41
Cohen et al. 2019 (S2R)	97.5	68.6
Cohen et al. 2019 (R2R)	97.7	75.9
DeepSphere (weighted loss)	97.8 ± 0.3	77.15 ± 1.94
DeepSphere (non-weighted loss)	87.8 ± 0.5	89.16 ± 1.37

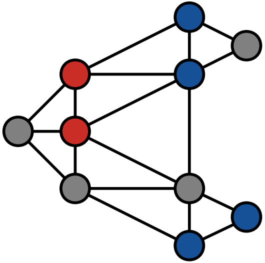
Mean accuracy (over TC, AR, BG) and mean average precision (over TC and AR).

- More generality is not necessarily helpful.
- Check your loss!

Take home

- DeepSphere is an efficient CNN for spherical data.
- Graphs encode the geometry. Graph NNs exploit that structure.
- Symmetries (invariants) are a principled way to design NNs.
- Measurements and computations are discrete.

THANKS QUESTIONS?



Slides `https://doi.org/10.5281/zenodo.3548192`

Papers Defferrard, Milani, Gusset, Perraudin, DeepSphere: a graph-based spherical CNN, under review at ICLR, 2020.

Defferrard, Perraudin, Kacprzak, Sgier, DeepSphere: towards an equivariant graph-based spherical CNN, RLGM workshop at ICLR, 2019.

Perraudin, Defferrard, Kacprzak, Sgier, DeepSphere: Efficient spherical Convolutional Neural Network with HEALPix sampling for cosmological applications, Astronomy and Computing, 2019.

Defferrard, Bresson, Vandergheynst, Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering, NIPS, 2016.

Codes `https://github.com/SwissDataScienceCenter/DeepSphere`
`https://github.com/mdeff/cnn_graph`
`https://github.com/epfl-lts2/pygsp`