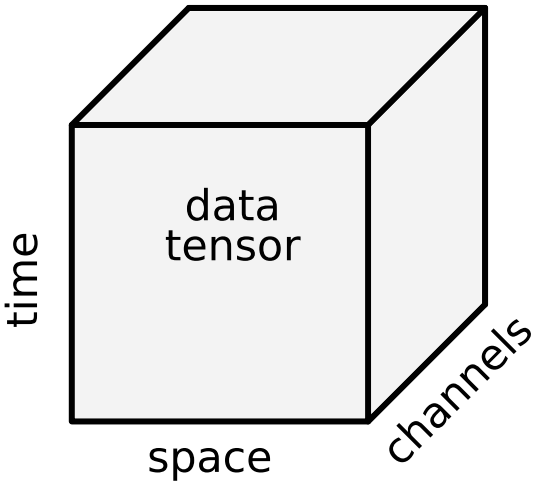
Qualcomm interview online, 2021-11-22

Leveraging topology, geometry, and symmetries for efficient Machine Learning

Michaël Defferrard

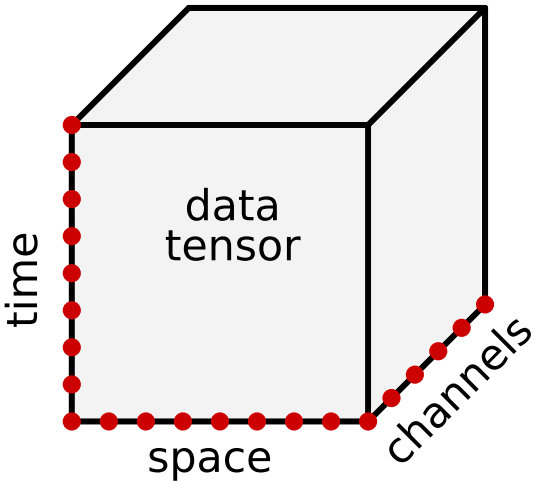
EPFL

Structured data



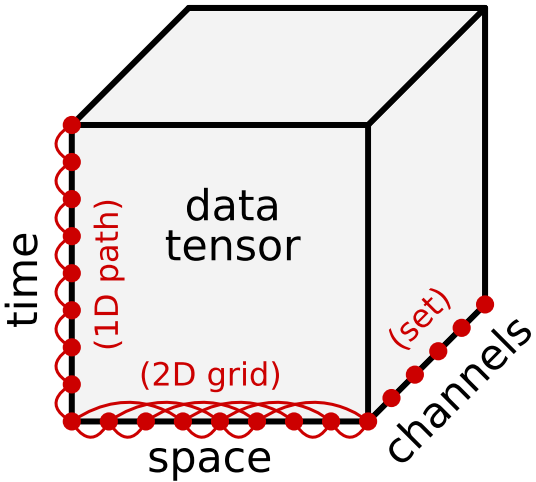
- Data is multi-dimensional.

Structured data



- Data is multi-dimensional.
- Measurements are discrete.

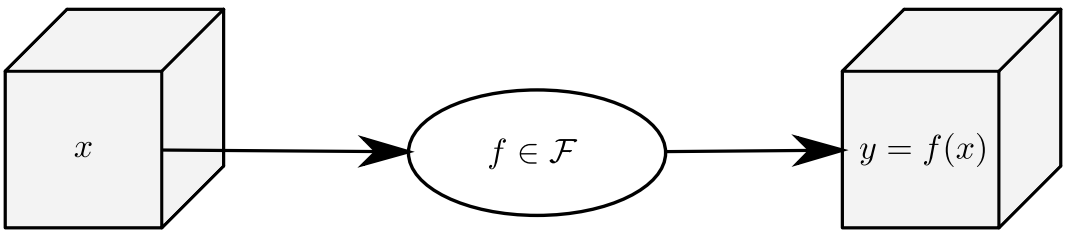
Structured data



- Data is multi-dimensional.
- Measurements are discrete.
- Dimensions are structured.

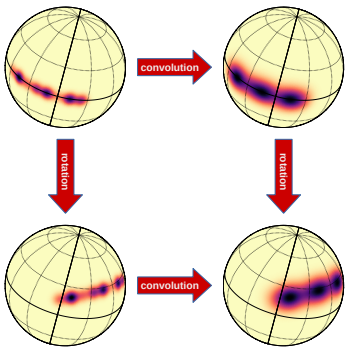
The (deep) learning revolution

From designing the solution f to designing the solution space $\mathcal{F}$.



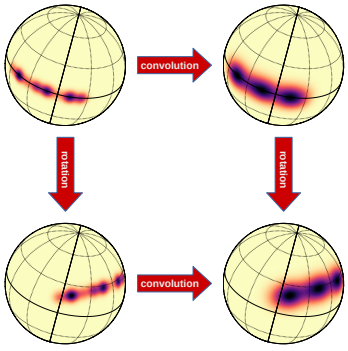
$\mathcal{F}$ is determined by the NN architecture. How to design it?

Design of solution spaces (NN architectures)

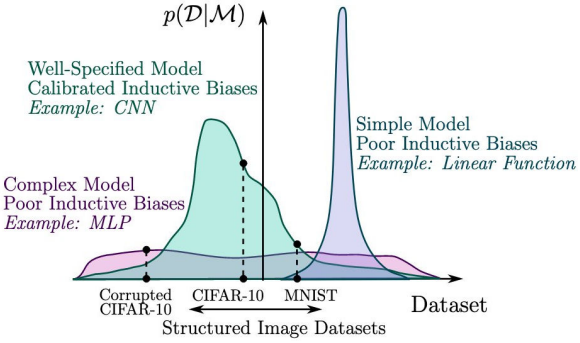


Constraints

Design of solution spaces (NN architectures)



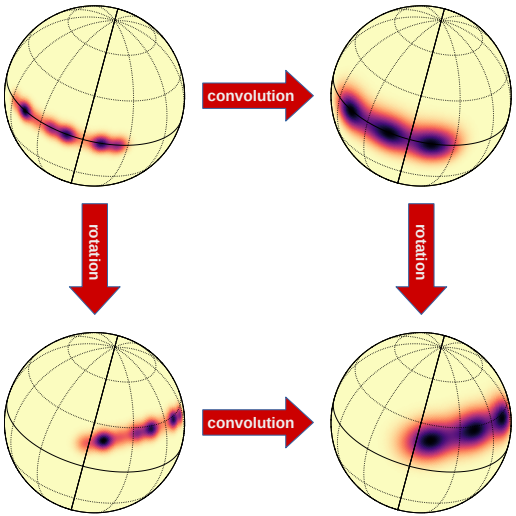
Constraints



Biases

Bias figure from Wilson and Izmailov 2020.

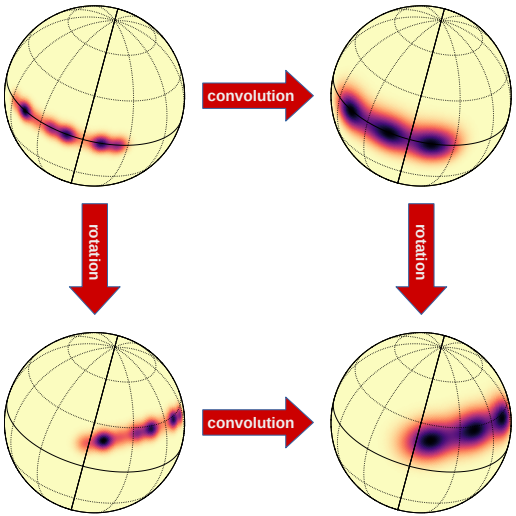
Example constraint: equivariance to rotations



- Equivariance for dense tasks:
  $f(g \cdot x) = g \cdot f(x)\ \forall g \in SO(3)$.
- Invariance for global tasks:
  $f(g \cdot x) = f(x)\ \forall g \in SO(3)$.

Why exploit symmetries?

Example constraint: equivariance to rotations



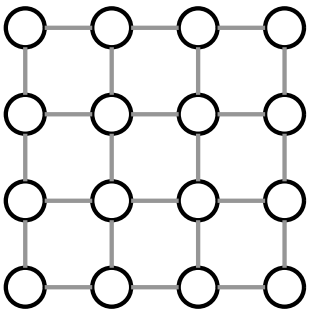
- Equivariance for dense tasks:
  $f(g \cdot x) = g \cdot f(x)\ \forall g \in SO(3)$.
- Invariance for global tasks:
  $f(g \cdot x) = f(x)\ \forall g \in SO(3)$.

Why exploit symmetries?

- Reduced sample complexity.
- Generalization guarantee.

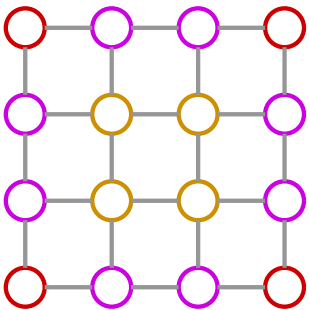
$\Rightarrow$ Principled convolution (weight sharing).

Symmetries might not be enough



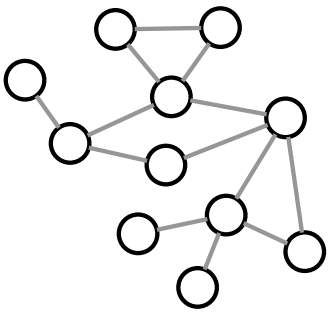
- What are the symmetries? Translations?

Symmetries might not be enough



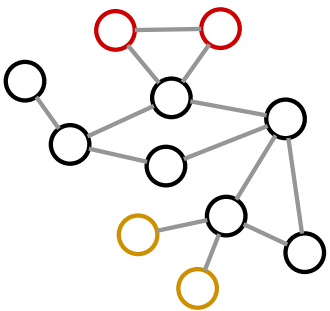
- What are the symmetries? Translations?
- Few symmetries.
- A solution: "cheat" by treating the grid as a discretization of the plane.

Symmetries might not be enough



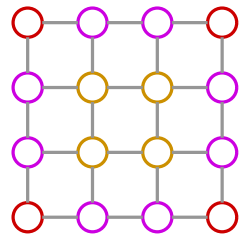
- What are the symmetries?

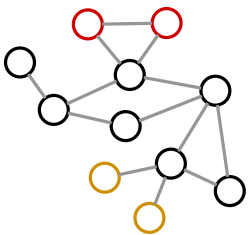
Symmetries might not be enough



- What are the symmetries?
- Asymmetric core with few symmetric motifs.
- Can't "cheat". No underlying continuous domain. Purely discrete.

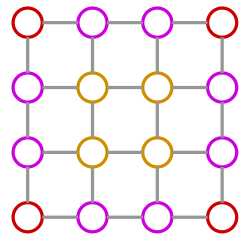
Symmetries might not be enough

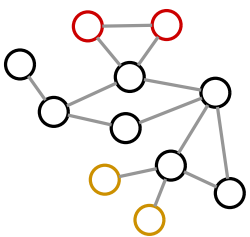




Why more weight sharing?

Symmetries might not be enough

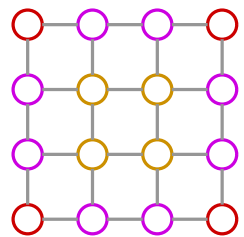


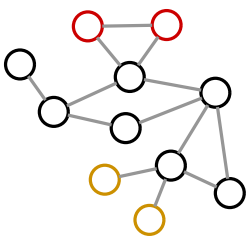


Why more weight sharing?

- More supervision, need less data.
- More generalization guarantee.
- Less powerful / general / flexible.

Symmetries might not be enough





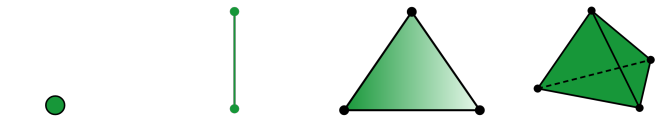
Why more weight sharing?

- More supervision, need less data.
- More generalization guarantee.
- Less powerful / general / flexible.

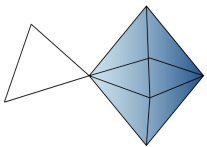
The bias–variance tradeoff.

A discrete calculus

Space: simplicial complexes



d-simplices.



Simplicial complex K.

- Simplex: set of vertices.
- Simplicial complex K: set of simplices.
  Single axiom: closed under taking subsets.
- K_d: set of all d-simplices.

Data

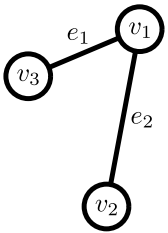
- Simplices naturally form a spatial basis.
- Vertex- ($d = 0$), edge- ($d = 1$), simplex-valued ($d > 1$) functions.
- Covariant d-chain $x_d \in \mathbb{R}^{|K_d|}$ and contravariant d-cochain $f_d \in \mathbb{R}^{|K_d|}$.

Duality:

$$\langle x_d, f_d \rangle = {x_d}^\top f_d$$

Topology: an incidence structure

$$K = \big\{\{v_1\}, \{v_2\}, \{v_3\}, \underbrace{\{v_3, v_1\}}_{e_1}, \underbrace{\{v_1, v_2\}}_{e_2}\big\}$$



$$B_1 = \begin{pmatrix} +1 & -1 \\ 0 & +1 \\ -1 & 0 \end{pmatrix}$$

- Ordering is arbitrary but necessary.
  $K_0 = \big\{\{v_1\}, \{v_2\}, \{v_3\}\big\}$ and $K_1 = \big\{e_1, e_2\big\}$.
- Orientation is arbitrary but necessary.
  $e_1 = \{v_3, v_1\}$ and $e_2 = \{v_1, v_2\}$.

Topology: an incidence structure

- Boundary operator ${B_d}^\top$:
  subdomain d-chain $x_d \to$ boundary $(d-1)$-chain ${B_d}^\top x_d$.

- Differential operator[1] B_d:
  data $(d-1)$-cochain $f_{d-1} \to$ finite difference d-cochain $B_d f_{d-1}$.

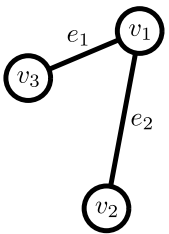
${B_d}^\top$ and B_d are adjoint:

$$\langle {B_d}^\top x_d, f_{d-1} \rangle = \langle x_d, B_d f_{d-1} \rangle \qquad \int_{\partial\Omega} \omega = \int_{\Omega} \mathrm{d}\omega$$

[1] Also known as the coboundary or exterior derivative.

Geometry: an inner product

$$\langle f_d, h_d \rangle_{M_d} = {f_d}^\top M_d h_d$$



$$M_0 = \begin{pmatrix} \text{weight}(v_1) & 0 & 0 \\ 0 & \text{weight}(v_2) & 0 \\ 0 & 0 & \text{weight}(v_3) \end{pmatrix}$$

$$M_1 = \begin{pmatrix} \text{weight}(e_1) & 0 \\ 0 & \text{weight}(e_2) \end{pmatrix}$$

Weights an represent similarities or distances/volumes.

Codifferential operator

$$\langle B_d f_{d-1}, h_d \rangle_{M_d} = \langle f_{d-1}, {B_d}^\dagger h_d \rangle_{M_{d-1}}$$

Codifferential operator ${B_d}^\dagger = {M_{d-1}}^{-1} {B_d}^\top M_d$.

- ${B_d}^\dagger$ is adjoint to B_d w.r.t. M_d.
- Gradient B_1, divergence ${B_1}^\dagger$, curl B_2.

Dirichlet energy: defines the Laplacian

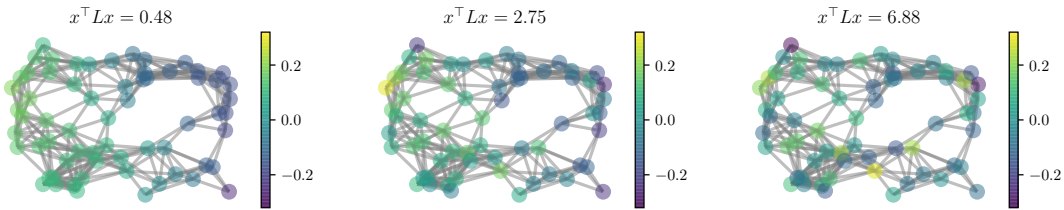
$$\langle {B_d}^\dagger f_d, {B_d}^\dagger h_d \rangle_{M_{d-1}} + \langle B_{d+1} f_d, B_{d+1} h_d \rangle_{M_{d+1}} = \langle f_d, L_d h_d \rangle_{M_d}$$

Laplacian as the second-order differential operator

$$L_d = B_d {B_d}^\dagger + {B_{d+1}}^\dagger B_{d+1}$$

Dirichlet energy: measure of variation

$$E(f_d) = \langle f_d, L_d f_d \rangle_{M_d} = \|B_d{}^\dagger f_d\|^2_{M_{d-1}} + \|B_{d+1} f_d\|^2_{M_{d+1}}$$



$$E(f_0) = \langle f_0, L_0 f_0 \rangle_{M_0} = \|B_1 f_0\|^2_{M_1}$$

Generalized convolutions

Graphs

- Graph G of $n = |K_0|$ vertices.
- Incidence matrix $B = B_1$.
- Unweighted vertices ($M_0 = I$) and edge weights $M = M_1$.
- Laplacian $L = L_0 = B^\dagger B = B^\top M B$.

Symmetries

$$\sigma \in \mathrm{Aut}(G) \subset S_n$$

- Automorphism σ.
- Automorphism group $\mathrm{Aut}(G)$.
- $0 \leq |\mathrm{Aut}(G)| \leq |S_n|$ symmetries.

Representation (spatial basis): permutation matrix P_σ.

Equivariance

$$P_\sigma{}^\top L P_\sigma = L \qquad\qquad LP_\sigma = P_\sigma L$$

- Symmetry preserves the adjacency structure.
- The Laplacian commutes with symmetry group actions.
- The Laplacian is an intrinsic and equivriant operator.

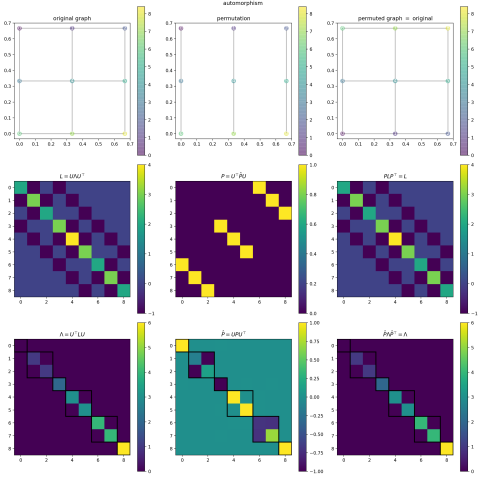
Fourier diagonalizes symmetry actions

$$L = U \Lambda U^{-1}$$

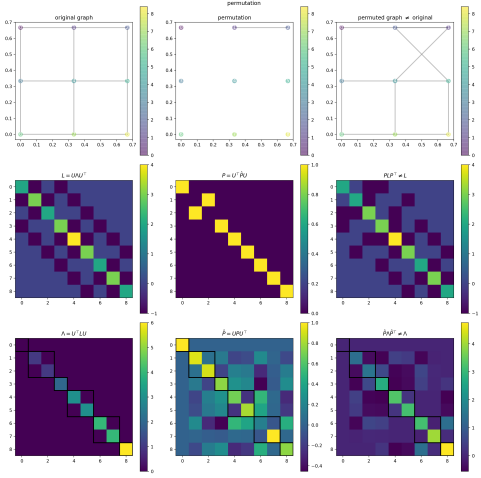
- Symmetries must act as rotations within the eigenspaces of L.
- Fourier jointly (block-)diagonalizes L and P_σ—without knowing the symmetries.

Special case of the Peter-Weyl theorem (compact groups) and Pontryagin duality (Abelian groups).

Fourier diagonalizes symmetry actions



Automorphism: $PLP^\top = L$.



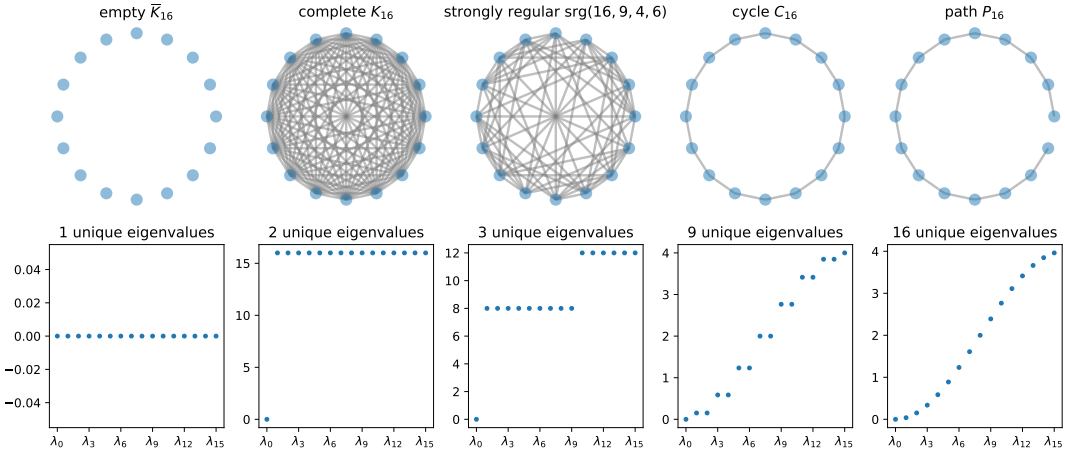
Permutation: $PLP^\top \neq L$.

Spectral basis

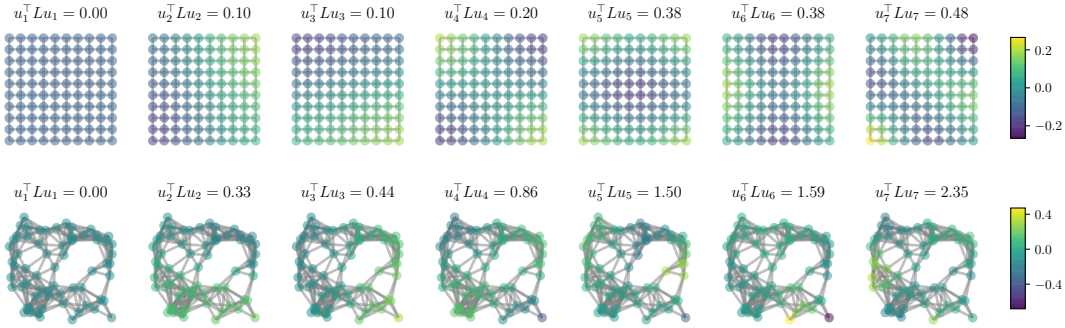
$$L = U \Lambda U^{-1}$$

- Fourier $U = [u_1, \ldots, u_n]$, eigenvectors u_i.
- Squared frequencies $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, eigenvalues $0 = \lambda_1 \leq \cdots \leq \lambda_n$.
- Because L is positive semi-definite [spectral theorem].
- Reduces to the discrete cosine (DCT) and Fourier (DFT) transforms.

Spectral basis: eigenvalues

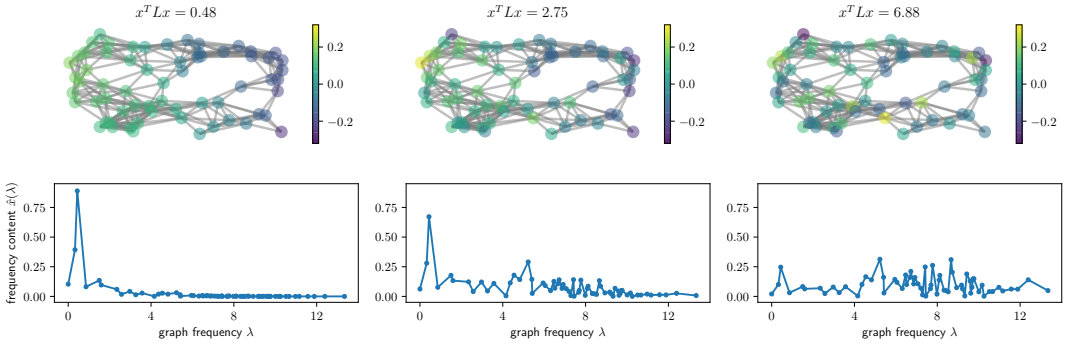


Spectral basis: eigenvectors



Spectral basis: Fourier transform

$$\hat{x} = U^{-1}x \qquad x = U\hat{x} \qquad E(x) = x^\top Lx = \hat{x}^\top \Lambda \hat{x}$$



Generalized convolutions

$$\Lambda = U^{-1} L U \qquad\qquad \Pi_\sigma = U^{-1} P_\sigma U$$

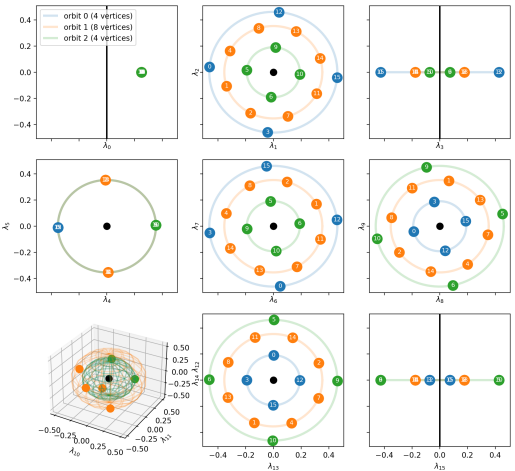
- The eigenspaces are the invariant subspaces of both operators.
- Λ is diagonal: one value per eigenspace.
- Π_σ is block-diagonal: one block per eigenspace.
- Each block implements a roto-reflection.

Generalized convolutions (spectral basis)

$$g(\Lambda) = \mathrm{diag}\big(g(\lambda_1), \ldots, g(\lambda_n)\big)$$

$$g(\Lambda)\Pi_\sigma = \Pi_\sigma g(\Lambda)$$

The action $g(\Lambda)$ of g (scaling) is orthogonal to the action Π_σ of σ (roto-reflection).



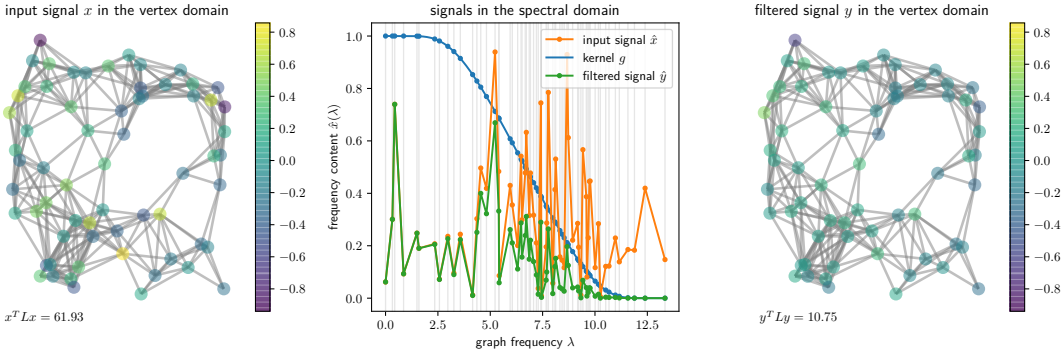
Generalized convolutions (spatial basis)

$$g(L) = U g(\Lambda) U^{-1}$$

- Multiplication operator $g(\Lambda)$ and convolution operator $g(L)$.
- $g(L)$ is an equivariant operator, the defining property of convolutions.
- Generalized because it commutes with more than symmetries.

Filtering

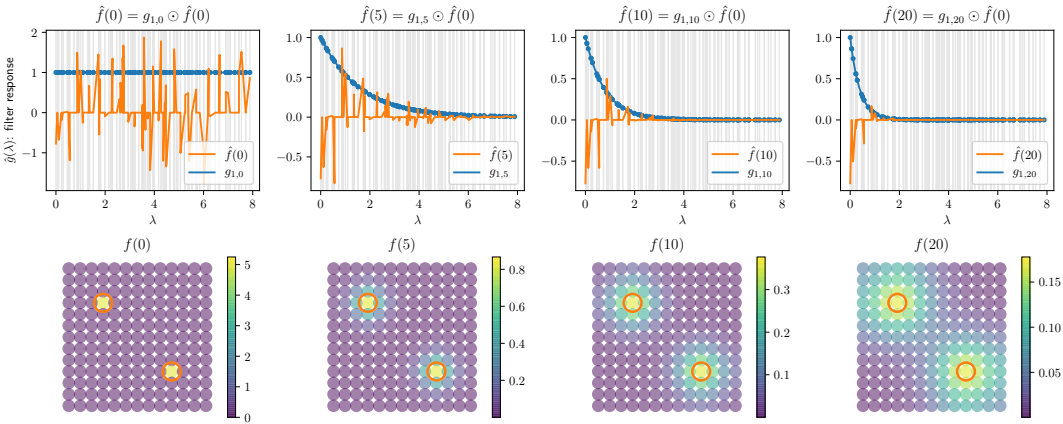
$$y = g(L)x = Ug(\Lambda)U^{-1}x$$



Left: data x in the spatial basis. Middle: data $\hat{x} = U^{-1}x$, concrete filter $\mathrm{diag}(g(\Lambda))$, and filtered data $\hat{y} = g(\Lambda)\hat{x}$ in the spectral basis. Right: filtered data $y = U\hat{y}$ in the spatial basis.

Filtering: heat diffusion

$$-\tau L f(t) = \partial_t f(t) \quad \Rightarrow \quad f(t) = g_{\tau t}(L) f(0) \text{ with } g_{\tau t}(\lambda) = \exp(-\tau t \lambda)$$



Designing g

Design a kernel $g : \mathbb{R} \to \mathbb{R}$ such that its action $y = g(L)x$ does something interesting.

- $g(\lambda) = \exp(-\tau t \lambda)$: heat diffusion.
- $g(\lambda) = \cos\left(t \arccos\left(1 - \frac{\tau^2}{2}\lambda\right)\right)$: wave propagation.
- $g(\lambda) = \begin{cases} 1 & \text{if } \lambda_{\text{min}} < \lambda < \lambda_{\text{max}}, \\ 0 & \text{otherwise.} \end{cases}$: projection on a subspace.
- $g(\lambda) = \frac{1}{1+\tau\lambda}$: denoising with $\arg\min_y \|y - x\|_2^2 + \tau y^\top L y$

But what if we don't know the process by which y depends on x? Learn g.

Convolution: symmetry action vs localization

Convolution with symmetry action.

$$\langle y, \delta_i \rangle = \langle T_i g, x \rangle$$

- $T_i g$ shifts g to the i^{th} vertex.
- x and g are the same objects.

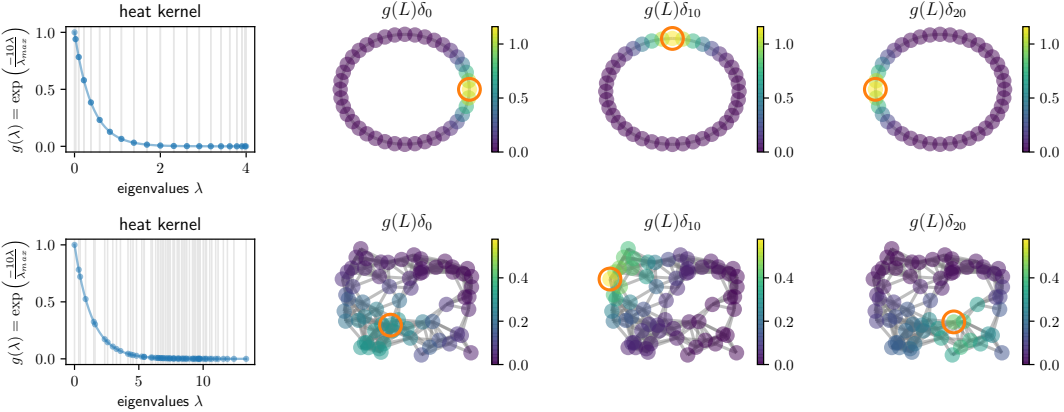
Convolution with localization.

$$\begin{aligned} \langle y, \delta_i \rangle &= \langle g(L)x, \delta_i \rangle \\ &= \langle x, g(L)\delta_i \rangle \end{aligned}$$

- $g(L)\delta_i$ localizes g at the i^{th} vertex.
- x and g are different.

Localization is a generalization of symmetry action to non-homogeneous spaces.

Convolution: symmetry action vs localization



Localization reduces to shift when there are symmetries.

Spectral embedding

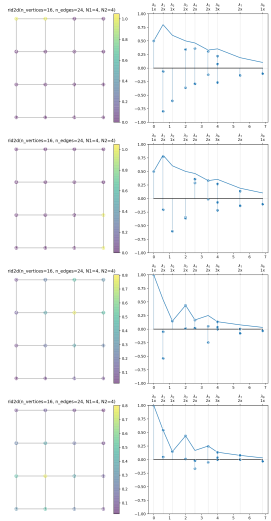
$$E_g(f) = \langle f, g(L)f \rangle = \|g^{1/2}(\Lambda)U^{-1}f\|_2^2$$

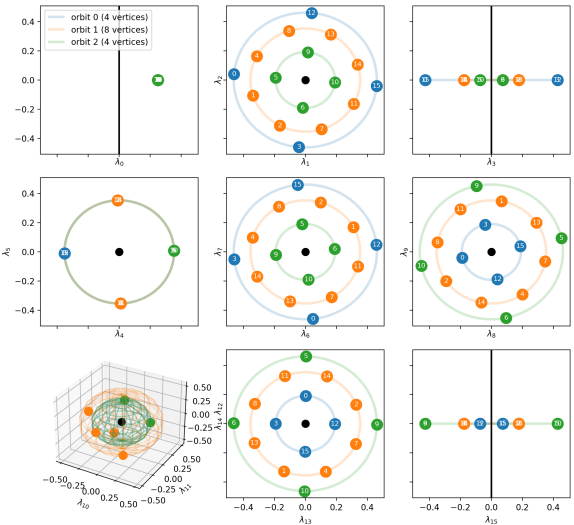
- Generalization of Dirichlet energy to $g \neq \mathrm{id}$.
- $g^{1/2}(\Lambda)U^{-1}f$ is an embedding of f in Euclidean space.

The embedding reproduces:

- the symmetries of the space encoded in L,
- a notion of distance set by g.

Spectral embedding





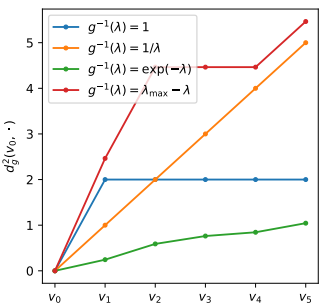
Network (vertex) embedding

$$Q = g^{1/2}(\Lambda)U^{-1}$$

- Embedding $Q = [q_1, \ldots, q_n]$, where $q_i \in \mathbb{R}^n$ represents the i^{th} vertex.
- Covariance $Q^\top Q = U g(\Lambda) U^{-1} = g(L)$.
  PCA with principal directions u_i and variances $g(\lambda_i)$.

Distance

$$d_g^2(v_i, v_j) = \|q_i - q_j\|_2^2 = E_g(\delta_i - \delta_j)$$



Distances on a path graph.

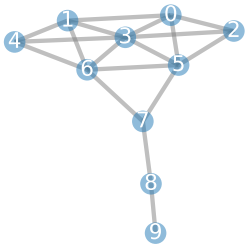
- $g^{-1}(\lambda) = 1$: Laplacian eigenmaps
  [Belkin & Niyogi '01]
- $g^{-1}(\lambda) = 1/\lambda$: resistance/commute-time distance
  [Klein & Randić '93] [Göbel & Jagers '74] [Fouss et al. '07]
- $g^{-1}(\lambda) = \exp(-2t\lambda)$: (heat) diffusion distance
  [Coifman & Lafon '06] [Kondor & Lafferty '02]
- $g^{-1}(\lambda) = (a - \lambda)^p$, $a \geq \lambda_{\max}$: p-step random-walk
  [PageRank, Brin & Page '98]

Centrality

$$C_g^2(v_i) = \|q_i\|_2^2 = E_g(\delta_i) = \big(g(L)\big)_{ii}$$

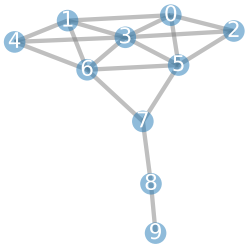
- Measures how close a vertex is to all others.
- Why? $\sum_j \|q_j - q_i\|^2 = \sum_j \|q_j\|_2^2 + n\|q_i\|_2^2 \propto \|q_i\|_2^2 = C_g^2(v_i)$.
- Closer to the origin (center of mass) implies closer to all other vertices.

Designing g

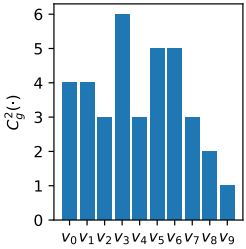


Krackhardt kite graph.

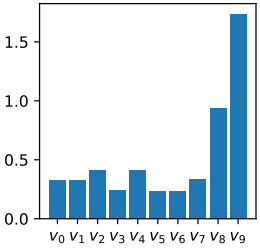
Designing g



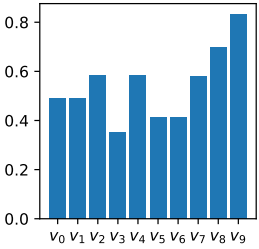
Krackhardt kite graph.



Degree centrality
$g(\lambda) = \lambda$.



Closeness centrality
$g(\lambda) = \lambda^{-1}$.



Diffusion centrality
$g(\lambda) = \exp(0.2\lambda)$.

Degree centrality is contravariant, the others are covariant.
Closeness centrality with resistance instead of the typical shortest-path distance.

Vertex representations

Spectral embedding q_i:

- Not invariant to automorphisms.
- Not even invariant to the arbitrary choice of basis in each eigenspace.

Vertex representations

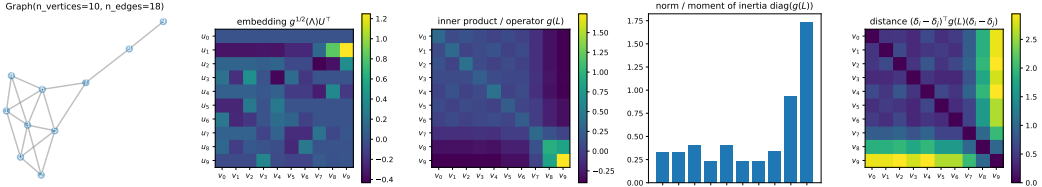
Spectral embedding q_i:

- Not invariant to automorphisms.
- Not even invariant to the arbitrary choice of basis in each eigenspace.

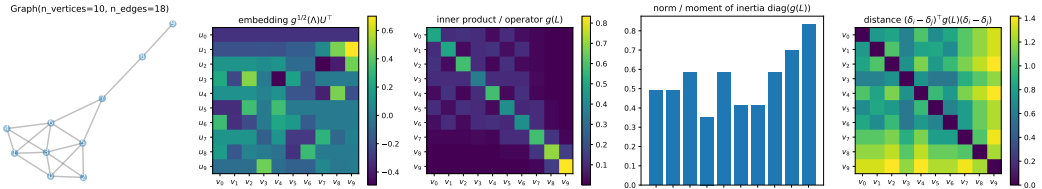
Better options: centrality $C_g^2(v_i)$ and multiset of distances $\{d_g^2(v_i, \cdot)\}$.

- Invariant to automorphisms.
- Only depends on $L = B^\top M B$—the space's topology B and geometry M—and the choice of distance g.

Designing g

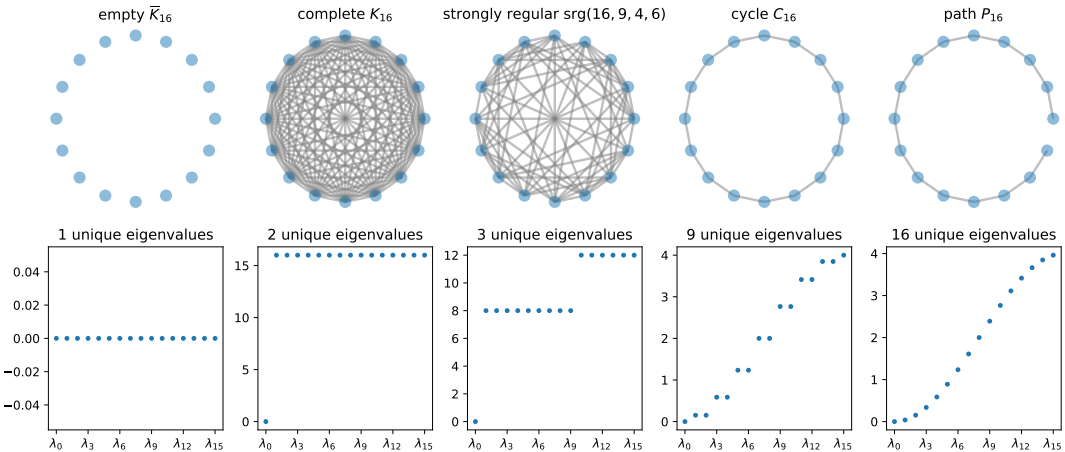


$g(\lambda) = \lambda^+ \approx \lambda^{-1}$ yields the resistance (or commute-time) distance.



$g(\lambda) = e^{-t\lambda}$ yields the diffusion distance and heat kernel signature (HKS) embedding.

Learning g: degrees of freedom



Number of independent distances: from 1 to n.

Non-limitation: it transfers across graphs

- g is a function sampled at the eigenvalues $\{\lambda_i\}$.

- The abstract kernel g does not depend on the graph G.
  Only the concrete representation $g(L) = g(B^\top M B) = U g(\Lambda) U^{-1}$ does.

Non-limitation: it scales

No need to compute the EVD $L = ULU^{-1}$.

1. Polynomial parameterization (better for local g):

$$g(\lambda) = \sum_{k<K} \alpha_k \lambda^k,$$
$$g(L) = \sum_{k<K} \alpha_k L^k = \sum_{k<K} \alpha_k \bar{x}_k, \quad \bar{x}_k = L\bar{x}_{k-1},\ \bar{x}_0 = x.$$

2. Partial spectrum (better for global g):

$$g(L) = \sum_{k \in \mathcal{K}} g(\lambda_k) u_k {u_k}^\top$$

Heisenberg's uncertainty principle: a g that is local in the vertex domain is smooth in the spectral domain and vice-versa.

Limitation

Neither centrality $C_g^2(v_i)$ nor distances $\{d_g^2(v_i, \cdot)\}$ are complete invariants w.r.t. automorphism/isomorphism.

Michaël Defferrard @m_deff · Dec 31, 2020

2020: I got the GI disease
2021: I will find a cure or get immune

Happy New Year y'all! 🎉

The graph isomorphism disease†

Ronald C. Read, Derek G. Corneil

First published: Winter 1977 | https://doi.org/10.1002/jgt.3190010410

† Dedicated to George Pólya on his 90th Birthday.

Summary

1. The kernel g defines a notion of distance.
2. It is represented by the generalized convolution $g(L)$.
3. $g(L) = g(B^\top M B)$ only depends on the domain's topology B and geometry M.
4. $g(L)$ is mostly constrained by the domain's symmetries and complexity, constraining the functional space to learn from.
5. $g(L)$ is equivariant to unknown symmetries.
6. Filtering and embedding are one and the same: the generalized convolution $g(L)$.
7. Design g if you know what you want, learn it if you don't.

Slides `https://doi.org/10.5281/zenodo.5718843`

Papers Defferrard, Generalized convolutions, In preparation, 2022.

Ebli, Defferrard, Spreemann, Simplicial Neural Networks, TDA@NeurIPS, 2020.

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

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

Code `https://github.com/epfl-lts2/pygsp`
`https://github.com/stefaniaebli/simplicial_neural_networks`
`https://github.com/deepsphere`
`https://github.com/mdeff/cnn_graph`