MATHEMATICS OF DEEP LEARNING

JOAN BRUNA , CIMS + CDS, NYU, SPRING'18

Lecture 5: Graph Neural Networks (cont'd), Unsupervised Learning with Geometric Priors

LECTURE 5/6 OVERVIEW

- Graph Neural Networks
  - Applications to Graph Inverse Problems.

- Unsupervised Learning with Geometric Priors
  - Mixture Models: Variational Autoencoders
  - Implicit Models: Flows and Generative Adversarial Networks
  - Wasserstein Models
  - Canonical and Microcanonical Models

- Open problems.

GRAPH NEURAL NETWORKS

[Scarselli et al.,'09], [Gori et al. '05]

- Given a signal $x \in \mathbb{R}^{V \times p}$, a Graph Neural Network (GNN) layer considers generators $[D, W]$ and trainable coefficients $\Theta = (\theta_1, \theta_2)$:

$$\tilde{x} = \rho\left(Dx\theta_1 + Wx\theta_2\right) \ .$$

$\theta_1, \theta_2 \in \mathbb{R}^{p \times \tilde{p}}$.

- Flexible model: does not require fixed input graphs.
- Initial version was inspired from the Message-Passing algorithm. Fixed point of a trainable, non-linear diffusion.
- Modernized in [Li et al.'15], [Duvenaud et al.'15], [Subkhaatar et al.'16],
- Authors also explored other forms of nonlinearity, e.g. *gating*.
- Similarly as in CNNs, we can also consider pooling layers, (*provided we have a graph coarsening scheme)*.

LAPLACIAN INTERPRETATION

- Since we are learning a linear combination of $A(G)$ and $D(G)$, we can reparametrize the generator in terms of the *Graph Laplacian:*

$$\Delta(G) = D(G) - A(G)$$

- If we consider generators of the form $[1, \Delta, \Delta^2, \dots]$, the resulting GNN layer is expressed as a polynomial in Δ:

$$\tilde{x} = \rho\left(\theta(\Delta)x\right) \ , \ \theta(\Delta) = \sum_{s=0} \theta_s \Delta^s \ .$$

- In Spectral Networks [B. et al'14], we train directly on the spectrum of the Laplacian:

$$\tilde{x} = \rho\left(V^T \mathrm{diag}(\alpha) V x\right) \ , \ \alpha = \mathcal{K}(\theta) \ , \qquad \mathcal{K} : \text{spline kernel}$$

  - Computationally expensive and unstable to deformations for varying graph.

EFFICIENT SPECTRAL NETWORKS

- These issues were addressed in [Defferrard et al.'16] by sticking with the polynomial representation $\theta(\Delta)$instead.
  - Since the spectrum of Δin finite dimensions is bounded, the optimal polynomial basis in the interval is given by Chebyshev polynomials:

GNN with generators $[F_j(\tilde{\Delta})]_{j\leq J}$, $\tilde{\Delta} = \frac{2}{\|\Delta\|}\Delta - \mathbf{1}$.

- [Kipf & Welling,'17] Further simplified the construction using only $J = 1$ and the normalized Laplacian

$$\bar{\Delta} = \mathbf{1} - D^{-1/2} W D^{-1/2} \ .$$

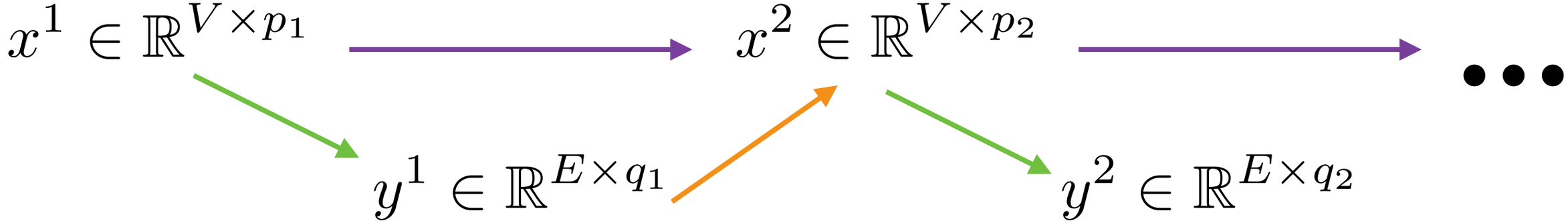
- Better frequency localization can be obtaining by replacing (real) Chebyshev polynomial with (complex) Cayley filters [Monti et al.'17].

EXTENSIONS/LIMITATIONS

- As opposed to Euclidean domains, in general graphs we only have an isotropic high-pass filter (Δ), but no oriented filters.

EXTENSIONS/LIMITATIONS

- As opposed to Euclidean domains, in general graphs we only have an isotropic high-pass filter (Δ), but no oriented filters.
- Inspired by Message-Passing algorithms, we can generalize GNNs to alternate between vertex and edge representations:



$$y^1(e) = \psi_\theta(x^1(i), x^1(j)) \ , \ e = (i,j) \in E \ .$$
$$x^2(i) = \phi_\theta(\{y^1(e)\}_{i \in e}) \ , i \in V \ .$$

- Used for example in [Battaglia et al.'16] for N-body prediction dynamics and [Gilmer et al.'17] for quantum chemistry.

SURFACE REPRESENTATIONS

[joint work with I. Kostrikov, D.Panozzo, D.Zorin (NYU)]

- In the particular case where G represents a 3D surface, we have a *mesh* representation:

$$M = (V, E, F) \ , \ \ F = \{(i, j, k)\} \text{ triangulation}$$

credit: jonathanpuckey

SURFACE REPRESENTATIONS

[joint work with I. Kostrikov, D.Panozzo, D.Zorin (NYU)]

- In the particular case where $\mathcal{G}$ represents a 3D surface, we have a *mesh* representation:

$$M = (V, E, F) \ , \ F = \{(i, j, k)\} \text{ triangulation}$$

credit: jonathanpuckey

- In that case, we can compute a "proper" square root of the Laplacian, the *Dirac* operator:

$$\Delta = D^* D \ , \ D \in \mathbb{H}^{V \times F}$$

  - Defined over quaternion space.
  - Captures principal curvature directions (ie orientation).

SURFACE REPRESENTATIONS

- In the particular case where G represents a 3D surface, we have a *mesh* representation:

$$M = (V, E, F) \ , \ F = \{(i, j, k)\} \text{ triangulation}$$

- In that case, we can compute a "proper" square root of the Laplacian, the *Dirac* operator:

$$\Delta = D^* D \ , \ D \in \mathbb{H}^{V \times F}$$

  - Defined over quaternion space.

credit: jonathanpuckey

LAPLACE NETWORK STABILITY

- Stable graph generators result in stable GNN representations:

Theorem: [B, K, P, Z'17] Let $G = (V, E)$ and suppose $V \subset \Omega \in \mathbb{R}^d$. Let $\Phi(x; \Delta)$ be a R-layer Laplace GNN with generators $\{I, \Delta\}$, and τ a deformation field on Ω. Then

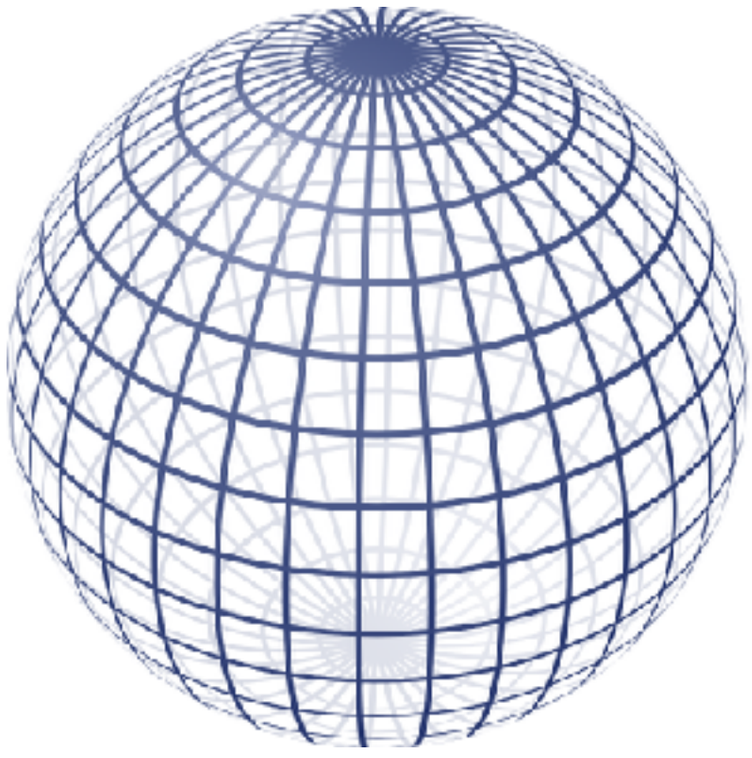
1. $\|\Phi(x; \Delta) - \Phi(x'; \Delta)\| \leq C(\Theta)\|x - x'\|^{h(\beta)}$,
2. $\|\Phi(x; \Delta) - \Phi(x; \tau(\Delta))\| \leq C'(\Theta)\|\nabla\tau\|^{h(\beta)}$,

where $h(\beta) = \prod_{r \leq R} \frac{\beta_r - 1}{\beta_r - 1/2}$ measures smoothness (Sobolev) of feature maps.

- In Euclidean graphs, the Laplacian is geometrically stable.
- *Caveat:* We currently require explicit smoothness decay of feature maps.
- *Future work:* Extension to intrinsic deformations.

NON-EUCLIDEAN BUT REGULAR DOMAINS

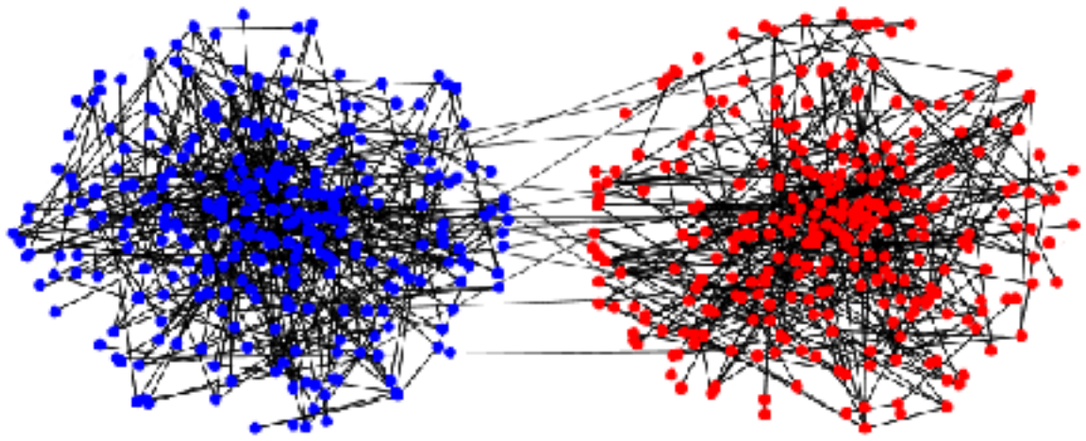
- One important particular case is when the domain Ω is regular, e.g the sphere:



- The set of transformations that leave the sphere S^2 invariant forms a Lie Group, $\mathrm{SO}(3)$.
- We can extend CNNs to such settings by replacing planar convolutions with group convolutions [Cohen & Welling]

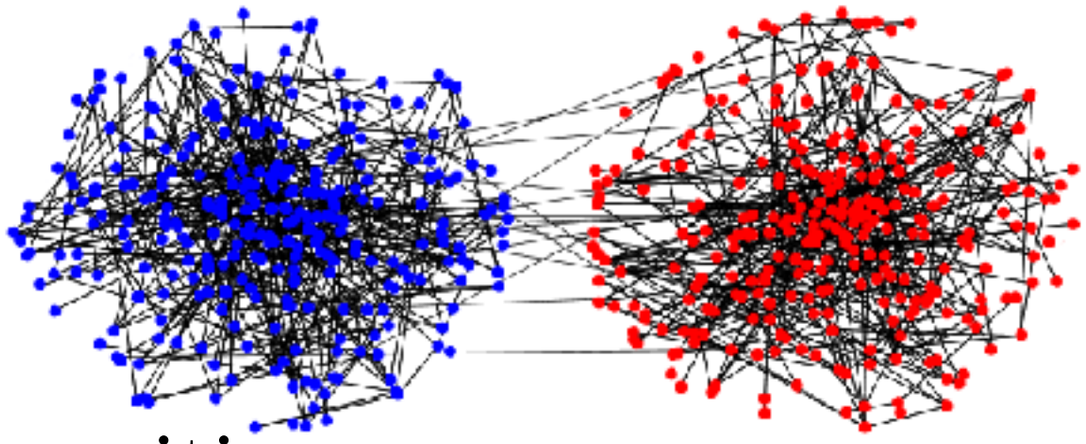
INVERSE PROBLEMS ON GRAPHS

- Consider the problem of inferring communities within a network:

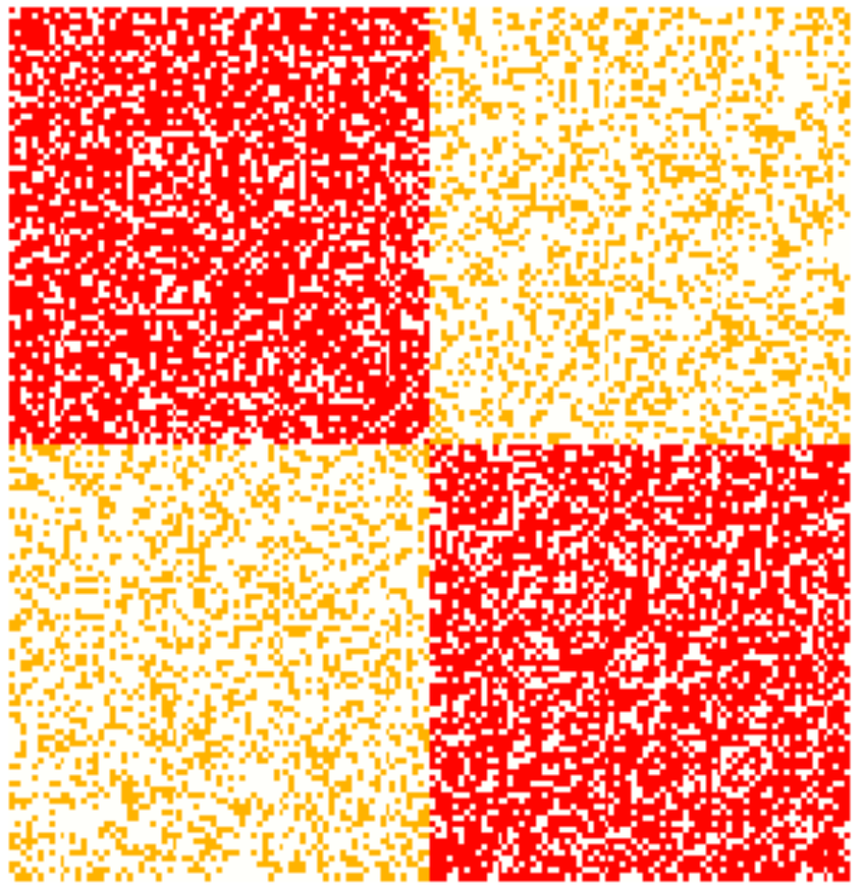


INVERSE PROBLEMS ON GRAPHS

- Consider the problem of inferring communities within a network:

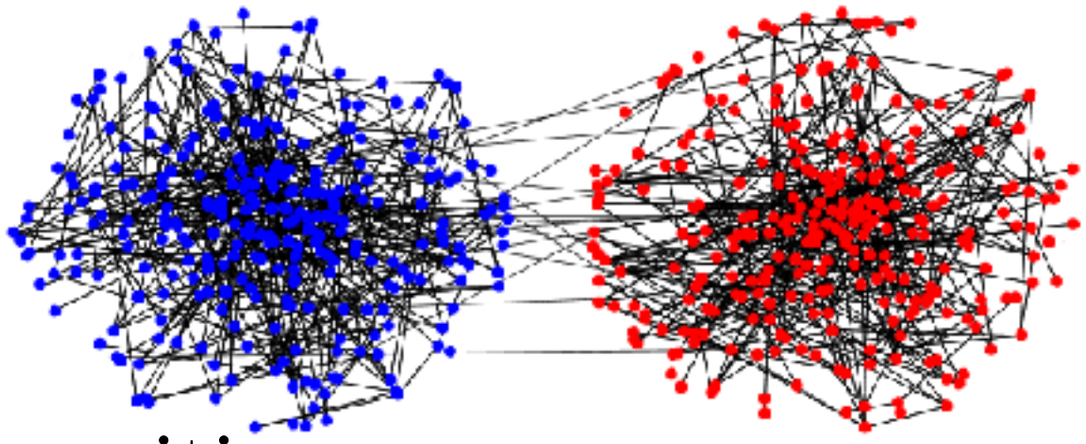


Adjacency matrix associative 2 communities

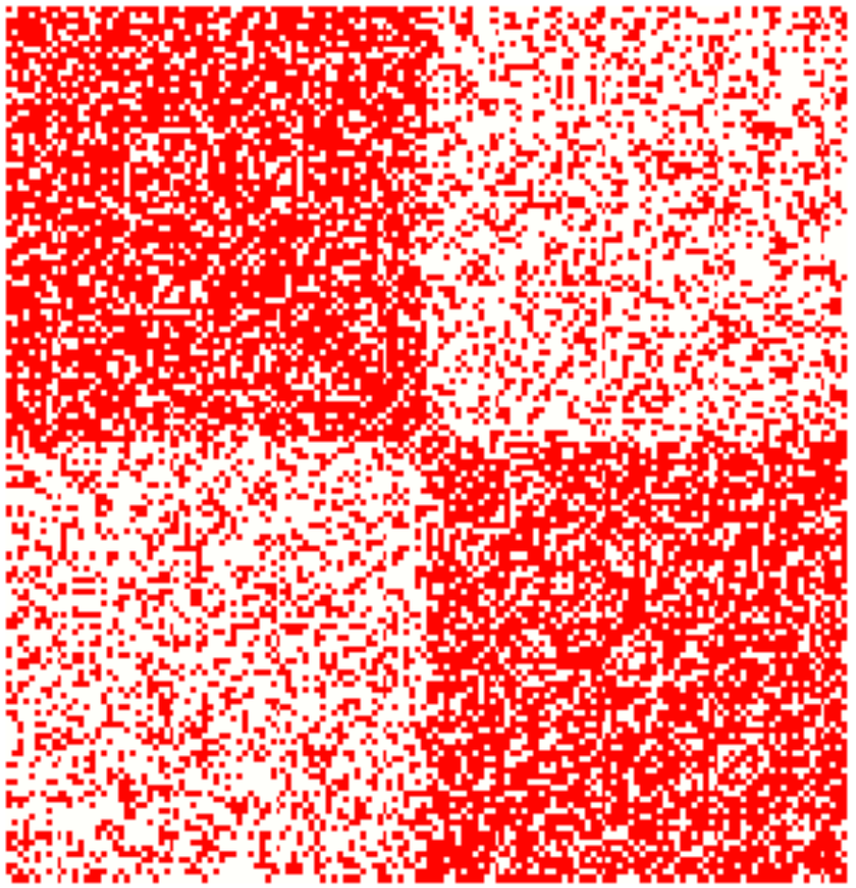


INVERSE PROBLEMS ON GRAPHS

- Consider the problem of inferring communities within a network:

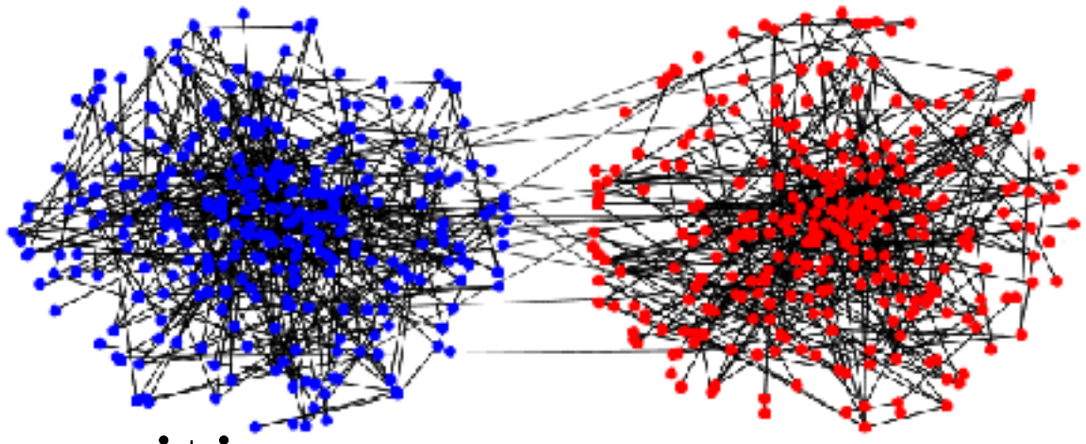


Adjacency matrix associative 2 communities

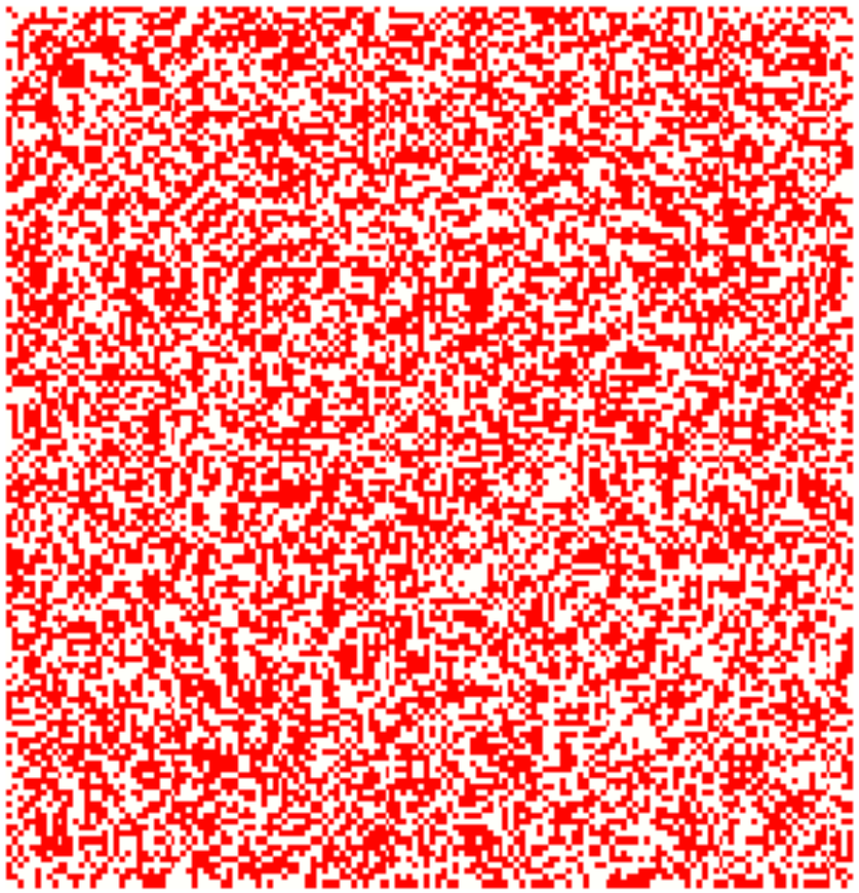


INVERSE PROBLEMS ON GRAPHS

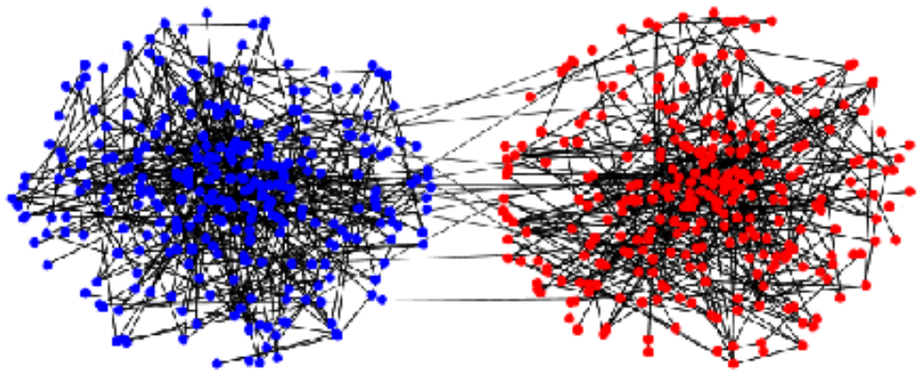
- Consider the problem of inferring communities within a network:



Adjacency matrix associative 2 communities



INVERSE PROBLEMS ON GRAPHS

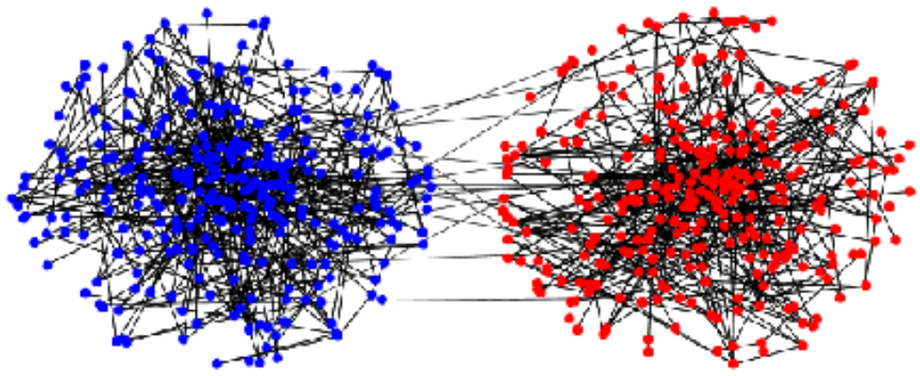


- Community Detection in graphs.
  - Studied in the Stochastic Block Model.
  - Hardness of estimation is controlled by a Signal-to-Noise Ratio:

$$\mathrm{SNR} = \frac{(a-b)^2}{k(a+(k-1)b)}$$

a: inner connection probability.
b: outer connection probability.

INVERSE PROBLEMS ON GRAPHS



- Community Detection in graphs.
  - Studied in the Stochastic Block Model.
  - Hardness of estimation is controlled by a Signal-to-Noise Ratio:

$$\mathrm{SNR} = \frac{(a-b)^2}{k(a+(k-1)b)}$$

a: inner connection probability.
b: outer connection probability.

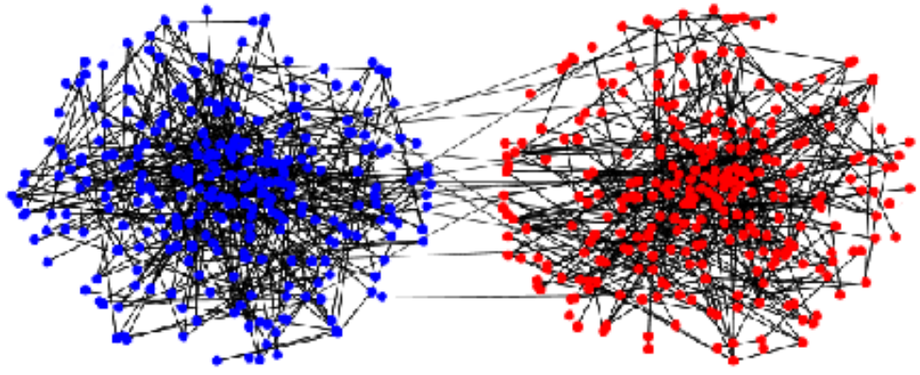
- Two major algorithmic frameworks:
  - Graph conductance/min-cut approach, leading to spectral clustering algorithms.

$$\min_{y_i=\pm 1; \bar{y}=0} \; y^T \mathcal{A}(G) y \; .$$

  - Probabilistic Graphical Models, leading to Belief Propagation.

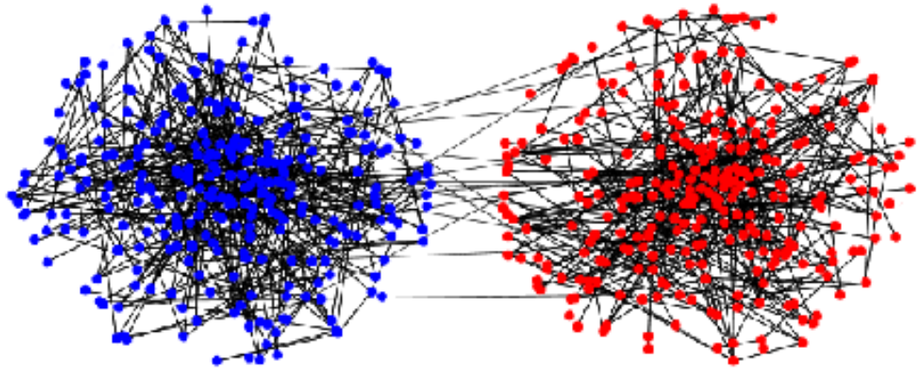
$$p(y|G) \propto \prod_{(i,j)\in E} \varphi_{(i,j)}(y_i, y_j) \prod_{v \in V} \psi_i(y_i)$$

INVERSE PROBLEMS ON GRAPHS



- Community Detection in graphs.
  - Studied in the Stochastic Block Model.
  - Hardness of estimation is controlled by a Signal-to-Noise Ratio.
- Recent research program has unified both approaches using tools from statistical physics, and identified computational and information theoretic thresholds:
  - When is the detection statistically possible?
  - When is the detection feasible with polynomial-time algorithms?

INVERSE PROBLEMS ON GRAPHS



- Community Detection in graphs.
  - Studied in the Stochastic Block Model.
  - Hardness of estimation is controlled by a Signal-to-Noise Ratio.
- Recent research program has unified both approaches using tools from statistical physics, and identified computational and information theoretic thresholds:
  - When is the detection statistically possible?
  - When is the detection feasible with polynomial-time algorithms?

- Q: Can we *learn* those algorithms from the data using graph neural networks? reaching detection thresholds?

DATA-DRIVEN COMMUNITY DETECTION

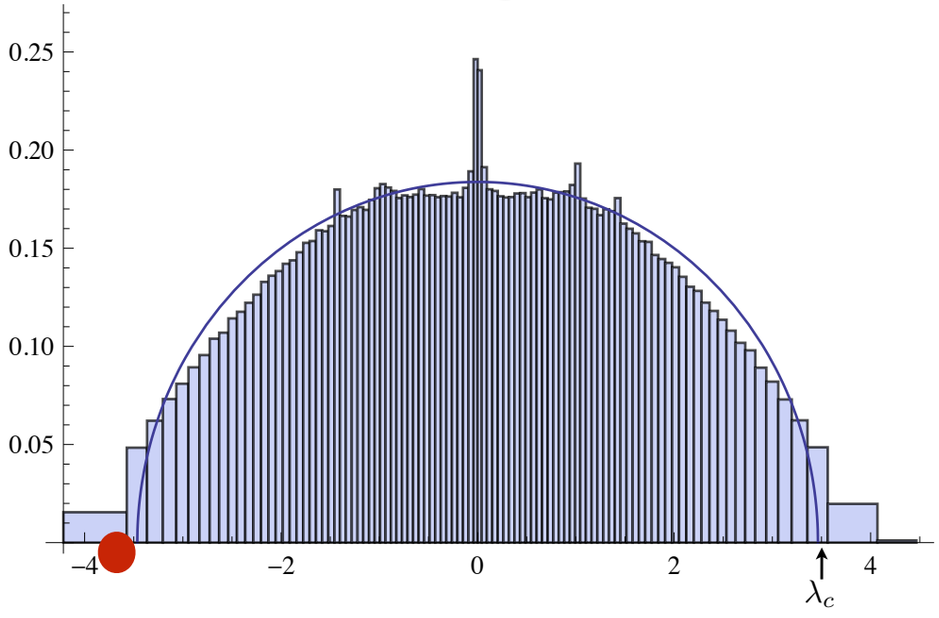
[joint work with Lisha Li (UC Berkeley)]

➤ $\mathcal{A}(G)$: linear operator defined on G, eg Laplacian $\Delta = D - A$.

➤ Spectral Clustering estimators (2-community case):

$$\hat{y} = \text{sign}\left(\text{Fiedler}(\mathcal{A}(G))\right) \ ,$$

Fiedler(M): eigenvector corresponding to 2nd smallest eigenvalue



➤ Iterative algorithm: projected power iterations on shifted $\mathcal{A}(G)$:

$$M = \|\mathcal{A}(G)\|\mathbf{1} - \mathcal{A}(G)$$

DATA-DRIVEN COMMUNITY DETECTION

[joint work with Lisha Li (UC Berkeley)]

➤ We consider a GNN generated by operators $\{\mathbf{1}, A, D\}$:

$$\tilde{x} = \rho\left(\theta_1 x + \theta_2 D x + \theta_3 A x\right) \ .$$

➤They generate the so-called *Bethe Hessian:*

$$BH(r) = (r^2 - 1)\mathbf{1} - rA + D$$

❖ Second-order approximation of Bethe Free energy at critical points of BP.

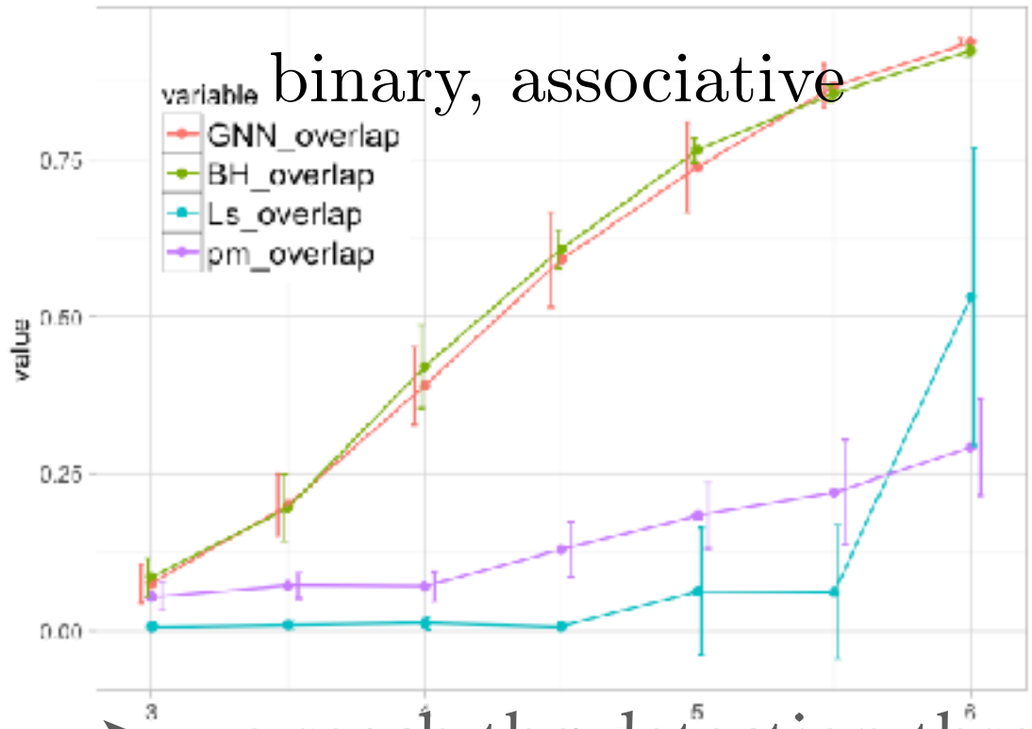
❖ In that case, Laplacian generator does not work: its spectrum is dominated by few nodes with dominant degree.

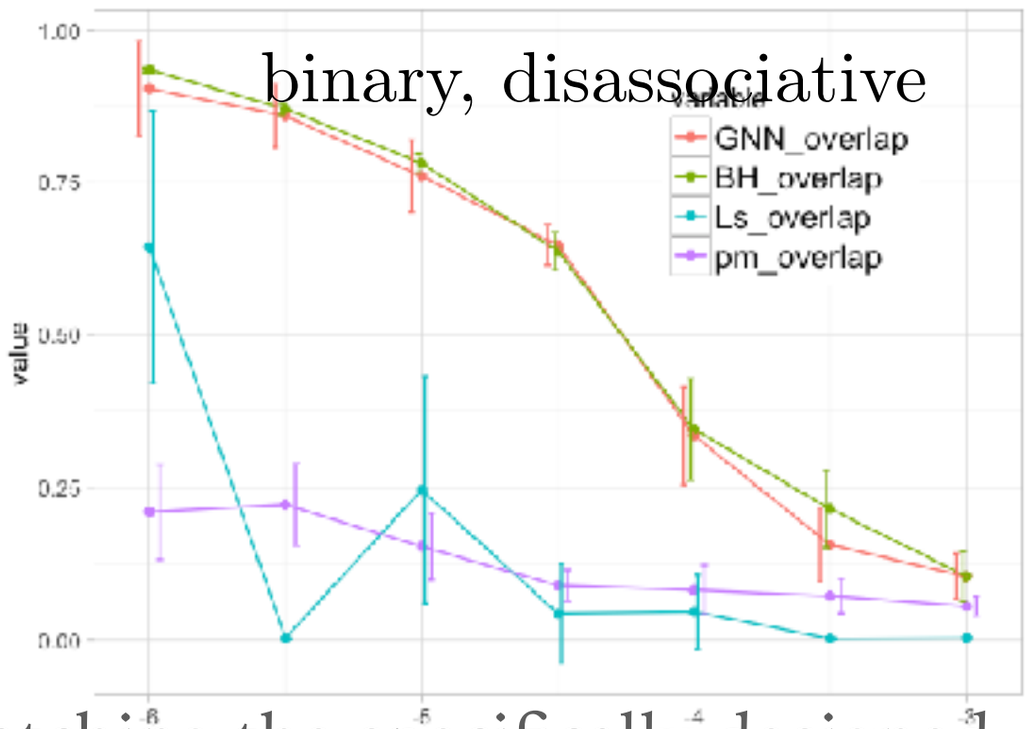
➤ We train it by back propagation using a loss that is globally invariant to label permutations.

REACHING DETECTION THRESHOLD ON SBM

[joint work with Lisha Li (UC Berkeley)]

- Stochastic Block Model Results:





  - we reach the detection threshold, matching the specifically designed spectral method.

- Real-world community detection results on SNAP data [Leskovec et al]

Table 1: Snap Dataset Performance Comparison between GNN and AGM

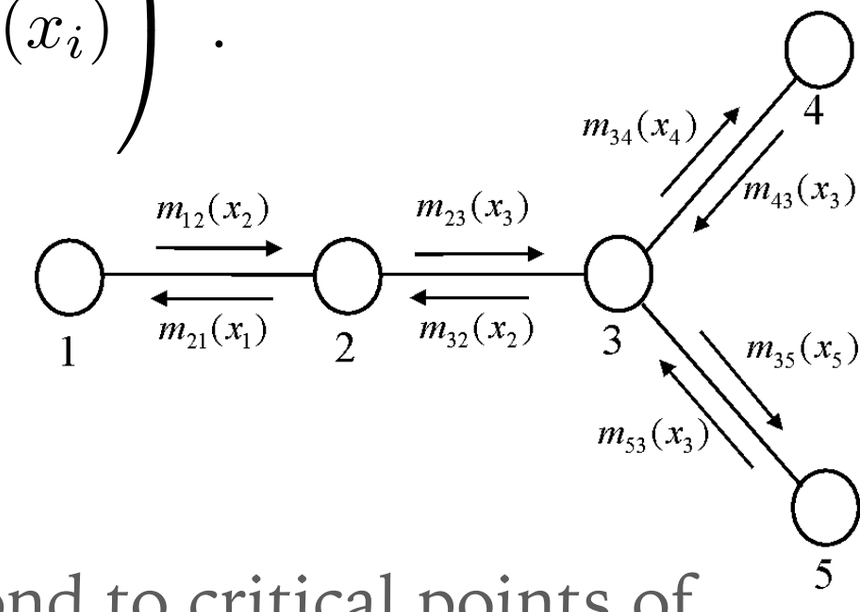
	Subgraph Instances			Overlap Comparison	
Dataset	(train/test)	Avg Vertices	Avg Edges	GNN	AGMFit
Amazon	315 / 35	60	346	$\mathbf{0.74 \pm 0.13}$	$\mathbf{0.76 \pm 0.08}$
DBLP	2831 / 510	26	164	$\mathbf{0.78 \pm 0.03}$	0.64 ± 0.01
Youtube	48402 / 7794	61	274	$\mathbf{0.9 \pm 0.02}$	0.57 ± 0.01

BELIEF PROPAGATION

- For small number of communities, the IT detection threshold is provably matched by (loopy) BP.
- BP performs message-passing updates of the form

$$m_{ij}(x_j) \leftarrow \sum_{x_i} \left(\tilde{\phi}_i(x_i; y) \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} m_{ki}(x_i) \right) .$$

$$b_j(x_j) = \frac{1}{Z_j} \tilde{\phi}_j(x_j; y) \prod_{i \in N(j)} m_{ij}(x_j) .$$



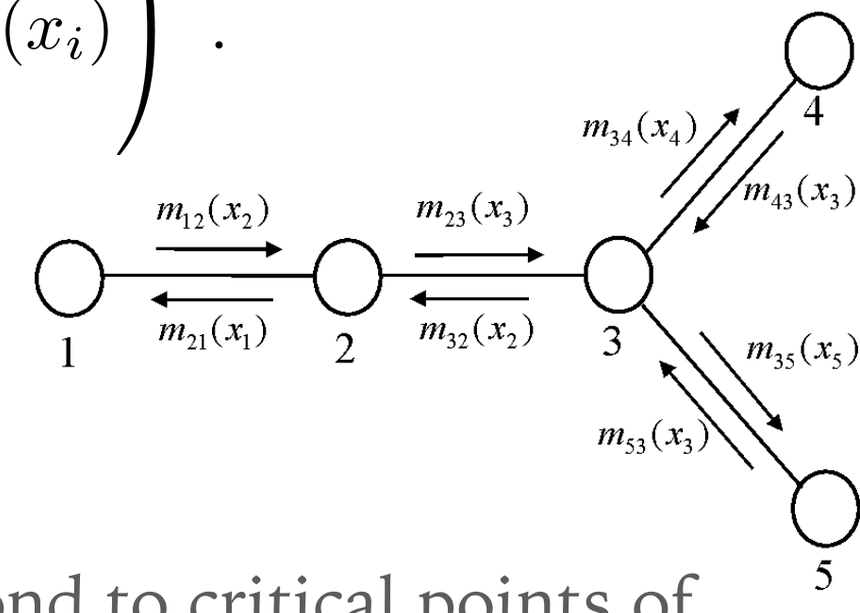
  - exact inference on simply connected graphs.
  - on general graphs, fixed points of BP correspond to critical points of the Bethe Free Energy.
- Messages are defined and propagated over edges of G, and account for non-backtracking paths.

BELIEF PROPAGATION

- For small number of communities, the IT detection threshold is provably matched by (loopy) BP.
- BP performs message-passing updates of the form

$$m_{ij}(x_j) \leftarrow \sum_{x_i} \left(\tilde{\phi}_i(x_i; y) \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} m_{ki}(x_i) \right) .$$

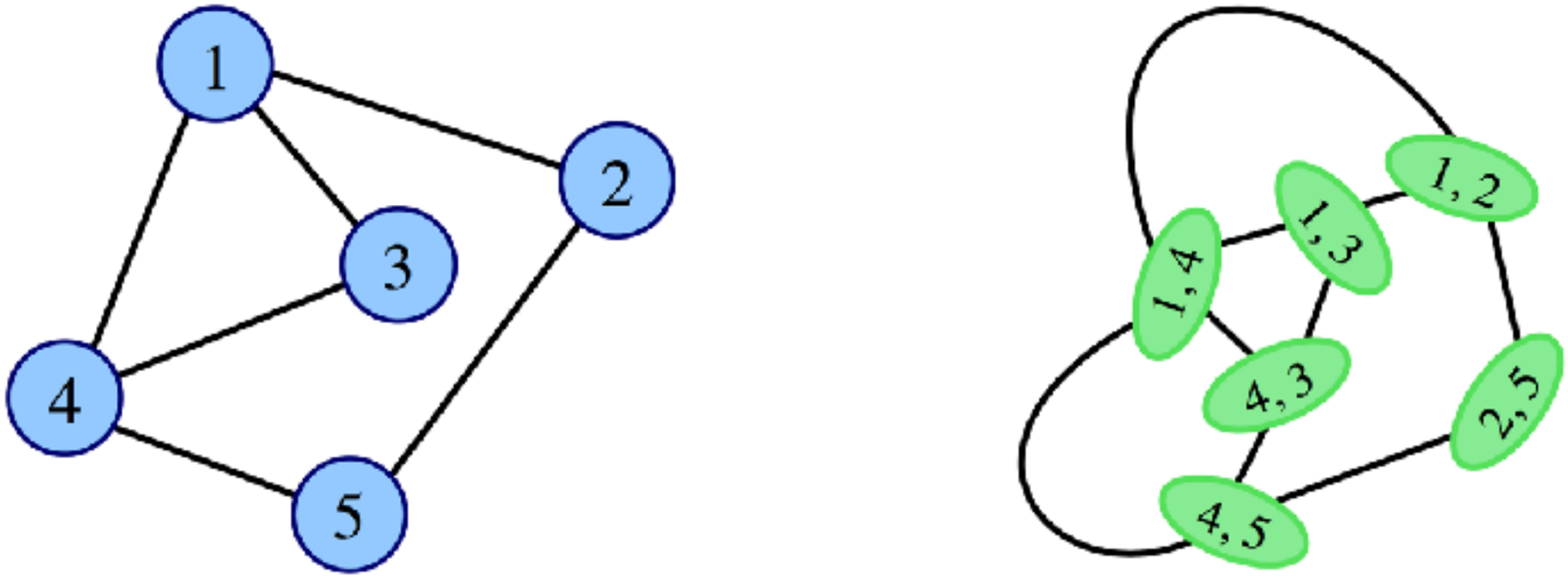
$$b_j(x_j) = \frac{1}{Z_j} \tilde{\phi}_j(x_j; y) \prod_{i \in N(j)} m_{ij}(x_j) .$$



  - exact inference on simply connected graphs.
  - on general graphs, fixed points of BP correspond to critical points of the Bethe Free Energy.
- Messages are defined and propagated over edges of G, and account for non-backtracking paths. GNN version?

GRAPH NEURAL NETWORKS ON GRAPH HIERARCHIES

- The *line* graph of $G = (V, E)$is a new graph $L(G) = (V', E')$ that models the adjacency of the edges: $V' \cong E$



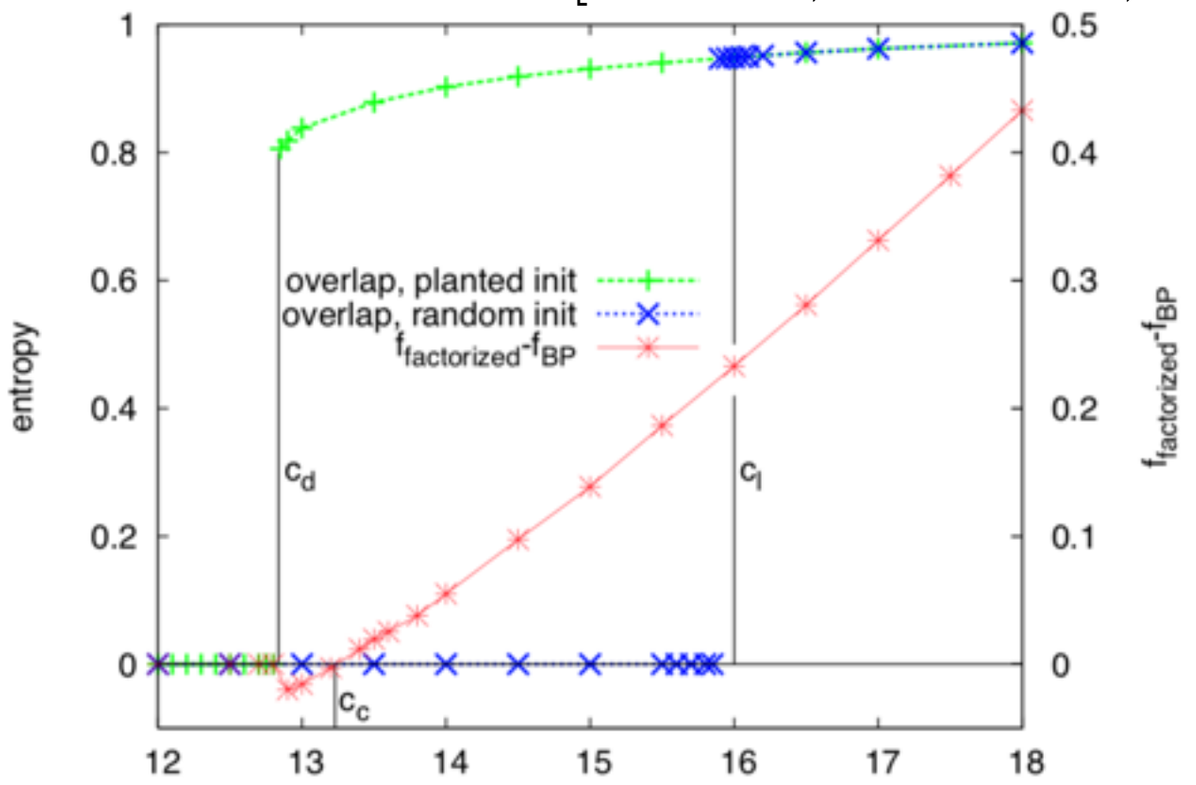
(source:wikipedia)

- We *augment* the GNN with analogous operations on $L(G)$.
- Related to Covariant Compositional Networks [Kondor et al'18], and neural message-passing [Gilmer et al.'17].

COMPUTATIONAL-TO-STATISTICAL GAPS

- When # of communities > 4, there is a gap between the information theoretical threshold and current known polynomial-time algorithms:

[Decelle, Krzakala, Moore, Zdeborova,'13]



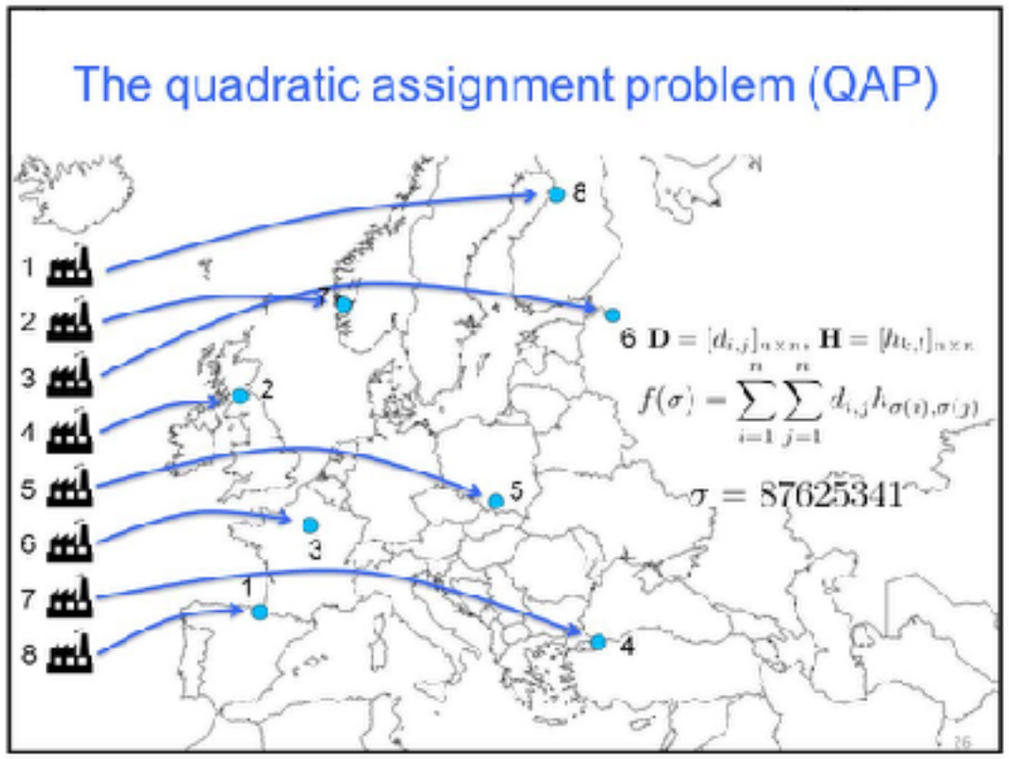
- Preliminary results for 5 communities, $\overline{\deg} = 14.5$:

$N = 10^3$	G	$\{G, L(G)\}$	BP
Overlap	29.5 ± 0.5	30.1 ± 0.5	30.4 ± 3

[with S. Villar (NYU), A. Nowak (NYU) and A. Bandeira (NYU)]

QUADRATIC ASSIGNMENT PROBLEM

- Find an assignment that optimizes the transportation cost between two graphs:



$$\min_{X\in\Pi_n} \ \mathrm{Tr}(A_1 X A_2 X^T) \ .$$

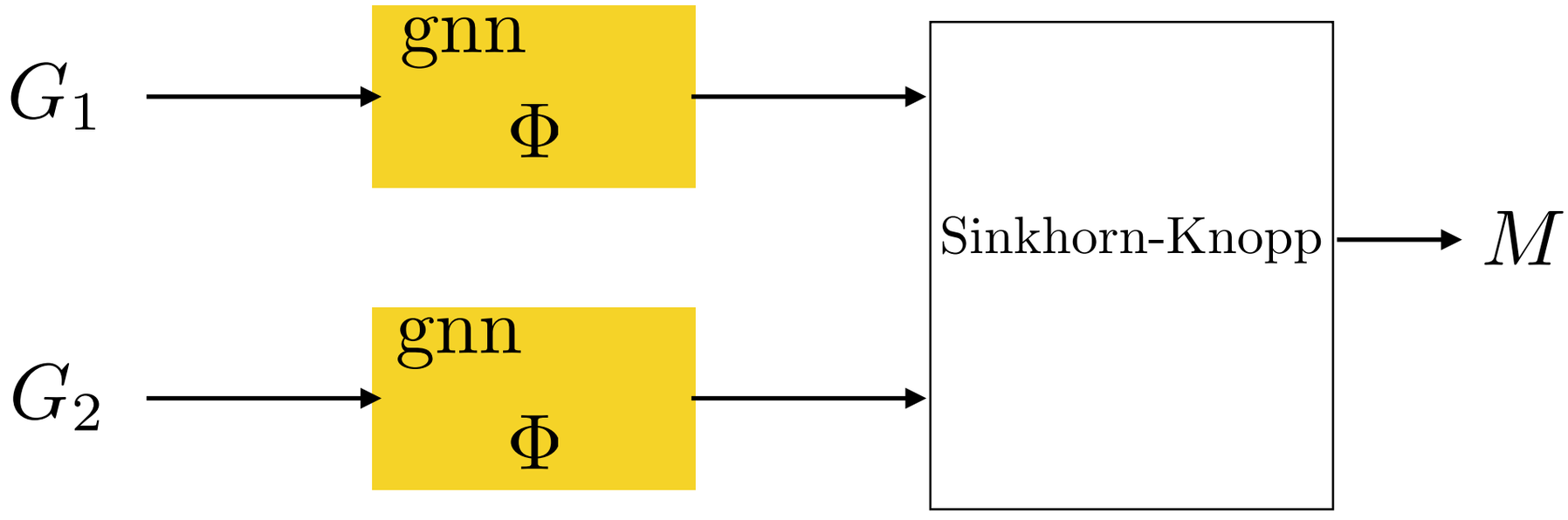
Π_n: space of $n \times n$ permutation matrices.

- NP-hard
- Contains the TSP as a particular instance.
- Relaxations using SDP and Spectral Approaches.

[with S. Villar (NYU), A. Nowak (NYU) and A. Bandeira (NYU)]

QUADRATIC ASSIGNMENT PROBLEM

- We learn approximate solutions using siamese graph neural networks:



$$M = \text{softmax}[\Phi(G_1)\Phi(G_2)^T)]$$

- We train the model to predict the correct permutation matrix on a dataset of *planted* solutions:

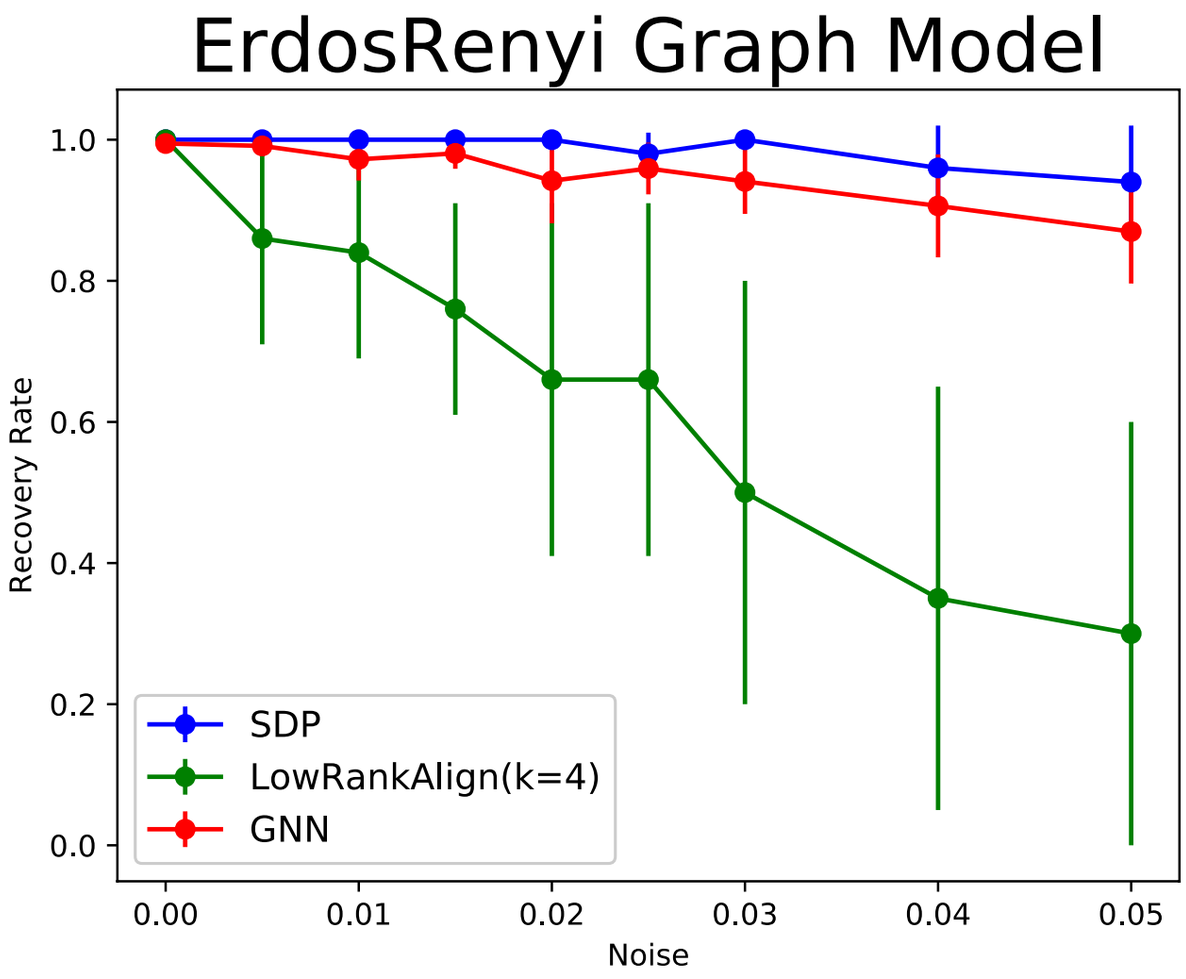
$$G_1 = PG_2 + N \qquad N \sim \text{Erdos-Renyi}$$

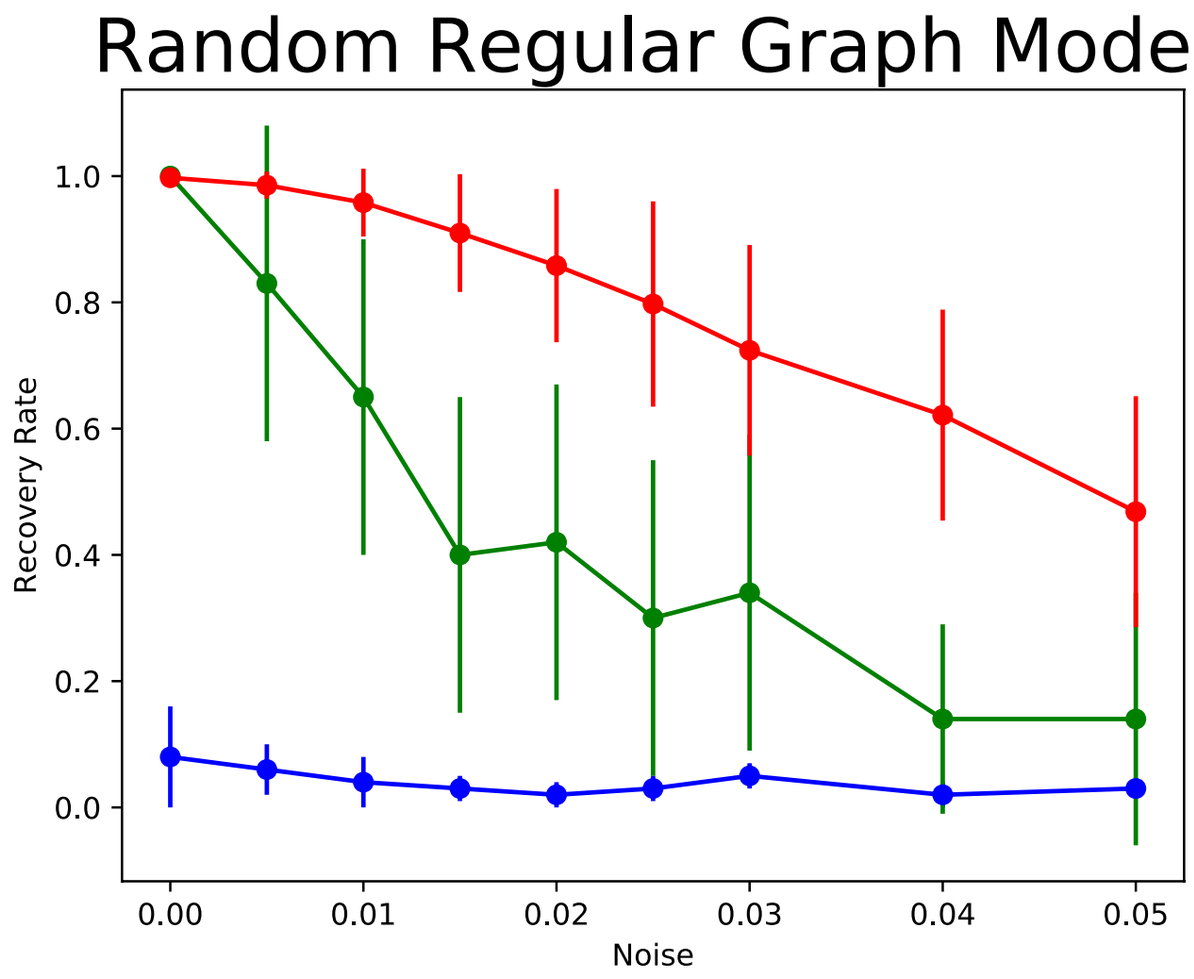
$G_2 \sim$ Erdos-Renyi

$G_2 \sim$ Random Regular

[with S. Villar (NYU), A. Nowak (NYU) and A. Bandeira (NYU)]

QUADRATIC ASSIGNMENT PROBLEM





- Our model runs in $o(n^2)$, LowRankAlign is $o(n^3)$ SDP in $o(n^4)$
- *Current*: What is the model learning? Link to *friendly* Graphs.
- *Current*: Applications to Shape Correspondence, Unaligned language translation

GIVENS FACTORIZATION OF UNITARY MATRICES

[with D. Folque (NYU)]

- Suppose we have a unitary matrix U (e.g. an eigenbasis) that we want to use extensively.
- Complexity of Matrix-vector multiplication: $\Theta(n^2)$ [Winograd]
- But structure on U can yield massive gains: FFT $\Theta(n \log n)$ [Tukey]

GIVENS FACTORIZATION OF UNITARY MATRICES

[with D. Folque (NYU)]

- Suppose we have a unitary matrix U (e.g. an eigenbasis) that we want to use extensively.
- Complexity of Matrix-vector multiplication: $\Theta(n^2)$ [Winograd]
- But structure on U can yield massive gains: FFT $\Theta(n \log n)$ [Tukey]
- General case?

GIVENS FACTORIZATION OF UNITARY MATRICES

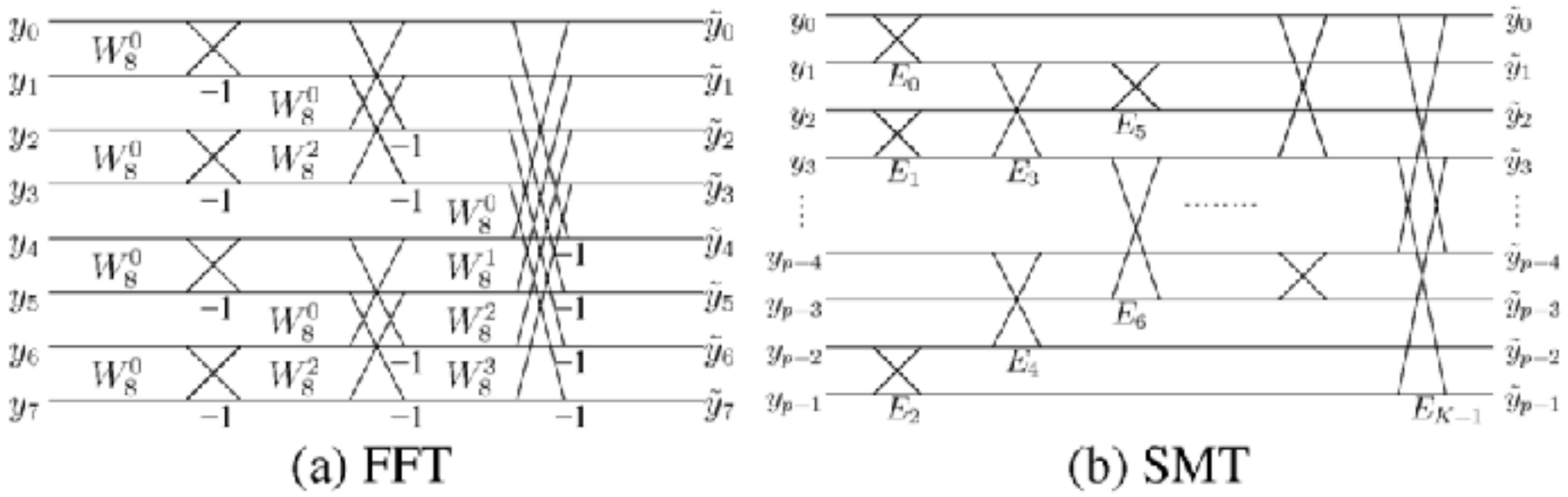
[with D. Folque (NYU)]

- We consider *Sparse Matrix Transforms* given by Givens plane rotations:

$$U = \prod_{k=1}^{K} \mathcal{O}(i_k, j_k, \alpha_k)$$

$\mathcal{O}(i_k, j_k, \alpha_k)$: Rotation of α_k in the plane $\{i_k, j_k\}$.

- $K = \dfrac{n(n-1)}{2}$ sufficient for any U $\quad \mathrm{FFT} : K = n \log n$
- NP-complete, non-commutative manifold-optimization problem.



- Use GNN model to *learn* such factors.

GIVENS FACTORIZATION OF UNITARY MATRICES

[with D. Folque (NYU)]

- We consider a simple inverse-problem setup with planted solution:

$$\left\{ U^{(l)} := \prod_{k=1}^{K} \mathcal{O}(i_k^{(l)}, j_k^{(l)}, \alpha_k^{(l)}) \right\}_{l \leq L}$$

input *labels*

- Model: a GNN on the fully-connected graph, where we learn both edge and node features.
  - Uses multiset loss to account for partial permutation invariance.
- Ongoing numerical experiments.

UNSUPERVISED LEARNING UNDER GEOMETRIC PRIORS

UNSUPERVISED LEARNING AS DATA-GENERATING MODELING

- We view the data as a sample from an underlying (and unknown) probability distribution Q defined over a Polish space $\mathcal{X}$ (complete and separable, whose topology comes from a distance function).
  - We denote by $\mathcal{P}_{\mathcal{X}}$ the space of probability measures μ defined on $(\mathcal{X}, \mathcal{U})$, $\mathcal{U}$: Borel σ-algebra generated by open sets of $\mathcal{X}$.

[Bottou et al.17]

UNSUPERVISED LEARNING AS DATA-GENERATING MODELING

- We view the data as a sample from an underlying (and unknown) probability distribution Q defined over a Polish space $\mathcal{X}$ (complete and separable, whose topology comes from a distance function).
  - We denote by $\mathcal{P}_{\mathcal{X}}$ the space of probability measures μ defined on $(\mathcal{X}, \mathcal{U})$, $\mathcal{U}$: Borel σ-algebra generated by open sets of $\mathcal{X}$.
- We need a measure to compare elements of $\mathcal{P}_{\mathcal{X}}$:

$$(P, Q) \mapsto D(P, Q) \in [0, \infty)$$

  - D can be a distance, but also a *pseudo-distance* (does not satisfy separation) or a *divergence* (not symmetric or no triangle inequality).

[Bottou et al.17]

UNSUPERVISED LEARNING AS DATA-GENERATING MODEL

- Given a parametric family of distributions $P_\theta \in \mathcal{P}_\mathcal{X}$, goal of data modeling is

$$\min_\theta L(\theta) = D(Q, P_\theta)$$

- Similarly as in supervised learning, we need to generalize from an empirical loss:

$$\min_\theta \hat{L}(\theta) = D(\hat{Q}, P_\theta) + \mathcal{R}(\theta) \ .$$

UNSUPERVISED LEARNING AS DATA-GENERATING MODEL

- Given a parametric family of distributions $P_\theta \in \mathcal{P}_\mathcal{X}$, goal of data modeling is

$$\min_\theta L(\theta) = D(Q, P_\theta)$$

- Similarly as in supervised learning, we need to generalize from an empirical loss:

$$\min_\theta \hat{L}(\theta) = D(\hat{Q}, P_\theta) + \mathcal{R}(\theta) \ .$$

- Which criteria D?
- Which model P_θ?

LATENT GRAPHICAL MODELS

- We assume first that both distributions admit a density with respect to a base measure μ :

$$Q(A) = \int_A q(x) d\mu(x) \ , \ P_\theta(A) = \int_A p_\theta(x) d\mu(x) \ .$$

LATENT GRAPHICAL MODELS

- We assume first that both distributions admit a density with respect to a base measure μ :

$$Q(A) = \int_A q(x) d\mu(x) \ , \ P_\theta(A) = \int_A p_\theta(x) d\mu(x) \ .$$

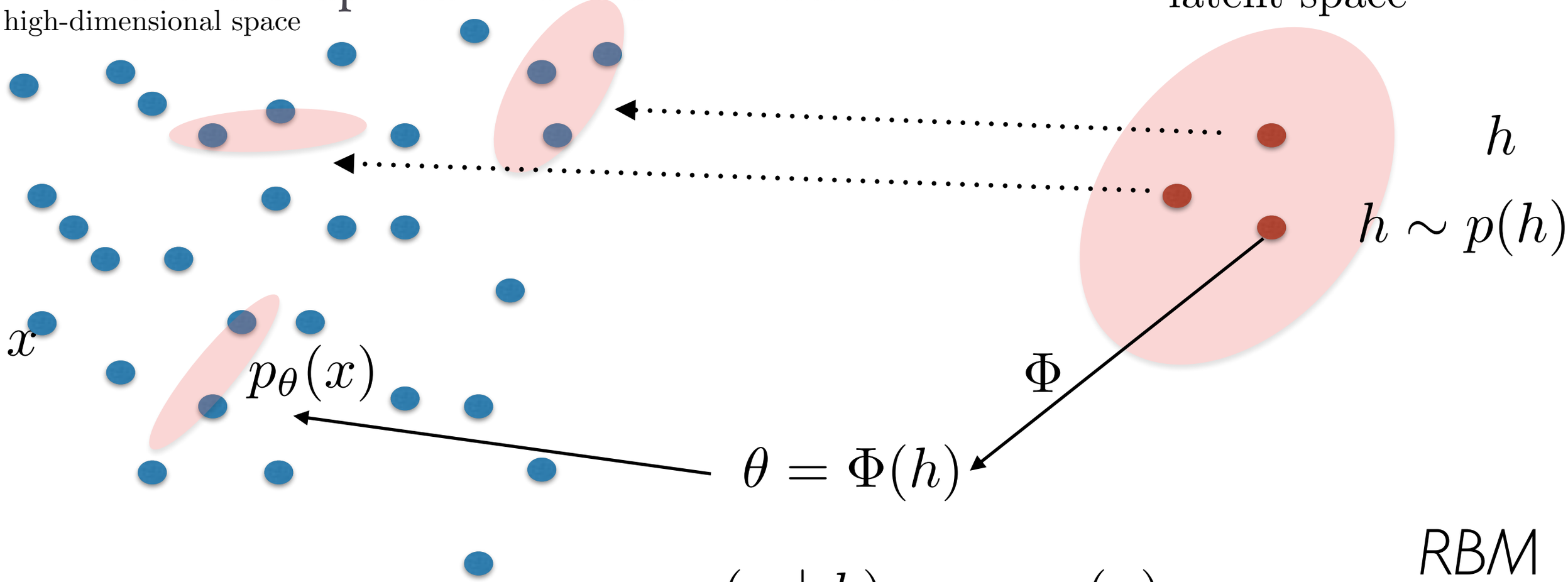
- A popular choice of criteria is the *Kullback-Leibler divergence*:

$$\begin{aligned} D_{KL}(Q, P_\theta) &= \mathbb{E}_{P_\theta} \left\{ f[q(x) p_\theta^{-1}(x)] \right\} \ , \ f(t) = t \log t \\ &= \int q(x) \log \left(\frac{q(x)}{p_\theta(x)} \right) d\mu(x) \ . \end{aligned}$$

- Learning is equivalent to $\max_\theta \mathbb{E}_Q \log p_\theta(x)$

LATENT GRAPHICAL MODELS

➤ Latent Graphical Models or *Mixtures*.



$$p(x \mid h) = p_{\Phi(h)}(x)$$

$$p(x) = \int p(x, h)dh = \int p(x \mid h)p(h)dh$$

RBM
DBN
DBM
VAE
...

Model: additive combination of simple parametric models