Deep Geometric Matrix Completion

Federico Monti

Università della Svizzera italiana
Switzerland

federico.monti@usi.ch

Joint work with M. M. Bronstein (USI) and X. Bresson (NTU)

Deep Geometric Matrix Completion

Federico Monti

Università della Svizzera italiana
Switzerland

federico.monti@usi.ch

Joint work with M. M. Bronstein (USI) and X. Bresson (NTU)

12 Years a Slave
2013

Gravity
2013

Frozen
2013

American Hustle
2013

Thor: The Dar...
2013

The Hunger Ga...
2013

Captain Phillips
2013

Dallas Buyers Club
2013

Reasonable Doubt
2014

Last Vegas
2013

Ender's Game
2013

Jackass Prese...
2013

Rush
2013

The Wolverine...
2013

About Time
2013

The Bag Man
2014

World War Z UNR...
2013

The Counselor
2013

We're the Mil...
2013

Man of Steel
2013

Now You See M...
2013

Cloudy with a...
2013

This Is the End
2013

The Returned
2013

Elysium

Pacific Rim

The Great Gatsby

The Lone Ranger

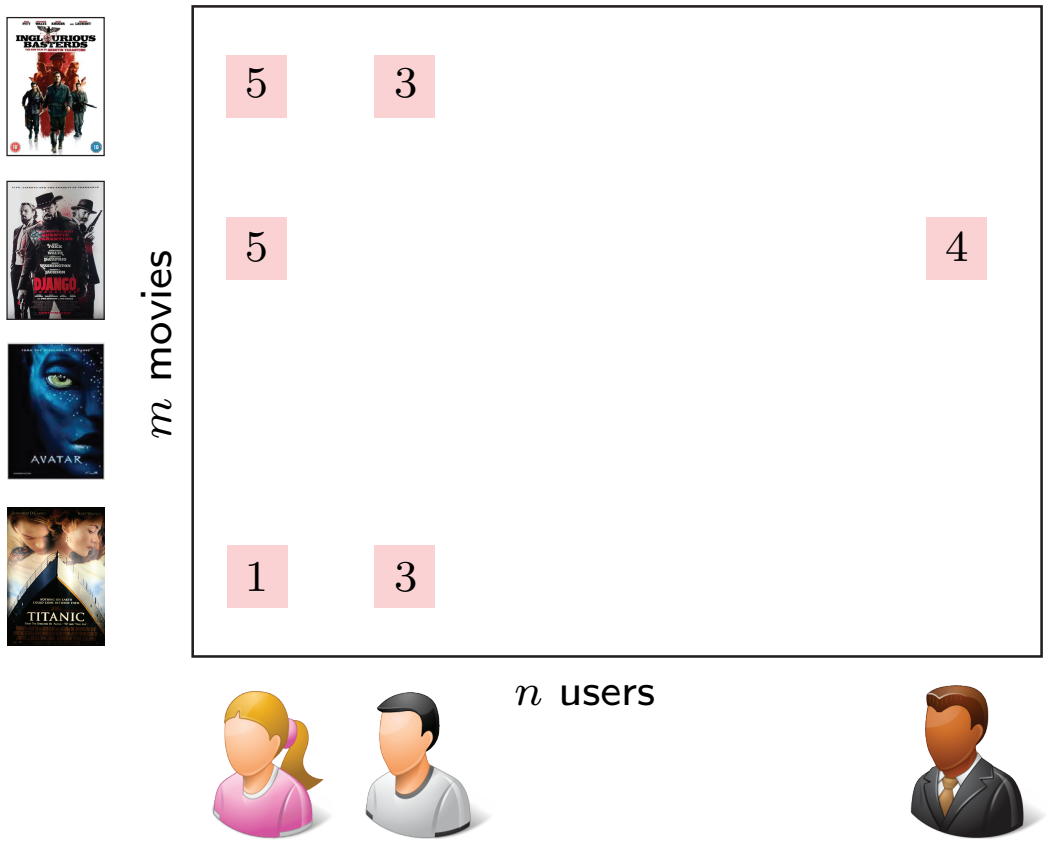
Blue Jasmine

2 Guns

Romeo and Juliet

Monsters Univer...

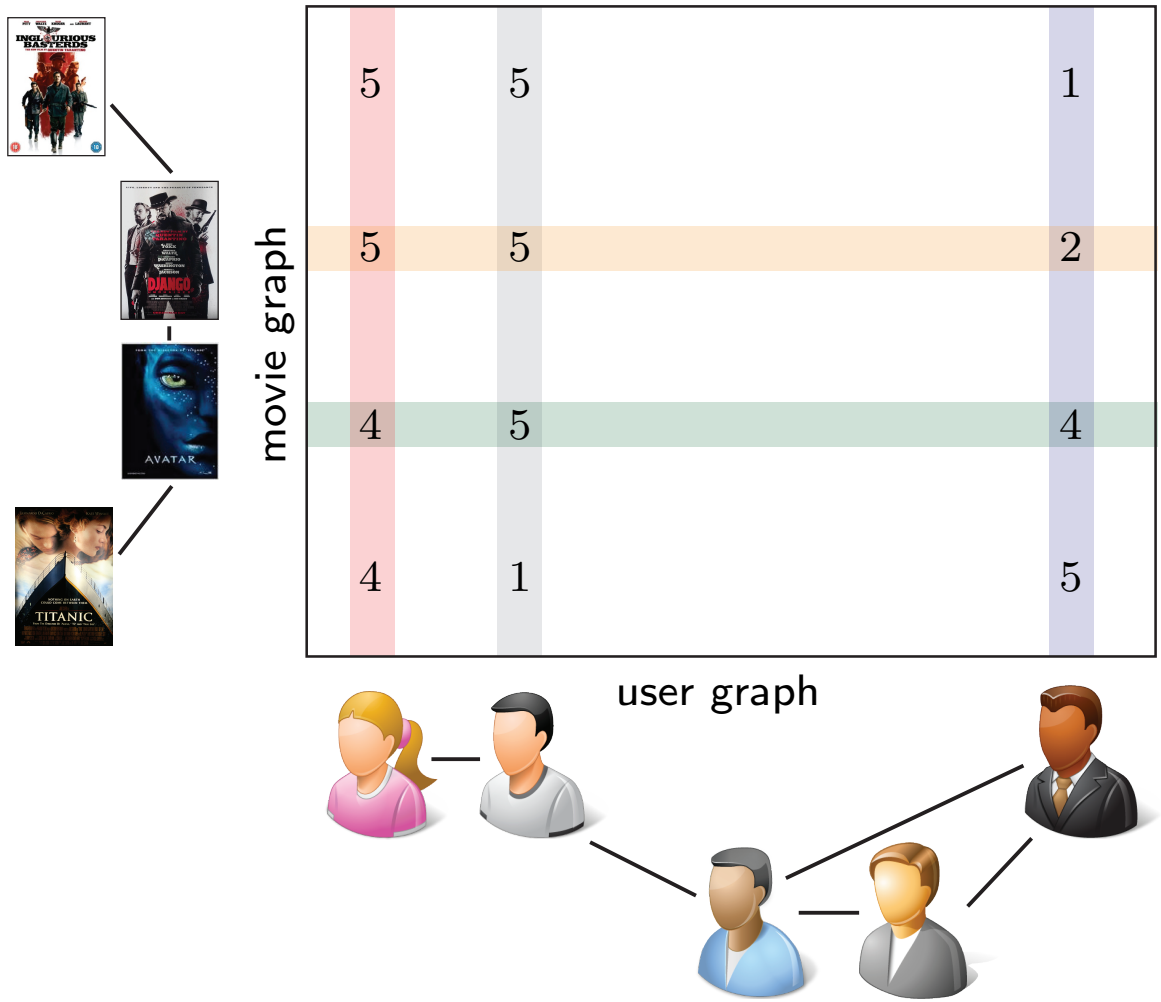
Matrix completion: 'Netflix challenge'



$$\min_{\mathbf{X} \in \mathbb{R}^{m \times n}} \quad \|\mathbf{X}\|_* + \mu \|\mathbf{\Omega} \circ (\mathbf{X} - \mathbf{A})\|_{\mathrm{F}}^2$$

Candès 2008

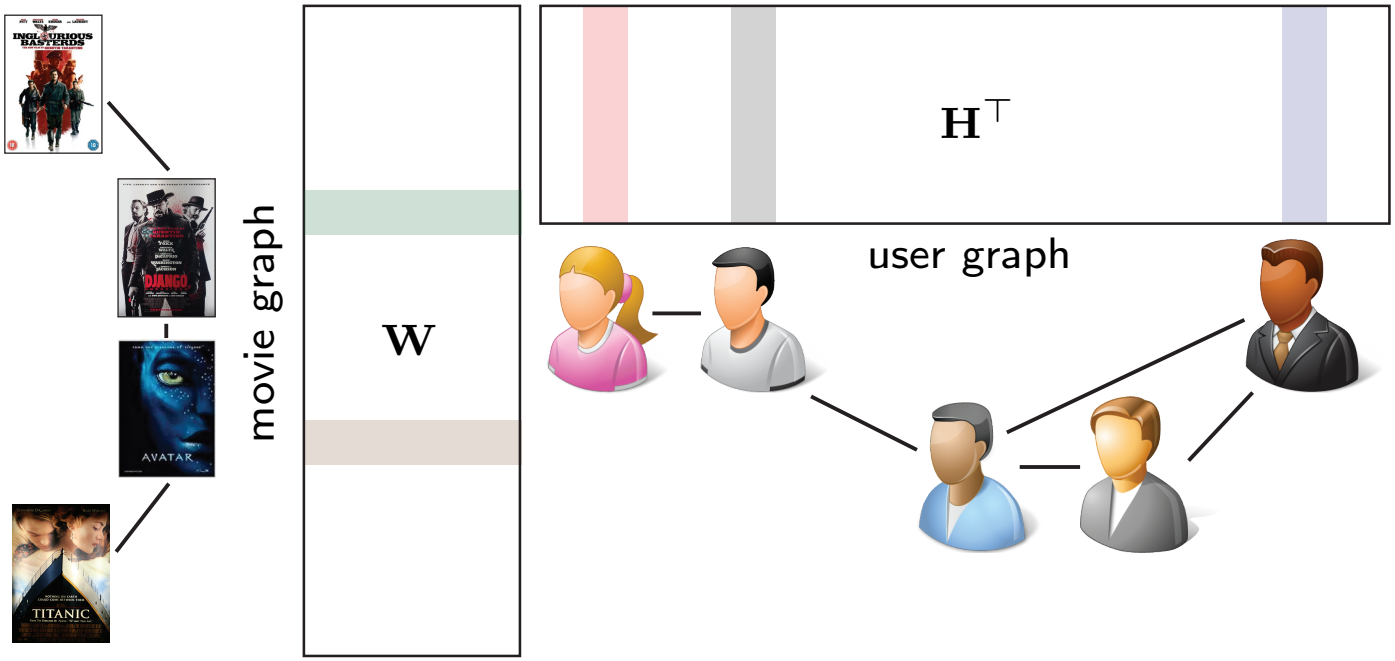
Geometric matrix completion: 'Netflix challenge'



$$\min_{\mathbf{X}\in\mathbb{R}^{m\times n}} \quad \|\boldsymbol{\Omega}\circ(\mathbf{X}-\mathbf{A})\|_{\mathrm{F}}^2 + \mu_{\mathrm{c}}\underbrace{\mathrm{tr}(\mathbf{X}\boldsymbol{\Delta}_{\mathrm{c}}\mathbf{X}^\top)}_{\|\mathbf{X}\|_{\mathcal{G}_{\mathrm{c}}}^2} + \mu_{\mathrm{r}}\underbrace{\mathrm{tr}(\mathbf{X}^\top\boldsymbol{\Delta}_{\mathrm{r}}\mathbf{X})}_{\|\mathbf{X}\|_{\mathcal{G}_{\mathrm{r}}}^2}$$

Kalofolias et al. 2014

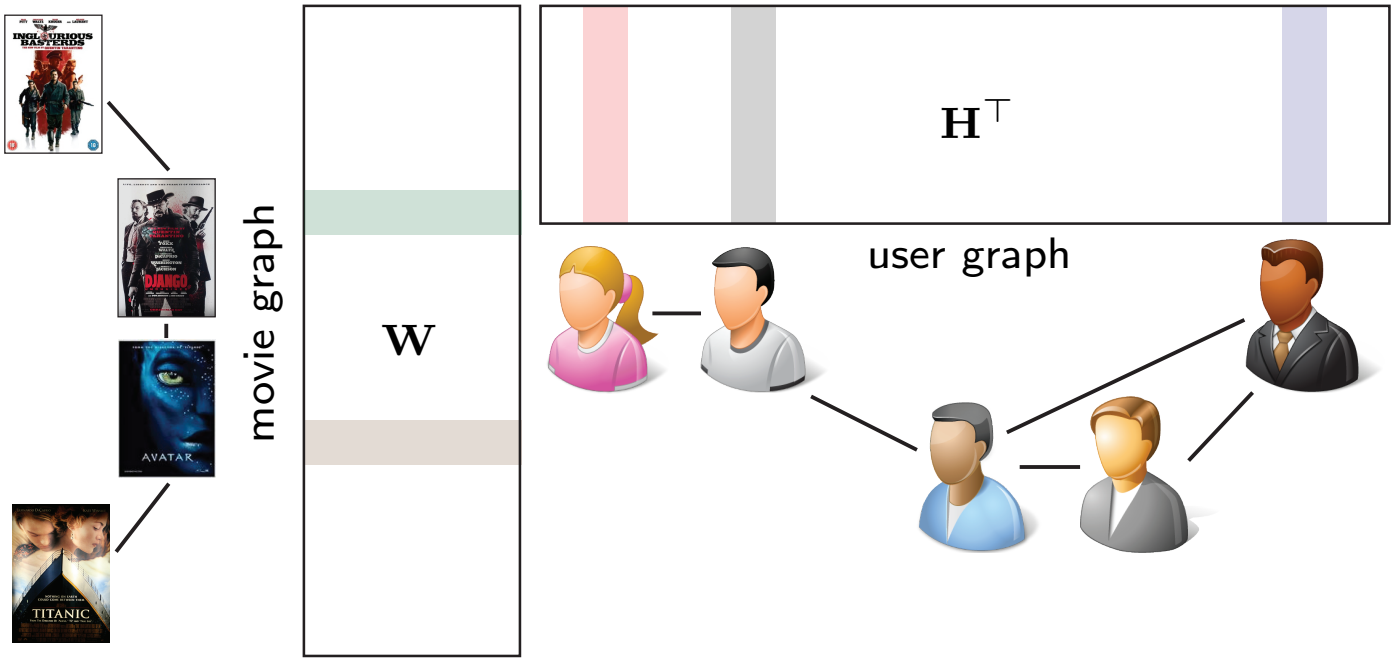
Factorized geometric matrix completion models



$$\min_{\substack{\mathbf{W}\in\mathbb{R}^{m\times s}\\ \mathbf{H}\in\mathbb{R}^{n\times s}}} \quad \|\mathbf{\Omega}\circ(\mathbf{X}-\mathbf{A})\|_{\mathrm{F}}^2+\mu_{\mathrm{c}}\mathrm{tr}(\mathbf{H}^\top\mathbf{\Delta}_{\mathrm{c}}\mathbf{H})+\mu_{\mathrm{r}}\mathrm{tr}(\mathbf{W}^\top\mathbf{\Delta}_{\mathrm{r}}\mathbf{W})$$

Rao et al. 2015

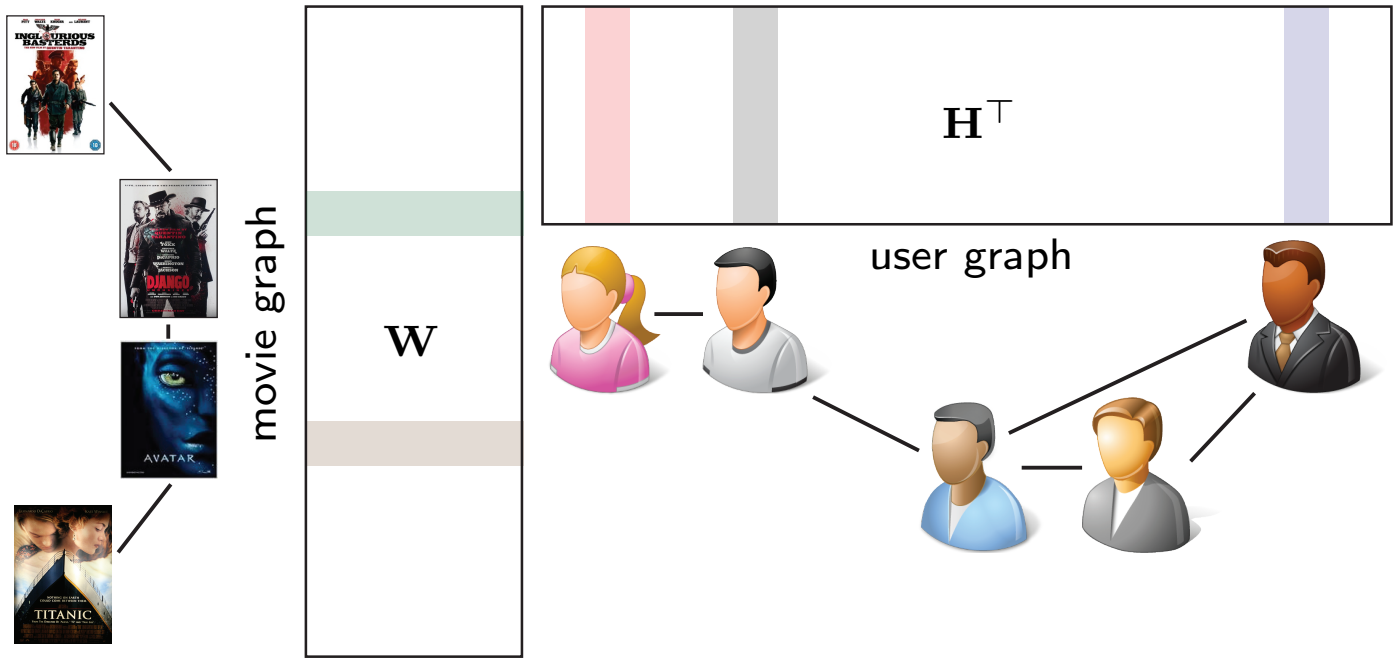
Factorized geometric matrix completion models



$$\min_{\substack{\mathbf{W}\in\mathbb{R}^{m\times s}\\ \mathbf{H}\in\mathbb{R}^{n\times s}}} \quad \|\mathbf{\Omega}\circ(\mathbf{X}-\mathbf{A})\|_{\mathrm{F}}^2+\mu_{\mathrm{c}}\mathrm{tr}(\mathbf{H}^\top\mathbf{\Delta}_{\mathrm{c}}\mathbf{H})+\mu_{\mathrm{r}}\mathrm{tr}(\mathbf{W}^\top\mathbf{\Delta}_{\mathrm{r}}\mathbf{W})$$

☹ Do not fully exploit the local stationary structures that users/items graphs present.

Factorized geometric matrix completion models



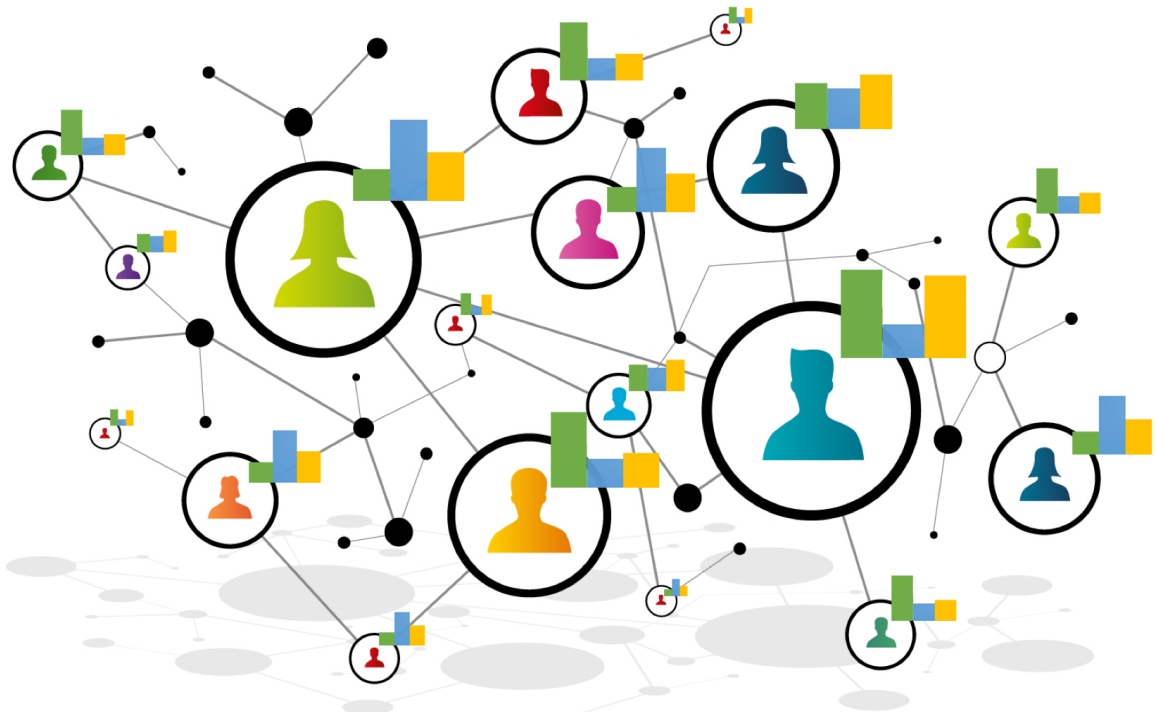
$$\min_{\substack{\mathbf{W}\in\mathbb{R}^{m\times s}\\ \mathbf{H}\in\mathbb{R}^{n\times s}}} \|\mathbf{\Omega}\circ(\mathbf{X}-\mathbf{A})\|_{\mathrm{F}}^2+\mu_{\mathrm{c}}\mathrm{tr}(\mathbf{H}^\top\mathbf{\Delta}_{\mathrm{c}}\mathbf{H})+\mu_{\mathrm{r}}\mathrm{tr}(\mathbf{W}^\top\mathbf{\Delta}_{\mathrm{r}}\mathbf{W})$$

- ☹ Do not fully exploit the local stationary structures that users/items graphs present.
- ☹ Number of parameters to train is at least linear wrt the number of users and item.

Graph Convolutional Neural Networks

A new challenge: geometric data

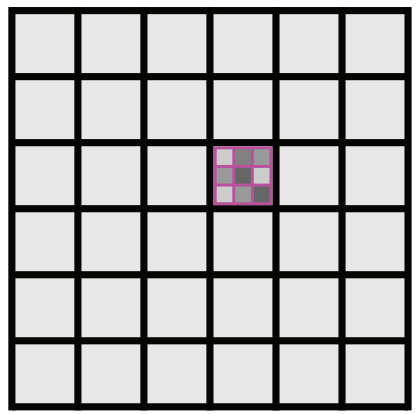
A new challenge: geometric data



Different formulations of CNN on graphs

Spectral domain[1,2,3]

Spatial domain[4,5,6]



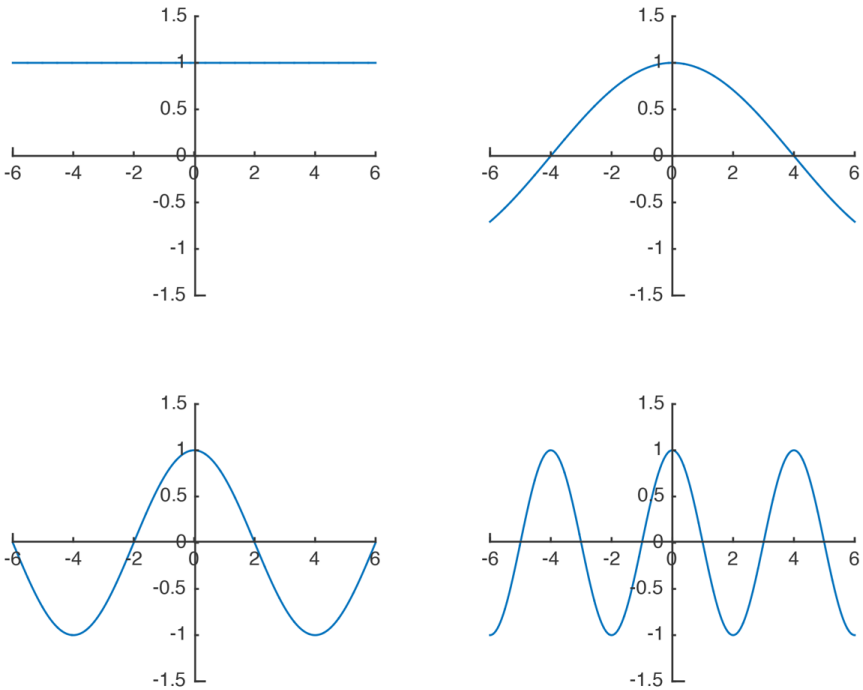
Embedding domain[7,8]

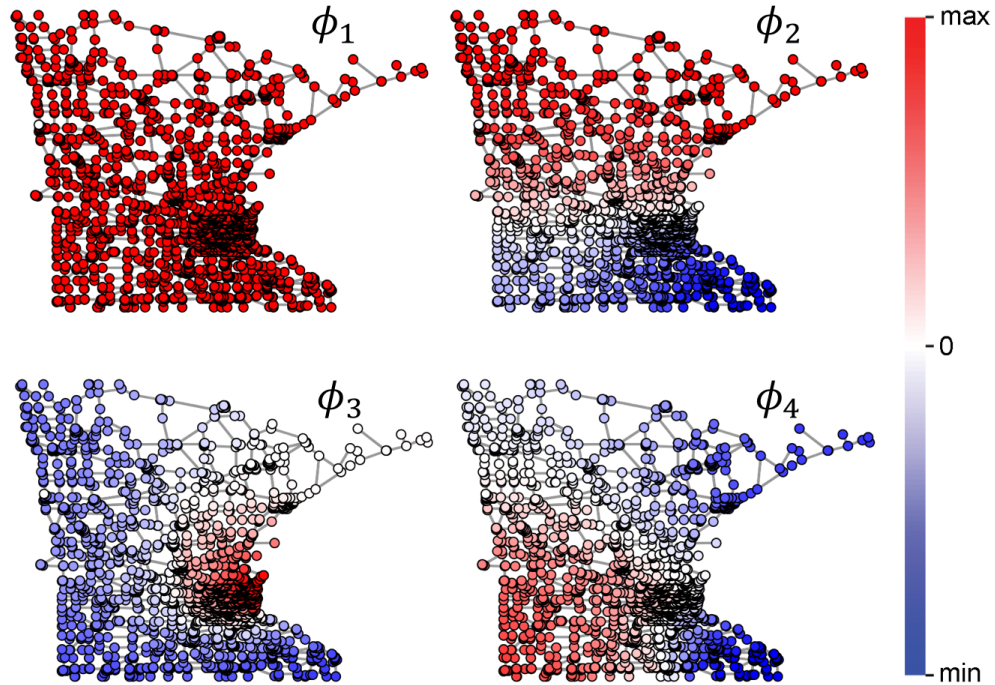
[1]Bruna et al. 2014; [2]Henaff, Bruna, LeCun 2015; [3]Defferrard, Bresson, Vandergheynst 2016; [4]Masci et al. 2015; [5]Boscaini et al. 2016; [6]Monti et al. 2017; [7]Sinha, Bai, Ramani 2016; [8]Maron et al. 2017

Laplacian eigenfunctions

$$\Delta e^{j\omega x} = \frac{d^2}{dx^2} e^{j\omega x} = -\omega^2 e^{j\omega x}$$

$$\mathbf{\Delta} = \mathbf{D} - \mathbf{W} = \mathbf{\Phi \Lambda \Phi}^{\mathrm{T}}$$





Convolution Theorem

- Given two functions $f, h : Z \to \mathbb{R}$ their convolution is a function

$$(f \star h)(x) = \sum_{x'=-K/2}^{K/2} f(x - x')h(x') = \mathcal{F}^{-1}\{\mathcal{F}\{f\} \cdot \mathcal{F}\{h\}\}$$

Convolution Theorem

- Given two functions $f, h : Z \to \mathbb{R}$ their convolution is a function

$$(f \star h)(x) = \sum_{x'=-K/2}^{K/2} f(x - x')h(x') = \mathcal{F}^{-1}\{\mathcal{F}\{f\} \cdot \mathcal{F}\{h\}\}$$

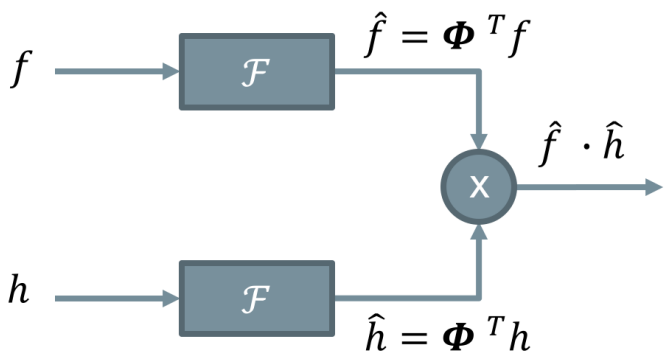
- Given two functions $f, h : \mathcal{V} \to \mathbb{R}$ their convolution is a function

Convolution Theorem

- Given two functions $f, h : Z \to \mathbb{R}$ their convolution is a function

$$(f \star h)(x) = \sum_{x'=-K/2}^{K/2} f(x - x')h(x') = \mathcal{F}^{-1}\{\mathcal{F}\{f\} \cdot \mathcal{F}\{h\}\}$$

- Given two functions $f, h : \mathcal{V} \to \mathbb{R}$ their convolution is a function

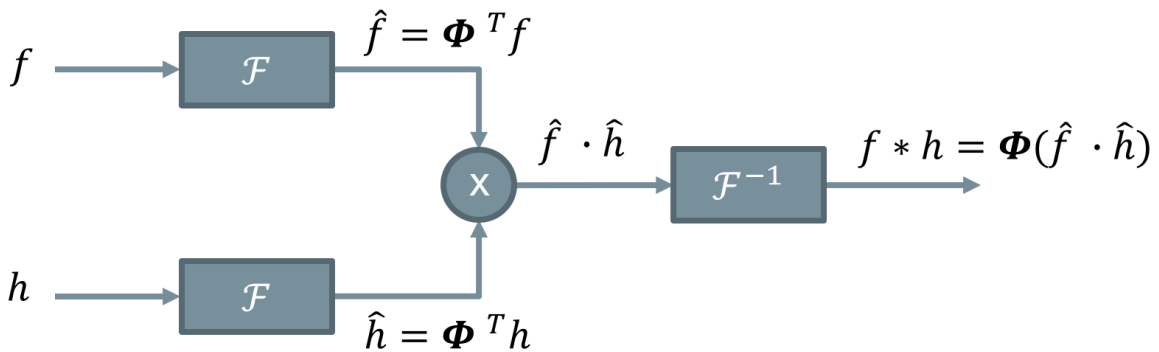


Convolution Theorem

- Given two functions $f, h : Z \to \mathbb{R}$ their convolution is a function

$$(f \star h)(x) = \sum_{x'=-K/2}^{K/2} f(x - x')h(x') = \mathcal{F}^{-1}\{\mathcal{F}\{f\} \cdot \mathcal{F}\{h\}\}$$

- Given two functions $f, h : \mathcal{V} \to \mathbb{R}$ their convolution is a function



$$f * h = \boldsymbol{\Phi}\,(\boldsymbol{\Phi}^T h) \cdot (\boldsymbol{\Phi}^T f)$$

Spectral graph CNN

Convolutional layer expressed in the spectral domain

$$\mathbf{g}_l = \xi\left(\sum_{l'=1}^{p} \mathbf{\Phi} \begin{bmatrix} \hat{h}_1^{(l,l')} & & \\ & \ddots & \\ & & \hat{h}_N^{(l,l')} \end{bmatrix} \mathbf{\Phi}^\top \mathbf{f}_{l'}\right) \qquad \begin{matrix} l = 1, \ldots, q \\ l' = 1, \ldots, p \end{matrix}$$

where q is the number of features in output and p in input.

Bruna et al. 2014

Spectral graph CNN

Convolutional layer expressed in the spectral domain

$$\mathbf{g}_l = \xi\left(\sum_{l'=1}^{p} \mathbf{\Phi} \begin{bmatrix} \hat{h}_1^{(l,l')} & & \\ & \ddots & \\ & & \hat{h}_N^{(l,l')} \end{bmatrix} \mathbf{\Phi}^\top \mathbf{f}_{l'}\right) \quad \begin{matrix} l = 1, \dots, q \\ l' = 1, \dots, p \end{matrix}$$

where q is the number of features in output and p in input.

- ☹ $\mathcal{O}(N)$ 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

Spectral graph CNN with polynomial filters

Represent spectral transfer function as a polynomial or order r

$$\tau_{\boldsymbol{\alpha}}(\lambda) = \sum_{j=0}^{r} \alpha_j \lambda^j$$

where $\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_r)^\top$ is the vector of filter parameters

Defferrard, Bresson, Vandergheynst 2016

Spectral graph CNN with polynomial filters

Represent spectral transfer function as a Chebyshev polynomial or order r

$$\tau_{\boldsymbol{\alpha}}(\tilde{\lambda}) = \sum_{j=0}^{r} \alpha_j T_j(\tilde{\lambda})$$

where $\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_r)^\top$ is the vector of filter parameters,

$$T_j(\tilde{\lambda}) = 2\tilde{\lambda} T_{j-1}(\tilde{\lambda}) - T_{j-2}(\tilde{\lambda}) \qquad T_0(\tilde{\lambda}) = 1, \quad T_1(\tilde{\lambda}) = \tilde{\lambda}$$

and $-1 \leq \tilde{\lambda} \leq 1$ is normalized frequency.

Defferrard, Bresson, Vandergheynst 2016

Spectral graph CNN with polynomial filters

Represent spectral transfer function as a Chebyshev polynomial or order r

$$\tau_{\boldsymbol{\alpha}}(\tilde{\lambda}) = \sum_{j=0}^{r} \alpha_j T_j(\tilde{\lambda})$$

where $\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_r)^\top$ is the vector of filter parameters,

$$T_j(\tilde{\lambda}) = 2\tilde{\lambda} T_{j-1}(\tilde{\lambda}) - T_{j-2}(\tilde{\lambda}) \qquad T_0(\tilde{\lambda}) = 1, \quad T_1(\tilde{\lambda}) = \tilde{\lambda}$$

and $-1 \leq \tilde{\lambda} \leq 1$ is normalized frequency.

Application of the filter to scaled Laplacian $\tilde{\mathbf{\Delta}} = \mathbf{D}^{-0.5}\mathbf{\Delta}\mathbf{D}^{-0.5} - \mathbf{I} = \mathbf{\Phi}\tilde{\mathbf{\Lambda}}\mathbf{\Phi}^T$:

$$\mathbf{g} = \mathbf{\Phi}(\sum_{j=0}^{r} \alpha_j T_j(\tilde{\mathbf{\Lambda}}))\mathbf{\Phi}^T\mathbf{f} = \sum_{j=0}^{r} \alpha_j T_j(\tilde{\mathbf{\Delta}})\mathbf{f}$$

Defferrard, Bresson, Vandergheynst 2016

Spectral graph CNN with polynomial filters

Represent spectral transfer function as a Chebyshev polynomial or order r

$$\tau_{\boldsymbol{\alpha}}(\tilde{\lambda}) = \sum_{j=0}^{r} \alpha_j T_j(\tilde{\lambda})$$

where $\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_r)^\top$ is the vector of filter parameters,

$$T_j(\tilde{\lambda}) = 2\tilde{\lambda} T_{j-1}(\tilde{\lambda}) - T_{j-2}(\tilde{\lambda}) \qquad T_0(\tilde{\lambda}) = 1, \quad T_1(\tilde{\lambda}) = \tilde{\lambda}$$

and $-1 \leq \tilde{\lambda} \leq 1$ is normalized frequency.

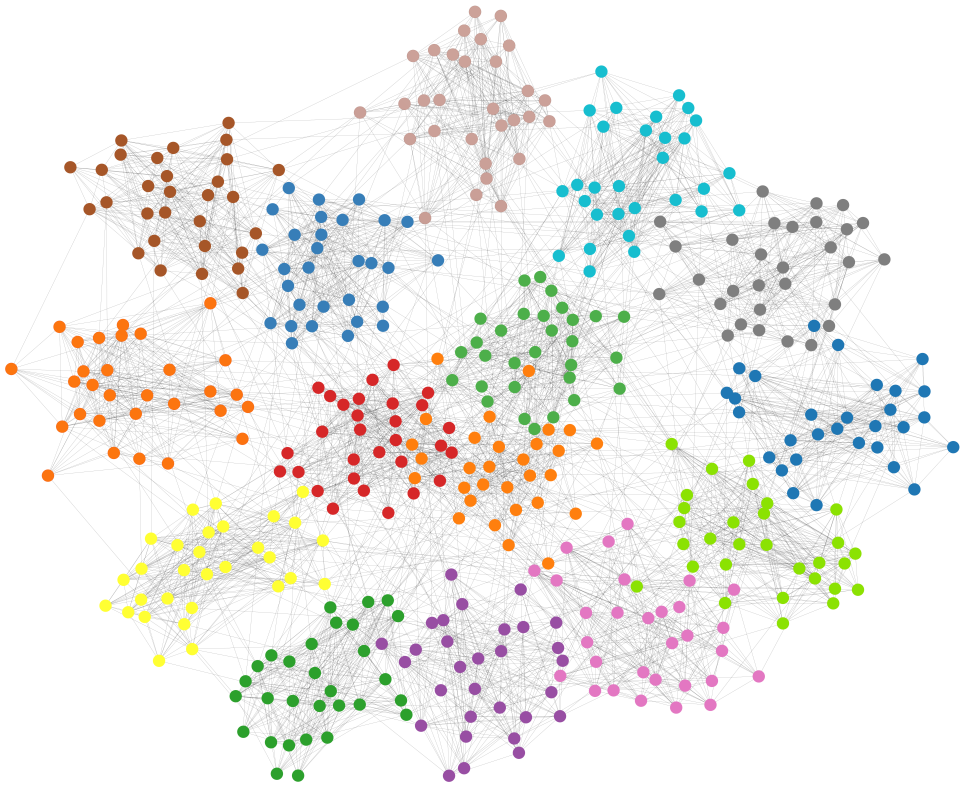
Application of the filter to scaled Laplacian $\tilde{\mathbf{\Delta}} = \mathbf{D}^{-0.5}\mathbf{\Delta}\mathbf{D}^{-0.5} - \mathbf{I} = \mathbf{\Phi}\tilde{\mathbf{\Lambda}}\mathbf{\Phi}^T$:

$$\mathbf{g} = \mathbf{\Phi}(\sum_{j=0}^{r} \alpha_j T_j(\tilde{\mathbf{\Lambda}}))\mathbf{\Phi}^T\mathbf{f} = \sum_{j=0}^{r} \alpha_j T_j(\tilde{\mathbf{\Delta}})\mathbf{f}$$

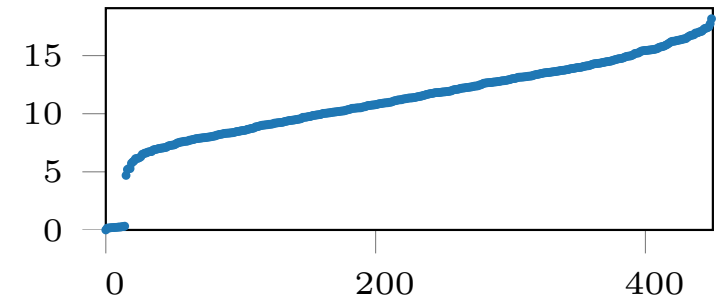
- ☺ $\mathcal{O}(1)$ parameters per layer
- ☺ Filters have guaranteed r-hops support
- ☺ No explicit computation of $\mathbf{\Phi}^\top, \mathbf{\Phi} \Rightarrow \mathcal{O}(r|\mathcal{E}|)$ computational complexity

Defferrard, Bresson, Vandergheynst 2016

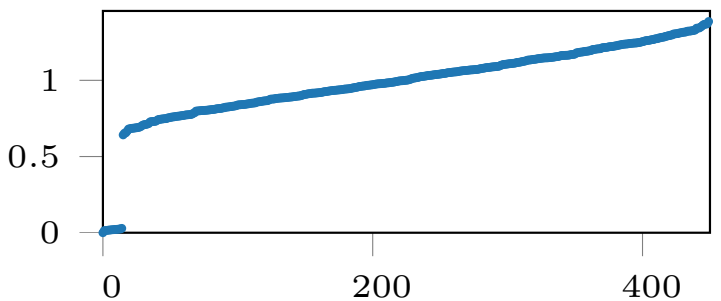
Community graph example



Synthetic graph with 15 communities



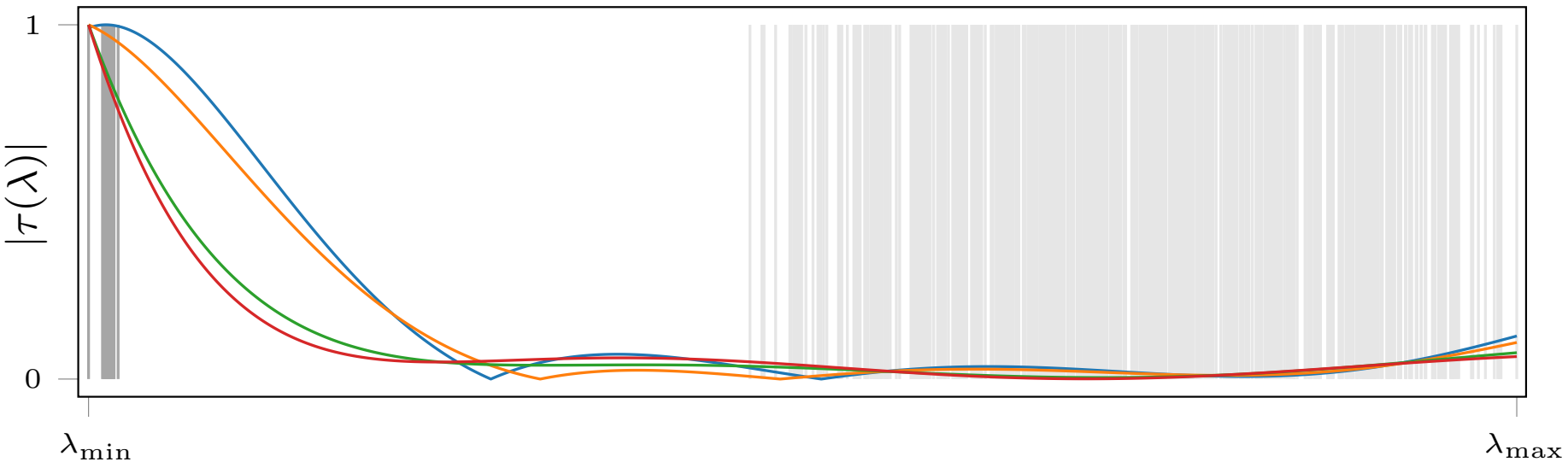
Unnorm. Laplacian eigenvalues



Normalized Laplacian eigenvalues

Levie et al. 2017

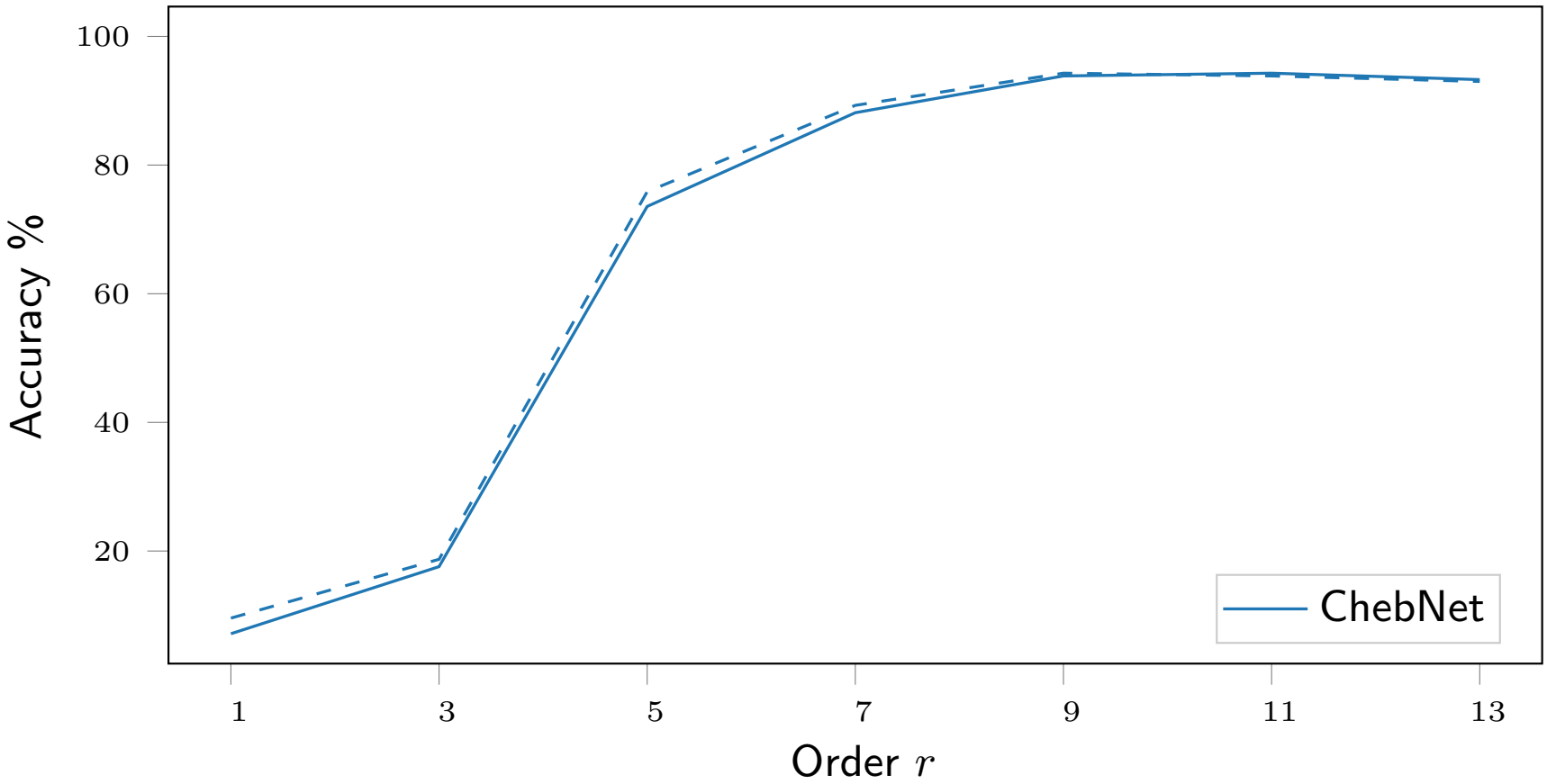
Community graph example



Example of Chebyshev filters learned on the 15-communities graph

Levie et al. 2017

Chebyshev: community graph example



Community detection accuracy of ChebNet on the synthetic 15-community graph

Levie et al. 2017

Spectral CNNs

- SplineNet: Bruna, Zaremba, Szlam, LeCun 2014; Henaff, Bruna, LeCun 2015
- ChebNet: Defferrard, Bresson, Vandergheynst 2016
- CayleyNet: Levie*, Monti*, Bresson, Bronstein 2017

Spectral zoom

Cayley transform $C(\lambda) = \frac{\lambda - i}{\lambda + i}$ is a smooth bijection from $\mathbb{R}$ to $e^{i\mathbb{R}} \setminus \{1\}$

Levie et al. 2017

Spectral zoom

Cayley transform $C(\lambda) = \frac{\lambda - i}{\lambda + i}$ is a smooth bijection from $\mathbb{R}$ to $e^{i\mathbb{R}} \setminus \{1\}$

Applying Cayley transform to the scaled eigenvalues $h\lambda$

$$C(h\lambda) = (h\lambda - i)(h\lambda + i)^{-1}$$

results in a non-linear transformation of the eigenvalues (spectral zoom)

Levie et al. 2017

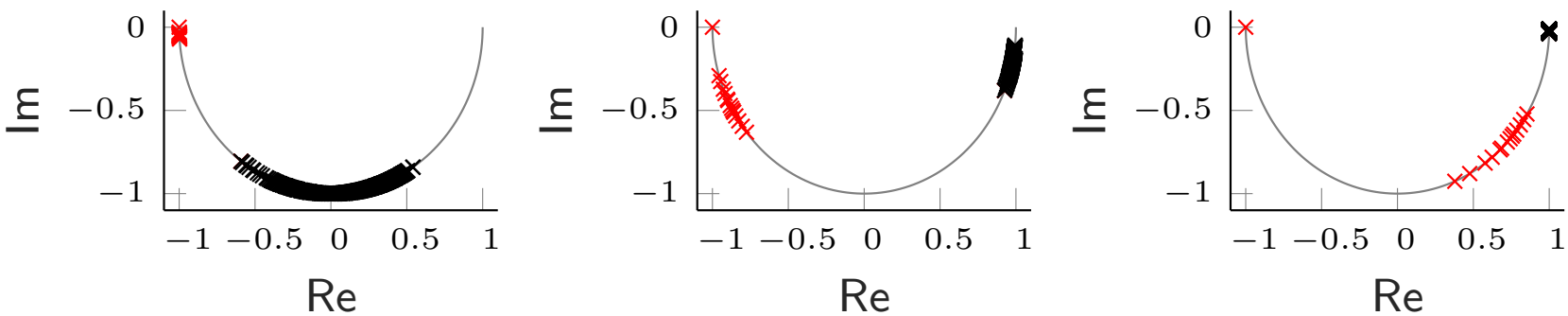
Spectral zoom

Cayley transform $C(\lambda) = \frac{\lambda - i}{\lambda + i}$ is a smooth bijection from $\mathbb{R}$ to $e^{i\mathbb{R}} \setminus \{1\}$

Applying Cayley transform to the scaled eigenvalues $h\lambda$

$$C(h\lambda) = (h\lambda - i)(h\lambda + i)^{-1}$$

results in a non-linear transformation of the eigenvalues (spectral zoom)



Cayley transform $C(h\lambda)$ for (left-to-right) $h = 0.1$, 1, and 10
of the 15-communities graph Laplacian spectrum

Levie et al. 2017

Cayley polynomials

Cayley polynomials of order r are a family of real-valued rational functions with complex coefficients c_j

$$\tau_{\mathbf{c}}(h\boldsymbol{\Delta}) = \boldsymbol{\Phi}\left(c_0 + \sum_{j=1}^{r} c_j C(h\boldsymbol{\Lambda})^j + \sum_{j=1}^{r} \bar{c_j} C(h\boldsymbol{\Lambda})^{-j}\right)\boldsymbol{\Phi}^T =$$

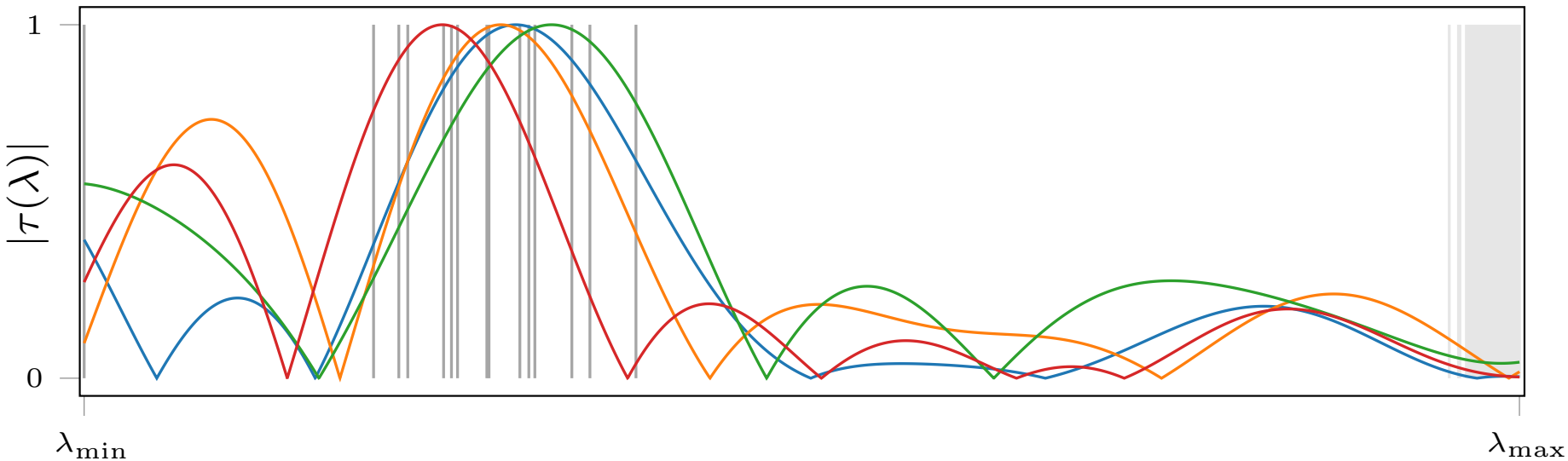
$$= c_0 + \sum_{j=1}^{r} c_j C(h\boldsymbol{\Delta})^j + \sum_{j=1}^{r} \bar{c_j} C(h\boldsymbol{\Delta})^{-j} =$$

$$= c_0 + 2\mathrm{Re}\Big\{\sum_{j=1}^{r} c_j C(h\boldsymbol{\Delta})^j\Big\}$$

Note that: $\boldsymbol{\Phi} C(h\Lambda)\boldsymbol{\Phi}^T = C(h\boldsymbol{\Delta}) = (h\boldsymbol{\Delta} - i\mathbf{I})(h\boldsymbol{\Delta} + i\mathbf{I})^{-1}$

(Cayley 1846); Levie et al. 2017

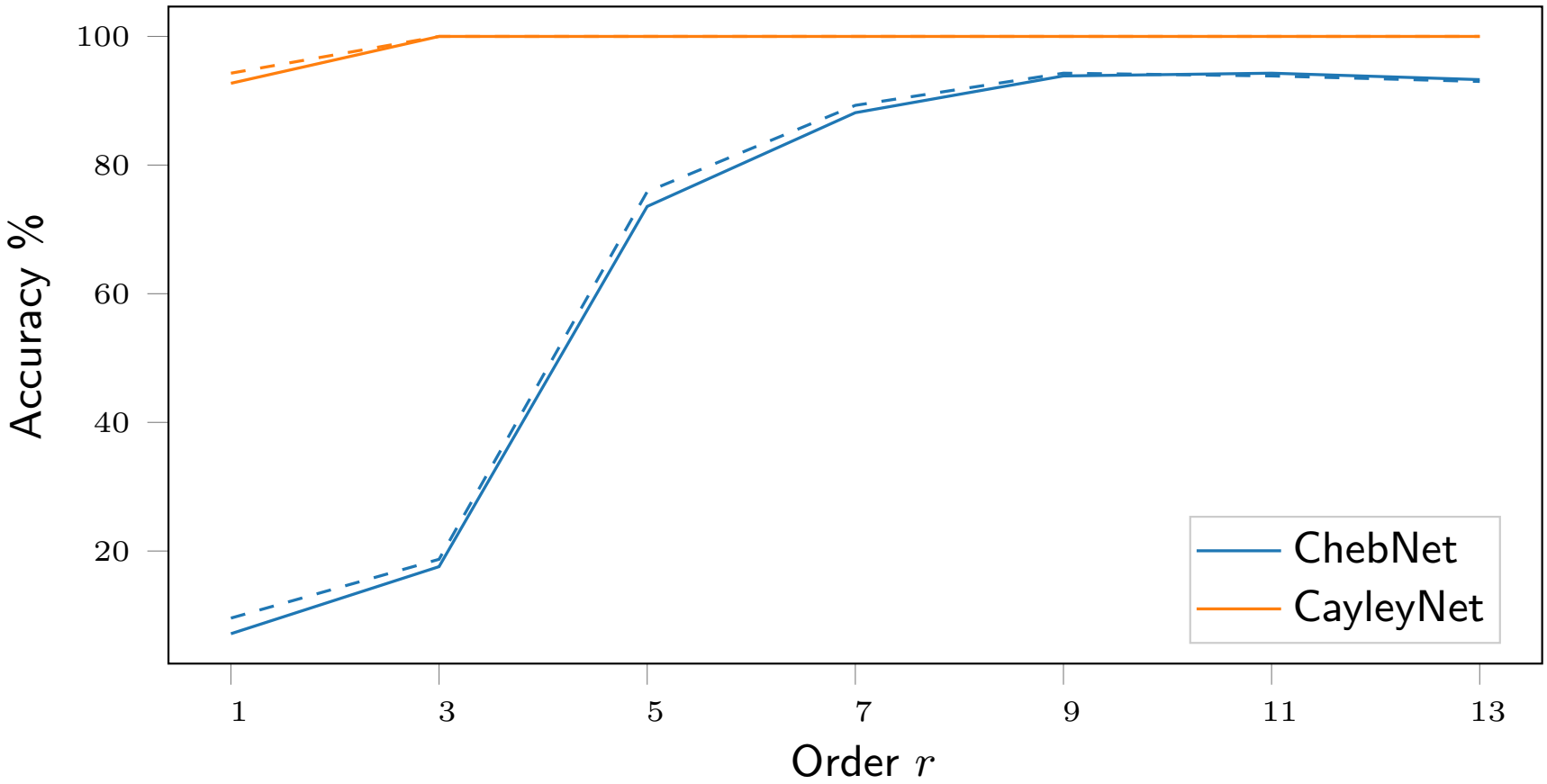
Chebyshev vs Cayley: community graph example



Example of Cayley filters learned on the 15-communities graph

Levie et al. 2017

Chebyshev vs Cayley: community graph example



Community detection accuracy of ChebNet and CayleyNet on the synthetic 15-community graph

Levie et al. 2017

Fast inversion

Application of Cayley filters $\tau_{\mathbf{c}}(h\boldsymbol{\Delta})\mathbf{f} = \mathrm{Re}\Big\{ \sum_{j=0}^{r} c_j C(h\boldsymbol{\Delta})^j \Big\}\mathbf{f}$ requires the computation of

$$
\begin{aligned}
\mathbf{y}_0 &= \mathbf{f} \\
\mathbf{y}_1 &= C(h\boldsymbol{\Delta})\mathbf{f} \\
\mathbf{y}_2 &= C(h\boldsymbol{\Delta})^2\mathbf{f} \\
&\vdots \\
\mathbf{y}_r &= C(h\boldsymbol{\Delta})^r\mathbf{f}
\end{aligned}
$$

with $C(h\boldsymbol{\Delta}) = (h\boldsymbol{\Delta} - i\mathbf{I})(h\boldsymbol{\Delta} + i\mathbf{I})^{-1} \to O(n^3)$ operations.

(Jacobi 1834); Levie et al. 2017

Fast inversion

Application of Cayley filters $\tau_{\mathbf{c}}(h\boldsymbol{\Delta})\mathbf{f} = \mathrm{Re}\Big\{ \sum_{j=0}^{r} c_j C(h\boldsymbol{\Delta})^j \Big\}\mathbf{f}$ requires the computation of

$$
\begin{aligned}
\mathbf{y}_0 &= \mathbf{f} \\
\mathbf{y}_1 &= C(h\boldsymbol{\Delta})\mathbf{f} = C(h\boldsymbol{\Delta})\mathbf{y}_0 \\
\mathbf{y}_2 &= C(h\boldsymbol{\Delta})^2\mathbf{f} = C(h\boldsymbol{\Delta})\mathbf{y}_1 \\
&\vdots \\
\mathbf{y}_r &= C(h\boldsymbol{\Delta})^r\mathbf{f} = C(h\boldsymbol{\Delta})\mathbf{y}_{r-1}
\end{aligned}
$$

with $C(h\boldsymbol{\Delta}) = (h\boldsymbol{\Delta} - i\mathbf{I})(h\boldsymbol{\Delta} + i\mathbf{I})^{-1} \to O(n^3)$ operations.

(Jacobi 1834); Levie et al. 2017

Fast inversion

Application of Cayley filters $\tau_{\mathbf{c}}(h\boldsymbol{\Delta})\mathbf{f} = \mathrm{Re}\Big\{ \sum_{j=0}^{r} c_j C(h\boldsymbol{\Delta})^j \Big\}\mathbf{f}$ requires the computation of

$$
\begin{aligned}
\mathbf{y}_0 &= \mathbf{f} \\
\mathbf{y}_1 &= C(h\boldsymbol{\Delta})\mathbf{f} = C(h\boldsymbol{\Delta})\mathbf{y}_0 \\
\mathbf{y}_2 &= C(h\boldsymbol{\Delta})^2\mathbf{f} = C(h\boldsymbol{\Delta})\mathbf{y}_1 \\
&\vdots \\
\mathbf{y}_r &= C(h\boldsymbol{\Delta})^r\mathbf{f} = C(h\boldsymbol{\Delta})\mathbf{y}_{r-1}
\end{aligned}
$$

For a generic power j we have therefore

$$
\mathbf{y}_j = C(h\boldsymbol{\Delta})\mathbf{y}_{j-1} = (h\boldsymbol{\Delta} - i\mathbf{I})(h\boldsymbol{\Delta} + i\mathbf{I})^{-1}\mathbf{y}_{j-1}
$$

$$
\downarrow
$$

$$
(h\boldsymbol{\Delta} + i\mathbf{I})\mathbf{y}_j = (h\boldsymbol{\Delta} - i\mathbf{I})\mathbf{y}_{j-1}
$$

(Jacobi 1834); Levie et al. 2017

Fast inversion

Approximate solution $\tilde{\mathbf{y}}_j \approx \mathbf{y}_j$ using K Jacobi iterations

$$\begin{aligned}\tilde{\mathbf{y}}_j^{(k+1)} &= \mathbf{J}\tilde{\mathbf{y}}_j^{(k)} + \mathrm{Diag}^{-1}(h\boldsymbol{\Delta} + i\mathbf{I})(h\boldsymbol{\Delta} - i\mathbf{I})\tilde{\mathbf{y}}_{j-1} \\ \tilde{\mathbf{y}}_j^{(0)} &= \mathrm{Diag}^{-1}(h\boldsymbol{\Delta} + i\mathbf{I})(h\boldsymbol{\Delta} - i\mathbf{I})\tilde{\mathbf{y}}_{j-1}\end{aligned}$$

with $\mathbf{J} = -\mathrm{Diag}^{-1}(h\boldsymbol{\Delta} + i\mathbf{I})\mathrm{Off}(h\boldsymbol{\Delta} + i\mathbf{I})$

(Jacobi 1834); Levie et al. 2017

Fast inversion

Approximate solution $\tilde{\mathbf{y}}_j \approx \mathbf{y}_j$ using K Jacobi iterations

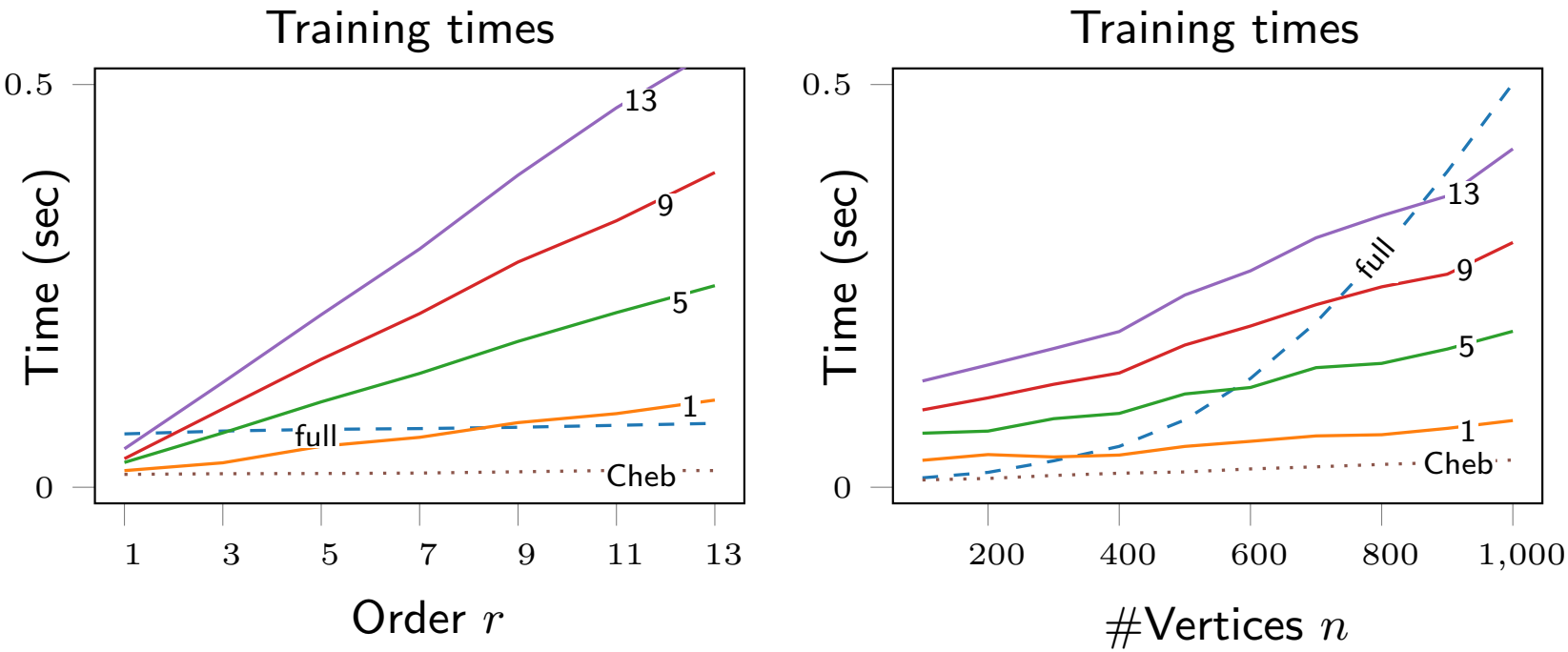
$$\begin{aligned}\tilde{\mathbf{y}}_j^{(k+1)} &= \mathbf{J}\tilde{\mathbf{y}}_j^{(k)} + \mathrm{Diag}^{-1}(h\boldsymbol{\Delta} + i\mathbf{I})(h\boldsymbol{\Delta} - i\mathbf{I})\tilde{\mathbf{y}}_{j-1} \\ \tilde{\mathbf{y}}_j^{(0)} &= \mathrm{Diag}^{-1}(h\boldsymbol{\Delta} + i\mathbf{I})(h\boldsymbol{\Delta} - i\mathbf{I})\tilde{\mathbf{y}}_{j-1}\end{aligned}$$

with $\mathbf{J} = -\mathrm{Diag}^{-1}(h\boldsymbol{\Delta} + i\mathbf{I})\mathrm{Off}(h\boldsymbol{\Delta} + i\mathbf{I})$

Cost: $\sum_{j=0}^{r} c_j \tilde{\mathbf{y}}_j \approx \tau(h\boldsymbol{\Delta})\mathbf{f}$ has $\mathcal{O}(rK|\mathcal{E}|)$ complexity for sparse graphs.

(Jacobi 1834); Levie et al. 2017

Computational complexity of approximate inversion



Training computational complexities of CayleyNet

Levie et al. 2017

Error bound

Unnormalized Laplacian: $d = \max_j d_{jj}$ and $\kappa = \|\mathbf{J}\|_\infty = \frac{hd}{\sqrt{h^2 d^2 + 1}} < 1$

Normalized Laplacian: Assume that $(h\tilde{\mathbf{\Delta}} + i\mathbf{I})$ is dominant diagonal s.t. $\kappa = \|\mathbf{J}\|_\infty < 1$

(Jacobi 1834); Levie et al. 2017

Error bound

Unnormalized Laplacian: $d = \max_j d_{jj}$ and $\kappa = \|\mathbf{J}\|_\infty = \frac{hd}{\sqrt{h^2 d^2 + 1}} < 1$

Normalized Laplacian: Assume that $(h\tilde{\mathbf{\Delta}} + i\mathbf{I})$ is dominant diagonal s.t. $\kappa = \|\mathbf{J}\|_\infty < 1$

Theorem 1 (approximation error) Under the above assumptions

$$\frac{\left\| \sum_{j=0}^{r} c_j \tilde{\mathbf{y}}_j - \tau(h\mathbf{\Delta})\mathbf{f} \right\|_2}{\|\tau(h\mathbf{\Delta})\mathbf{f}\|_2} < M\kappa^K$$

where $M = \sqrt{n}\sum_{j=1}^{r} j|c_j|$ for a general graph and $M = \sum_{j=1}^{r} j|c_j|$ for a regular graph

(Jacobi 1834); Levie et al. 2017

Exponential decay

Exponential decay on graphs $f \in L^p(\mathcal{V})$, $1 \leq p \leq \infty$ has exponential decay about vertex m if $\exists \gamma \in (0,1)$ and $c > 0$ such that for any k

$$\|f|_{\mathcal{N}_{k,m}^c}\|_p \leq c\gamma^k \|f\|_p$$

where $\mathcal{N}_{k,m}$ is the k-hop neighborhood of vertex m

Levie et al. 2017

Exponential decay

Exponential decay on graphs $f \in L^p(\mathcal{V})$, $1 \leq p \leq \infty$ has exponential decay about vertex m if $\exists \gamma \in (0,1)$ and $c > 0$ such that for any k

$$\|f|_{\mathcal{N}_{k,m}^c}\|_p \leq c\gamma^k \|f\|_p$$

where $\mathcal{N}_{k,m}$ is the k-hop neighborhood of vertex m

Compare to

Exponential decay in $\mathbb{R}$ $f(x)$ has exponential decay (about 0) if $\exists \gamma \in (0,1)$ and $c > 0$ such that for any $\rho > 0$

$$\|f|_{B_\rho^c}\|_\infty \leq c\gamma^{-\rho} \|f\|_\infty$$

where B_ρ is a ball of radius ρ about 0.

Levie et al. 2017

Exponential decay

Exponential decay on graphs $f \in L^p(\mathcal{V})$, $1 \leq p \leq \infty$ has exponential decay about vertex m if $\exists \gamma \in (0,1)$ and $c > 0$ such that for any k

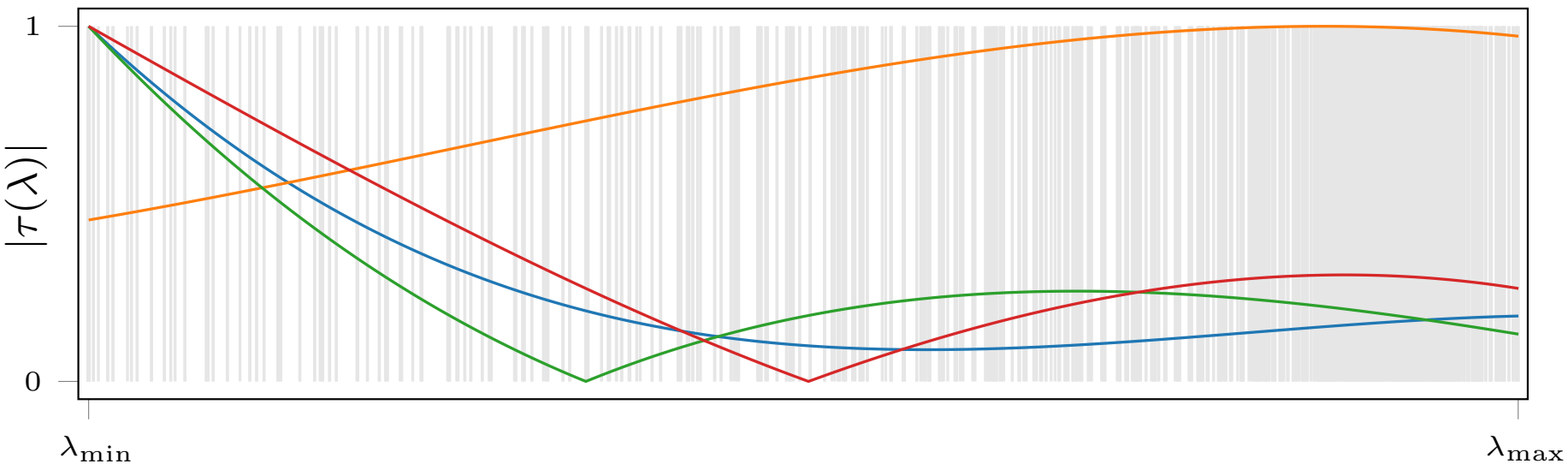
$$\|f|_{\mathcal{N}_{k,m}^c}\|_p \leq c\gamma^k \|f\|_p$$

where $\mathcal{N}_{k,m}$ is the k-hop neighborhood of vertex m

Theorem 2 (exponential decay of Cayley filters) Let $\tau(h\boldsymbol{\Delta})$ be a Cayley filter of order r and δ_m a delta-function at vertex m of the graph. Then, $\tau(h\boldsymbol{\Delta})\boldsymbol{\delta}_m$ has exponential decay about m with $p = 2$, $c = 2M/\|\tau(\boldsymbol{\Delta})\boldsymbol{\delta}_m\|_2$ and $\gamma = \kappa^{1/r}$ (where $M = \sqrt{n}\sum_{j=1}^{r} j|c_j|$, $\kappa = \|\mathbf{J}\|_\infty$ as in Theorem 1)

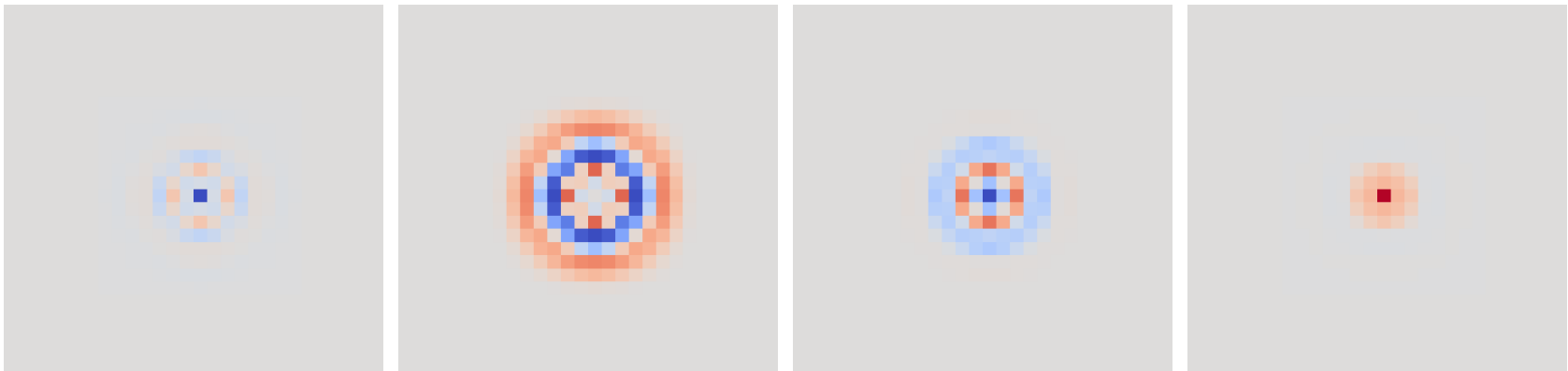
Levie et al. 2017

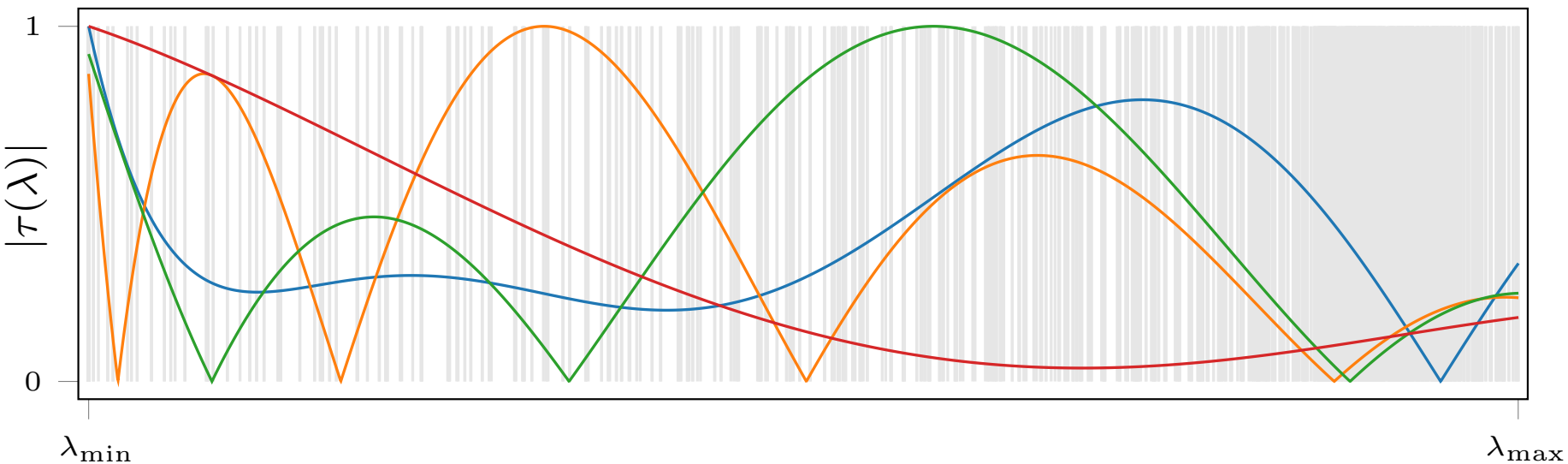
Chebyshev vs Cayley



Example of Chebyshev filters (order $r = 3$) on Euclidean grid

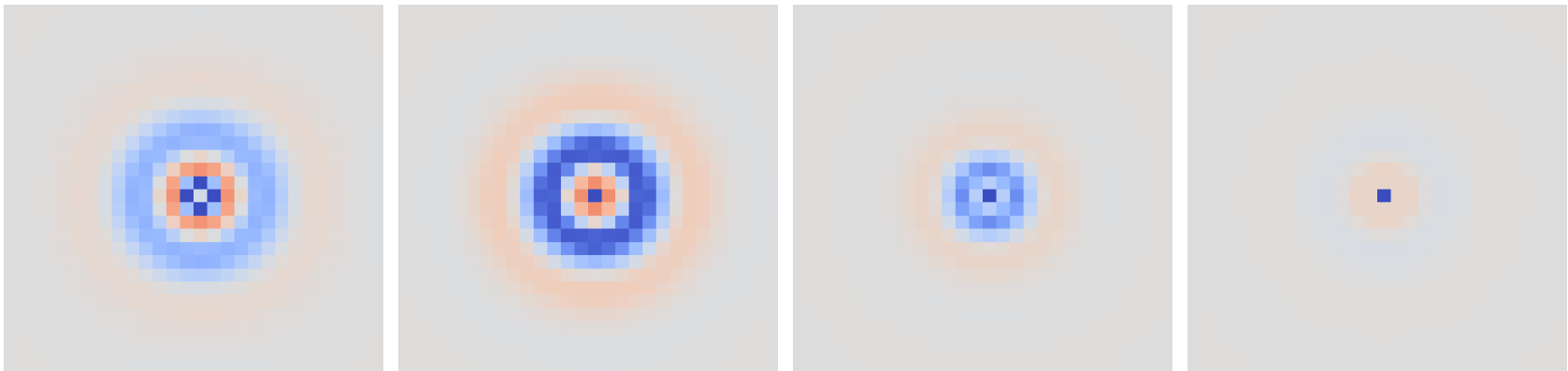
Chebyshev vs Cayley

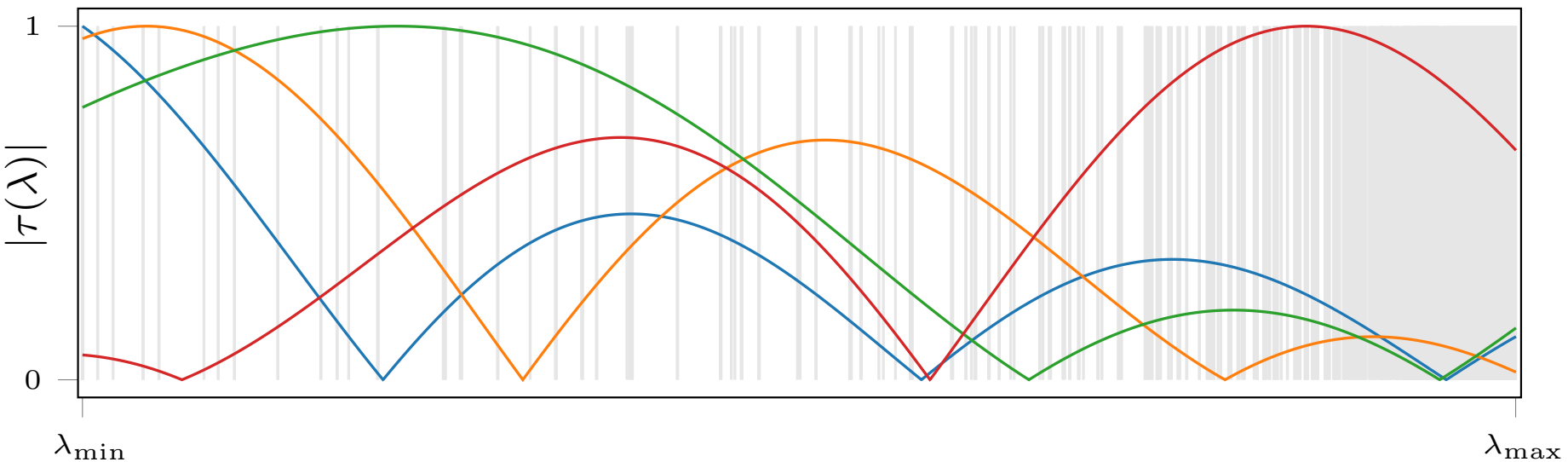




Example of Chebyshev filters (order $r = 7$) on Euclidean grid

Chebyshev vs Cayley





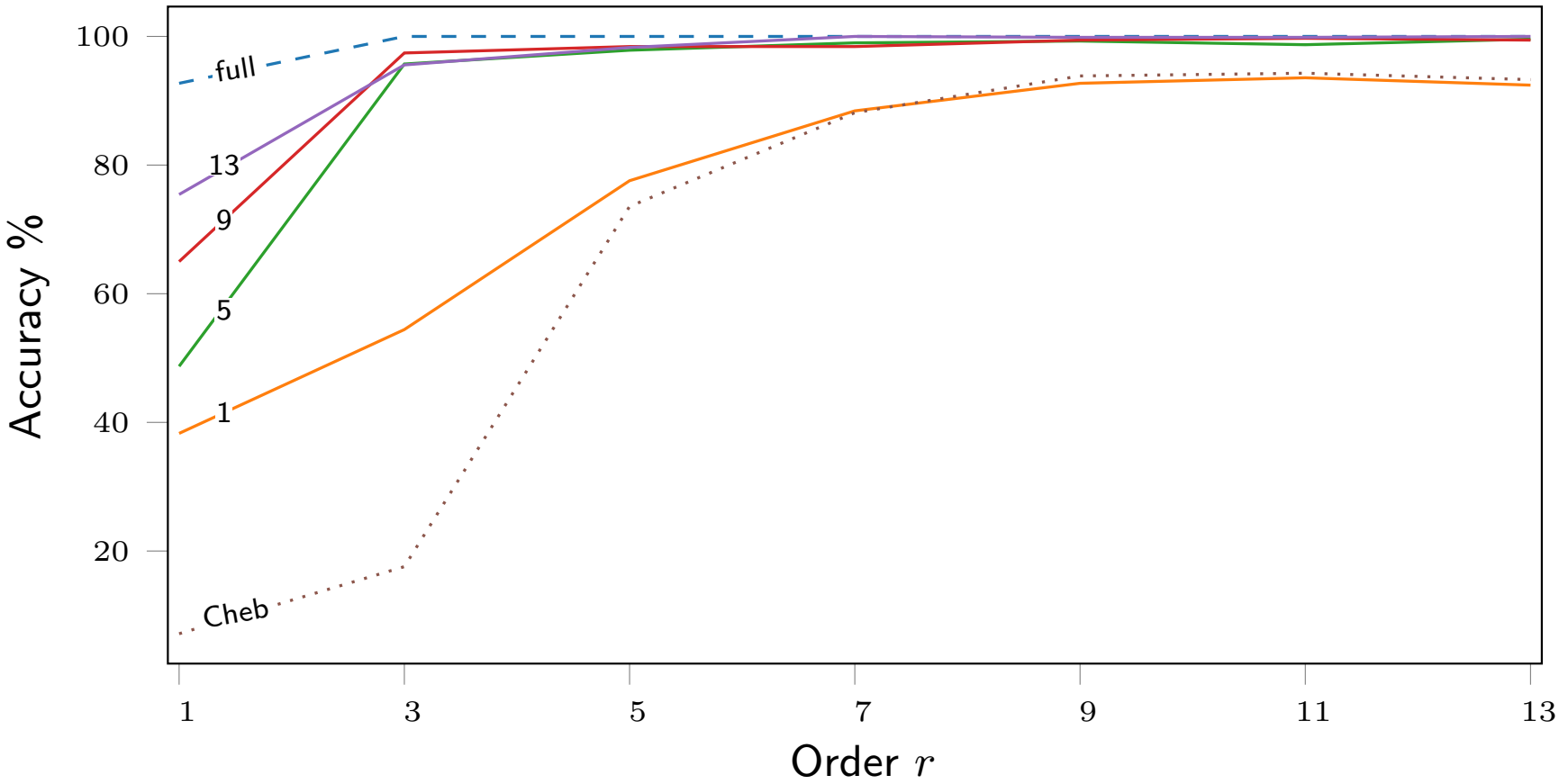
Example of Cayley filters (order $r = 3$) on Euclidean grid

Poster

MotifNet: a motif-based Graph Convolutional Network for directed graphs

F. Monti, K. Otness, M. M. Bronstein

Accuracy of approximate inversion

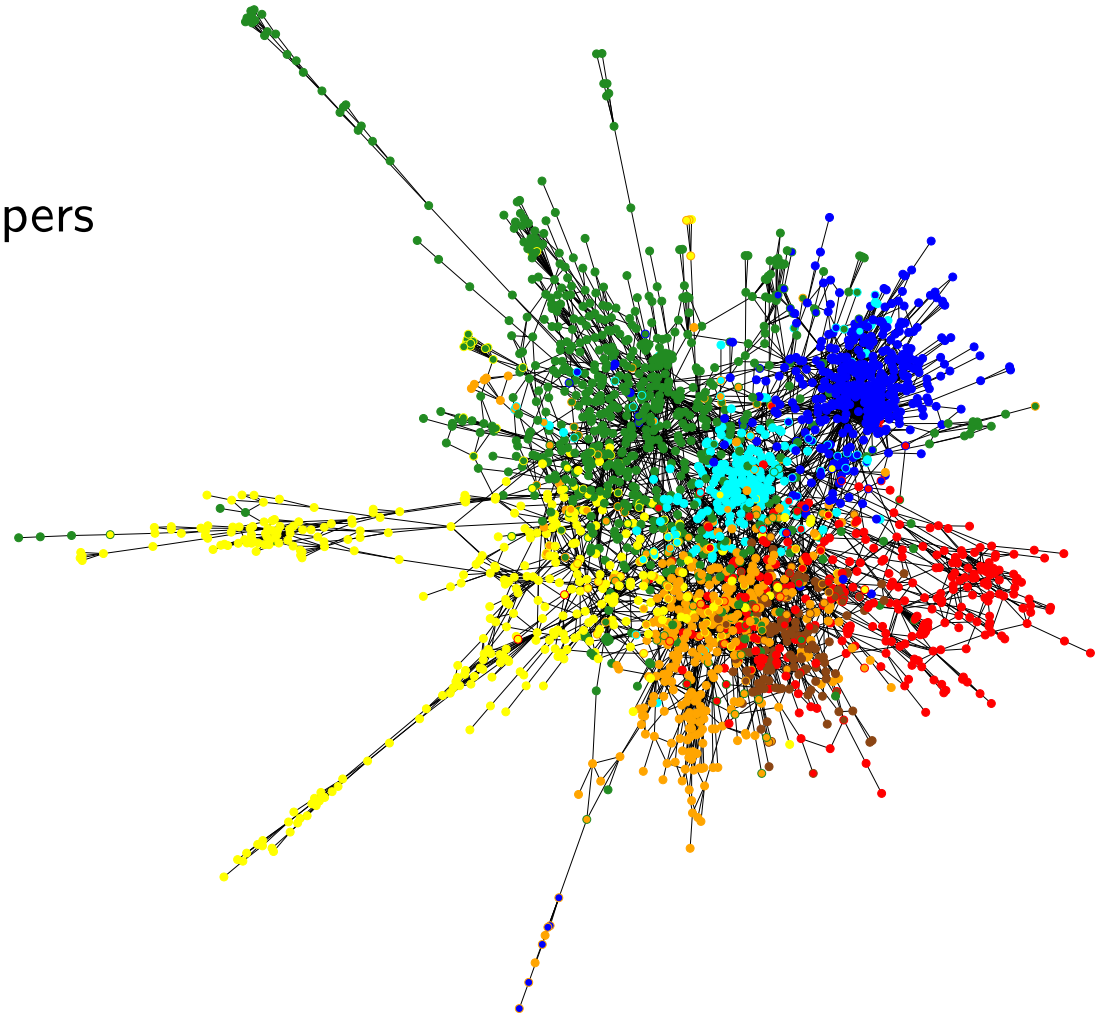


Community detection accuracy of CayleyNet using approximate Jacobi inversion on the synthetic 15-community graph

Levie et al. 2017

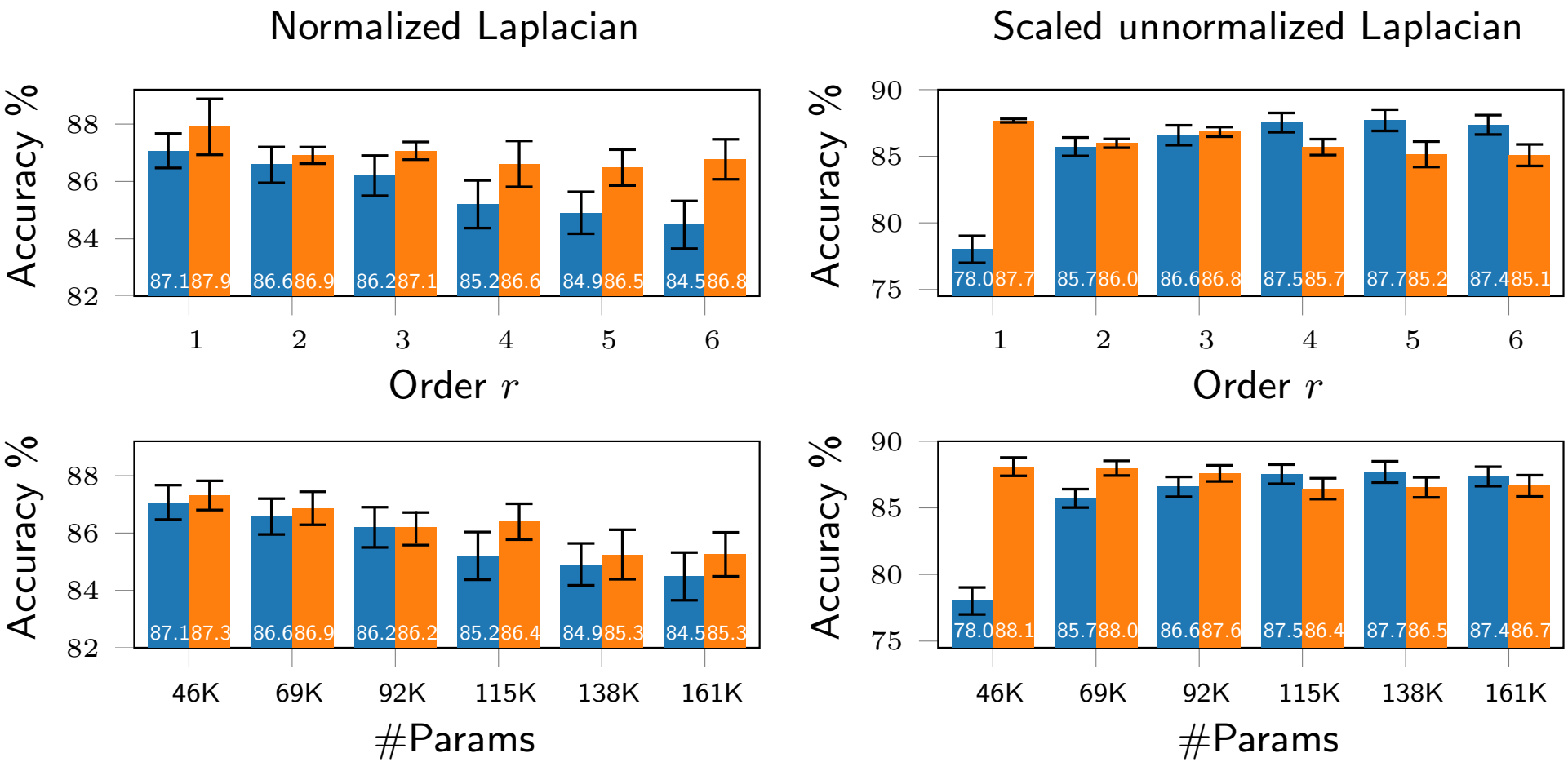
Cora dataset

- Citations network representing papers (vertices) and citations (edges).
- Goal: vertex-wise classification (paper topic).
- 2708 documents, 7 topics.
- Training set: 1,708 vertices; validation set: 500 vertices; test set: 500 vertices.



Set et al. 2008

Chebyshev vs Cayley: Cora example



ChebNet (blue) and CayleyNet (orange) test accuracies on the CORA dataset. Polynomials with complex coefficients (top) and real coefficients (bottom) have been exploited with CayleyNet in the two analysis.

Levie et al. 2017; data: Set et al. 2008

CayleyNet: Spectral graph CNN with Cayley polynomials

Represent spectral transfer function as a Cayley polynomial or order r

$$\tau_{\mathbf{c},h}(\lambda) = c_0 + 2\mathrm{Re}\Big\{ \sum_{j=1}^{r} c_j (h\lambda - i)^j (h\lambda + i)^{-j} \Big\}$$

where the filter parameters are the vector of real/complex coefficients $\mathbf{c} = (c_0, \ldots, c_r)^\top$ and the spectral zoom h

Levie et al. 2017

CayleyNet: Spectral graph CNN with Cayley polynomials

Represent spectral transfer function as a Cayley polynomial or order r

$$\tau_{\mathbf{c},h}(\lambda) = c_0 + 2\mathrm{Re}\Big\{ \sum_{j=1}^{r} c_j (h\lambda - i)^j (h\lambda + i)^{-j} \Big\}$$

where the filter parameters are the vector of real/complex coefficients $\mathbf{c} = (c_0, \ldots, c_r)^\top$ and the spectral zoom h

- ☺ $\mathcal{O}(1)$ parameters per layer
- ☺ Filters have guaranteed exponential spatial decay
- ☺ $\mathcal{O}(r|\mathcal{E}|)$ computational complexity with Jacobi approximate inversion (assuming sparsely-connected graph)

Levie et al. 2017

CayleyNet: Spectral graph CNN with Cayley polynomials

Represent spectral transfer function as a Cayley polynomial or order r

$$\tau_{\mathbf{c},h}(\lambda) = c_0 + 2\mathrm{Re}\Big\{ \sum_{j=1}^{r} c_j (h\lambda - i)^j (h\lambda + i)^{-j} \Big\}$$

where the filter parameters are the vector of real/complex coefficients $\mathbf{c} = (c_0, \ldots, c_r)^\top$ and the spectral zoom h

- ☺ $\mathcal{O}(1)$ parameters per layer
- ☺ Filters have guaranteed exponential spatial decay
- ☺ $\mathcal{O}(r|\mathcal{E}|)$ computational complexity with Jacobi approximate inversion (assuming sparsely-connected graph)
- ☺ Spectral zoom property allowing to better localize in frequency

Levie et al. 2017

CayleyNet: Spectral graph CNN with Cayley polynomials

Represent spectral transfer function as a Cayley polynomial or order r

$$\tau_{\mathbf{c},h}(\lambda) = c_0 + 2\mathrm{Re}\Big\{ \sum_{j=1}^{r} c_j (h\lambda - i)^j (h\lambda + i)^{-j} \Big\}$$

where the filter parameters are the vector of real/complex coefficients $\mathbf{c} = (c_0, \dots, c_r)^\top$ and the spectral zoom h

- ☺ $\mathcal{O}(1)$ parameters per layer
- ☺ Filters have guaranteed exponential spatial decay
- ☺ $\mathcal{O}(r|\mathcal{E}|)$ computational complexity with Jacobi approximate inversion (assuming sparsely-connected graph)
- ☺ Spectral zoom property allowing to better localize in frequency
- ☺ Richer class of filters than Chebyshev for the same order

Levie et al. 2017

MGCNN

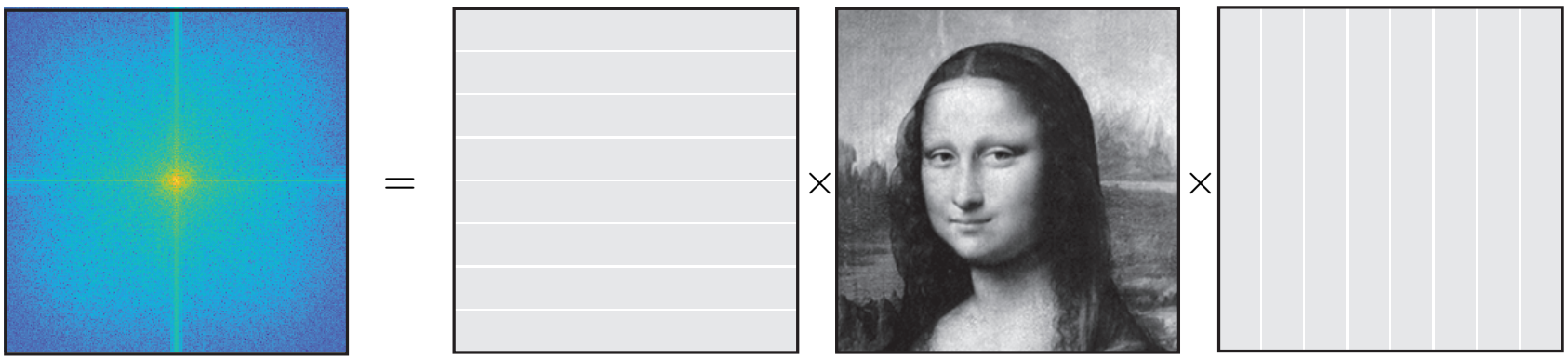
2D Fourier transform

$\mathbf{X}$

2D Fourier transform

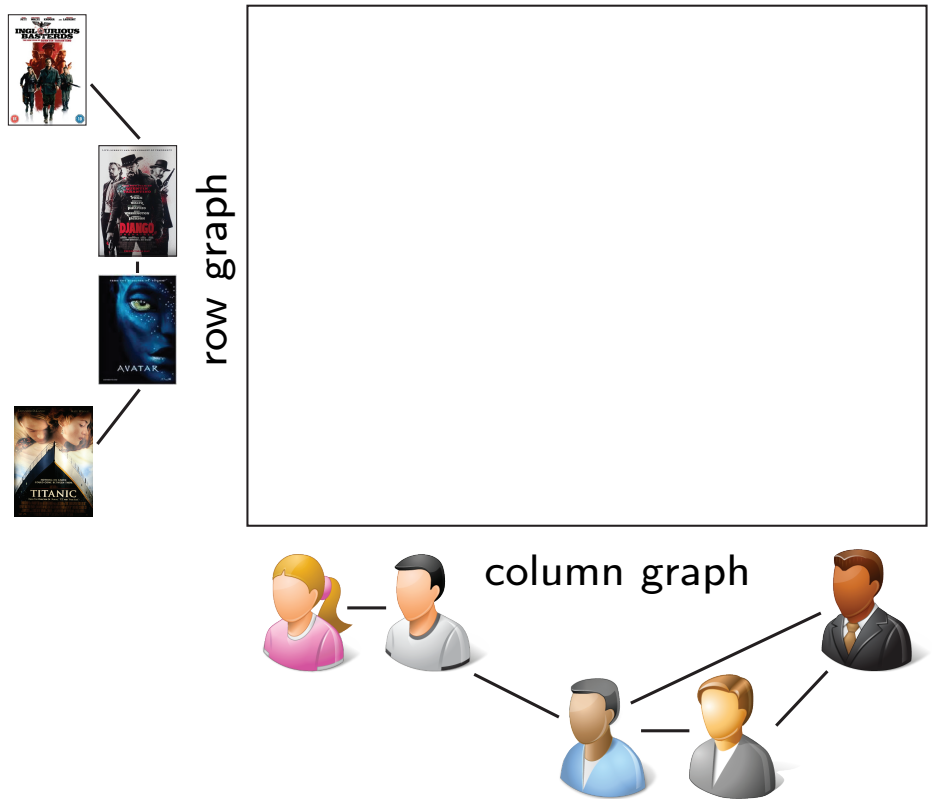
Column-wise trasform

2D Fourier transform



Column-wise trasform + Row-wise transform = 2D transform

Multi-graph Fourier transform



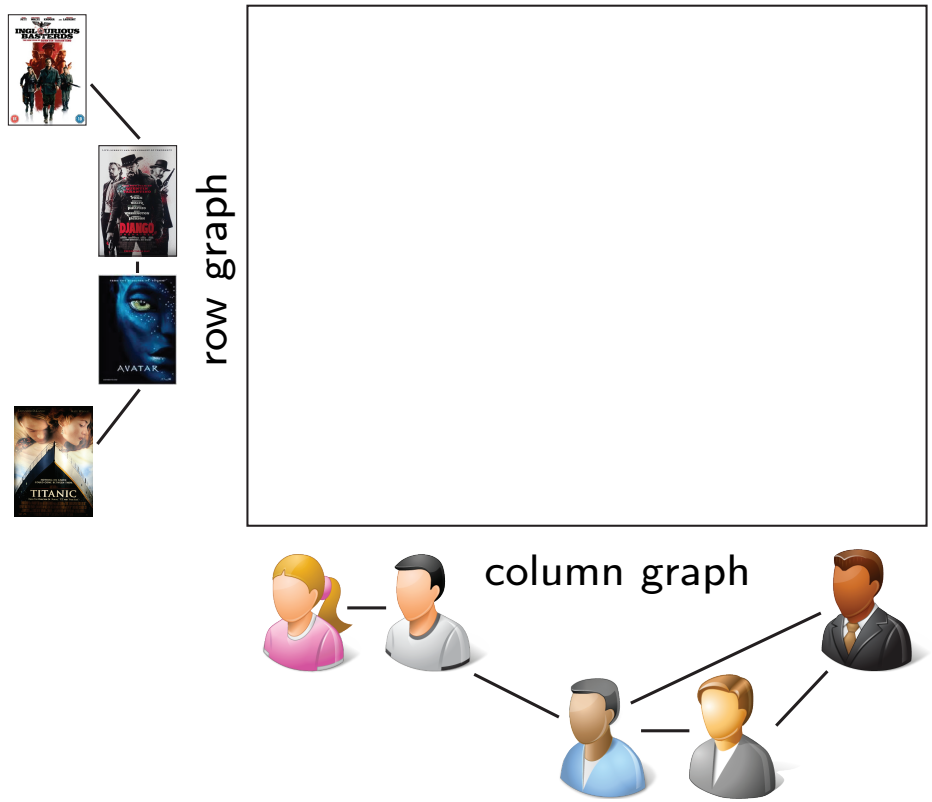
Multi-graph Fourier transform

$$\hat{\mathbf{X}} = \mathbf{\Phi}_{\mathrm{r}}^{\top} \mathbf{X} \mathbf{\Phi}_{\mathrm{c}}$$

where $\mathbf{\Phi}_{\mathrm{c}}$ and $\mathbf{\Phi}_{\mathrm{r}}$ are the eigenvectors of the column- and row-graph Laplacians $\mathbf{\Delta}_{\mathrm{c}}$ and $\mathbf{\Delta}_{\mathrm{r}}$, respectively

Monti, Bresson, Bronstein 2017

Multi-graph convolution



Multi-graph spectral convolution

$$\mathbf{X} \star \mathbf{Y} = \mathbf{\Phi}_{\mathrm{r}}(\hat{\mathbf{X}} \circ \hat{\mathbf{Y}})\mathbf{\Phi}_{\mathrm{c}}^{\top}$$

Monti, Bresson, Bronstein 2017

Multi-Graph CNN

Multi-graph spectral coefficient parametrization

$$\tau_{\mathbf{\Theta}}(\tilde{\lambda}_c, \tilde{\lambda}_r) = \sum_{j,j'=0}^{r} \theta_{jj'} T_j(\tilde{\lambda}_c) T_{j'}(\tilde{\lambda}_r)$$

Monti, Bresson, Bronstein 2017

Multi-Graph CNN

Multi-graph spectral convolutional layer

$$\mathbf{Y}_l = \xi\left(\sum_{l'=1}^{p}\sum_{j,j'=0}^{r}\theta_{jj'll'}T_j(\tilde{\mathbf{\Delta}}_{\mathrm{r}})\mathbf{X}_{l'}T_{j'}(\tilde{\mathbf{\Delta}}_{\mathrm{c}})\right) \quad \begin{array}{l} l=1,\dots,q \\ l'=1,\dots,p \end{array}$$

applied to p input channels ($m \times n$ matrices $\mathbf{X}_1,\dots,\mathbf{X}_p$) and producing q output channels ($m \times n$ matrices $\mathbf{Y}_1,\dots,\mathbf{Y}_q$)

Monti, Bresson, Bronstein 2017

Multi-Graph CNN

Multi-graph spectral convolutional layer

$$\mathbf{Y}_l = \xi\left(\sum_{l'=1}^{p}\sum_{j,j'=0}^{r}\theta_{jj'll'}T_j(\tilde{\mathbf{\Delta}}_{\mathrm{r}})\mathbf{X}_{l'}T_{j'}(\tilde{\mathbf{\Delta}}_{\mathrm{c}})\right) \quad \begin{matrix} l=1,\dots,q \\ l'=1,\dots,p \end{matrix}$$

applied to p input channels ($m \times n$ matrices $\mathbf{X}_1,\dots,\mathbf{X}_p$) and producing q output channels ($m \times n$ matrices $\mathbf{Y}_1,\dots,\mathbf{Y}_q$)

- ☺ $\mathcal{O}(1)$ parameters per layer
- ☺ Filters have guaranteed r-hops support on both graphs

Monti, Bresson, Bronstein 2017

Multi-Graph CNN

Multi-graph spectral convolutional layer

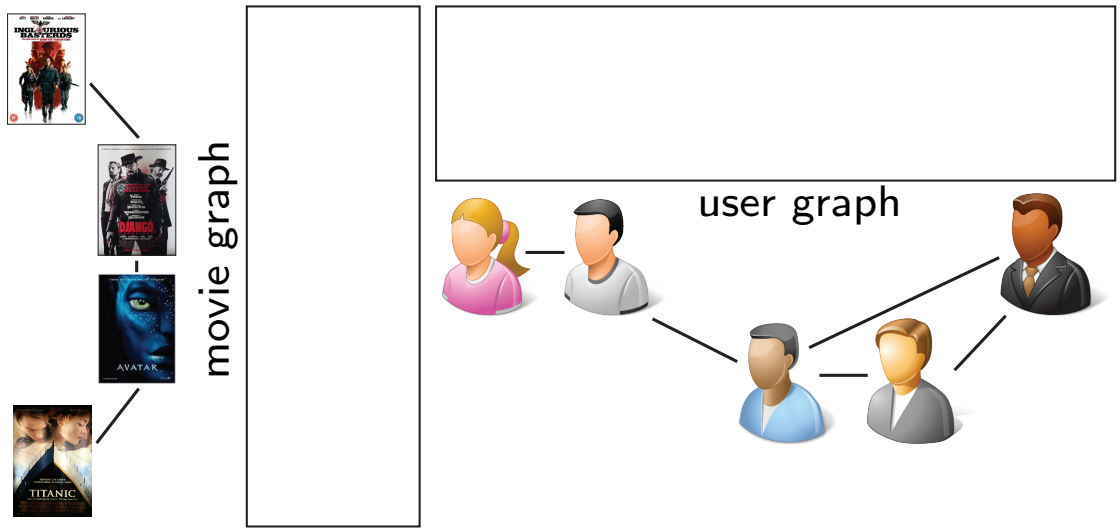
$$\mathbf{Y}_l = \xi\left(\sum_{l'=1}^{p}\sum_{j,j'=0}^{r}\theta_{jj'll'}T_j(\tilde{\mathbf{\Delta}}_{\mathrm{r}})\mathbf{X}_{l'}T_{j'}(\tilde{\mathbf{\Delta}}_{\mathrm{c}})\right) \quad \begin{matrix} l=1,\dots,q \\ l'=1,\dots,p \end{matrix}$$

applied to p input channels ($m \times n$ matrices $\mathbf{X}_1,\dots,\mathbf{X}_p$) and producing q output channels ($m \times n$ matrices $\mathbf{Y}_1,\dots,\mathbf{Y}_q$)

- ☺ $\mathcal{O}(1)$ parameters per layer
- ☺ Filters have guaranteed r-hops support on both graphs
- ☹ $\mathcal{O}(nm)$ computational complexity

Monti, Bresson, Bronstein 2017

Separable multi-graph spectral filters



Separable filters applied to row- and column factors independently

$$\mathbf{u}_l = \sum_{j=0}^{r} \theta_{\mathrm{r},j} T_j(\tilde{\mathbf{\Delta}}_\mathrm{r}) \mathbf{w}_l \qquad \mathbf{v}_l = \sum_{j'=0}^{r} \theta_{\mathrm{c},j'} T_{j'}(\tilde{\mathbf{\Delta}}_\mathrm{c}) \mathbf{h}_l \qquad l = 1, \dots, s$$

where $\boldsymbol{\theta}_\mathrm{r} = (\theta_{\mathrm{r},0}, \dots, \theta_{\mathrm{r},r})$ and $\boldsymbol{\theta}_\mathrm{c} = (\theta_{\mathrm{c},0}, \dots, \theta_{\mathrm{c},r})$ are the parameters of the row- and column- filters, $\mathbf{W} = (\mathbf{w}_1, \dots, \mathbf{w}_s)$ and $\mathbf{H} = (\mathbf{h}_1, \dots, \mathbf{h}_s)$

Monti, Bresson, Bronstein 2017

Separable Multi-Graph CNN

Two spectral convolutional layers applied to each of the factors $\mathbf{W}, \mathbf{H}$

$$\mathbf{u}_l = \xi\left(\sum_{l'=1}^{p}\sum_{j=0}^{r}\theta_{\mathrm{r},jll'}T_j(\tilde{\mathbf{\Delta}}_{\mathrm{r}})\mathbf{w}_{l'}\right) \qquad \begin{array}{l} l=1,\dots,q \\ l'=1,\dots,p \end{array}$$

$$\mathbf{v}_l = \xi\left(\sum_{l'=1}^{p}\sum_{j'=0}^{r}\theta_{\mathrm{c},j'll'}T_{j'}(\tilde{\mathbf{\Delta}}_{\mathrm{c}})\mathbf{h}_{l'}\right)$$

Monti, Bresson, Bronstein 2017

Separable Multi-Graph CNN

Two spectral convolutional layers applied to each of the factors $\mathbf{W}, \mathbf{H}$

$$\mathbf{u}_l = \xi\left(\sum_{l'=1}^{p}\sum_{j=0}^{r}\theta_{\mathrm{r},jll'}T_j(\tilde{\mathbf{\Delta}}_{\mathrm{r}})\mathbf{w}_{l'}\right) \qquad \begin{array}{l} l=1,\dots,q \\ l'=1,\dots,p \end{array}$$

$$\mathbf{v}_l = \xi\left(\sum_{l'=1}^{p}\sum_{j'=0}^{r}\theta_{\mathrm{c},j'll'}T_{j'}(\tilde{\mathbf{\Delta}}_{\mathrm{c}})\mathbf{h}_{l'}\right)$$

☺ $\mathcal{O}(1)$ parameters per layer

☺ Filters have guaranteed r-hops support on both graphs

Monti, Bresson, Bronstein 2017

Separable Multi-Graph CNN

Two spectral convolutional layers applied to each of the factors $\mathbf{W}, \mathbf{H}$

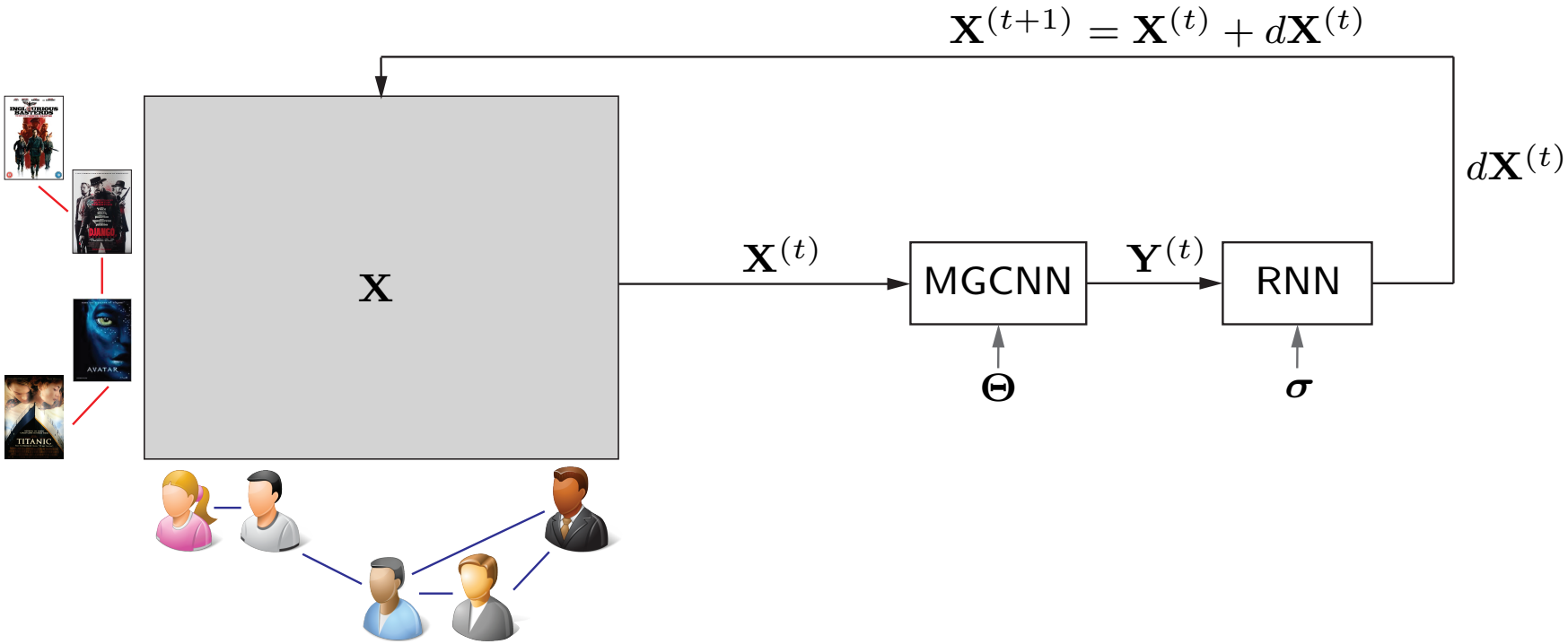
$$\mathbf{u}_l = \xi\left(\sum_{l'=1}^{p}\sum_{j=0}^{r}\theta_{\mathrm{r},jll'}T_j(\tilde{\mathbf{\Delta}}_{\mathrm{r}})\mathbf{w}_{l'}\right) \qquad \begin{matrix} l=1,\dots,q \\ l'=1,\dots,p \end{matrix}$$

$$\mathbf{v}_l = \xi\left(\sum_{l'=1}^{p}\sum_{j'=0}^{r}\theta_{\mathrm{c},j'll'}T_{j'}(\tilde{\mathbf{\Delta}}_{\mathrm{c}})\mathbf{h}_{l'}\right)$$

- ☺ $\mathcal{O}(1)$ parameters per layer
- ☺ Filters have guaranteed r-hops support on both graphs
- ☺ $\mathcal{O}(n+m)$ computational complexity (assuming sparse graph)

Monti, Bresson, Bronstein 2017

Matrix completion with Recurrent Multi-Graph CNN

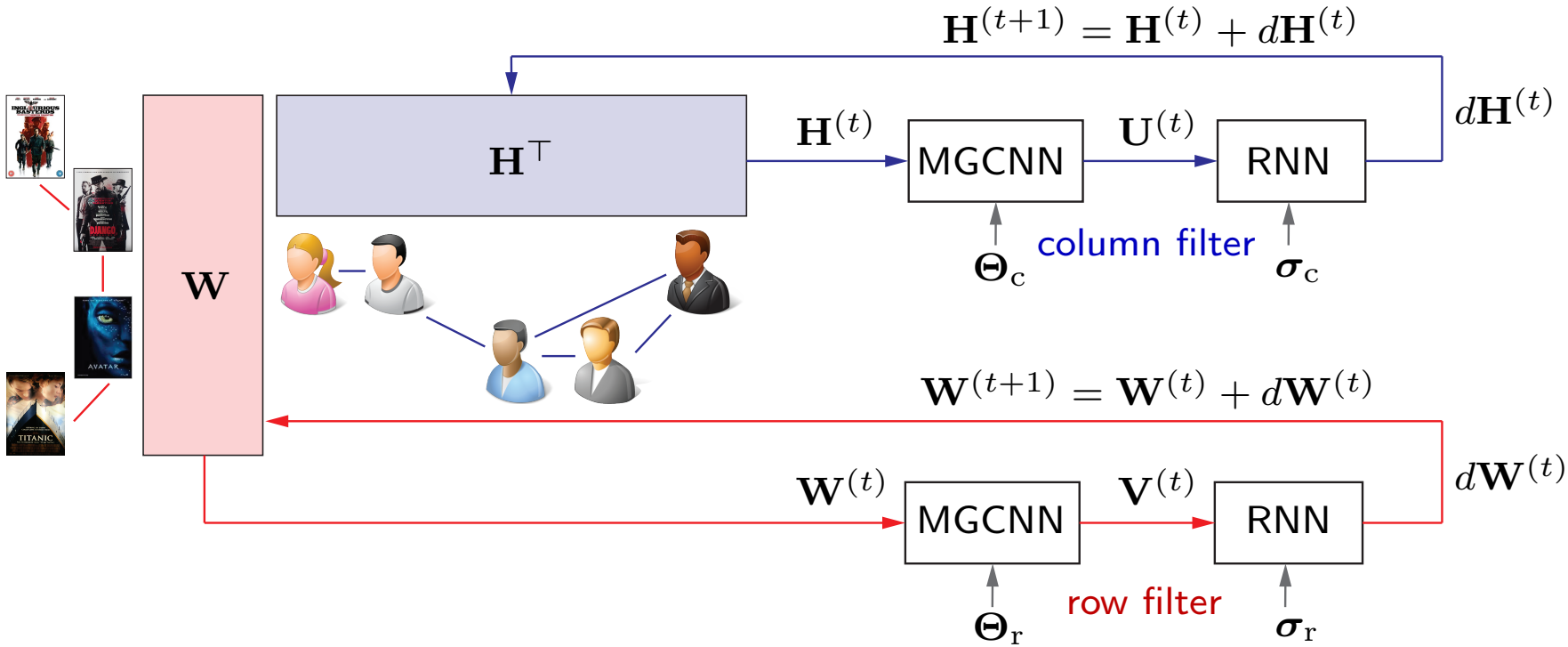


Recurrent multigraph CNN (RMCNN) architecture for matrix completion

$$\min_{\boldsymbol{\Theta},\boldsymbol{\sigma}} \|\mathbf{X}^{(T)}_{\boldsymbol{\Theta},\boldsymbol{\sigma}}\|^2_{\mathcal{G}_\mathrm{r}} + \|\mathbf{X}^{(T)}_{\boldsymbol{\Theta},\boldsymbol{\sigma}}\|^2_{\mathcal{G}_\mathrm{c}} + \frac{\mu}{2}\|\boldsymbol{\Omega} \circ (\mathbf{X}^{(T)}_{\boldsymbol{\Theta},\boldsymbol{\sigma}} - \mathbf{Y})\|^2_\mathrm{F}$$

Monti, Bresson, Bronstein 2017

Matrix completion with Recurrent Multi-Graph CNN



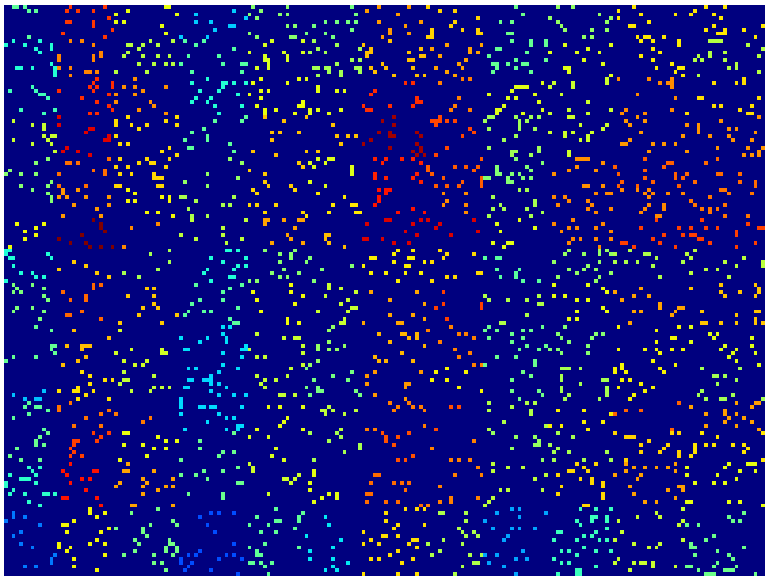
Separable recurrent multigraph CNN (sRMCNN) architecture for matrix completion in factorized form

$$\min_{\boldsymbol{\theta}_{\mathrm{r}},\boldsymbol{\theta}_{\mathrm{r}},\boldsymbol{\sigma}_{\mathrm{r}},\boldsymbol{\sigma}_{\mathrm{c}}} \|\mathbf{W}^{(T)}_{\boldsymbol{\theta}_{\mathrm{r}},\boldsymbol{\sigma}_{\mathrm{r}}}\|^2_{\mathcal{G}_{\mathrm{r}}} + \|\mathbf{H}^{(T)}_{\boldsymbol{\theta}_{\mathrm{c}},\boldsymbol{\sigma}_{\mathrm{c}}}\|^2_{\mathcal{G}_{\mathrm{c}}} + \frac{\mu}{2}\|\boldsymbol{\Omega} \circ (\mathbf{W}^{(T)}_{\boldsymbol{\theta}_{\mathrm{r}},\boldsymbol{\sigma}_{\mathrm{r}}}(\mathbf{H}^{(T)}_{\boldsymbol{\theta}_{\mathrm{c}},\boldsymbol{\sigma}_{\mathrm{c}}})^\top - \mathbf{Y})\|^2_{\mathrm{F}}$$

Monti, Bresson, Bronstein 2017

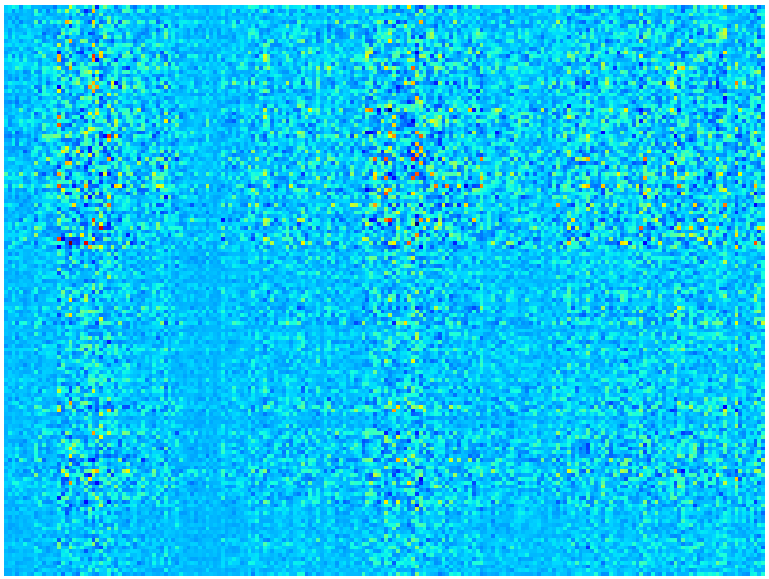
Incremental updates with RNN

Non-factorized (**RMGCNN**)



RMSE=2.26

Factorized (**sRMGCNN**)



RMSE=1.15

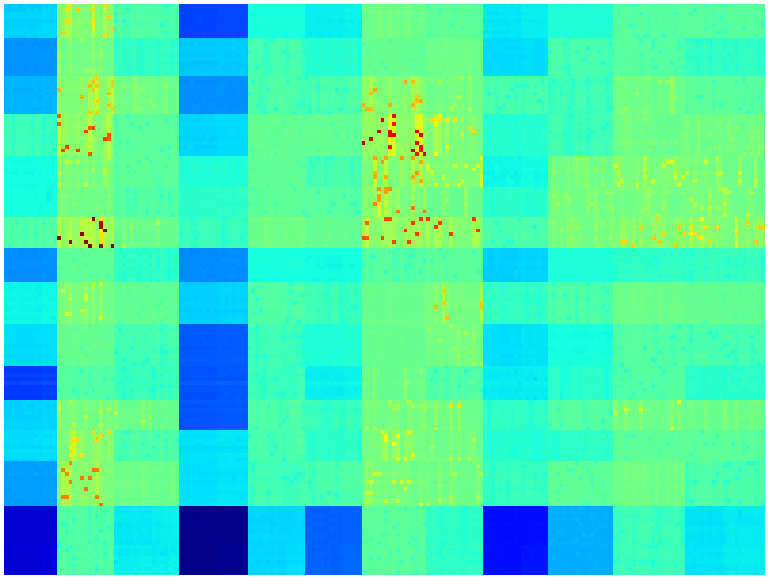
Matrix completion results on a synthetic dataset.

$$\mathbf{t = 0}$$

Monti, Bresson, Bronstein 2017

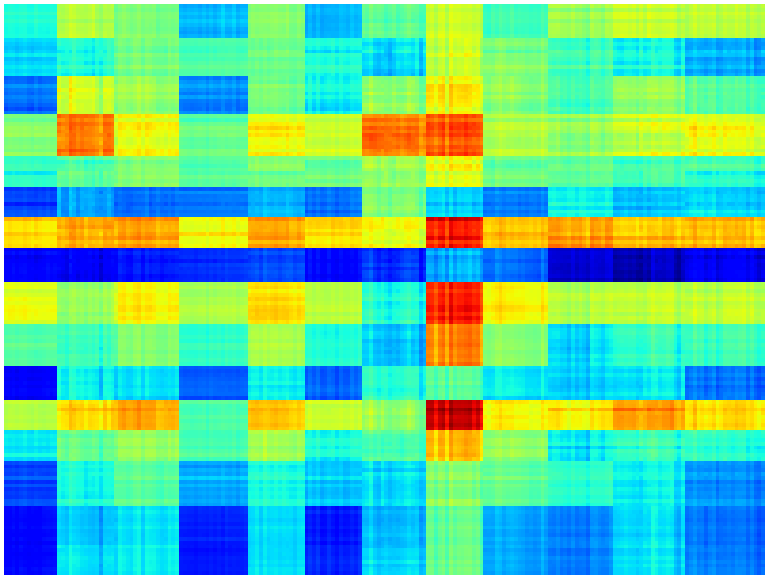
Incremental updates with RNN

Non-factorized (**RMGCNN**)



RMSE=0.52

Factorized (**sRMGCNN**)



RMSE=0.76

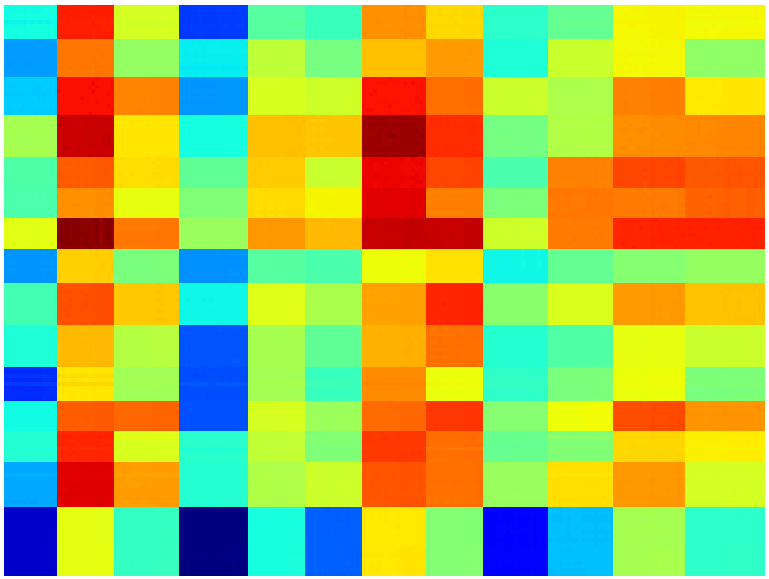
Matrix completion results on a synthetic dataset.

$$\mathbf{t = 5}$$

Monti, Bresson, Bronstein 2017

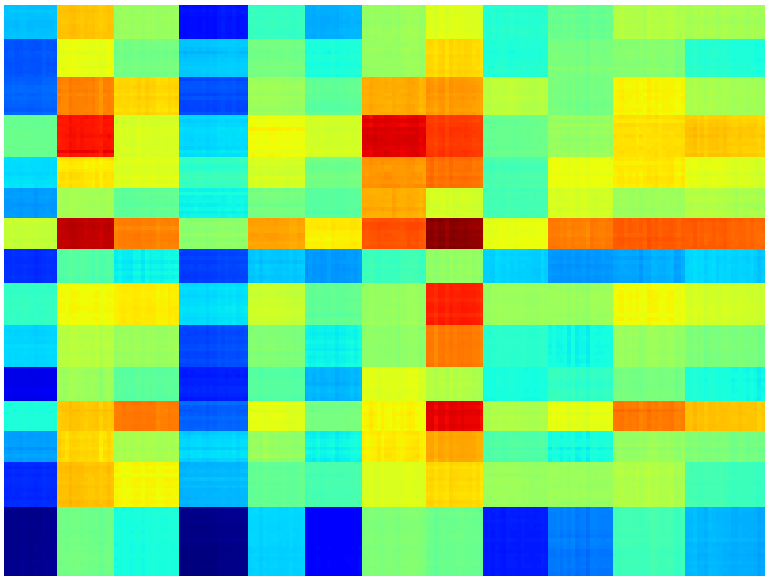
Incremental updates with RNN

Non-factorized (**RMGCNN**)



RMSE=0.38

Factorized (**sRMGCNN**)



RMSE=0.27

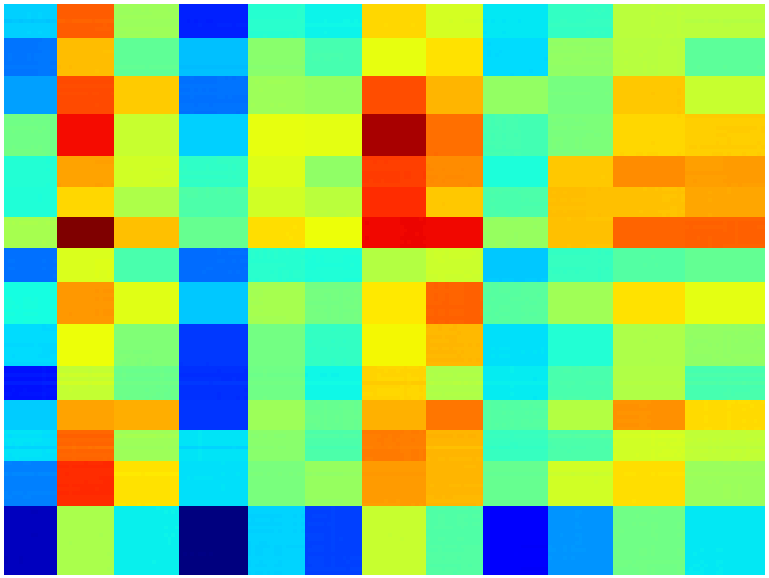
Matrix completion results on a synthetic dataset.

$$\mathbf{t = 8}$$

Monti, Bresson, Bronstein 2017

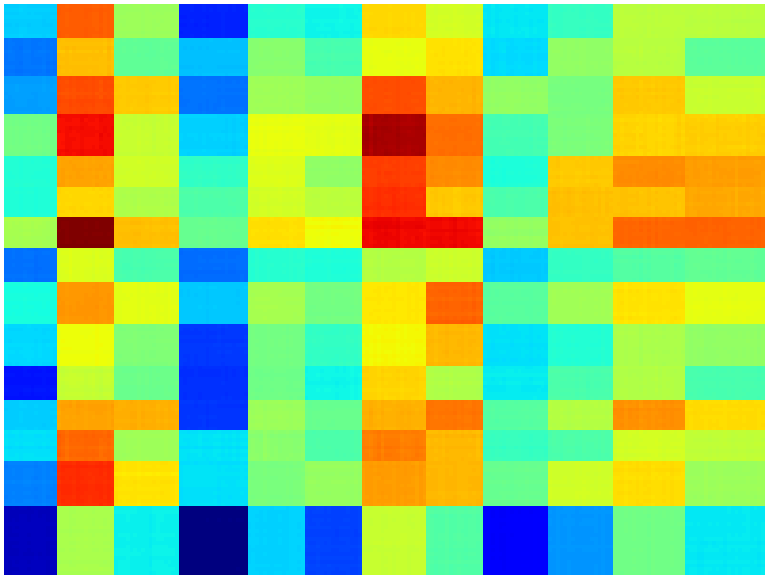
Incremental updates with RNN

Non-factorized (**RMGCNN**)



RMSE=0.01

Factorized (**sRMGCNN**)



RMSE=0.01

Matrix completion results on a synthetic dataset.

$$\mathbf{t = 10}$$

Monti, Bresson, Bronstein 2017

Matrix completion methods comparison

Method	**#Params**	**Complexity**	**RMSE**
GMC[1]	mn	mn	0.3693
GRALS[2]	$m+n$	$m+n$	0.0114
sRMGCNN[3]	$\mathbf{1}$	$\boldsymbol{m+n}$	**0.0106**
RMGCNN[3]	$\mathbf{1}$	$\boldsymbol{mn}$	**0.0053**

Comparison of geometric matrix completion methods on synthetic data using both users and movies graphs

Method	Params	Architecture	RMSE
$\text{MGCNN}_{\text{3layers}}$	$9K$	1MGC32, 32MGC10, 10MGC1	0.0116
$\text{MGCNN}_{\text{4layers}}$	$53K$	1MGC32, 32MGC32 $\times$ 2, 32MGC1	0.0073
$\text{MGCNN}_{\text{5layers}}$	$78K$	1MGC32, 32MGC32 $\times$ 3, 32MGC1	0.0074
$\text{MGCNN}_{\text{6layers}}$	$104K$	1MGC32, 32MGC32 $\times$ 4, 32MGC1	0.0064
RMGCNN[3]	$\mathbf{9K}$	1MGC32 + LSTM	**0.0053**

Reconstruction errors with multi-layer MGCNNs and the proposed solution. q'MGCq denotes a multi-graph convolution with q'/q input/output features.

Methods: [1]Kalofolias et al. 2014; [2]Rao et al. 2015; [3]Monti, Bresson, Bronstein 2017;

Matrix completion methods comparison

Method	**MovieLens**[1]	**Flixster**[2]	**Douban**[3]	**Yahoo**[4]
IMC[5]	1.653	–	–	–
GMC[6]	0.996	–	–	–
MC[7]	0.973	–	–	–
GRALS[8]	0.945	1.313/1.245	0.833	38.042
sRMGCNN (Cheby, r=4)[9]	**0.929**	**1.179/0.926**	**0.801**	**22.415**
sRMGCNN (Cheby, r=8)[9]	**0.925**	–	–	–
sRMGCNN (Cayley, r=4)[10]	**0.922**	–	–	–

Performance (RMS error) on several datasets. For Douban and YahooMusic, a single graph (of users and items respectively) was used. For Flixster, two settings are shown: users+items graphs / only users graph.

Data: [1]Miller et al. 2003; [2]Jamali, Ester 2010; [3]Ma et al. 2011; [4]Dror et al. 2012
Methods: [5]Jain, Dhillon 2013; [6]Kalofolias et al. 2014; [7]Candès, Recht 2012; [8]Rao et al. 2015; [9]Monti, Bresson, Bronstein 2017; [10]Levie et al. 2017

Conclusions

- We presented a new spectral approach with spectral zoom properties (CayleyNet).
- We introduced MGCNN, the first Multi-Graph Convolutional Neural Network.
- We showed how coupling MGCNN with a RNN a learnable diffusion process can be realized for reconstructing missing information.
- Our Geometric Deep Learning approach outperforms previous state of the art solutions on the matrix completion problem.

Thank You!