Contributions to graph learning and change point detection

Magnet seminar

Batiste Le Bars

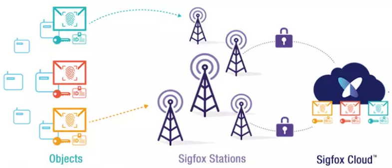
Magnet, Inria Lille
Before: Centre Borelli, ENS Paris-Saclay

Jeudi 13 Janvier 2022

Industrial context

What is Sigfox?

- Internet-of-Things network
- 28k Base Stations (BS)

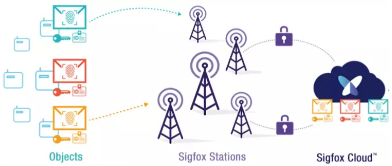


- A message can be received by all nearby BS
- $\sim$ 56M messages/day
- 72 countries

Industrial context

What is Sigfox?

- Internet-of-Things network
- 28k Base Stations (BS)



- A message can be received by all nearby BS
- $\sim$ 56M messages/day
- 72 countries

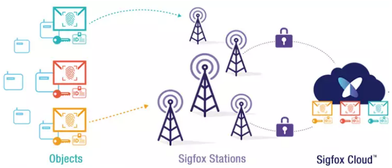
Objective

Detect BS failure using the data collected in the network

Industrial context

What is Sigfox?

- Internet-of-Things network
- 28k Base Stations (BS)



- A message can be received by all nearby BS
- $\sim$ 56M messages/day
- 72 countries

Objective

Detect BS failure using the data collected in the network

Which data?

- Only reception information
- For each message, which BS received it (1) or not (0)

	BS#1	BS#2	BS#3	…
Message #1	0	1	1	…
Message #2	1	0	0	…
Message #3	0	0	1	…
…	…	…	…	…

- "Pure" data: almost no processing
- Collected at the level of nodes in a network: Graph vectors!

General context

Graph vectors

- Let $G = (\mathcal{V}, \mathcal{E})$ be a graph, $y : \mathcal{V} \to \mathbb{R}$ is a graph vector
- Also referred as *graph signals* or *graph data*
- Examples: Sigfox data, Social network data, EEG etc.

General context

Graph vectors

- Let $G = (\mathcal{V}, \mathcal{E})$ be a graph, $y : \mathcal{V} \to \mathbb{R}$ is a graph vector
- Also referred as *graph signals* or *graph data*
- Examples: Sigfox data, Social network data, EEG etc.

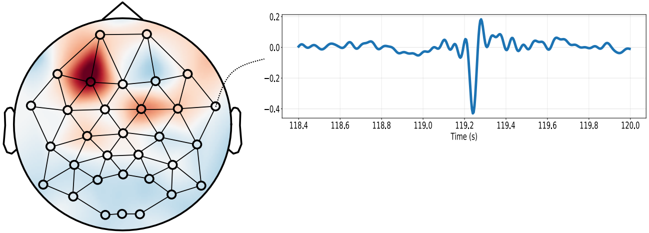


Figure: Electroencephalogram (EEG) seen as a set of graph data

General context

Graph vectors

- Let $G = (\mathcal{V}, \mathcal{E})$ be a graph, $y : \mathcal{V} \to \mathbb{R}$ is a graph vector
- Also referred as *graph signals* or *graph data*
- Examples: Sigfox data, Social network data, EEG etc.

Problems

- Graph known:
  $\rightarrow$ improve the performance of your learning/statistical tasks
- Graph unknown:
  $\rightarrow$ learn it to better understand the data

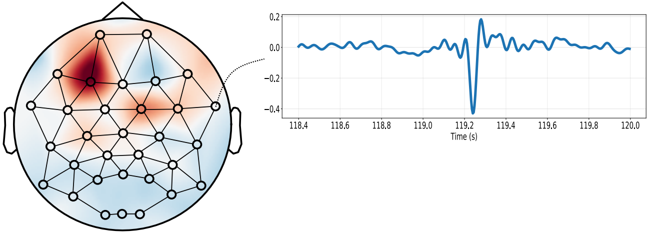
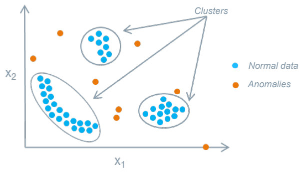


Figure: Electroencephalogram (EEG) seen as a set of graph data

Objectives and motivations

Event detection for graph vectors

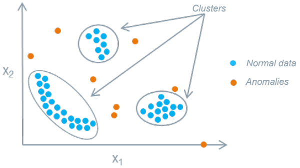
- *Anomaly* or *Change-point* detection
- Motivated by Sigfox application (BS failure)
- Applications: network security, sensor's breakdown, etc.



Objectives and motivations

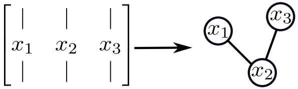
Event detection for graph vectors

- *Anomaly* or *Change-point* detection
- Motivated by Sigfox application (BS failure)
- Applications: network security, sensor's breakdown, etc.



Graph learning

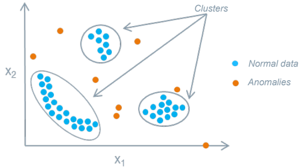
- Infer the relationship between variables (similarity, dependency, etc.)
- Visualize and model the vectors. Apply graph-based learning algorithms
- Applications: gene co-expression, movie recommendation, etc.



Objectives and motivations

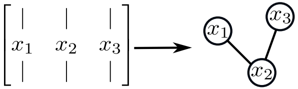
Event detection for graph vectors

- *Anomaly* or *Change-point* detection
- Motivated by Sigfox application (BS failure)
- Applications: network security, sensor's breakdown, etc.



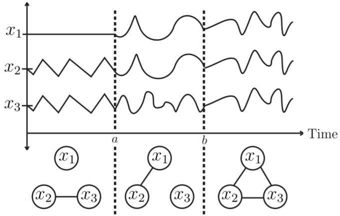
Graph learning

- Infer the relationship between variables (similarity, dependency, etc.)
- Visualize and model the vectors. Apply graph-based learning algorithms
- Applications: gene co-expression, movie recommendation, etc.



Detect changes in the underlying graph

- Combination of graph learning and change-point detection
- More difficult
- Keeps the advantages of the previous tasks



Related works

Event detection for graph vectors

- Use the graph to build features (Chen *et al.* 2018 [3], Egilmez *et al.* 2014 [6])
- Different levels of detection
  → node-level (Ji *et al.* 2013 [11]), subgraph-level (Neil *et al.* 2013 [15]), graph-level (Chen *et al.* 2018 [3])

Related works

Event detection for graph vectors

- Use the graph to build features (Chen *et al.* 2018 [3], Egilmez *et al.* 2014 [6])
- Different levels of detection
  $\rightarrow$ node-level (Ji *et al.* 2013 [11]), subgraph-level (Neil *et al.* 2013 [15]), graph-level (Chen *et al.* 2018 [3])

Graph learning

- Statistical framework: estimating parameters of Markov Random Fields
  $\rightarrow$ Gaussian model (Friedman *et al.* 2008 [7]), Ising model (Ravikumar *et al.* 2010 [16], Goel *et al.* 2019 [9])
- Graph signal processing framework
  $\rightarrow$ Smoothness (Dong *et al.* 2016 [5]), sparsity of the graph spectral domain (Sardellitti *et al.* 2019 [18])

Related works

Event detection for graph vectors

- Use the graph to build features (Chen *et al.* 2018 [3], Egilmez *et al.* 2014 [6])
- Different levels of detection
  $\rightarrow$ node-level (Ji *et al.* 2013 [11]), subgraph-level (Neil *et al.* 2013 [15]), graph-level (Chen *et al.* 2018 [3])

Graph learning

- Statistical framework: estimating parameters of Markov Random Fields
  $\rightarrow$ Gaussian model (Friedman *et al.* 2008 [7]), Ising model (Ravikumar *et al.* 2010 [16], Goel *et al.* 2019 [9])
- Graph signal processing framework
  $\rightarrow$ Smoothness (Dong *et al.* 2016 [5]), sparsity of the graph spectral domain (Sardellitti *et al.* 2019 [18])

Detect changes in the underlying graph

- Statistical framework (Roy *et al.* 2017 [17], Londschien *et al.* 2020 [17] [14])
- Graph signal processing framework (Yamada *et al.* 2020 [20])
- Known (Bybee and Atchadé, 2018 [1]) vs Unknown (Gibberd and Nelson, 2017 [8]) number of change-points

Outline

Part 1

-

Node-level anomaly detection in networks: application to Sigfox

Model

- Let N be the number of considered BS

Definition (Fingerprint)

The *fingerprint* of a Sigfox message is $X = (X_1, \ldots, X_N) \in \{0, 1\}^N$, where $X_j = 1$ if BS j received the message, 0 otherwise

- Assumption 1: Sigfox messages are independent random vectors

Model

- Let N be the number of considered BS

Definition (Fingerprint)

The *fingerprint* of a Sigfox message is $X = (X_1, \ldots, X_N) \in \{0,1\}^N$, where $X_j = 1$ if BS j received the message, 0 otherwise

- Assumption 1: Sigfox messages are independent random vectors

Definition (Conditional probability function)

Let a BS $j \in [N]$, The conditional probability function of j is

$$\eta_j^*(x_{\setminus j}) \triangleq \mathbb{P}(X_j = 1 | X_{\setminus j} = x_{\setminus j}),$$

where $X_{\setminus j}$ is the vector X without its j-th component

- Assumption 2: The conditional probability function of a BS j doesn't change over time

Objective and scoring function

Goal: Given a set $\mathcal{D}_n = \{X^{(i)}\}_{i=1}^n$ and its realization $\{x^{(i)}\}_{i=1}^n$, fix a BS $j \in [N]$ and determine if $m_j = \sum_{i=1}^n x_j^{(i)}$ is abnormally low

- Assumption 3: We have access to a set of normal communication behaviors $\mathcal{D}_{train}$

Objective and scoring function

Goal: Given a set $\mathcal{D}_n = \{X^{(i)}\}_{i=1}^n$ and its realization $\{x^{(i)}\}_{i=1}^n$, fix a BS $j \in [N]$ and determine if $m_j = \sum_{i=1}^n x_j^{(i)}$ is abnormally low

- Assumption 3: We have access to a set of normal communication behaviors $\mathcal{D}_{train}$

A natural scoring function

- Use values of the other BS
- Knowing $X_{\setminus j}^{(i)} = x_{\setminus j}^{(i)}$, $M_j = \sum_{i=1}^n X_j^{(i)}$ is a Poisson Binomial distribution with parameter $\{\eta_j^*(x_{\setminus j}^{(i)})\}_{i=1}^n$
- Given η_j^*, its cumulative distribution function (cdf) $F_{M_j}(\cdot)$ can be computed efficiently (Hong, 2013 [10])

Objective and scoring function

Goal: Given a set $\mathcal{D}_n = \{X^{(i)}\}_{i=1}^n$ and its realization $\{x^{(i)}\}_{i=1}^n$, fix a BS $j \in [N]$ and determine if $m_j = \sum_{i=1}^n x_j^{(i)}$ is abnormally low

- Assumption 3: We have access to a set of normal communication behaviors $\mathcal{D}_{train}$

A natural scoring function

- Use values of the other BS
- Knowing $X_{\setminus j}^{(i)} = x_{\setminus j}^{(i)}$, $M_j = \sum_{i=1}^n X_j^{(i)}$ is a Poisson Binomial distribution with parameter $\{\eta_j^*(x_{\setminus j}^{(i)})\}_{i=1}^n$
- Given η_j^*, its cumulative distribution function (cdf) $F_{M_j}(\cdot)$ can be computed efficiently (Hong, 2013 [10])

Definition (Anomaly scoring function)

A natural score of abnormality for m_j is given by:

$$s(m_j) = \mathbb{P}(M_j > m_j) = 1 - F_{M_j}(m_j),$$

where close to 1 value means m_j stands in a low-density region (left-hand tail).

- In practice we do not have access to η_j^*. What can be done?

A supervised-learning solution

Solution

- Learn $\hat{\eta}_j$, estimator of η_j^*, using a regression algorithm (logistic, random forest, etc.) over $\mathcal{D}_{train}$
- Use $\hat{\eta}_j$ instead of η_j^* to build F_{M_j}, and compute the previous anomaly score
- Fix a threshold above which m_j is considered abnormal (e.g. 0.99 or 0.95)

Algorithm: Regression-based anomaly detection

Input: $\mathcal{D}_{train}$, $\mathcal{D}_n$, node j, threshold s
Regression algorithm: $\texttt{Regressor}(\cdot)$
Output: 1 if anomaly, 0, otherwise

$\hat{\eta} \longleftarrow \texttt{Regressor}\left(\mathcal{D}_{train} = \{\tilde{x}_{\backslash j}^{(i)}, \tilde{x}_j^{(i)}\}\right)$
for $i = 1 \dots, n$ **do**
  $\hat{p}_i \longleftarrow \hat{\eta}(x_{\backslash j}^{(i)})$
end for
$\hat{F} \longleftarrow \texttt{PoiBin}\left(\sum x_j^{(i)}; \hat{p}_j^{(1)}, \dots, \hat{p}_j^{(n)}\right)$
$\hat{s} \longleftarrow \max(\hat{F}, 1 - \hat{F})$
if $\hat{s} > s$ **then**
  Output 1: Abnormal node
else
  Output 0: Normal node
end if

Sigfox application

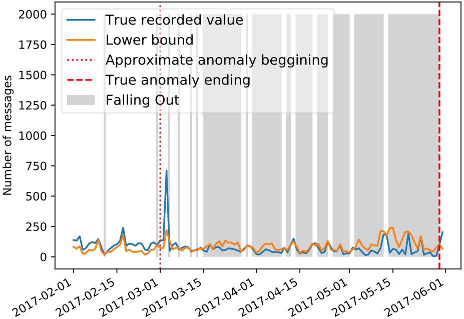
Dataset

- 34 BS, 232000 messages over 5 months
- Training set: first month($\sim$ 35000 messages)
- Daily prediction over the 4 other months: 120 testing data sets ($\sim$ 1600 messages/day in average)
- 1 failing BS, approximately from March, i.e. 30 normal days, 90 abnormal
- Dataset available online

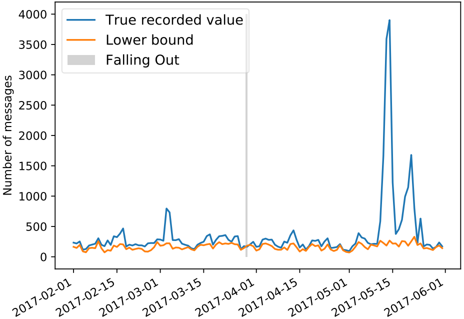
Setup

- Regressor: Random forest from scikit-learn, by default hyperparameters (no tuning)
- Threshold fixed via CV over the training set s.t. False positive rate ~ 0.01
- Baseline: basic feature engineering + One-class SVM (Schölkopf *et al.* 2001 [19])

Results



(a) Abnormal Base Station



(b) Normal Base Station

Results (2)

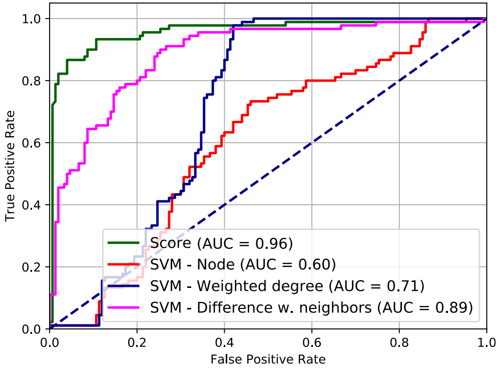


Figure: ROC curves and their respective AUC.

- Convincing results: the proposed approach seems adapted
- Larger-scale experiments, performed internally at Sigfox, corroborate those results
- A more general presentation of the method in [Le Bars and Kalogeratos, INFOCOM 2019]

Part 2

-

Graph inference from smooth and bandlimited graph signals

Background

Graph Signal Processing (GSP)

- Generalizes signal processing concepts for graph signals (smoothness, Fourier transform, sampling, filtering, etc.)
- Temporal signals and images are graph signals with specific graph (cycles and grid)
- Having access to the graph is a strong assumption: graph learning

Background

Graph Signal Processing (GSP)

- Generalizes signal processing concepts for graph signals (smoothness, Fourier transform, sampling, filtering, etc.)
- Temporal signals and images are graph signals with specific graph (cycles and grid)
- Having access to the graph is a strong assumption: graph learning

Definition (Graph Laplacian)

The graph Laplacian of a graph $G = (\mathcal{V}, \mathcal{E})$ with weight matrix W and degree matrix D is the matrix $L = D - W$

Definition (Graph Fourier Transform)

Let $G = (\mathcal{V}, \mathcal{E})$ and $L = X\Lambda X^\top$ be the eigenvalue decomposition of its Laplacian matrix. The Graph Fourier Transform (GFT) of a graph signal $y \in \mathbb{R}^p$ is given by

$$h = X^\top y$$

Problem Statement

Goal: Learn the Laplacian L that best explains the structure of n graph signals $Y = [y^{(1)}, \dots, y^{(n)}]$ of size N.

- Need for structural assumptions that link L to Y

Problem Statement

Goal: Learn the Laplacian L that best explains the structure of n graph signals $Y = [y^{(1)}, \ldots, y^{(n)}]$ of size N.

- Need for structural assumptions that link L to Y

Assumptions:

- G is undirected and has a single connected component

Problem Statement

Goal: Learn the Laplacian L that best explains the structure of n graph signals $Y = [y^{(1)}, \ldots, y^{(n)}]$ of size N.

- Need for structural assumptions that link L to Y

Assumptions:

- G is undirected and has a single connected component
- The graph signals are **smooth** with respect to G i.e. $y^{(i)\mathsf{T}} L y^{(i)} = \frac{1}{2} \sum w_{kl} (y_k^{(i)} - y_l^{(i)})^2$ is small

Problem Statement

Goal: Learn the Laplacian L that best explains the structure of n graph signals $Y = [y^{(1)}, \dots, y^{(n)}]$ of size N.

- Need for structural assumptions that link L to Y

Assumptions:

- G is undirected and has a single connected component
- The graph signals are **smooth** with respect to G i.e. $y^{(i)\mathsf{T}} L y^{(i)} = \frac{1}{2} \sum w_{kl} (y_k^{(i)} - y_l^{(i)})^2$ is small
- They have a **bandlimited** spectrum i.e. $\forall i$, $h^{(i)} = X^{\mathsf{T}} y^{(i)}$ has some zero-valued coefficients at same dimensions

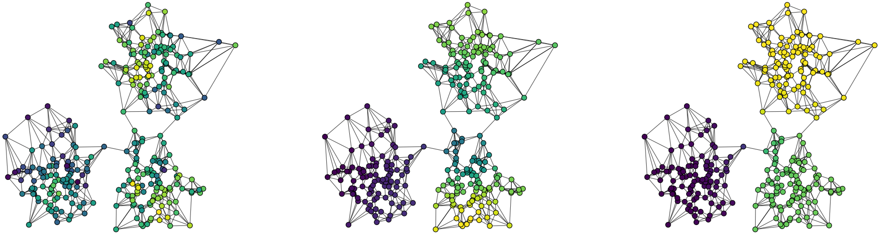
Problem Statement

Goal: Learn the Laplacian L that best explains the structure of n graph signals $Y = [y^{(1)}, \dots, y^{(n)}]$ of size N.

- Need for structural assumptions that link L to Y

Assumptions:

- G is undirected and has a single connected component
- The graph signals are **smooth** with respect to G i.e. $y^{(i)\intercal} L y^{(i)} = \frac{1}{2} \sum w_{kl}(y_k^{(i)} - y_l^{(i)})^2$ is small
- They have a **bandlimited** spectrum i.e. $\forall i,\ h^{(i)} = X^\intercal y^{(i)}$ has some zero-valued coefficients at same dimensions
  $\rightarrow$ Basic assumption of sampling methods (Chen *et al.* 2015 [2])
  $\rightarrow$ Different notion of smoothness
  $\rightarrow$ Also relies to the cluster structure of the graph (Sardellitti *et al.* 2019 [18])



(a) (b) (c)

Figure: Three smooth graph signals ($N = 300$) with decreasing bandlimitedness: (a) 150-sparse, (b) 6-sparse, (c) 3-sparse.

Optimization program

$$\min_{H,X,\Lambda} \|Y - XH\|_F^2 + \alpha\|\Lambda^{1/2}H\|_F^2 + \beta\|H\|_S$$

$$\text{s.t.} \quad \begin{cases} X^{\mathsf{T}}X = I_N,\, x_1 = \frac{1}{\sqrt{N}}\mathbf{1}_N & \text{(a)} \\ (X\Lambda X^{\mathsf{T}})_{k,\ell} \leq 0 \quad k \neq \ell & \text{(b)} \\ \Lambda = \mathrm{diag}(0, \lambda_2, \ldots, \lambda_N) \succeq 0 & \text{(c)} \\ \mathrm{tr}(\Lambda) = N \in \mathbb{R}_*^+ & \text{(d)} \end{cases}$$

- Learn $X\Lambda X^{\mathsf{T}}$ instead of L
- Y are assumed to be noisy version of some true graph vectors XH
- H stands for the graph Fourier transform of the true graph vectors

Optimization program

$$\min_{H,X,\Lambda} \|Y - XH\|_F^2 + \alpha\|\Lambda^{1/2}H\|_F^2 + \beta\|H\|_S$$

$$\text{s.t.} \quad \begin{cases} X^\mathsf{T} X = I_N,\, x_1 = \frac{1}{\sqrt{N}}\mathbf{1}_N & \text{(a)} \\ (X\Lambda X^\mathsf{T})_{k,\ell} \leq 0 \quad k \neq \ell & \text{(b)} \\ \Lambda = \mathrm{diag}(0, \lambda_2, \ldots, \lambda_N) \succeq 0 & \text{(c)} \\ \mathrm{tr}(\Lambda) = N \in \mathbb{R}_*^+ & \text{(d)} \end{cases}$$

- $\|Y - XH\|_F^2$ stands for the reconstruction error
- $\|\Lambda^{1/2}H\|_F^2$ controls the smoothness of the approximation XH
- $\|H\|_S = \|H\|_{2,0}$ or $\|H\|_{2,1}$ enforces the GFT to be 0 at the same dimensions
- α and β are positive hyperparameters

Optimization program

$$\min_{H,X,\Lambda} \|Y - XH\|_F^2 + \alpha \|\Lambda^{1/2} H\|_F^2 + \beta \|H\|_S$$

$$\text{s.t.} \quad \begin{cases} X^\top X = I_N, x_1 = \frac{1}{\sqrt{N}} \mathbf{1}_N & \text{(a)} \\ (X \Lambda X^\top)_{k,\ell} \leq 0 \quad k \neq \ell & \text{(b)} \\ \Lambda = \mathrm{diag}(0, \lambda_2, \ldots, \lambda_N) \succeq 0 & \text{(c)} \\ \mathrm{tr}(\Lambda) = N \in \mathbb{R}_*^+ , & \text{(d)} \end{cases}$$

- $\|Y - XH\|_F^2$ stands for the reconstruction error
- $\|\Lambda^{1/2} H\|_F^2$ controls the smoothness of the approximation XH
- $\|H\|_S = \|H\|_{2,0}$ or $\|H\|_{2,1}$ enforces the GFT to be 0 at the same dimensions
- α and β are positive hyperparameters

Optimization program

$$\min_{H,X,\Lambda} \|Y - XH\|_F^2 + \alpha\|\Lambda^{1/2}H\|_F^2 + \beta\|H\|_S$$

$$\text{s.t.} \quad \begin{cases} X^{\mathsf{T}}X = I_N, x_1 = \frac{1}{\sqrt{N}}\mathbf{1}_N & \text{(a)} \\ (X\Lambda X^{\mathsf{T}})_{k,\ell} \leq 0 \quad k \neq \ell & \text{(b)} \\ \Lambda = \operatorname{diag}(0, \lambda_2, \dots, \lambda_N) \succeq 0 & \text{(c)} \\ \operatorname{tr}(\Lambda) = N \in \mathbb{R}_*^+ & \text{(d)} \end{cases}$$

- $\|Y - XH\|_F^2$ stands for the reconstruction error
- $\|\Lambda^{1/2}H\|_F^2$ controls the smoothness of the approximation XH
- $\|H\|_S = \|H\|_{2,0}$ or $\|H\|_{2,1}$ enforces the GFT to be 0 at the same dimensions
- α and β are positive hyperparameters that controls smoothness and bandlimitedness

Optimization program

$$\min_{H,X,\Lambda} \|Y - XH\|_F^2 + \alpha\|\Lambda^{1/2}H\|_F^2 + \beta\|H\|_S$$

$$\text{s.t.} \quad \begin{cases} X^\mathsf{T}X = I_N,\ x_1 = \frac{1}{\sqrt{N}}\mathbf{1}_N & \text{(a)} \\ (X\Lambda X^\mathsf{T})_{k,\ell} \leq 0 \quad k \neq \ell & \text{(b)} \\ \Lambda = \operatorname{diag}(0, \lambda_2, \ldots, \lambda_N) \succeq 0 & \text{(c)} \\ \operatorname{tr}(\Lambda) = N \in \mathbb{R}_*^+ & \text{(d)} \end{cases}$$

- $\|Y - XH\|_F^2$ stands for the reconstruction error
- $\|\Lambda^{1/2}H\|_F^2$ controls the smoothness of the approximation XH
- $\|H\|_S = \|H\|_{2,0}$ or $\|H\|_{2,1}$ enforces the GFT to be 0 at the same dimensions
- α and β are positive hyperparameters that controls smoothness and bandlimitedness
- (a), (b) and (c) ensure $X\Lambda X^\mathsf{T}$ to be a Laplacian
- (d) makes sure the graph has edges

Optimization program

$$\min_{H,X,\Lambda} \|Y - XH\|_F^2 + \alpha\|\Lambda^{1/2}H\|_F^2 + \beta\|H\|_S$$

$$\text{s.t.} \quad \begin{cases} X^\mathsf{T} X = I_N,\, x_1 = \frac{1}{\sqrt{N}}\mathbf{1}_N & \text{(a)} \\ (X\Lambda X^\mathsf{T})_{k,\ell} \leq 0 \quad k \neq \ell & \text{(b)} \\ \Lambda = \mathrm{diag}(0, \lambda_2, \ldots, \lambda_N) \succeq 0 & \text{(c)} \\ \mathrm{tr}(\Lambda) = N \in \mathbb{R}_*^+ & \text{(d)} \end{cases}$$

- $\|Y - XH\|_F^2$ stands for the reconstruction error
- $\|\Lambda^{1/2}H\|_F^2$ controls the smoothness of the approximation XH
- $\|H\|_S = \|H\|_{2,0}$ or $\|H\|_{2,1}$ enforces the GFT to be 0 at the same dimensions
- α and β are positive hyperparameters that controls smoothness and bandlimitedness
- (a), (b) and (c) ensure $X\Lambda X^\mathsf{T}$ to be a Laplacian
- (d) makes sure the graph has edges

Solving the program (overview)

- Optimization program not convex + very difficult to updates all variables directly
  $\longrightarrow$ Use block-coordinate descent
- Other problem: constraint (b) $(X\Lambda X^{\top})_{kl} \leq 0$ difficult to handle at the X-step
  $\longrightarrow$ Solution: **IGL-3SR** and **FGL-3SR** [Le Bars *et al.*, ICASSP 2019, Humbert *et al.*, JMLR 2021]
- Both relax (b) and use block-coordinate descent over X, Λ and H

$$\min_{H,X,\Lambda} \|Y - XH\|_F^2 + \alpha\|\Lambda^{1/2}H\|_F^2 + \beta\|H\|_S$$

$$\text{s.t.} \quad \begin{cases} X^{\top}X = I_N, x_1 = \frac{1}{\sqrt{N}}\mathbf{1}_N & \text{(a)} \\ (X\Lambda X^{\top})_{k,\ell} \leq 0 \quad k \neq \ell & \text{(b)} \\ \Lambda = \operatorname{diag}(0, \lambda_2, \ldots, \lambda_N) \succeq 0 & \text{(c)} \\ \operatorname{tr}(\Lambda) = N \in \mathbb{R}_*^+ & \text{(d)} \end{cases}$$

Solving the program (overview)

- Optimization program not convex + very difficult to updates all variables directly
  $\longrightarrow$ Use block-coordinate descent
- Other problem: constraint (b) $(X\Lambda X^{\top})_{kl} \leq 0$ difficult to handle at the X-step
  $\longrightarrow$ Solution: **IGL-3SR** and **FGL-3SR** [Le Bars *et al.*, ICASSP 2019, Humbert *et al.* JMLR 2021]
- Both relax (b) and use block-coordinate descent over X, Λ and H

IGL-3SR

Relaxation: use a log-barrier function to put (b) in the objective

+ Each sub-problem is solvable using known techniques
+ Decrease at each step and stays in the constraint set
+ Iterates are ensured to converge
- High complexity

FGL-3SR

Relaxation: get rid of (b), only at the X-step

+ Lower complexity
+ 2/3 steps has closed-form
+ Returns a Laplacian even with the relaxation
- Objective function value can increase

Synthetic data

- Large simulation study in [Le Bars *et al.*, ICASSP 2019, Humbert *et al.*, JMLR 2021]
- True graphs: Random Geometric or Erdös-Renyi
- Y sampled via factor analysis model
- Comparison with two GSP baselines:
  $\rightarrow$ GL-SigRep (Dong *et al.* 2016 [5]): Only smoothness
  $\rightarrow$ ESA-GL (Sardellitti *et al.* 2019 [18]): Bandlimitedness

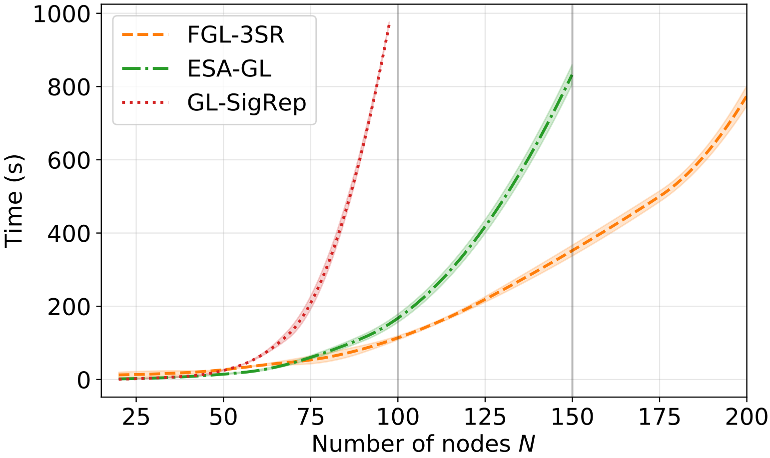
Synthetic data

- Large simulation study in [Le Bars *et al.*, ICASSP 2019, Humbert *et al.*, JMLR 2021]
- True graphs: Random Geometric or Erdös-Renyi
- Y sampled via factor analysis model
- Comparison with two GSP baselines:
  $\rightarrow$ GL-SigRep (Dong *et al.* 2016 [5]): Only smoothness
  $\rightarrow$ ESA-GL (Sardellitti *et al.* 2019 [18]): Bandlimitedness

Results and conclusion

- IGL-3SR outperforms baselines and FGL-3SR in terms of true graph recovery
- It is very slow, not practical for $\gtrsim$ 20 nodes
- FGL-3SR is a good compromise between graph recovery and time before convergence

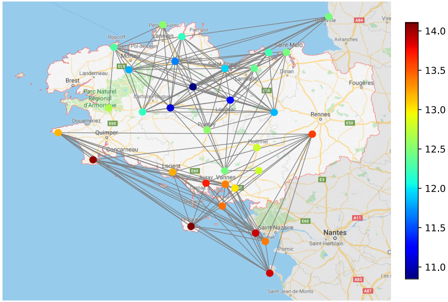
Synthetic data - Results



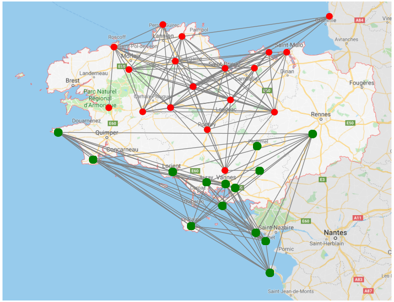
A real-world illustration

- Temperature data in Brittany (Chepuri *et al.* 2017 [4])
- $N = 32$ weather station
- spectral clustering to ascess the quality

- $n = 744$ measurements
- $\alpha = 10^{-4}$, β s.t 2-bandlimited



(a) A measurement example and the learned graph.



(b) Spectral clustering with the learned graph.

- Coherent with the spatial distribution. Splits the north from the south of Brittany

Part 3

-

Detecting changes in the graph structure of a varying Ising model

Background

Context

- Probabilistic modeling, the data come from a Markov Random Field (MRF)
- Binary vector data: **Ising model**
- Change-point detection with *unknown* number of change-points
- Related works:
  - Detection in Gaussian graphical models (Gibberd and Nelson, 2017 [8])
  - Detection in Ising with *known* number of change-points (Roy *et al.* 2017 [17])

Background

Context

- Probabilistic modeling, the data come from a Markov Random Field (MRF)
- Binary vector data: **Ising model**
- Change-point detection with *unknown* number of change-points
- Related works:
  - Detection in Gaussian graphical models (Gibberd and Nelson, 2017 [8])
  - Detection in Ising with *known* number of change-points (Roy *et al.* 2017 [17])

Ising model

Let $G = (V, E)$ and $\Omega \in \mathbb{R}^{p \times p}$ **symmetric** whose non-zero elements correspond to the set of edges E. The probability distribution function (pdf) of an Ising random vector X:

$$\mathbb{P}_\Omega(X = x) = \frac{1}{Z(\Omega)} \exp\left\{\sum_{a<b} x_a x_b \omega_{ab}\right\}$$

- $Z(\Omega)$: Normalizing constant
- $x \in \{-1, 1\}^p$

Model and objectives

Piece-wise constant Ising model

- Time-series of n independent Ising vectors $X^{(i)}$ with parameter $\Omega^{(i)}$
- Piecewise constant evolving structure:

$$\Omega^{(i)} = \sum_{k=0}^{D} \Theta^{(k+1)} \mathbf{1}\{T_k \leq i < T_{k+1}\}$$

$T_0 = 1$ and $T_{D+1} = n + 1$.

- D change-points appearing a time $T_1, \ldots, T_D$
- $D + 1$ sub-model parametrized by $\Theta^{(1)}, \ldots, \Theta^{(D+1)}$

Model and objectives

Piece-wise constant Ising model

- Time-series of n independent Ising vectors $X^{(i)}$ with parameter $\Omega^{(i)}$
- Piecewise constant evolving structure:

$$\Omega^{(i)} = \sum_{k=0}^{D} \Theta^{(k+1)} \mathbf{1}\{T_k \leq i < T_{k+1}\}$$

$T_0 = 1$ and $T_{D+1} = n + 1$.

- D change-points appearing a time $T_1, \ldots, T_D$
- $D + 1$ sub-model parametrized by $\Theta^{(1)}, \ldots, \Theta^{(D+1)}$

Objectives:

- Learn for each $X^{(i)}$ its associated parameter $\Omega^{(i)}$
- Infer the number of change-points D and their time instances

Learning

- Can we use standard maximum likelihood approach ?
  $\rightarrow$ No, due to the intractability of $Z(\cdot)$ and the high-dimensional scenario

- Instead, penalized **neighborhood selection** strategy: **TVI-FL** [Le Bars *et al.*, ICML 2020]

TVI-FL

For each node $j = 1, \ldots, p$, we solve

$$\widehat{\omega}_j = \underset{\omega \in \mathbb{R}^{p-1 \times n}}{\operatorname{argmin}} \; \mathcal{L}_j(\omega) + pen_{\lambda_1, \lambda_2}(\omega)$$

- A column $\widehat{\omega}_j^{(i)}$ of $\widehat{\omega}_j$ corresponds to the j-th row/column of $\widehat{\Omega}^{(i)}$
  $\longrightarrow$ The neighborhood's weights of node j at time i

Learning

TVI-FL

For each node $j = 1, \ldots, p$, we solve

$$\widehat{\omega}_j = \underset{\omega \in \mathbb{R}^{p-1 \times n}}{\operatorname{argmin}} \; \mathcal{L}_j(\omega) + pen_{\lambda_1, \lambda_2}(\omega)$$

$$\begin{aligned}\mathcal{L}_j(\omega) &\triangleq -\sum_{i=1}^{n} \log\left(\mathbb{P}_{\omega^{(i)}}(x_j^{(i)} | x_{\setminus j}^{(i)})\right) \\ &= \sum_{i=1}^{n} \log\left\{\exp\left(\omega^{(i)\top} x_{\setminus j}^{(i)}\right) + \exp\left(-\omega^{(i)\top} x_{\setminus j}^{(i)}\right)\right\} - \sum_{i=1}^{n} x_j^{(i)} \omega^{(i)\top} x_{\setminus j}^{(i)}\end{aligned}$$

- Conditional log-likelihood of node j knowing the other nodes values
- Convex function

Learning

TVI-FL

For each node $j = 1, \ldots, p$, we solve

$$\widehat{\omega}_j = \underset{\omega \in \mathbb{R}^{p-1 \times n}}{\operatorname{argmin}} \; \mathcal{L}_j(\omega) + pen_{\lambda_1,\lambda_2}(\omega)$$

$$pen_{\lambda_1,\lambda_2}(\omega) \triangleq \lambda_1 \sum_{i=2}^{n} \|\omega^{(i)} - \omega^{(i-1)}\|_2 + \lambda_2 \sum_{i=1}^{n} \|\omega^{(i)}\|_1$$

- λ_1 and λ_2 are positive hyperparameters
- The first term - **fused penalty** - controls the piece-wise constant structure and the number of change-points
- The second term - **lasso penalty** - imposes sparsity in the learnt neighborhood

Learning

TVI-FL

For each node $j = 1, \dots, p$, we solve

$$\widehat{\omega}_j = \underset{\omega \in \mathbb{R}^{p-1 \times n}}{\operatorname{argmin}} \; \mathcal{L}_j(\omega) + pen_{\lambda_1,\lambda_2}(\omega)$$

$$pen_{\lambda_1,\lambda_2}(\omega) \triangleq \lambda_1 \sum_{i=2}^{n} \|\omega^{(i)} - \omega^{(i-1)}\|_2 + \lambda_2 \sum_{i=1}^{n} \|\omega^{(i)}\|_1$$

- λ_1 and λ_2 are positive hyperparameters
- The first term - **fused penalty** - controls the piece-wise constant structure and the number of change-points
- The second term - **lasso penalty** - imposes sparsity in the learnt neighborhood

In conclusion:

- Non-differentiable but **convex function**
- TVI-FL solvable by convex programming tools and software
- Set of estimated change-points : $\widehat{\mathcal{D}} = \left\{ \widehat{T}_k \in \{2, \dots, n\} : \|\widehat{\omega}_j^{(\widehat{T}_k)} - \widehat{\omega}_j^{(\widehat{T}_k - 1)}\|_2 \neq 0 \right\}$

Theoretical analysis

Assumptions:

- **(A1)** $\exists \phi_{\min} > 0$ and $\phi_{\max} < \infty$ s.t. $\phi_{\min} \leq \Lambda_{\min}\left(\mathbb{E}_{\Theta^{(k)}}[X_{\setminus j}X_{\setminus j}^\top]\right)$ and $\phi_{\max} \geq \Lambda_{\max}\left(\mathbb{E}_{\Theta^{(k)}}[X_{\setminus j}X_{\setminus j}^\top]\right)$
- **(A2)** There exists $M \geq 0$ s.t. $\max_{k\in[D+1]}\|\theta_j^{(k)}\|_2 \leq M$
- **(A3)** For all $k = 1, \ldots, D$, $T_k = \lfloor n\tau_k \rfloor$ with unknown $\tau_k \in [0, 1]$

Theoretical analysis

Assumptions:

- **(A1)** $\exists \phi_{\min} > 0$ and $\phi_{\max} < \infty$ s.t. $\phi_{\min} \leq \Lambda_{\min}\left(\mathbb{E}_{\Theta^{(k)}}[X_{\setminus j} X_{\setminus j}^\top]\right)$ and $\phi_{\max} \geq \Lambda_{\max}\left(\mathbb{E}_{\Theta^{(k)}}[X_{\setminus j} X_{\setminus j}^\top]\right)$
- **(A2)** There exists $M \geq 0$ s.t. $\max_{k \in [D+1]} \|\theta_j^{(k)}\|_2 \leq M$
- **(A3)** For all $k = 1, \ldots, D$, $T_k = \lfloor n\tau_k \rfloor$ with unknown $\tau_k \in [0, 1]$

Theorem - Change-Point consistency

Consider (A1-A3) and let $\{\delta_n\}_{n\geq 1}$ be a non-increasing sequence that converges to 0 and s.t. $n\delta_n \to \infty$.
If $\widehat{D} = D$, we have:

$$\mathbb{P}\left(\max_{k=1,\ldots,D} |\hat{T}_k - T_k| \leq n\delta_n\right) \underset{n\to\infty}{\longrightarrow} 1$$

- Drawback: $\widehat{D} = D$ difficult to verify

Change-Point consistency 2

- $d(A \| B) = \sup_{b \in B} \inf_{a \in A} |b - a|$

Proposition

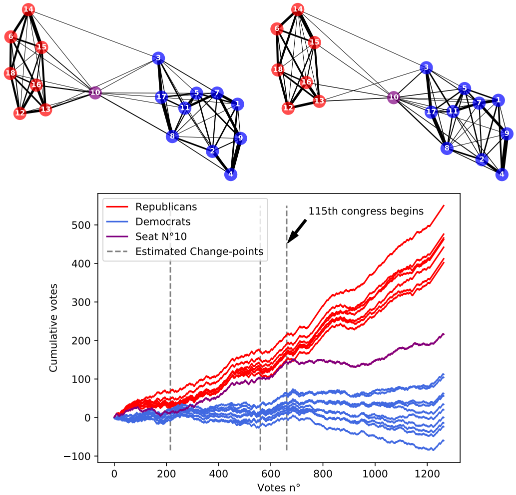
Under the same conditions, if $D \leq \widehat{D}$ then:

$$\mathbb{P}(d(\widehat{\mathcal{D}} \| \mathcal{D}) \leq n\delta_n) \underset{n \to \infty}{\longrightarrow} 1$$

- Overestimated number of change-points
- Asymptotically, all the true change-points belong to the estimated set of change-points

Voting data set

- Votes (yes/no) in Illinois house of representatives (Lewis *et al.* 2020 [13])
- 18 seats $\rightarrow$ 18 nodes
- 1264 votes
- 114-th and 115-th US Congresses (2015-2019)
- λ_1 and λ_2 minimizing AIC



Results:

- Party structure: Republican vs Democrat
- Biggest change-point: End of congress
- Seat 10 change party
- Brings knowledge: seat 10 is a *super-collaborator*

Conclusion

Conclusion

A diverse work ...

- Anomaly detection, change-point detection, graph learning, optimization
- GSP framework, probabilistic framework
- Not discussed: robust kernel density estimation [Le Bars *et al.*, 2020]
- Codes available online at `github.com/BatisteLB`

... with open questions

- Online version for change-point detection of part 3
- Better theoretical understanding: consistent graph recovery?
- Improve optimization of part 2 and 3
- Make a better use of the graph in part 1

What about my postdoc?

Fully decentralized federated learning

- Decentralized algorithms depend on a graph topology
  $\rightarrow$ also impacts the convergence!
- Impact increases when data are non iid
- Objective: learning data-dependent graphs that can speed-up convergence

Learning with privacy

- Learning graphs under privacy constraints
- Privately learning the graph proposed above
- Markov Random Fields inference under (local) differential privacy

Publications and preprints

- B. Le Bars, and A. Kalogeratos. A Probabilistic Framework to Node-level Anomaly Detection in Communication Networks. *In 2019 IEEE Conference on Computer Communications (INFOCOM)*, 2019
- B. Le Bars, P. Humbert, L. Oudre, and A. Kalogeratos. Learning Laplacian Matrix from Bandlimited Graph Signals. *In 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019
- B. Le Bars, P. Humbert, A. Kalogeratos, and N. Vayatis. Learning the piece-wise constant graph structure of a varying Ising model. *In 2020 International Conference on Machine Learning (ICML)*, 2020
- B. Le Bars, P. Humbert, L. Minvielle, and N. Vayatis. Robust Kernel Density Estimation with Median-of-Means principle. *Arxiv preprint*, 2020
- P. Humbert, B. Le Bars, L. Oudre, A. Kalogeratos, and N. Vayatis. Learning Laplacian Matrix from Graph Signals with Sparse Spectral Representation. *In Journal of Machine Learning Research (JMLR)*, 2021

References

[1] Bybee, L. and Atchadé, Y. (2018). Change-point computation for large graphical models: a scalable algorithm for gaussian graphical models with change-points. *J. of Machine Learning Research*, 19(1):440–477.

[2] Chen, S., Sandryhaila, A., and Kovačević, J. (2015). Sampling theory for graph signals. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3392–3396.

[3] Chen, Y., Mao, X., Ling, D., and Gu, Y. (2018). Change-point detection of gaussian graph signals with partial information. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3934–3938. IEEE.

[4] Chepuri, S. P., Liu, S., Leus, G., and Hero, A. O. (2017). Learning sparse graphs under smoothness prior. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6508–6512.

[5] Dong, X., Thanou, D., Frossard, P., and Vandergheynst, P. (2016). Learning laplacian matrix in smooth graph signal representations. *Trans. Signal Processing*, 64(23):6160–6173.

[6] Egilmez, H. E. and Ortega, A. (2014). Spectral anomaly detection using graph-based filtering for wireless sensor networks. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1085–1089. IEEE.

[7] Friedman, J., Hastie, T., and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441.

[8] Gibberd, A. J. and Nelson, J. D. (2017). Regularized estimation of piecewise constant gaussian graphical models: The group-fused graphical lasso. *Journal of Computational and Graphical Statistics*, 26(3):623–634.

[9] Goel, S., Kane, D. M., and Klivans, A. R. (2019). Learning ising models with independent failures. In *Conference on Learning Theory*, pages 1449–1469.

[10] Hong, Y. (2013). On computing the distribution function for the poisson binomial distribution. *Computational Statistics & Data Analysis*, 59:41–51.

[11] Ji, T., Yang, D., and Gao, J. (2013). Incremental local evolutionary outlier detection for dynamic social networks. In *Proc. of the Joint European Conf. on Machine Learning and Knowledge Discovery in Databases*, pages 1–15. Springer.

References

[12] Kolar, M., Song, L., Ahmed, A., and Xing, E. P. (2010). Estimating time-varying networks. *The Annals of Applied Statistics*, 4(1):94–123.

[13] Lewis, J. B., Poole, K., Rosenthal, H., Boche, A., Rudkin, A., and Sonnet, L. (2020). Voteview: Congressional roll-call votes database. *https://voteview.com/*.

[14] Londschien, M., Kovács, S., and Bühlmann, P. (2020). Change point detection for graphical models in the presence of missing values. *Journal of Computational and Graphical Statistics*, pages 1–32.

[15] Neil, J., Hash, C., Brugh, A., Fisk, M., and Storlie, C. (2013). Scan statistics for the online detection of locally anomalous subgraphs. *Technometrics*, 55(4):403–414.

[16] Ravikumar, P., Wainwright, M. J., and Lafferty, J. D. (2010). High-dimensional ising model selection using l1-regularized logistic regression. *The Annals of Statistics*, 38(3):1287–1319.

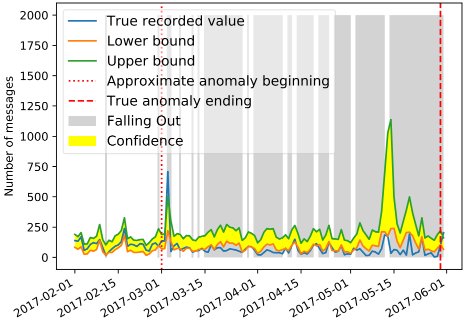
[17] Roy, S., Atchadé, Y., and Michailidis, G. (2017). Change point estimation in high dimensional markov random-field models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(4):1187–1206.

[18] Sardellitti, S., Barbarossa, S., and Di Lorenzo, P. (2019). Graph topology inference based on sparsifying transform learning. *IEEE Transactions on Signal Processing*, 67(7):1712–1727.

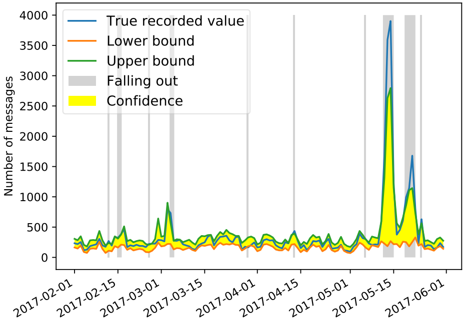
[19] Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., and Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471.

[20] Yamada, K., Tanaka, Y., and Ortega, A. (2020). Time-varying graph learning with constraints on graph temporal variation. *arXiv preprint arXiv:2001.03346*.

Results bilateral



(a) Abnormal Base Station - Bilateral intervals



(b) Normal Base Station - Bilateral intervals

FGL-3SR

***H*-step**

$$\min_{H} \|Y - XH\|_F^2 + \alpha\|\Lambda^{1/2}H\|_F^2 + \beta\|H\|_S$$

- No constraint
- Equivalent to multiple sparse linear regression problems
- Closed-form solutions
- $\|\cdot\|_S = \|\cdot\|_{2,0}$: hard-thresholding
- $\|\cdot\|_S = \|\cdot\|_{2,1}$: solf-thresholding

FGL-3SR

X-step

$$\min_X \|Y - XH\|_F^2 \qquad \text{s.t.} \quad X^\top X = I_N,\ x_1 = \frac{1}{\sqrt{N}}\mathbf{1}_N \qquad \text{(a)}$$

- (b) is out
- Non-convex but has a closed-form:

$$X^{(t+1)} = X^{(t)} \begin{bmatrix} 1 & \mathbf{0}_{N-1}^\top \\ \mathbf{0}_{N-1} & PQ^\top \end{bmatrix},$$

where the columns in P and Q are the left- and right-singular vectors of $(X^{(t+1)\top} YH^\top)_{2:,2:}$

FGL-3SR

Λ-step

$$\min_{\Lambda}\ \alpha\ \underbrace{\mathrm{tr}(HH^{\mathsf{T}}\Lambda)}_{\|\Lambda^{1/2}H\|_F^2} \qquad \text{s.t.} \quad \begin{cases} (X\Lambda X^{\mathsf{T}})_{i,j} \leq 0 \quad i \neq j\ , & \text{(b)} \\ \Lambda = \mathrm{diag}(0, \lambda_2, \ldots, \lambda_N) \succeq 0\ , & \text{(c)} \\ \mathrm{tr}(\Lambda) = N \in \mathbb{R}_*^+\ , & \text{(d)} \end{cases}$$

- (b) is back
- Linear program: can be solved via solvers
- Property: for all X that satisfies (a), there exist Λ that satisfies (b), (c) and (d)
  $\longrightarrow$ Need to finish by this step

Synthetic data graph learning - Results

N	Metrics	RG graph model IGL-3SR	RG graph model FGL-3SR	RG graph model ESA-GL	RG graph model GL-SigRep	ER graph model IGL-3SR	ER graph model FGL-3SR	ER graph model ESA-GL	ER graph model GL-SigRep
20	F_1-measure	**0.97 (±0.03)**	**0.97 (±0.03)**	0.93 (±0.03)	0.95 (±0.04)	**0.94 (±0.03)**	0.82 (±0.07)	**0.94 (±0.04)**	0.78 (±0.07)
	$\rho(L, \hat{L})$	**0.94 (±0.05)**	0.90 (±0.03)	0.92 (±0.05)	0.79 (±0.04)	**0.92 (±0.03)**	0.73 (±0.06)	0.90 (±0.04)	0.20 (±0.07)
	Time	< 1min	< 10s	< **5s**	< **5s**	< 1min	< 10s	< **5s**	< **5s**
50	F_1-measure	**0.90 (±0.01)**	0.81 (±0.02)	0.87 (±0.04)	0.75 (±0.01)	0.81 (±0.02)	0.76 (±0.03)	**0.84 (±0.02)**	0.61 (±0.03)
	$\rho(L, \hat{L})$	**0.86 (±0.02)**	0.74 (±0.03)	0.83 (±0.03)	0.55 (±0.02)	0.78 (±0.03)	0.73 (±0.02)	**0.82 (±0.06)**	0.06 (±0.01)
	Time	< 17mins	< **40s**	< 60s	< **40s**	< 17mins	< **40s**	< 60s	< **40s**
100	F_1-measure	**0.73 (±0.03)**	0.64 (±0.01)	0.70 (±0.01)	–	**0.62 (±0.01)**	0.59 (±0.02)	0.59 (±0.02)	–
	$\rho(L, \hat{L})$	**0.61 (±0.04)**	0.48 (±0.01)	0.60 (±0.03)	–	0.55 (±0.02)	0.51 (±0.022)	**0.64 (±0.02)**	–
	Time	< 50mins	< **2mins**	< 4mins	–	< 50mins	< **2mins**	< 4mins	–

Synthetic data

- $n = 100$, $p = 20$, 2 Change-Points
- Random Regular Graphs with degree $\in \{2, 3, 4\}$
- Competitor: Tesla (Kolar *et al.* [12])
- Metrics, F_1-score and h-score:

$$h(\mathcal{D}, \widehat{\mathcal{D}}) \triangleq \frac{1}{n} \max \left\{ \max_{t \in \mathcal{D}} \min_{\hat{t} \in \widehat{\mathcal{D}}} |t - \hat{t}|, \max_{\hat{t} \in \widehat{\mathcal{D}}} \min_{t \in \mathcal{D}} |t - \hat{t}| \right\}.$$

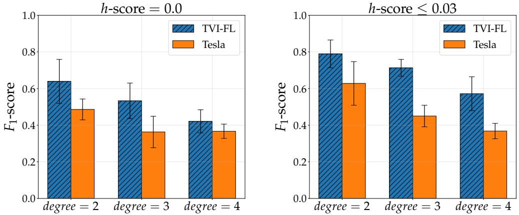


Figure: Average F_1-score obtained when the h-score is below a certain threshold.

- Outperforming Telsa, not designed for proper CP detection
- Complete results in the main paper

Sigfox data set TVI-FL

- Same data set as in part 1
- λ_1 and λ_2 selected via AIC
- Several change-points, but an important one around the 30th day

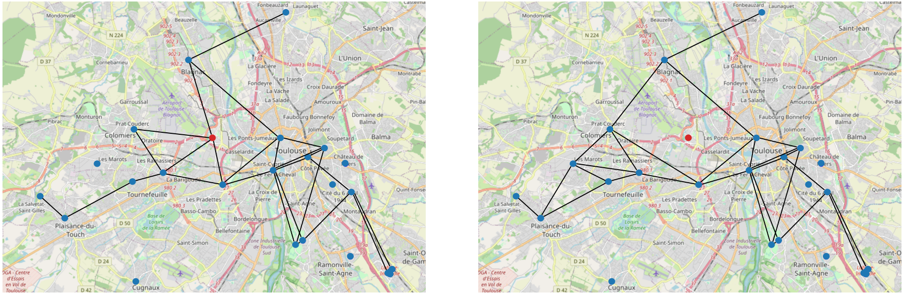


Figure: (Left) A graph learned before the BS failure, recorded on the 30th day. (Right) A graph learned after this day

Robust Kernel Density Estimator

Classical framework

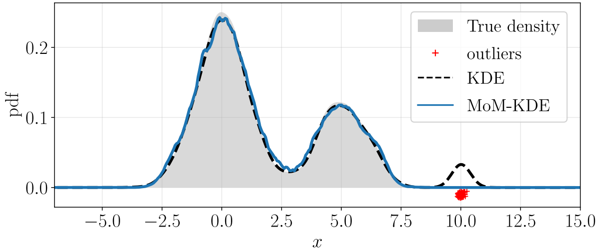
- $\{X_1, \dots, X_n\}$
- $\forall i = 1, \dots, n,\, X_i \sim f$
- Kernel Density Estimator (KDE):

$$\hat{f}_n(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{X_i - x}{h}\right)$$

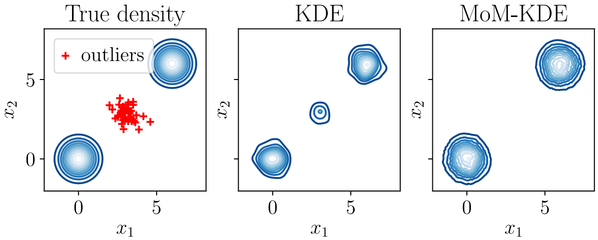
Outlier framework

- $\{X_1, \dots, X_n\} = \mathcal{O} \cup \mathcal{I}$
- $\forall i \in \mathcal{I},\, X_i \sim f$
- $B_1, \dots, B_S$: random partition of $[n]$
- $n_s = |B_s|$
- Median-of-Means KDE:
  $\hat{f}_{MoM}(x_0) \propto \text{Median}\left(\hat{f}_{n_1}(x_0), \dots, \hat{f}_{n_S}(x_0)\right)$

Robust Kernel Density Estimator



(a) One-dimensional



(b) Two-dimensional