# Signal Processing on Expanding Graphs

Bishwadeep Das

TU Delft

*Thanks to* Elvin Isufi, Samuel Rey Escudero, Alan Hanjalic

May 1, 2025

# Graphs and graph signals

## Graphs (Networks)

- Captures pair-wise and higher order relationships between entities
- Examples: Social networks, sensor networks, citation networks
- Irregular domain, unlike time, grid.

## Graph Signals

- Data associated with a graph
- Node level data: User opinions, temperature readings, paper label
- Different way to interpret data w.r.t an underlying topology

# Graph signal processing

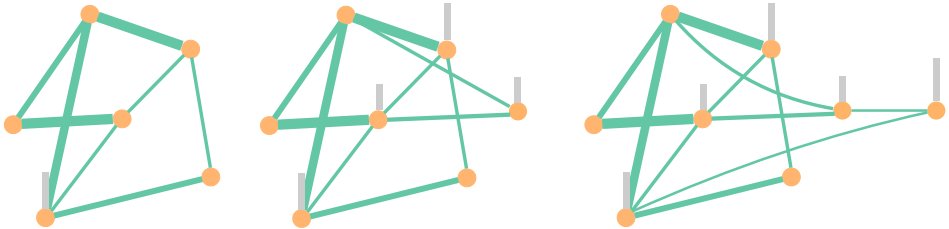
Processing data w.r.t the underlying graph topology for a specific task.

- Filtering: Topology aware filtering out certain variations in the signal [1]
- Interpolation: Filling in missing values at certain nodes from other node values [2]
- Classification: Assign a node to a class based on graph signals [3]
- Topology Identification: Estimate the graph structure (Edges) from graph signals [4]

# Rating prediction in Recommender System

- Nodes: Users in a recommender system with known ratings
- Graph built from rating similarities between the users
- Graph signal: All user ratings for a movie
- Task: Predict unobserved ratings (Interpolation) [5], predicting associations between users and items (topology identification) [6]

# Expanding graphs



- Example: New users joining a recommender system
- Received attention from network science via well-known growth models [7]
- In GSP, majority of works focus on graphs (static and dynamic) with fixed number of nodes [8, 9]
- Lacking a principled approach from a signal processing POV

# Expanding graphs: Challenges

- **Dependence on connectivity**
  - Often the connections of new nodes are not known
  - Inability to use GSP tools to perform rating prediction (Cold start users)

# Expanding graphs: Challenges

- **Dependence on connectivity**
  - Often the connections of new nodes are not known
  - Inability to use GSP tools to perform rating prediction (Cold start users)
- **Streaming nature of data**
  - No Batch type data
  - Need to efficiently re-train as new users arrive over time

# Expanding graphs: Challenges

- **Dependence on connectivity**
  - Often the connections of new nodes are not known
  - Inability to use GSP tools to perform rating prediction (Cold start users)
- **Streaming nature of data**
  - No Batch type data
  - Need to efficiently re-train as new users arrive over time
- **Non-stationarity**
  - New nodes may represent different entities with different distribution of data
  - Signal processing algorithms to adapt to the preferences of different users with different tastes

## Overview of this talk

- Task aware GSP with filters under uncertainty
- Online graph filter design
- Online topology identification

## Commonly used notation

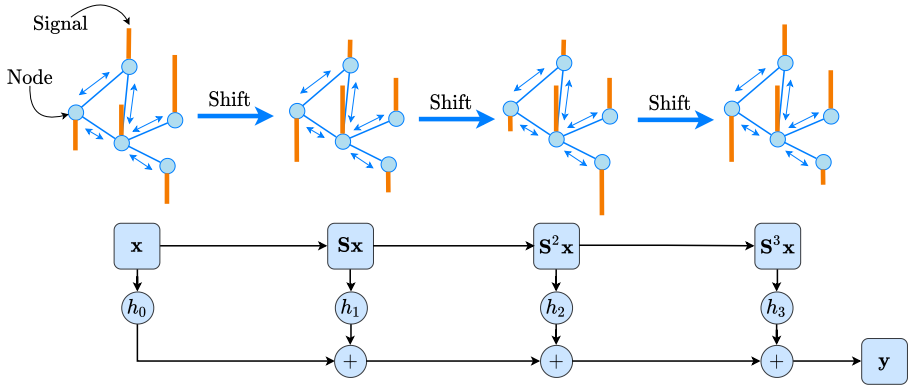
- $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$: Graph with node set $\mathcal{V}$ and edge set $\mathcal{E}$
- $\mathbf{S} \in \mathbb{R}^{N \times N}$: Shift operator of an $N$ node graph with $S_{ij} \neq 0$ if an edge exists between node $i$ and $j$; $S_{ij} = 0$ otherwise
- $\mathbf{x} \in \mathbb{R}^{N \times 1}$: Graph signal
- $\mathbf{A}$: Adjacency matrix, $\mathbf{L}$: Graph Laplacian matrix
- Graphs are undirected without self-loops, unless specified otherwise

# Task aware GSP with filters under uncertainty

## Graph filters

- Flexible, parametric, and localized operator for processing signals on graphs.
- $\mathbf{Sx}$: Shifts $\mathbf{x}$. each node weighs signals from immediate neighbours
- $\mathbf{S}^k\mathbf{x}_k$ shifts signal $k$ times over the graph and accumulates information at each node up to its $k$-hop neighborhood
- Combine shifts to get output $\mathbf{y} = \mathbf{H}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} h_k \mathbf{S}^k \mathbf{x}$
- $h_k$ weighs the $k$th hop, filtering combines them.
- Used in a variety of applications for graphs of fixed size [10, 1]

# Graph filters: Illustration



## Stochastic attachment

- Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with $N$ nodes $\{v_1, \ldots, v_N\}$ with $\mathbf{A} \in \mathbb{R}^{N \times N}$ as adjacency matrix
- $v_+$ incoming node forms graph $\mathcal{G}_+ = \{\mathcal{V} \cup v_+, \mathcal{E}_+\}$
- However, we do not know how $v_+$ will connect
- Let $\mathbf{a}_+ \in \mathbb{R}^N$ be a random vector containing attachment pattern of $v_+$
- $$[\mathbf{a}_+]_i = \begin{cases} w_i & \text{with probability } p_i \\ 0 & \text{with probability } (1 - p_i) \end{cases} \tag{1}$$
- $v_+$ attaches to $v_i$ with probability $p_i$ with edge weight $w_i$

## Modeling unknown connectivity

- $\mathbf{w}$: vector of edge weights $\mathbf{p}$ : vector of probabilities characterise the attachment behaviour of $v_+$
- The updated adjacency and Laplacian matrices are

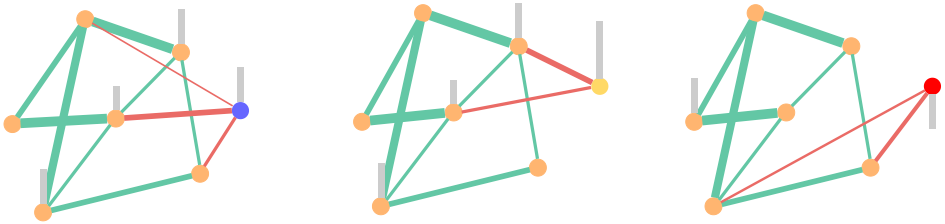
$$\mathbf{A}_+ = \begin{bmatrix} \mathbf{A} & \mathbf{a}_+ \\ \mathbf{a}_+^\top & 0 \end{bmatrix}, \ \mathbf{L}_+ = \begin{bmatrix} \mathbf{L} + \mathrm{diag}(\mathbf{a}_+) & -\mathbf{a}_+ \\ -\mathbf{a}_+^\top & \mathbf{a}_+^\top \mathbf{1} \end{bmatrix} \tag{2}$$

- Statistics of $\mathbf{a}_+$: $\mathbb{E}[\mathbf{a}_+] = \mathbf{w} \circ \mathbf{p}, \quad \mathrm{cov}(\mathbf{a}_+) = \mathbf{\Sigma}_+ = \mathrm{diag}(\mathbf{w}^{\circ 2} \circ \mathbf{p} \circ (\mathbf{1} - \mathbf{p}))$
- What is a good $\mathbf{w}$, $\mathbf{p}$? Our answer: It depends on the task!
- **Target**: Find $\mathbf{w}$ and $\mathbf{p}$ which learns attachment behviour of new nodes

## Processing data with this attachment model

- Let $\mathbf{x} = [x_1, \ldots, _N]^\top$ be the existing graph signal on $\mathcal{G}$
- We want to predict $x_+$ at $v_+$
- For this we use a graph filter $\mathbf{h} \in \mathbb{R}^{K+1}$ of order $K$
- Goal: Learn $\mathbf{w}$, $\mathbf{p}$ for predicting $x_+$
- Done via minimizing a task-aware loss

# Key Idea



Key Idea: Learning from how other new users have already interacted with the network with their data

## Key Idea: More concrete

Learn empirically from a batch of new nodes, each with its own attachment pattern $\mathbf{a}_+$ and signal $x_+$

## Key Idea: More concrete

Learn empirically from a batch of new nodes, each with its own attachment pattern $\mathbf{a}_+$ and signal $x_+$

$$\begin{aligned} &\min_{\mathbf{p},\mathbf{w}} \ \mathbb{E}\big[f_{\mathcal{T}}(\mathbf{p},\mathbf{w},\mathbf{a}_{t+},\mathbf{x},x_+)\big] + g_{\mathcal{T}}(\mathbf{p},\mathbf{b}_{t+}) + h_{\mathcal{T}}(\mathbf{w},\mathbf{a}_{t+}) \\ &\text{subject to} \quad \mathbf{p}\in\mathcal{P},\mathbf{w}\in\mathcal{W} \end{aligned} \tag{3}$$

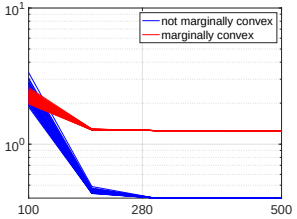
- $f_{\mathcal{T}}(\mathbf{p},\mathbf{w},\mathbf{a}_{t+},\mathbf{x},x_+)$: Task-aware loss
- $g_{\mathcal{T}}(\mathbf{p},\mathbf{b}_{t+})$: Priors on the attachment probabilities
- $h_{\mathcal{T}}(\mathbf{w},\mathbf{a}_{t+})$: Priors on the weights
- Nonconvex problem, learnt by alternating projected gradient descent
- Analysis of convergence and perturbation behaviour in [11].

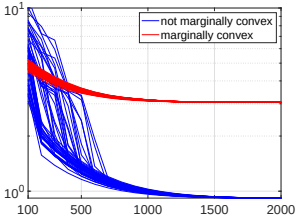
# Experimental Setup

We compare with the following baselines:

- *Uniformly-at-random*: Heuristic, $\mathbf{p} = \frac{1}{N}\mathbf{1}$
- *Degree-biased preferential attachment*, Heuristic, $\mathbf{p} = \frac{\mathbf{d}}{\mathbf{1}^\top \mathbf{d}}$
- *Data-based attachment*: Learnt empirically from the $\mathbf{a}_+$s during training
- **Erdos Renyi** and **Barabasi Albert** graphs for synthetic data
- **Movielens 100K** for real world data
- All hyperparameters found via cross validation

# Synthetic Results




| **MSE** | Erdős-Rényi graph | | | Barabasi-Albert graph | | |
|---|---|---|---|---|---|---|
| **Rule** | Prop. | Pref. | Rand. | Prop. | Pref. | Rand. |
| **Mean Error** | **0.37** | 0.64 | 0.73 | **0.84** | 1.72 | 1.35 |
| **Std Dev.** | **0.04** | 0.04 | 0.04 | **0.11** | 0.11 | 0.11 |

- Outperforms heuristical attachment models
- Typically, $\ell_2$ regularizers on $\mathbf{p}$ and $\mathbf{w}$ work better
- We also found that training both $\mathbf{p}$ and $\mathbf{w}$ is more beneficial

## Rating Prediction on Movielens 100K data

- Existing 35 nearest neighbor graph of $N = 50$ users for each movie
- Trained on individual movie with varying degrees of ratings (personalized)
- Trained on all movies as well

| | Proposed | Attachment Only | Random | Preferential | Mean Prediction |
|---|---|---|---|---|---|
| Item 1 | **0.494** | 0.537(+8.7) | 0.5417(+9.7) | 0.527(+6.7) | 0.669(+35.4) |
| Item 48 | **0.492** | 0.611(+24.2) | 0.53(+7.7) | 0.62(+26) | 0.55(+11.8) |
| Item 459 | 0.462 | 0.49(+6) | 0.52(+12.5) | **0.40(+12.5)** | 1.07(+131) |
| Item 550 | **0.512** | 0.678(+32.4) | 0.66(+29) | 0.692(+35.2) | 0.643(+25.6) |
| Item 57 | **0.049** | 0.057(+16.3) | 0.41(+736) | 0.20(+308) | 0.32(+553) |
| Item 877 | **0.99** | 1.04(+5) | 1.07(+8) | 1.05(+6) | 1.01(+1.72) |
| All Items | **0.799** | 0.802(+0.38) | 0.821(+2.75) | 0.820(+2.63) | 0.832(+4.1) |

- Proposed approach does much better in personalized cases
- Differences evened out for all items

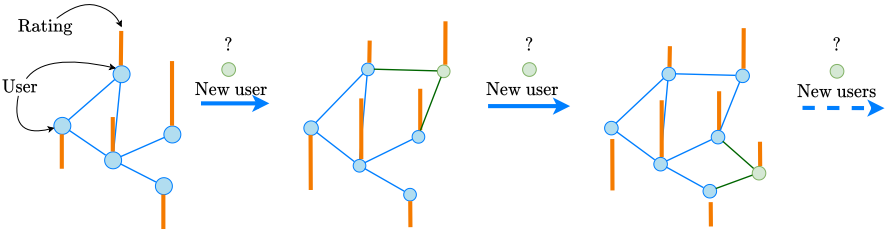
# Overview

- Learning task-aware stochastic attachment model
- We can design graph filters under the same scenarios. More details in [12]
- Some results using graph convolutional neural networks[1]
- Also possible to learn the attachment model and filters jointly
- However, all these works consider one incoming node added with replacement for training

# Online graph filter design

# Sequentially expanding graphs

- Graphs with growing number of nodes
- Delay-sensitive applications
- Time-varying nature of data at incoming node



**Challenges**

- Streaming nature of the growing graph and data over it
- Lack of connectivity of the incoming nodes (i.e., cold start in recommendation)

**Targets**

- How to adapt the graph filter to the change in topology?
- How to design the filter when the connectivity is not known and the graph is growing?

# Proposed Solution

- Train graph filters online, i.e., update with each incoming node
- Use stochastic attachment when node information is not known and update filters online
- Adapt the stochastic attachment pattern to predict the attachment over time
- Obtain performance bounds w.r.t known attachment patterns

## Problem Formulation: Deterministic attachment

- Starting graph $\mathcal{G}_0 = \{\mathcal{V}_0, \mathcal{E}_0\}$: $N_0$ nodes, $M_0$ edges, adj. matrix $\mathbf{A}_0$.
- $v_1, \ldots, v_T$ be $T$ sequentially incoming nodes, node $v_t$ attaches to graph $\mathcal{G}_{t-1}$
- Vector $\mathbf{a}_t = [a_1, \ldots, a_{N_{t-1}}]^\top \in \mathbb{R}^{N_{t-1}}$ represents directed connectivity of $v_t$ at time $t$
- The expanded adjacency matrix at time $t$ $\mathbf{A}_t \in \mathbb{R}^{N_t \times N_t}$ equals

$$\mathbf{A}_t = \begin{bmatrix} \mathbf{A}_{t-1} & \mathbf{0} \\ \mathbf{a}_t^\top & 0 \end{bmatrix}$$

## Problem Formulation: Stochastic attachment

- Node $v_t$ attaches to $v_i \in \mathcal{V}_{t-1}$ obeying Prob. vector $\mathbf{p}_t \in \mathbb{R}^{N_{t-1}}$ and weights $\mathbf{w}_t \in \mathbb{R}^{N_{t-1}}$

## Problem Formulation: Stochastic attachment

- Node $v_t$ attaches to $v_i \in \mathcal{V}_{t-1}$ obeying Prob. vector $\mathbf{p}_t \in \mathbb{R}^{N_{t-1}}$ and weights $\mathbf{w}_t \in \mathbb{R}^{N_{t-1}}$

- Time-varying statistics of $\mathbf{a}_t$: $\mathbb{E}[\mathbf{a}_t] = \mathbf{p}_t \circ \mathbf{w}_t$ ; $\mathbf{\Sigma}_t = \mathsf{diag}(\mathbf{w}_t^{\circ 2} \circ \mathbf{p}_t \circ (\mathbf{1} - \mathbf{p}_t))$

## Problem Formulation: Stochastic attachment

- Node $v_t$ attaches to $v_i \in \mathcal{V}_{t-1}$ obeying Prob. vector $\mathbf{p}_t \in \mathbb{R}^{N_{t-1}}$ and weights $\mathbf{w}_t \in \mathbb{R}^{N_{t-1}}$

- Time-varying statistics of $\mathbf{a}_t$: $\mathbb{E}[\mathbf{a}_t] = \mathbf{p}_t \circ \mathbf{w}_t$ ; $\mathbf{\Sigma}_t = \mathsf{diag}(\mathbf{w}_t^{\circ 2} \circ \mathbf{p}_t \circ (\mathbf{1} - \mathbf{p}_t))$

- Parameter of interest: an order $K$ graph filter $\mathbf{h} = [h_0, \ldots, h_K]^\top$

## Problem Formulation: Stochastic attachment

- Node $v_t$ attaches to $v_i \in \mathcal{V}_{t-1}$ obeying Prob. vector $\mathbf{p}_t \in \mathbb{R}^{N_{t-1}}$ and weights $\mathbf{w}_t \in \mathbb{R}^{N_{t-1}}$

- Time-varying statistics of $\mathbf{a}_t$: $\mathbb{E}[\mathbf{a}_t] = \mathbf{p}_t \circ \mathbf{w}_t$ ; $\mathbf{\Sigma}_t = \mathsf{diag}(\mathbf{w}_t^{\circ 2} \circ \mathbf{p}_t \circ (\mathbf{1} - \mathbf{p}_t))$

- Parameter of interest: an order $K$ graph filter $\mathbf{h} = [h_0, \ldots, h_K]^\top$

- Output at new node $[\tilde{\mathbf{y}}_t]_{N_t} := \hat{x}_t = \mathbf{a}_t^\top \sum_{k=1}^{K} h_k \mathbf{A}_{t-1}^{k-1} \tilde{\mathbf{x}}_t = \mathbf{a}_t^\top \mathbf{A}_{x,t-1} \mathbf{h}$

## Online graph filtering

- At time $t$, $v_t$ appears
- $\mathbf{a}_t$ used in the deterministic setting

## Online graph filtering

- At time $t$, $v_t$ appears
- $\mathbf{a}_t$ used in the deterministic setting
- Use current filter $\mathbf{h}(t-1)$ to predict $\hat{x}_t$

## Online graph filtering

- At time $t$, $v_t$ appears
- $\mathbf{a}_t$ used in the deterministic setting
- Use current filter $\mathbf{h}(t-1)$ to predict $\hat{x}_t$
- Known loss incurred w.r.t true signal $x_t$ revealed as

$$l_t(\mathbf{h}, x_t) = f_t(\mathbf{h}, x_t) + r(\mathbf{h})$$

## Online graph filtering

- At time $t$, $v_t$ appears
- $\mathbf{a}_t$ used in the deterministic setting
- Use current filter $\mathbf{h}(t-1)$ to predict $\hat{x}_t$
- Known loss incurred w.r.t true signal $x_t$ revealed as

$$l_t(\mathbf{h}, x_t) = f_t(\mathbf{h}, x_t) + r(\mathbf{h})$$

- Update $\mathbf{h}(t)$ based on the loss and current estimate $\mathbf{h}(t-1)$.

## Online graph filtering

- At time $t$, $v_t$ appears
- $\mathbf{a}_t$ used in the deterministic setting
- Use current filter $\mathbf{h}(t-1)$ to predict $\hat{x}_t$
- Known loss incurred w.r.t true signal $x_t$ revealed as

$$l_t(\mathbf{h}, x_t) = f_t(\mathbf{h}, x_t) + r(\mathbf{h})$$

- Update $\mathbf{h}(t)$ based on the loss and current estimate $\mathbf{h}(t-1)$.
- $\mathbf{a}_t$ revealed in the stochastic setting

## Deterministic online filtering

- Loss function $l_t(\mathbf{h}, x_t) = \frac{1}{2}(\mathbf{a}_t^\top \mathbf{A}_{x,t-1}\mathbf{h} - x_t)^2 + \mu||\mathbf{h}||_2^2$ with $\mu > 0$.

- Update: Projected online gradient descent [13]

$$\mathbf{h}(t) = \prod_{\mathcal{H}}(\mathbf{h}(t-1) - \eta \nabla_h l_t(\mathbf{h}, x_t)|_{\mathbf{h}(t-1)})$$

  where $\prod_{\mathcal{H}}(\cdot)$ denotes projection on $\mathcal{H}$.

- Computational complexity of order $\mathcal{O}(K(M_t + M_{\max}))$, $M_{\max}$: maximum edges formed by $v_t$

## Stochastic online filtering

- Predefined $\mathbf{p}_t$ and $\mathbf{w}_t$ based on stochastic attachment rule and $\mathbf{w}_t$ for all $t$
- Loss function $l_t(\mathbf{h}, x_t) = \mathbb{E}\left[\frac{1}{2}(\mathbf{a}_t^\top \mathbf{A}_{x,t-1}\mathbf{h} - x_t)^2\right] + \mu||\mathbf{h}||_2^2$ with $\mu > 0$.
- Computational complexity at time $t$ of order $\mathcal{O}(K(M_t + N_t))$
- Potential drawback: Reliance on one type of attachment rule

## Adaptive online filtering

- Update $\mathbf{p}_t$ and $\mathbf{w}_t$ for all $t$, i.e., adapt to the incoming node behaviour

## Adaptive online filtering

- Update $\mathbf{p}_t$ and $\mathbf{w}_t$ for all $t$, i.e., adapt to the incoming node behaviour

- Dictionary of attachment rules $\mathbf{P}_t$ and weights $\mathbf{W}_t$ updated across time

## Adaptive online filtering

- Update $\mathbf{p}_t$ and $\mathbf{w}_t$ for all $t$, i.e., adapt to the incoming node behaviour
- Dictionary of attachment rules $\mathbf{P}_t$ and weights $\mathbf{W}_t$ updated across time
- At time $t$ we have $\bar{\mathbf{p}}_t = \mathbf{P}_{t-1}\mathbf{m}$ and $\bar{\mathbf{w}}_t = \mathbf{W}_{t-1}\mathbf{n}$

## Adaptive online filtering

- Update $\mathbf{p}_t$ and $\mathbf{w}_t$ for all $t$, i.e., adapt to the incoming node behaviour
- Dictionary of attachment rules $\mathbf{P}_t$ and weights $\mathbf{W}_t$ updated across time
- At time $t$ we have $\bar{\mathbf{p}}_t = \mathbf{P}_{t-1}\mathbf{m}$ and $\bar{\mathbf{w}}_t = \mathbf{W}_{t-1}\mathbf{n}$
- Projected online gradient descent but update $\mathbf{h}(t)$, $\mathbf{m}(t)$, and $\mathbf{n}(t)$

## Adaptive online filtering

- Update $\mathbf{p}_t$ and $\mathbf{w}_t$ for all $t$, i.e., adapt to the incoming node behaviour
- Dictionary of attachment rules $\mathbf{P}_t$ and weights $\mathbf{W}_t$ updated across time
- At time $t$ we have $\bar{\mathbf{p}}_t = \mathbf{P}_{t-1}\mathbf{m}$ and $\bar{\mathbf{w}}_t = \mathbf{W}_{t-1}\mathbf{n}$
- Projected online gradient descent but update $\mathbf{h}(t)$, $\mathbf{m}(t)$, and $\mathbf{n}(t)$
- Computational complexity at time $t$ of order $\mathcal{O}(K(M_0 + N_t) + N_t(M))$

## Regret

Regret: $R_T(\mathbf{h}^*) = \sum_{t=1}^{T} l_t(\mathbf{h}_t, x_t) - l_t(\mathbf{h}^*, x_t)$

## Regret

Regret: $R_T(\mathbf{h}^*) = \sum_{t=1}^{T} l_t(\mathbf{h}_t, x_t) - l_t(\mathbf{h}^*, x_t)$

Need to bound it!

# Regret

Regret: $R_T(\mathbf{h}^*) = \sum_{t=1}^{T} l_t(\mathbf{h}_t, x_t) - l_t(\mathbf{h}^*, x_t)$

Need to bound it!

**Deterministic regret**

$$\frac{1}{T} R_T(\mathbf{h}^\star) \leq \frac{||\mathbf{h}^\star||_2^2}{2\eta T} + \frac{\eta}{2} L_d^2$$

## Regret

Regret: $R_T(\mathbf{h}^*) = \sum_{t=1}^{T} l_t(\mathbf{h}_t, x_t) - l_t(\mathbf{h}^*, x_t)$

Need to bound it!

**Deterministic regret**

$$\frac{1}{T} R_T(\mathbf{h}^\star) \leq \frac{||\mathbf{h}^\star||_2^2}{2\eta T} + \frac{\eta}{2} L_d^2$$

**Stochastic regret**

$$\frac{1}{T} R_{s,T}(\mathbf{h}^\star) \leq \frac{1}{T} \Bigg( \sum_{t=1}^{T} w_h^2 Y^2 (||\mathbf{p}_t||_2^2 + M_{max}) + 2R w_h Y \sqrt{||\mathbf{p}_t||_2^2 + M_{max}} + w_h^2 Y^2 \bar{\sigma}_t^2 + L_d ||\mathbf{h}^s(t-1) - \mathbf{h}^d(t-1)|| \Bigg) + \frac{||\mathbf{h}^\star||_2^2}{2\eta} + \frac{\eta}{2} L_d^2 T \tag{4}$$

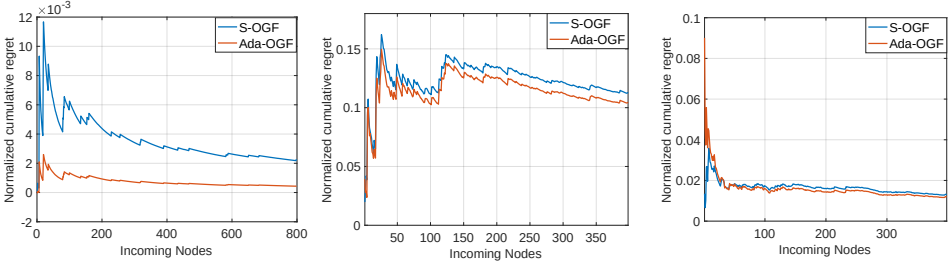
Stochastic regret depends on choice of attachment rule

# Results

Compare with (1) Batch filter, (2) pre-trained filter, (3) Online kernel-based methods OKL and OMHKL, (4) Prediction Correction-based Online filtering (PC-OGF)

| | Synthetic Data | | | | | | Real Data | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Method | Filter | | WMean | | Kernel | | Movielens100K | | COVID | |
| | NRMSE | Sdev | NRMSE | Sdev | NRMSE | Sdev | NRMSE | Sdev | NRMSE | Sdev |
| D-OGF (ours) | **0.038** | 0.04 | **0.02** | 0.02 | **0.28** | 0.05 | **0.26** | 0.01 | 0.21 | 0.02 |
| S-OGF (ours) | 0.26 | 0.04 | 0.3 | 0.06 | 0.36 | 0.11 | 0.28 | 0.007 | 0.31 | 0.02 |
| Ada-OGF (ours) | 0.26 | 0.04 | 0.33 | 0.07 | 0.48 | 0.21 | 0.28 | 0.007 | 0.26 | 0.007 |
| PC-OGF (ours) | 0.22 | 0.04 | 0.26 | 0.04 | 0.32 | 0.06 | 0.29 | 0.01 | 0.26 | 0.003 |
| Batch | 0.05 | 0.03 | 0.08 | 0.04 | 1.12 | 0.26 | 6.7 | 0.1 | **0.17** | 0.03 |
| pre-trained | 0.13 | 0.07 | 0.10 | 0.05 | 0.59 | 0.36 | 0.84 | 0.02 | 2.5 | 0.9 |
| OKL | 0.24 | 0.03 | 0.27 | 0.05 | 0.30 | 0.06 | 0.27 | 0.01 | 0.25 | 0.02 |
| OMHKL | 0.25 | 0.04 | 0.4 | 0.05 | 0.5 | 0.2 | 0.27 | 0.01 | 0.25 | 0.02 |

# Results



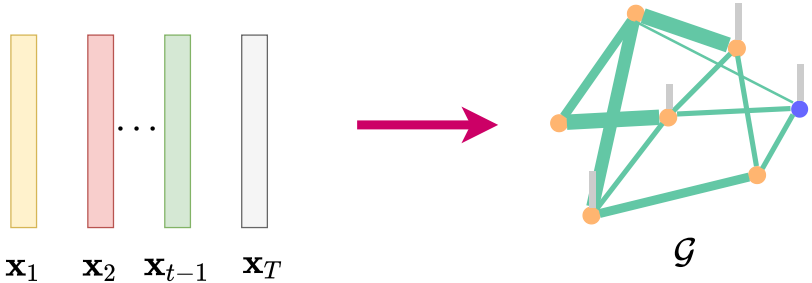

Normalized cumulative regret for stochastic online filtering on synthetic data

## Overview

- We proposed online filtering over graphs that grow sequentially over time. More details in [14]
- Extension to growing simplicial complexes in [15]
- Provide regret analysis in terms of attachment behaviour of incoming nodes
- Numerical results indicate online filters perform collectively better than kernel methods which do not utilize the data, pre-trained filters, and even a batch filter.

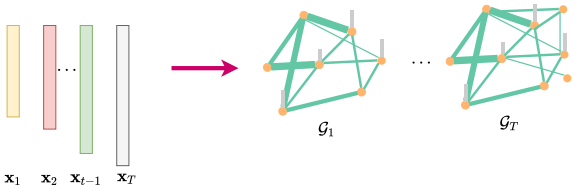
# Online topology identification

- Graph topology is often unknown or unavailable
- Topology identification concerns learning the topology associated with observed data [4]
- Typically uses prior information linking the graph with the topology, e.g., Gaussianity [16], smoothness [17], stationarity [18]



## Contribution

- Existing works focus on estimating graphs (static or dynamic) of fixed size [19, 20, 21]
- Instead, we focus on cases where the underlying graph grows steadily in size over time
- Examples: New companies joining financial networks [22]
- **Target** Learn the growing topology



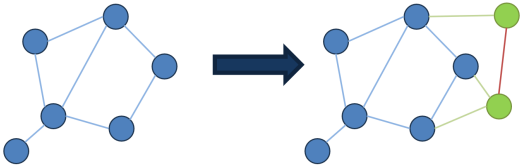
- $\mathbf{X} \in \mathbb{R}^{N \times T}$: Signals collected over $T$ time instances
- $\mathcal{G}$ involved via the graph-shift operator $\mathbf{S} \in \mathbb{R}^{N \times N}$ (e.g. $\mathbf{A}$, $\mathbf{L}$)
- Estimate $\mathbf{S}$ from observed signals $\mathbf{X} \in \mathbb{R}^{N \times T}$ solving

$$\min_{\mathbf{S} \in \mathcal{S}} \quad \mathcal{L}(\mathbf{S}, \hat{\mathbf{C}}) + \lambda \|\mathbf{S}\|_1 \tag{5}$$

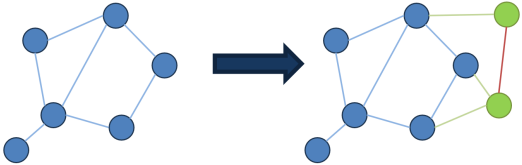
- Loss $\mathcal{L}$ relates $\mathbf{S}$ and sample covariance $\hat{\mathbf{C}} = \frac{1}{T}\mathbf{X}\mathbf{X}^\top$
- $\ell_1$ norm promotes sparsity

# Expanding graphs

- In the expanding graph setting new nodes may arrive every time instant $t$
- The size of the graph increases, i.e., $|\mathcal{V}_t| = N_t \leq |\mathcal{V}_{t+1}| = N_t + 1$



# Expanding graphs




- Sequential arrival of nodes results in the following block structure

$$\mathbf{S}_{t+1} = \begin{bmatrix} [\mathbf{S}_{t+1}]_{\mathcal{V}_t} & [\mathbf{S}_{t+1}]_{\mathcal{V}_t,\mathcal{I}_{t+1}} \\ [\mathbf{S}_{t+1}]_{\mathcal{I}_{t+1},\mathcal{V}_t} & [\mathbf{S}_{t+1}]_{\mathcal{I}_t} \end{bmatrix}, \qquad \mathbf{x}_{t+1} = \begin{bmatrix} [\mathbf{x}_{t+1}]_{\mathcal{V}_t} \\ [\mathbf{x}_{t+1}]_{\mathcal{I}_t} \end{bmatrix}$$

- Where $\mathcal{I}_t$ denoting the indexes of incoming nodes at time $t$

## Batch Solution

- One option is to learn the new topology $\mathbf{S}_t^*$ every time a new node and its data appears
- We calculate the sample covariance $\hat{\mathbf{C}}_t$ and solve

$$\mathbf{S}_t^* = \underset{\mathbf{S}\in\mathcal{S}_{N_t}}{\operatorname{argmin}} \mathcal{L}(\mathbf{S}, \hat{\mathbf{C}}_t) + \alpha d([\mathbf{S}]_{\mathcal{V}_{t-1}}, \mathbf{S}_{t-1}^*) + \lambda\|\mathbf{S}\|_1 \tag{6}$$

- Second term accounts for smooth variation between old nodes
- Provide high-quality estimates $\mathbf{S}_t^*$
- Excessive computational cost, not fit for delay-sensitive applications

- **Step 1**: At time $t$, $\mathbf{x}_t \in \mathbb{R}^{N_t}$ is available
- Update the covariance matrix as

$$\hat{\mathbf{C}}_t = \mathbf{M}_1 \circ \hat{\underline{\mathbf{C}}}_{t-1}^{(N_t)} + \mathbf{M}_2 \circ \mathbf{x}_t \mathbf{x}_t^\top \tag{7}$$

- Mask $\mathbf{M}_1$ scales importance of past observations, $\mathbf{M}_2$ that of the latest

## Online Algorithm

- **Step 1**: At time $t$, $\mathbf{x}_t \in \mathbb{R}^{N_t}$ is available
- Update the covariance matrix as

$$\hat{\mathbf{C}}_t = \mathbf{M}_1 \circ \underline{\hat{\mathbf{C}}}_{t-1}^{(N_t)} + \mathbf{M}_2 \circ \mathbf{x}_t \mathbf{x}_t^\top \tag{7}$$

- Mask $\mathbf{M}_1$ scales importance of past observations, $\mathbf{M}_2$ that of the latest
- **Step 2**: Loss function at time $t$ given by

$$f_t(\mathbf{S}) = \mathcal{L}(\mathbf{S}, \hat{\mathbf{C}}_t) + \alpha d([\mathbf{S}]_{\mathcal{V}_{t-1}}, \hat{\mathbf{S}}_{t-1}) + ||\mathbf{S}||_1 \tag{8}$$

- Proximal gradient step: $\check{\mathbf{S}}_t = \Pi_{\mathcal{S}_{N_t}} \left( T_{\eta\lambda} \left( \underline{\hat{\mathbf{S}}}_{t-1}^{(N_t)} - \eta \nabla f(\underline{\hat{\mathbf{S}}}_{t-1}^{(N_t)}) \right) \right)$
- Projection on to constraint set of $N_t \times N_t$ shift operators

## Online Algorithm

- **Step 1**: At time $t$, $\mathbf{x}_t \in \mathbb{R}^{N_t}$ is available
- Update the covariance matrix as

$$\hat{\mathbf{C}}_t = \mathbf{M}_1 \circ \underline{\hat{\mathbf{C}}}_{t-1}^{(N_t)} + \mathbf{M}_2 \circ \mathbf{x}_t \mathbf{x}_t^\top \tag{7}$$

- Mask $\mathbf{M}_1$ scales importance of past observations, $\mathbf{M}_2$ that of the latest
- **Step 2**: Loss function at time $t$ given by

$$f_t(\mathbf{S}) = \mathcal{L}(\mathbf{S}, \hat{\mathbf{C}}_t) + \alpha d([\mathbf{S}]_{\mathcal{V}_{t-1}}, \hat{\mathbf{S}}_{t-1}) + ||\mathbf{S}||_1 \tag{8}$$

- Proximal gradient step: $\check{\mathbf{S}}_t = \Pi_{\mathcal{S}_{N_t}} \left( T_{\eta\lambda} \left( \underline{\hat{\mathbf{S}}}_{t-1}^{(N_t)} - \eta \nabla f(\underline{\hat{\mathbf{S}}}_{t-1}^{(N_t)}) \right) \right)$
- Projection on to constraint set of $N_t \times N_t$ shift operators
- **Step 3**: update estimate GSO as $\hat{\mathbf{S}}_t = h\check{\mathbf{S}}_t + (1-h)\underline{\hat{\mathbf{S}}}_{t-1}^{(N_t)}$

## The Gaussian case

- Assume signals are drawn from Gaussian distribution $\mathbf{x}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{S}_t^{-1})$
- GSO estimated maximizing the regularized log-likelihood [16]

$$\nabla f_t(\mathbf{S}) = \hat{\mathbf{C}}_t - (\mathbf{S} + \epsilon \mathbf{I})^{-1} + \alpha \nabla d([\mathbf{S}]_{\mathcal{V}_{t-1}}, \hat{\mathbf{S}}_{t-1})$$
$$\Pi_{\mathcal{S}_{N_t}}(\mathbf{S}) = \mathbf{V} \min \left\{ \max \{\mathbf{\Lambda}, 0\}, \sigma^{1/2} \right\} \mathbf{V}^\top$$

- Gradient from log-likelihood and projection to PSD matrices set
- Computational complexity of $\mathcal{O}(N_t^3)$ with over-the-shelf methods

## Regret

- We want to know how accurately the online approach can track the growing topology
- Cumulative regret measures error made over a sequence

$$\sum_{t=1}^{T} \|\hat{\mathbf{S}}_t - \mathbf{S}_t^*\|_F \tag{9}$$

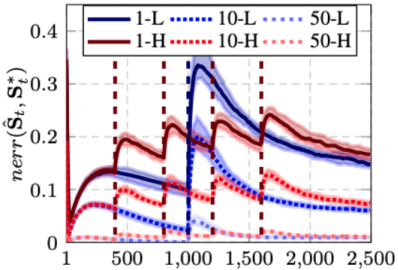
- With step size $\eta \leq \epsilon^2$, dynamic cumulative regret is upper bounded by

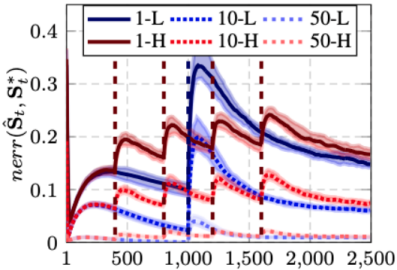
$$\sum_{t=1}^{T} \|\hat{\mathbf{S}}_t - \mathbf{S}_t^*\|_F \leq K_1 + K_2 \sum_{t=2}^{T} \|\mathbf{S}_t^* - \underline{\mathbf{S}}_{t-1}^{*(N_t)}\|_F \tag{10}$$

- Bounded by how much the optimal topology changes.

## Numerical evaluation

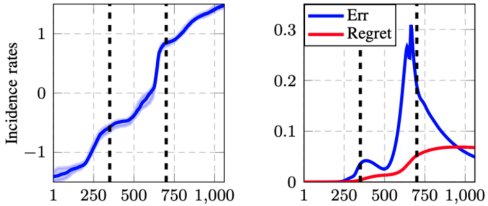
- ER graphs growing from 100 to 120 nodes with different frequencies
- Low-frequency setting: 20 new nodes arrive simultaneously
- High-frequency setting: nodes arrive in 4 groups of 5 nodes




- Recovers from disruptions in both settings, insufficient recovery tme $\implies$ error accumulation

- COVID-19 dataset with incidence rates reported by various U.S. states and territories.
- Start with 46 nodes and reports from additional states are available at $t = \{312, 652\}$.




- Error rapidly increases after $t = 350$ coinciding with a rise in incident rates.
- Our algorithm adapts to the new topology and the error decreases.

## Overview

- An online graph learning method tailored to expanding graphs
- Define efficient node-dependent covariance updates
- Propose an algorithm based on PPG descent to learn from streaming data
- Bounded dynamic regret

## Overview

- A SP-based principled framework for expanding graphs
- Stochastic model-based attachments for unknown connectivity
- Online approaches for SP and Topology identification with some theoretical analysis
- Relatively well compared to known topology

## Overview

- A SP-based principled framework for expanding graphs
- Stochastic model-based attachments for unknown connectivity
- Online approaches for SP and Topology identification with some theoretical analysis
- Relatively well compared to known topology

## Challenges and further possibilities

- Modeling data on expanding graphs a challenge
- More refined stochastic attachment models
- Potential for using physics-driven models involving the spectrum
- Extension to expanding higher order networks possible

END

# References I


- E. Isufi, F. Gama, D. I. Shuman, and S. Segarra, "Graph filters for signal processing and machine learning on graphs," *IEEE Transactions on Signal Processing*, vol. 72, pp. 4745–4781, 2024.
- X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," in *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 639–648, 2020.
- F. Gama, A. G. Marques, G. Leus, and A. Ribeiro, "Convolutional neural network architectures for signals supported on graphs," *IEEE Transactions on Signal Processing*, vol. 67, no. 4, pp. 1034–1049, 2018.
- G. Mateos, S. Segarra, A. G. Marques, and A. Ribeiro, "Connecting the Dots: Identifying Network Structure via Graph Signal Processing," *IEEE Signal Process. Mag.*, vol. 36, pp. 16–43, May 2019.
- W. Huang, A. G. Marques, and A. R. Ribeiro, "Rating Prediction via Graph Signal Processing," *IEEE Trans. Signal Process.*, vol. 66, pp. 5066–5081, Oct. 2018.

# References II


- Z. Huang, X. Li, and H. Chen, "Link prediction approach to collaborative filtering," in *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, pp. 141–142, 2005.
- A.-L. Barabási *et al.*, *Network science*. Cambridge university press, 2016.
- A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, "Graph Signal Processing: Overview, Challenges, and Applications," *Proc. IEEE*, vol. 106, pp. 808–828, May 2018.
- D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, 2013.
- E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Filtering random graph processes over random time-varying graphs," *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4406–4421, 2017.

# References III


- B. Das, A. Hanjalic, and E. Isufi, "Task-aware connectivity learning for incoming nodes over growing graphs," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 8, pp. 894–906, 2022.
- B. Das and E. Isufi, "Graph filtering over expanding graphs," in *IEEE Data Science Learning Workshop Processing (DSLW)*, May 2022.
- F. Orabona, "A modern introduction to online learning," *arXiv preprint arXiv:1912.13213*, 2019.
- B. Das and E. Isufi, "Online graph filtering over expanding graphs," *IEEE Transactions on Signal Processing*, 2024.
- M. Yang, B. Das, and E. Isufi, "Online edge flow prediction over expanding simplicial complexes," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, IEEE, 2023.
- J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

# References IV


- V. Kalofolias, "How to learn a graph from smooth signals," in *Art. Intel. Stat.*, pp. 920–929, PMLR, 2016.
- A. Buciulea, M. Navarro, S. Rey, S. Segarra, and A. G. Marques, "Online network inference from graph-stationary signals with hidden nodes," *arXiv preprint arXiv:2409.08760*, 2024.
- D. Hallac, Y. Park, S. Boyd, and J. Leskovec, "Network inference via the time-varying graphical lasso," in *Intl. Conf. Knowledge Discovery Data Mining*, pp. 205–213, 2017.
- R. Shafipour, A. Hashemi, G. Mateos, and H. Vikalo, "Online topology inference from streaming stationary graph signals," in *2019 IEEE Data Science Workshop (DSW)*, pp. 140–144, IEEE, 2019.
- A. Natali, M. Coutino, E. Isufi, and G. Leus, "Online time-varying topology identification via prediction-correction algorithms," in *IEEE Intl. Conf. Acoustics, Speech and Signal Process. (ICASSP)*, pp. 5400–5404, 2021.
- J. V. De Miranda Cardoso and D. P. Palomar, "Learning undirected graphs in financial markets," in *Conf. Signals, Syst., Computers (Asilomar)*, pp. 741–745, IEEE, 2020.