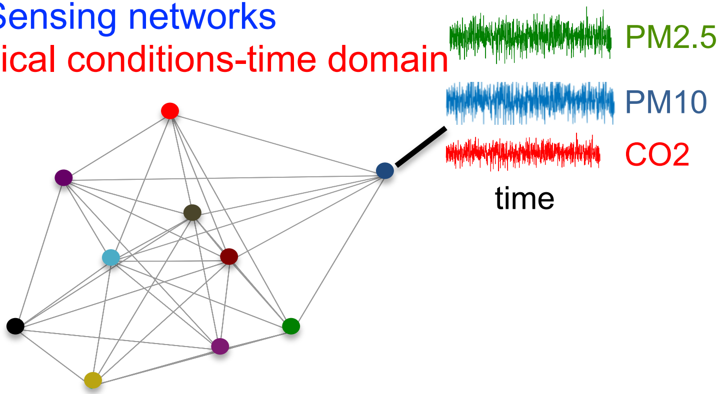
Learning Product Graphs from Multidomain Signals

Sai Kiran Kadambari **Sundeep Prabhakar Chepuri**

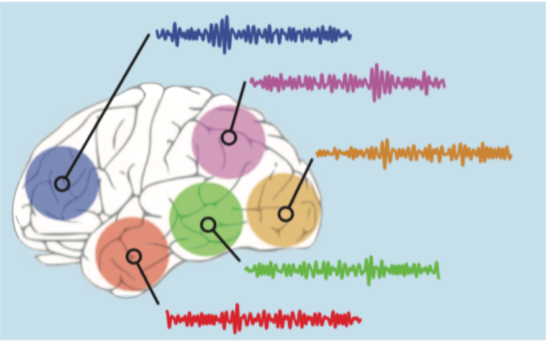
Indian Institute of Science

Email: kadambarik@iisc.ac.in

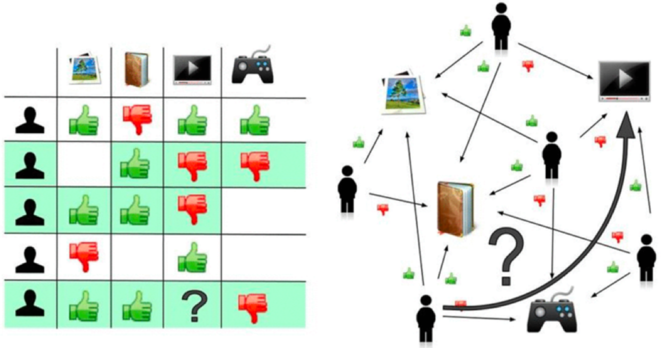
Multidomain graph data



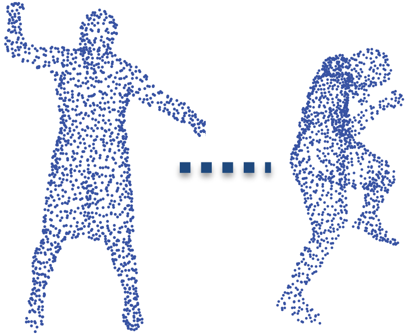
Sensing networks
Space-physical conditions-time domain



Brain networks (fMRI, EEG time-series)
Voxel-time domain



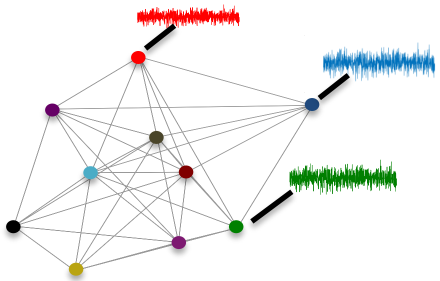
Recommendation systems
Item-user domain



Dynamic point clouds
Space-time domain

Graphs and graph signals

➢ Datasets with *irregular support* can be represented using a graph



Graph signal

$$\mathcal{G}_N = (\mathcal{V}_N, \mathcal{E}_N, \mathbf{L}_N)$$

- $\mathcal{V}_N$ is the set of nodes
- $\mathcal{E}_N$ is the set of edges

➢ Graph with N nodes is represented using a matrix $\mathbf{L}_N \in \mathbb{R}^{N \times N}$

- $\mathbf{L}_N$ could be a graph Laplacian matrix, adjacency matrix, or its variants
- $[\mathbf{L}_N]_{i,j}$ is non zero only if $i = j$ and/or $(i, j) \in \mathcal{E}$

Smooth signals on a graph

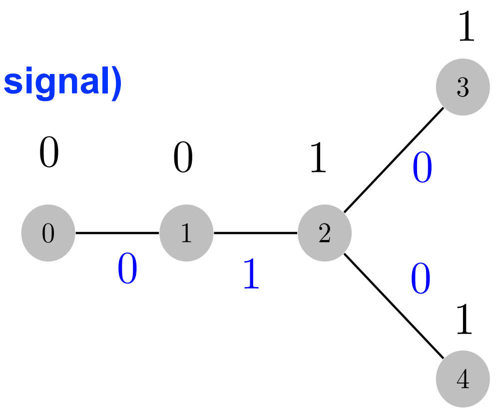
- Smoothness of a graph signal is quantified using the Laplacian quadratic form

$$\mathbf{x}^T \mathbf{L}_N \mathbf{x} = \sum_{(i,j) \in \mathcal{E}_N} (x_i - x_j)^2$$

- Smaller the quadratic term, smoother is the graph signal

(graph signal)

$\mathbf{x} \in \mathbb{R}^N :$



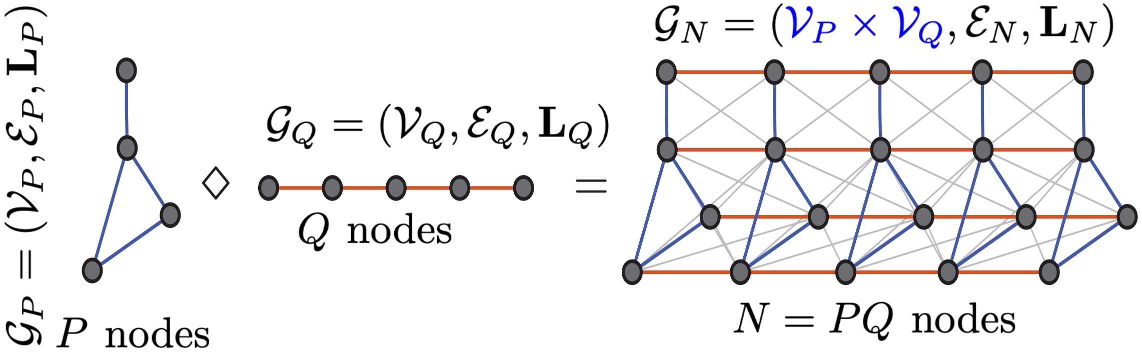
$$\mathbf{x}^T \mathbf{L}_N \mathbf{x} = \sum_{(i,j) \in \mathcal{E}_N} (x_i - x_j)^2$$

$$= 1$$

Sum of squares of differences across edges

Product graphs

- Many graphs can be factorized into two or more smaller graphs



- Cartesian product (colored edges)
- Kronecker product (gray edges)
- Strong product (all edges)

- We will focus on the Cartesian product $\mathcal{G}_N = \mathcal{G}_P \oplus \mathcal{G}_Q$

- Then the Laplacian matrices can be related as

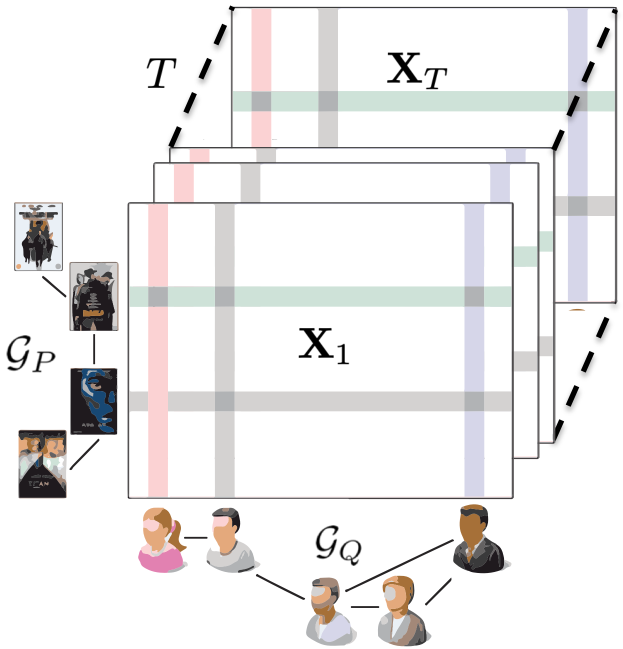
$$\mathbf{L}_N = \mathbf{L}_P \oplus \mathbf{L}_Q = \mathbf{I}_Q \otimes \mathbf{L}_P + \mathbf{L}_Q \otimes \mathbf{I}_P$$

$\otimes$ is the Kronecker product
$\oplus$ is the Kronecker sum

Product graph signals

- Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T] \in \mathbb{R}^{N \times T}$ denote the graph data on $\mathcal{G}_N$
- Each node in the graph $\mathcal{G}_N$ is represented by a pair of vertices in its graph factors
- Any multidomain graph signal can be represented as

$$\mathbf{x}_i = \mathrm{vec}(\mathbf{X}_i), \mathbf{X}_i \in \mathbb{R}^{P \times Q}, i = \{1, 2, \cdots T\}$$



The question!

Given the graph data $\mathbf{X} \in \mathbb{R}^{N \times T}$, estimate the graph Laplacian matrices of the graph factors $\mathcal{G}_P$ and $\mathcal{G}_Q$ that best explain the data

- We develop solvers assuming that

  1. The graph $\mathcal{G}_N$ can be factorized as $\mathcal{G}_N = \mathcal{G}_P \oplus \mathcal{G}_Q$
  2. The graph signals are smooth on the underlying graph

Solution by adapting existing works

Two-step method

Step 1: graph learning

- Ignoring the product structure in $\mathcal{G}_N$ we can compute $\mathbf{L}_N$ using *[Dong et al. 2016]*

$$\underset{\mathbf{L}_N \in \mathcal{L}_N}{\text{minimize}} \quad \text{tr}(\mathbf{X}^T \mathbf{L}_N \mathbf{X}) + \beta \|\mathbf{L}_N\|_F^2$$

Set of valid Laplacian matrices: $\mathcal{L}_N := \{\mathbf{L} \in \mathbb{R}^{N \times N} | \mathbf{L}\mathbf{1} = \mathbf{0}, \text{tr}(\mathbf{L}) = N, L_{ij} = L_{ji} \leq 0, i \neq j\}$

Step 2: product graph factorization

Avoids trivial solution

- Solve a convex program to obtain the graph factors

$$\underset{\mathbf{L}_P \in \mathcal{L}_P, \mathbf{L}_Q \in \mathcal{L}_Q}{\text{minimize}} \|\mathbf{L}_N - \mathbf{L}_P \oplus \mathbf{L}_Q\|_F^2$$

Remarks:

- Two step approach
- Requires computing a size-N Laplacian matrix in step 1

X. Dong et al. “Learning Laplacian matrix in smooth graph signal representations.” *IEEE Trans. Signal Process.*, vol. 64, no. 23, pp. 6160–6173, Dec 2016.

Task-cognizant product graph learning

- Typically, we might not have access to the original data $\mathbf{X}$
- But we observe $\mathbf{Y} \in \mathbb{R}^{N \times T}$ that is related to $\mathbf{X}$
- The Laplacian matrices of the graph factors can be jointly estimated by solving

$$\underset{\mathbf{L}_P \in \mathcal{L}_P, \mathbf{L}_Q \in \mathcal{L}_Q, \mathbf{X}}{\text{minimize}} \; f(\mathbf{X}, \mathbf{Y}) + \alpha \text{tr}(\mathbf{X}^T (\mathbf{L}_P \oplus \mathbf{L}_Q) \mathbf{X}) + \beta_1 \|\mathbf{L}_P\|_F^2 + \beta_2 \|\mathbf{L}_Q\|_F^2$$

Promotes smoothness

Promotes sparsity

- Recall the set of valid Laplacian matrices: $\mathcal{L}_N := \{\mathbf{L} \in \mathbb{R}^{N \times N} | \mathbf{L1} = \mathbf{0}, \text{tr}(\mathbf{L}) = N, L_{ij} = L_{ji} \leq 0, i \neq j\}$

Avoids trivial solution

- Task-specific loss function: $f(\mathbf{X}, \mathbf{Y})$

For noisy observations: $f(\mathbf{X}, \mathbf{Y}) = \|\mathbf{X} - \mathbf{Y}\|_F^2$

Proposed method – one step approach

With $\mathbf{X} = \mathbf{Y}$

- The optimization problem

$$\underset{\mathbf{L}_P \in \mathcal{L}_P, \mathbf{L}_Q \in \mathcal{L}_P}{\text{minimize}} \quad \alpha \text{tr}\left(\mathbf{X}^T (\mathbf{L}_P \oplus \mathbf{L}_Q) \mathbf{X}\right) + \beta_1 \|\mathbf{L}_P\|_F^2 + \beta_2 \|\mathbf{L}_Q\|_F^2$$

is equivalent to the following convex quadratic program

$$\underset{\mathbf{L}_P \in \mathcal{L}_P, \mathbf{L}_Q \in \mathcal{L}_P}{\text{minimize}} \quad \alpha \sum_{i=1}^{T} \left[\text{tr}\left(\mathbf{X}_i^T \mathbf{L}_P \mathbf{X}_i\right) + \text{tr}\left(\mathbf{X}_i \mathbf{L}_Q \mathbf{X}_i^T\right)\right] + \beta_1 \|\mathbf{L}_P\|_F^2 + \beta_2 \|\mathbf{L}_Q\|_F^2$$

- ✓ Symmetric structure of the Laplacian matrices can be leveraged
- ✓ Admits an explicit solution based on a water-filling-like algorithm (obtained by solving the KKT conditions)

Results on synthetic data

- We generate a graph with $N = 150$ nodes

- Obtained by the Cartesian product of two community graphs $P = 10$ and $Q = 15$

- We generate $T = 50$ graph signals on $\mathcal{G}_N$

- Given $\mathbf{X}$, we obtain $\mathcal{G}_P, \mathcal{G}_Q$ using
  - Solver 1: This paper, proposed water-filling method
  - Solver 2: Factorizing the graph obtained from *[Dong et al. 2016]*

- Estimation performance in terms of the F-measure

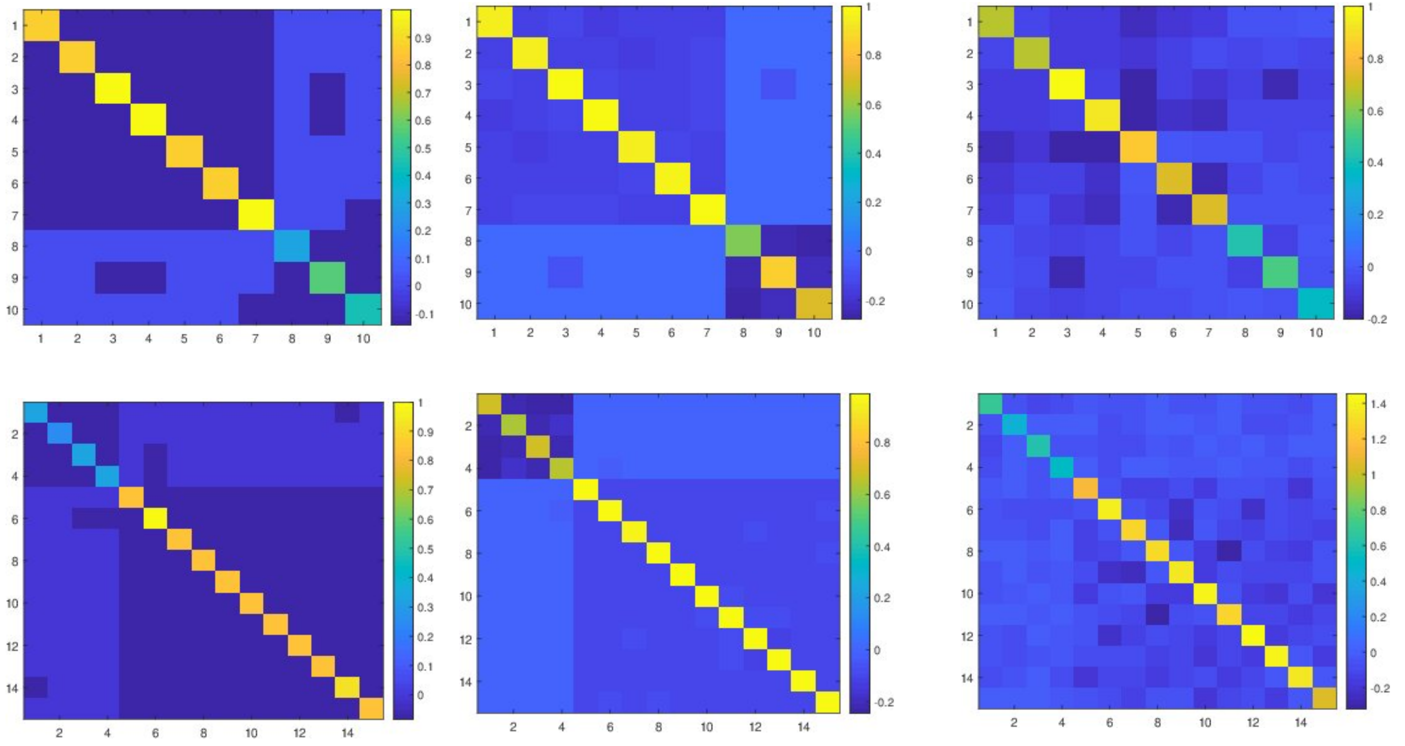
Method	$\mathbf{L}_P$	$\mathbf{L}_Q$	$\mathbf{L}_N$
Solver 1	0.9615	0.9841	0.9755
Solver 2	0.7556	0.7842	0.7612

Results on synthetic data

Ground truth

Solver 1
This paper (one step water-filling)

Solver 2
(Dong et al. 2016 + factorization)



$\mathbf{L}_P$

$\mathbf{L}_Q$

Community structures

Preserved

Lost

Results on real data – air quality dataset

- We use $\mathrm{PM}_{2.5}$ data, collected over 40 stations for each day in year 2018 in India

- The data has many missing entries. Therefore, we perform joint matrix completion and product graph learning

$$\underset{\mathbf{L}_P \in \mathcal{L}_P, \mathbf{L}_Q \in \mathcal{L}_P, \mathbf{X}}{\text{minimize}} \quad f(\mathbf{X}, \mathbf{Y}) + \alpha \sum_{i=1}^{T} \left[\operatorname{tr}\left(\mathbf{X}_i^T \mathbf{L}_P \mathbf{X}_i \right) + \operatorname{tr}\left(\mathbf{X}_i \mathbf{L}_Q \mathbf{X}_i^T \right) \right] + \beta_1 \|\mathbf{L}_P\|_F^2 + \beta_2 \|\mathbf{L}_Q\|_F^2$$

with $f(\mathbf{X}, \mathbf{Y}) := \sum_{i=1}^{T} \|\mathcal{A}(\mathbf{X}_i - \mathbf{Y}_i)\|_F^2 + \|\mathbf{X}_i\|_*$ and observation mask $\mathcal{A}$

- We use alternating minimization to solve the above non-convex problem method to solve

Step 1:

Solve for $\{\mathbf{L}_P, \mathbf{L}_Q\}$ by fixing $\mathbf{X}$ using Solver 1

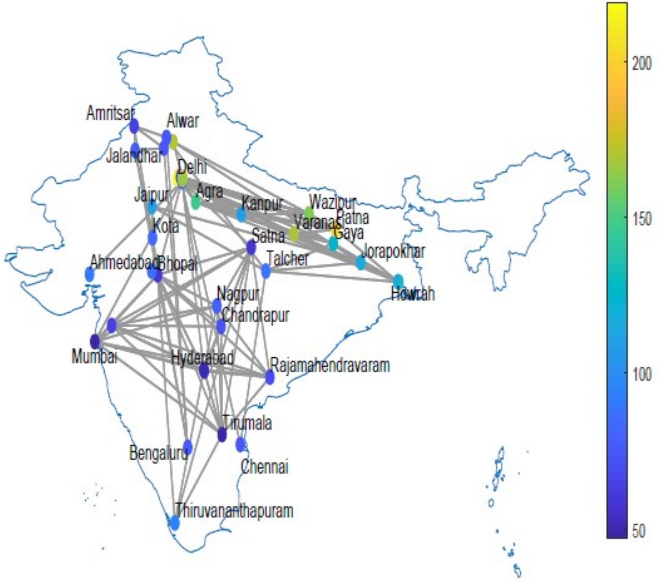
Step 2:

Solve for $\mathbf{X}$ by fixing $\{\mathbf{L}_P, \mathbf{L}_Q\}$ *[Kalofalias et al. 2014]*

$$\underset{\{\mathbf{X}_i\}_{i=1}^{T}}{\text{minimize}} f(\mathbf{X}, \mathbf{Y}) + \alpha \sum_{i=1}^{T} \left[\operatorname{tr}\left(\mathbf{X}_i^T \mathbf{L}_P \mathbf{X}_i \right) + \operatorname{tr}\left(\mathbf{X}_i \mathbf{L}_Q \mathbf{X}_i^T \right) \right]$$

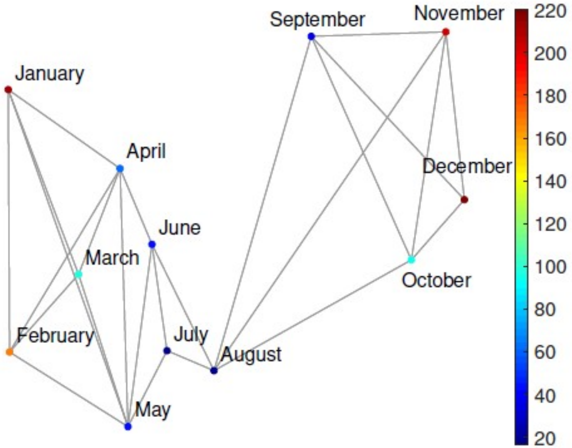
V. Kalofolias et al., “Matrix completion on graphs,” *arXiv preprint,* arXiv:1408.1717, 2014.

Results on real data



$\mathcal{G}_P$

Weather stations



$\mathcal{G}_Q$

Seasons/months

- Close by weather stations are not necessarily connected
- Seasonal variation of the PM2.5 concentration is captured in the temporal graph

Conclusions

- We proposed a framework for learning graphs that can be factorized as the Cartesian product of two smaller graphs from multidomain datasets

- We have shown that the product graph learning can be posed as a convex optimization problem with an explicit and efficient water-filling-like solution

- We applied the developed framework to real air pollution data collected across different locations in India to impute the missing entries and to leverage the underlying graph structure, we estimate the underlying graph factors

Thank You!

This work was supported by the grant from Robert Bosch Centre for Cyber Physical Systems, IISc, India.