BGU
Ben-Gurion University of the Negev

Sparsity-Aware Extended Kalman Filter for Tracking Dynamic Graphs

Lital Dabush, Nir Shlezinger & **Tirza Routtenberg**

School of Electrical and Computer Engineering
Ben-Gurion University of the Negev, Israel

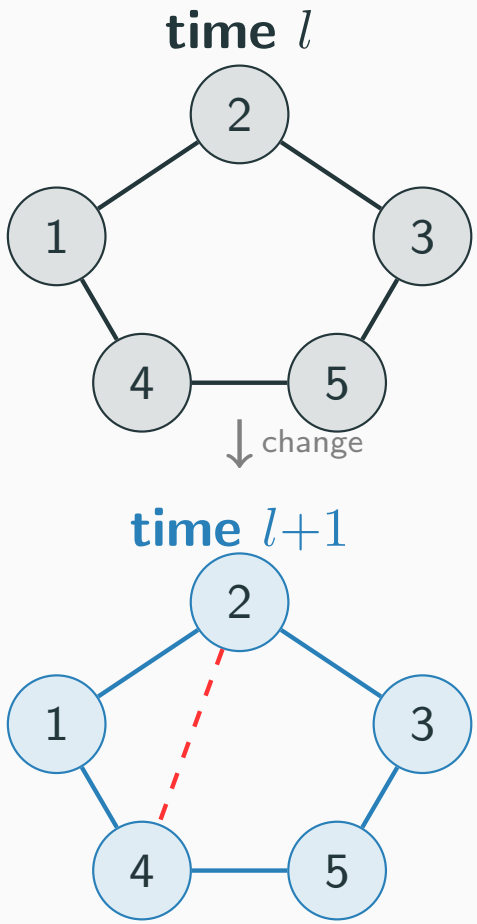
Outline

1. Motivation
2. Problem Formulation
3. Observability Analysis
4. GSP-EKF Algorithm
5. Numerical Results
6. Conclusions

Why Track Dynamic Graph Topologies?

Challenge: In many systems the **topology evolves over time**.

1. Power grids: line switching, impedance variation
2. Brain-machine interfaces: dynamic functional connectivity
3. Communication networks: link failures & additions
4. Traffic networks: congestion-dependent routing



Goal

Track the sparse, time-varying topology (i.e., Laplacian matrix) from **nodal graph-signal observations** (no direct topology measurements).

(Some of the) Existing Approaches

- **Stationary topology learning:** Graphical Lasso and smoothness-based graph learning ⇒ learns a fixed graph; not tracking
- **Linear dynamic topology models:** Time-varying Graphical Lasso[1] and SVAR-type models[2] ⇒ dynamic, but mostly 1st-order interactions; no polynomial graph-filter effects
- **Windowed dynamic graph learning:** temporally-regularized / sliding-window methods[3] ⇒ useful for smooth changes, but batch-based and slow to react to recent changes
- **Online RLS graph-filter methods:** Graph RLS[4] ⇒ online, but uses fixed forgetting; no explicit process/measurement noise model
- **Deep-learning methods:**[5] tracking dynamic topologies from data ⇒ require sufficient labeled events/training data

[1] Hallac, Park, Boyd, and Leskovec, 2017.
[2] Ioannidis, Shen, and Giannakis, 2019.
[3] Kalofolias, Loukas, Thanou, and Frossard, 2017.
[4] Ramezani-Mayami and Beferull-Lozano, 2017.
[5] Alippi and Zambon, 2023.

In this work

Gap addressed in this work

- **Nonlinear** polynomial Laplacian observations
- **Online**, sample-by-sample sequential tracking
- **Sparsity-aware** joint support + weight estimation

Kalman-type methods are natural candidates for online nonlinear tracking, but applying them to sparse dynamic graphs raises non-trivial challenges.

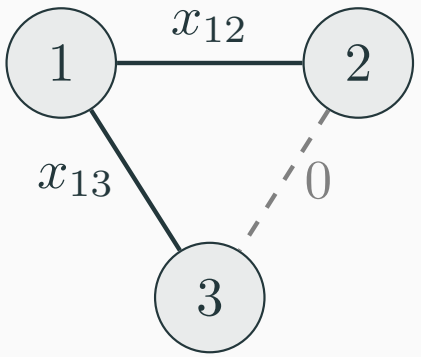
Notation: Weighted Undirected Graph

We consider a **weighted undirected graph**

$$\mathcal{G}_l = (\mathcal{V}, \xi_l), \qquad |\mathcal{V}| = N,$$

where N is the number of nodes and ξ_l is the time-varying edge set.

- The topology is represented by the graph Laplacian $\mathbf{L}_l \in \mathbb{R}^{N \times N}$
- $\mathbf{x}_l$ contains the edge weights of the **complete graph**; zero entries correspond to absent edges.
- $\mathbf{B} \in \mathbb{R}^{N \times N(N-1)/2}$ is the incidence matrix of the complete graph.



Example: $N = 3$
edge $(2, 3)$ absent

$$\mathbf{x}_l = \begin{bmatrix} x_{1,2} \\ x_{1,3} \\ x_{2,3} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & -1 \end{bmatrix}$$

$$\mathbf{L}_l = \mathbf{B}\, \mathrm{diag}(\mathbf{x}_l)\, \mathbf{B}^T$$

Signal Model & State-Space Formulation

- **Observed graph signal at time l:**

$$\mathbf{y}_l = h(\mathbf{L}_l)\,\mathbf{q}_l + \mathbf{v}_l, \qquad \mathbf{v}_l \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$$

  - $h(\cdot)$ - known polynomial Laplacian filter.
  - $\mathbf{y}_l, \mathbf{q}_l \in \mathbb{R}^N$ - observed (known) output and input signals.
  - $\mathbf{L}_l \in \mathbb{R}^{N \times N}$ - unknown, time-varying Laplacian.

Signal Model & State-Space Formulation

- **Observed graph signal at time l:**

$$\mathbf{y}_l = h(\mathbf{L}_l)\,\mathbf{q}_l + \mathbf{v}_l, \qquad \mathbf{v}_l \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$$

  - $h(\cdot)$ - known polynomial Laplacian filter.
  - $\mathbf{y}_l, \mathbf{q}_l \in \mathbb{R}^N$ - observed (known) output and input signals.
  - $\mathbf{L}_l \in \mathbb{R}^{N\times N}$ - unknown, time-varying Laplacian.

- **Incidence-matrix parameterization:**

$$\mathbf{L}_l = \mathbf{B}\ \mathrm{diag}(\mathbf{x}_l)\,\mathbf{B}^T, \qquad \mathbf{x}_l \in \mathbb{R}_+^{N(N-1)/2} \text{ (sparse)}$$

- **Nonlinear State-Space Model (SSM):**

$$\mathbf{x}_l = \mathbf{f}_l(\mathbf{x}_{l-1}) + \mathbf{e}_l, \qquad \mathbf{e}_l \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$$
$$\mathbf{y}_l = h\big(\mathbf{B}\,\mathrm{diag}(\mathbf{x}_l)\mathbf{B}^T\big)\,\mathbf{q}_l + \mathbf{v}_l$$

- **Latent state** = sparse edge-weight vector $\mathbf{x}_l$

Model Structure and Assumptions

Modeling Assumptions

- Noise covariances $\mathbf{Q}$, $\mathbf{R}$ are known
- **Non-negativity** $(\mathbf{x}_l \geq \mathbf{0})$: required for a valid $\mathbf{L}_l$; enforced post-hoc via clipping; covariance propagation ignores this constraint
- **Sparsity/Gaussianity:** sparse $\mathbf{x}_l$ is non-Gaussian; ℓ_1-regularization (update step only; prediction is Gaussian)

Model Structure and Assumptions

Modeling Assumptions

- Noise covariances $\mathbf{Q}$, $\mathbf{R}$ are known
- **Non-negativity** ($\mathbf{x}_l \geq \mathbf{0}$): required for a valid $\mathbf{L}_l$; enforced post-hoc via clipping; covariance propagation ignores this constraint
- **Sparsity/Gaussianity:** sparse $\mathbf{x}_l$ is non-Gaussian; ℓ_1-regularization (update step only; prediction is Gaussian)

Special Case - Linear SSM

For $P = 1$ using $\mathrm{diag}(\mathbf{a})\mathbf{b} = \mathrm{diag}(\mathbf{b})\mathbf{a}$ we obtain

$$\mathbf{x}_l = \mathbf{F}_l \mathbf{x}_{l-1} + \mathbf{e}_l,$$

$$\mathbf{y}_l = \underbrace{a_1 \mathbf{B}\, \mathrm{diag}(\mathbf{B}^T \mathbf{q}_l)}_{\mathbf{H}_l} \mathbf{x}_l + a_0 \mathbf{q}_l + \mathbf{v}_l$$

Observability of the Linear SSM

Observation		State
$\mathbf{y}_l,\, \mathbf{q}_l \in \mathbb{R}^N$	$\ll$	$\mathbf{x}_l \in \mathbb{R}^{N(N-1)/2}$
(N scalars per step)		($\sim N^2/2$ unknowns)

The T-step Observability Matrix

Stacking T consecutive output vectors $\mathbf{y}_l, \ldots, \mathbf{y}_{l+T-1} \in \mathbb{R}^N$ yields TN scalar equations in $\frac{N(N-1)}{2}$ unknowns:

$$\mathbf{O}_l^{(T)} \triangleq \begin{bmatrix} \mathbf{H}_l \\ \mathbf{H}_{l+1}\mathbf{F}_{l+1} \\ \vdots \\ \mathbf{H}_{l+T-1}\mathbf{F}_{l+T-1}\cdots\mathbf{F}_{l+1} \end{bmatrix} \in \mathbb{R}^{TN \times \frac{N(N-1)}{2}}$$

- TN rows (stacked observations)
- $\frac{N(N-1)}{2}$ columns (one per possible edge)

Observability: Theorem and Implications

Theorem 1 (T-step Observability)

T-step observability holds if and only if

$$\operatorname{rank}\left(\mathbf{O}_l^{(T)}\right) = \frac{N(N-1)}{2}$$

Corollary: A necessary condition is $T \geq \left\lceil \frac{N-1}{2} \right\rceil$.

Proof: $\mathbf{O}_l^{(T)} \in \mathbb{R}^{NT \times N\frac{(N-1)}{2}} \Rightarrow T \geq \frac{N-1}{2} \Rightarrow T \geq \left\lceil \frac{N-1}{2} \right\rceil$

Implication: Topology **cannot change faster** than once every $\left\lceil \frac{N-1}{2} \right\rceil$ steps (e.g. $N = 20 \Rightarrow T \geq 10$)

Solution: Under **sparsity** $|\xi_l| = K \ll \frac{N(N-1)}{2}$, compressed sensing requires only $TN = \mathcal{O}(K \log N)$ equations

$\Rightarrow$ a factor of $\frac{N}{2K}$ improvement over the unstructured case.

Extended Kalman Filter (EKF)

Standard EKF update solves:

$$\hat{\mathbf{x}}_l = \underset{\mathbf{x}}{\operatorname{argmin}} \underbrace{\|\mathbf{y}_l - \hat{\mathbf{y}}_{l|l-1} - \hat{\mathbf{H}}_l(\mathbf{x} - \hat{\mathbf{x}}_{l|l-1})\|^2_{\mathbf{R}^{-1}}}_{\text{linearized data fidelity}} + \underbrace{\|\mathbf{x} - \hat{\mathbf{x}}_{l|l-1}\|^2_{\hat{\mathbf{\Sigma}}^{-1}_{l|l-1}}}_{\text{Gaussian prior}},$$

- $\hat{\mathbf{x}}_{l|l-1}, \hat{\mathbf{y}}_{l|l-1}, \hat{\mathbf{\Sigma}}_{l|l-1}$ - estimated state, observation, & covariance
- $\hat{\mathbf{H}}_l = \nabla_{\mathbf{x}} \mathbf{h}_l(\hat{\mathbf{x}}_{l|l-1})$ - observation Jacobian matrix
- $\mathbf{R}$ - observation noise covariance.

Extended Kalman Filter (EKF)

Standard EKF update solves:

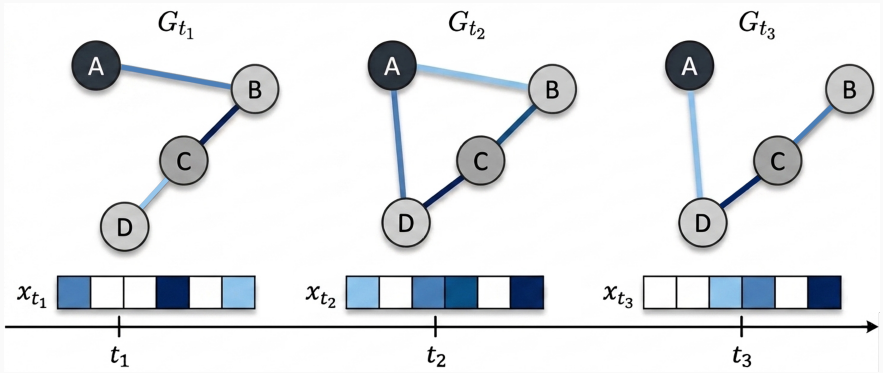
$$\hat{\mathbf{x}}_l = \underset{\mathbf{x}}{\operatorname{argmin}} \underbrace{\|\mathbf{y}_l - \hat{\mathbf{y}}_{l|l-1} - \hat{\mathbf{H}}_l(\mathbf{x} - \hat{\mathbf{x}}_{l|l-1})\|^2_{\mathbf{R}^{-1}}}_{\text{linearized data fidelity}} + \underbrace{\|\mathbf{x} - \hat{\mathbf{x}}_{l|l-1}\|^2_{\hat{\mathbf{\Sigma}}^{-1}_{l|l-1}}}_{\text{Gaussian prior}},$$

- $\hat{\mathbf{x}}_{l|l-1}, \hat{\mathbf{y}}_{l|l-1}, \hat{\mathbf{\Sigma}}_{l|l-1}$ - estimated state, observation, & covariance
- $\hat{\mathbf{H}}_l = \nabla_{\mathbf{x}} \mathbf{h}_l(\hat{\mathbf{x}}_{l|l-1})$ - observation Jacobian matrix
- $\mathbf{R}$ - observation noise covariance.

EKF uses the closed-form solution:

$$\hat{\mathbf{x}}_l = \hat{\mathbf{x}}_{l|l-1} + \mathbf{K}_l \Delta \mathbf{y}_l$$

No sparsity structure $\Rightarrow$ dense estimates



Sparsity-Regularized Update (ISTA-Update)

Proposed update: Add ℓ_1 regularization **within** the filter:

$$\hat{\mathbf{x}}_l = \underset{\mathbf{x}\in\mathbb{R}^{N(N-1)/2}}{\operatorname{argmin}} \; \phi_l(\mathbf{x}) + \underbrace{\mu\|\mathbf{x}\|_1}_{\text{sparsity-inducing regularization}}$$

Solved via **proximal gradient descent (ISTA)**; each iteration:

$$\hat{\mathbf{x}}_l^{(m+1)} = \mathcal{T}_{\mu\rho^{(m)}}\left(\hat{\mathbf{x}}_l^{(m)} - \rho^{(m)}\nabla_{\mathbf{x}}\phi_l(\hat{\mathbf{x}}_l^{(m)})\right)$$

$\mathcal{T}_\beta(x) = \operatorname{sign}(x)\max(0, |x| - \beta)$ is the soft-thresholding operator.

Sparsity-Regularized Update (ISTA-Update)

Proposed update: Add ℓ_1 regularization **within** the filter:

$$\hat{\mathbf{x}}_l = \underset{\mathbf{x}\in\mathbb{R}^{N(N-1)/2}}{\operatorname{argmin}} \; \phi_l(\mathbf{x}) + \underbrace{\mu\|\mathbf{x}\|_1}_{\text{sparsity-inducing regularization}}$$

Solved via **proximal gradient descent (ISTA)**; each iteration:

$$\hat{\mathbf{x}}_l^{(m+1)} = \mathcal{T}_{\mu\rho^{(m)}}\left(\hat{\mathbf{x}}_l^{(m)} - \rho^{(m)}\nabla_{\mathbf{x}}\phi_l(\hat{\mathbf{x}}_l^{(m)})\right)$$

$\mathcal{T}_\beta(x) = \operatorname{sign}(x)\max(0, |x| - \beta)$ is the soft-thresholding operator.

- **Warm start:** $\hat{\mathbf{x}}_l^{(0)} \leftarrow \underbrace{\hat{\mathbf{x}}_{l|l-1} + \mathbf{K}_l\Delta\mathbf{y}_l}_{\text{unregularized EKF}} \Rightarrow \nabla_{\mathbf{x}}\phi_l(\hat{\mathbf{x}}_l^{(0)}) = \mathbf{0}$
  $\Rightarrow$ 1st iter. = pure soft-thresholding
- $M_{\text{ISTA}} = 1$ is sufficient in practice + Woodbury identity avoids $\hat{\boldsymbol{\Sigma}}_{l|l-1}^{-1}$ inversion $\Rightarrow$ **low complexity**

ISTA-Update Algorithm

Algorithm 1: ISTA-Update at time l

Input: $\hat{\mathbf{x}}_{l|l-1}$, $\hat{\mathbf{\Sigma}}_{l|l-1}$, $\hat{\mathbf{H}}_l$, $\mathbf{K}_l$, $\mathbf{y}_l$, $\mathbf{q}_l$, μ, $\{\rho^{(m)}\}$

Initialize: $\hat{\mathbf{x}}_l^{(0)} \leftarrow$ unregularized EKF estimate

For $m = 0, 1, \ldots, M_{\text{ISTA}} - 1$:

1. Compute gradient $\nabla_{\mathbf{x}} \phi_l(\hat{\mathbf{x}}_l^{(m)})$ (closed form)
2. $\hat{\mathbf{x}}_l^{(m+1)} = \mathcal{T}_{\mu\rho^{(m)}}\left(\hat{\mathbf{x}}_l^{(m)} - \rho^{(m)} \nabla_{\mathbf{x}} \phi_l(\hat{\mathbf{x}}_l^{(m)})\right)$

Output: $\hat{\mathbf{x}}_l^{(M_{\text{ISTA}})}$

Closed-form gradient (key enabler):

$$\nabla_{\mathbf{x}} \phi_l = 2 \underbrace{\left(\hat{\mathbf{H}}_l^T \mathbf{R}^{-1} \hat{\mathbf{H}}_l + \hat{\mathbf{\Sigma}}_{l|l-1}^{-1}\right)}_{\text{effective Hessian}} (\mathbf{x} - \hat{\mathbf{x}}_{l|l-1}) - 2\hat{\mathbf{H}}_l^T \mathbf{R}^{-1} \Delta \mathbf{y}_l$$

Efficient Jacobian via Dynamic Programming

Polynomial filter representation: $\mathbf{h}_l(\mathbf{x}_l) = \sum_{p=0}^{P} a_p \mathbf{L}_l^p \mathbf{q}_l$

Jacobian: $[\hat{\mathbf{H}}_l]_{:,m} = \sum_{p=1}^{P} a_p \sum_{k=0}^{p-1} \mathbf{L}_l^k (\mathbf{B}_{:,m} \mathbf{B}_{:,m}^T) \mathbf{L}_l^{p-k-1} \mathbf{q}_l$

Alg. 2: Efficient Jacobian via Dynamic Programming

Input: $\mathbf{L}_l$, $\mathbf{q}_l$, coefficients $\{a_p\}_{p=0}^{P}$, edge mapping ψ

Precompute $\{\mathbf{c}_p, \mathbf{D}_p\}_{p=0}^{P-1}$:

$\mathbf{c}_0 \leftarrow \mathbf{q}_l, \quad \mathbf{D}_{P-1} \leftarrow a_P \mathbf{I}$

For $p = 1, \ldots, P-1$:

$\mathbf{c}_p \leftarrow \mathbf{L}_l \mathbf{c}_{p-1}, \quad \mathbf{D}_{P-1-p} \leftarrow a_{P-p} \mathbf{I} + \mathbf{L}_l \mathbf{D}_{P-p}$

Compute columns of $\hat{\mathbf{H}}_l$:

For $m = 1, \ldots, \frac{N(N-1)}{2}$:

$(n, k) \leftarrow \psi^{-1}(m), \quad [\hat{\mathbf{H}}_l]_{:,m} \leftarrow \sum_{p=0}^{P-1} ([\mathbf{c}_p]_n - [\mathbf{c}_p]_k) \cdot ([\mathbf{D}_p]_{:,n} - [\mathbf{D}_p]_{:,k})$

Output: $\hat{\mathbf{H}}_l$

Complexity: $O(P^3 N^4)$ naive $\rightarrow$ $O(PN^3)$ proposed

Algorithm Overview: Full GSP-EKF

GSP-EKF loop at each time step l

Prediction:

$\hat{\mathbf{x}}_{l|l-1} = \mathbf{f}_l(\hat{\mathbf{x}}_{l-1})$

Compute $\hat{\mathbf{H}}_l$ via **Alg. 2 (Dynamic Programming)**

$\hat{\mathbf{\Sigma}}_{l|l-1} = \hat{\mathbf{F}}_l \hat{\mathbf{\Sigma}}_{l-1} \hat{\mathbf{F}}_l^T + \mathbf{Q}, \quad \hat{\mathbf{S}}_{l|l-1} = \hat{\mathbf{H}}_l \hat{\mathbf{\Sigma}}_{l|l-1} \hat{\mathbf{H}}_l^T + \mathbf{R}$

Update:

Kalman gain $\mathbf{K}_l = \hat{\mathbf{\Sigma}}_{l|l-1} \hat{\mathbf{H}}_l^T \hat{\mathbf{S}}_{l|l-1}^{-1}$

Solve via **Alg. 1 (ISTA)**: $\hat{\mathbf{x}}_l = \arg\min_{\mathbf{x}} \ \phi_l(\mathbf{x}) + \mu \|\mathbf{x}\|_1$

Update $\hat{\mathbf{\Sigma}}_l$;

Output: $\hat{\mathbf{x}}_l$

Special Cases

1. **Linear model** ($P = 1$):

The first ISTA iteration coincides with the Graph Tracking-KF.[1]

2. **Oracle GSP-EKF (known support ξ_l, unknown weights):**

- Known support $\rightarrow$ masking matrix $\mathbf{M}_l$
- Constrained update: $\hat{\mathbf{x}}_{\bar{\xi}_l} = \mathbf{0}$

Prediction with masking: $\hat{\mathbf{x}}_{l|l-1} = \mathbf{M}_l(\mathbf{f}_l(\hat{\mathbf{x}}_{l-1}) + \bar{\mathbf{e}}_l)$
Covariance: $\hat{\mathbf{\Sigma}}_{l|l-1} = \mathbf{M}_l\hat{\mathbf{F}}_l\hat{\mathbf{\Sigma}}_{l-1}\hat{\mathbf{F}}_l^T\mathbf{M}_l + \mathbf{M}_l\mathbf{Q}\mathbf{M}_l$

- Serves as oracle performance benchmark
- Practical examples: power grids (fixed lines, varying impedances), traffic networks (fixed roads, varying flow)

[1] Dabush and Routtenberg 2024.

Simulation Setup & Baselines

Compared methods:

GSP-EKF	Proposed (Alg. 3)
EKF	No sparsity prior
Change-det[2]	Sliding window graph learning, temporal regularization
GRLS[3]	RLS estimation of graph filters
Prob-SSM[4]	Discrete Bayes, Probabilistic SSM
Oracle	Known support

Performance measures:

NMSE (edge weight estimation): $\frac{2}{N(N-1)} \times \|\mathbf{x}_l - \hat{\mathbf{x}}_l\|_2^2$,

EIER (edge identification error rate): $\frac{|(\xi_l \backslash \hat{\xi}_l) \cup (\hat{\xi}_l \backslash \xi_l)|}{N(N-1)} \times 100\%$.

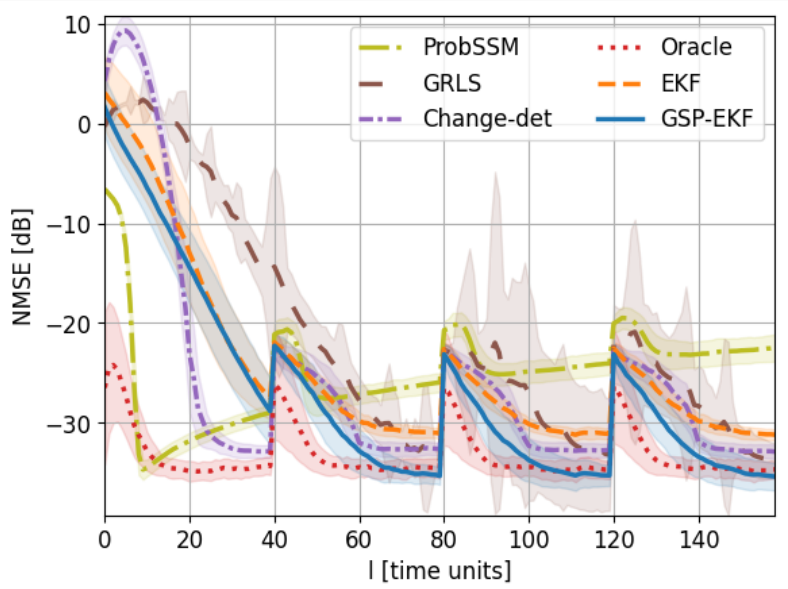
[2] Hu and Xiao 2023.

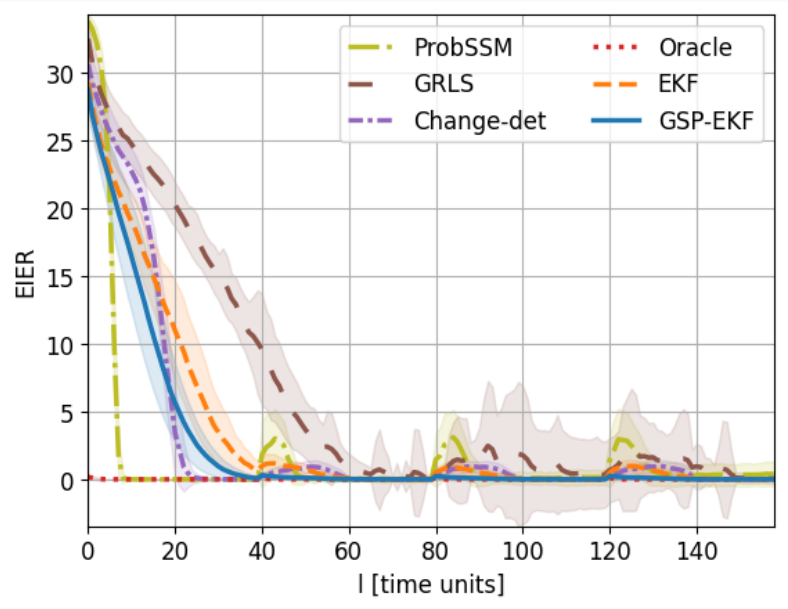
[3] Ramezani-Mayiami and Beferull-Lozano 2017.

[4] Tenorio, Isufi, Leus, and Marques, 2024

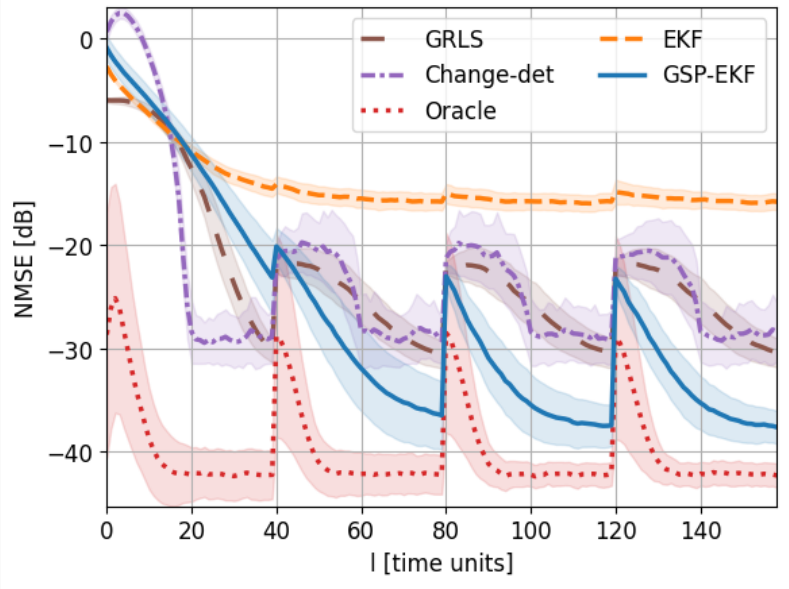
Performance vs. Time (Synthetic Data, $N = 20$)

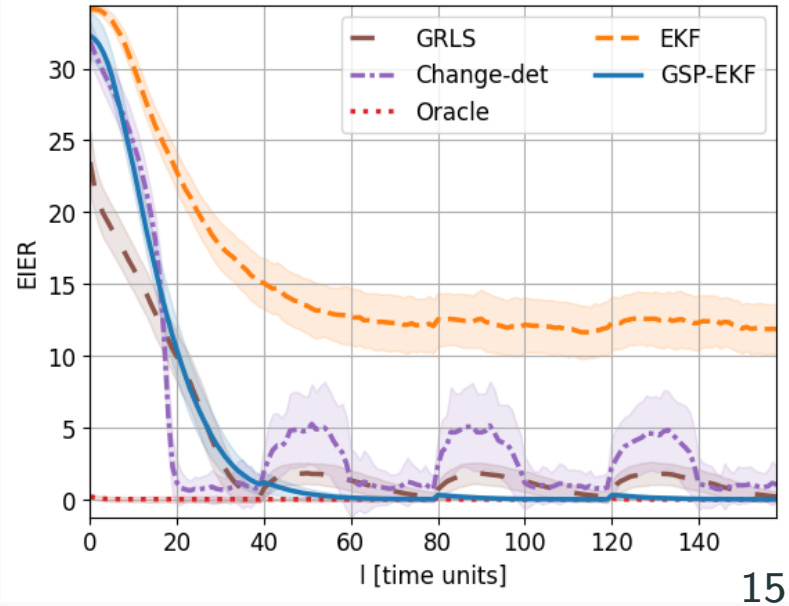
$h(\mathbf{L}) = \mathbf{L}$





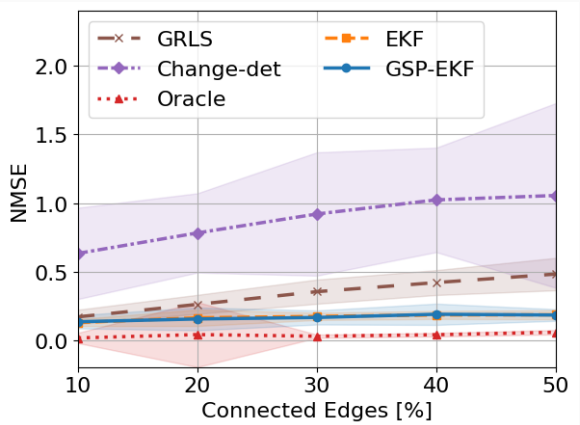
$h(\mathbf{L}) = \mathbf{I} + \mathbf{L} + \mathbf{L}^2 + 0.1\mathbf{L}^3 + \mathbf{L}^4$

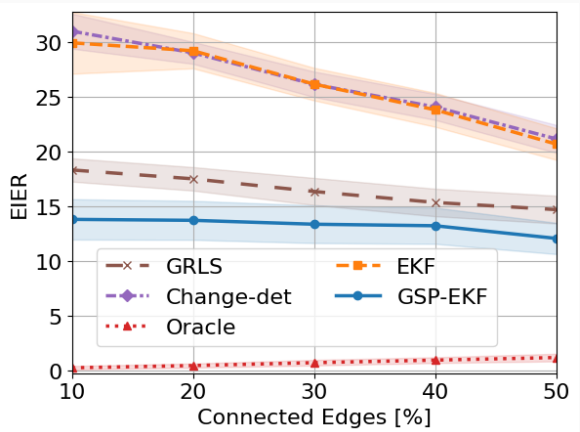




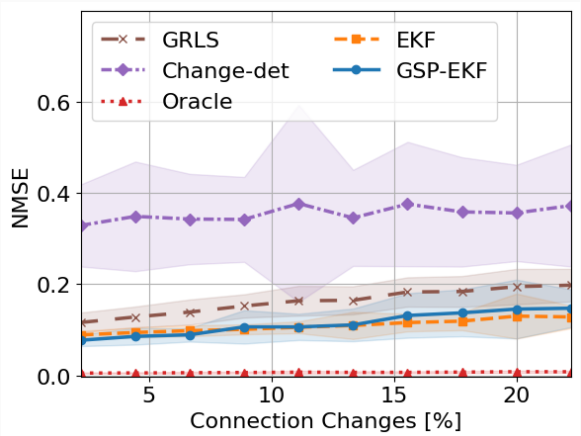
Performance vs. Graph Parameters (Synthetic Data, $N = 10$, $h(\mathbf{L}) = \mathbf{I} + \mathbf{L} + 0.8\mathbf{L}^2 + 0.6\mathbf{L}^3 + 0.4\mathbf{L}^4 + 0.2\mathbf{L}^5$)

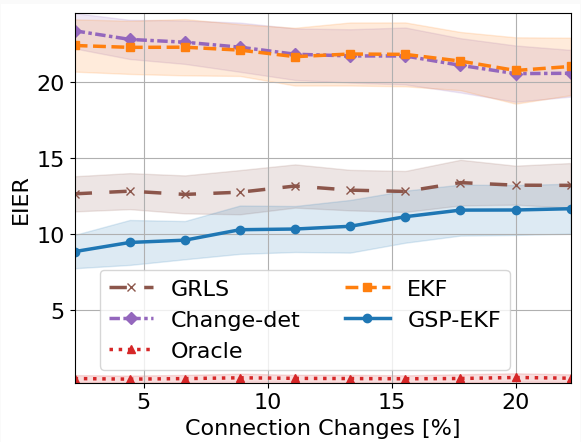
vs. Sparsity



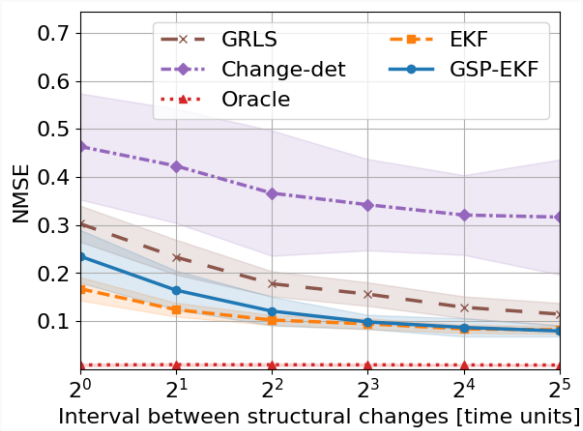


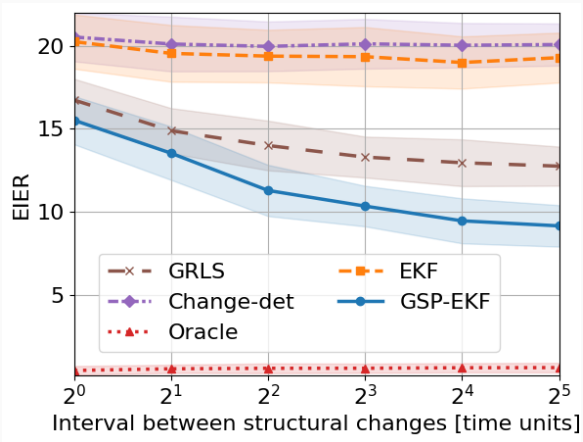
vs. Edge-change %





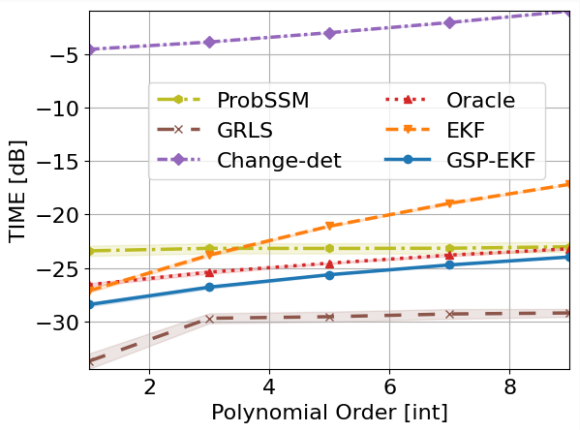
vs. Change rate



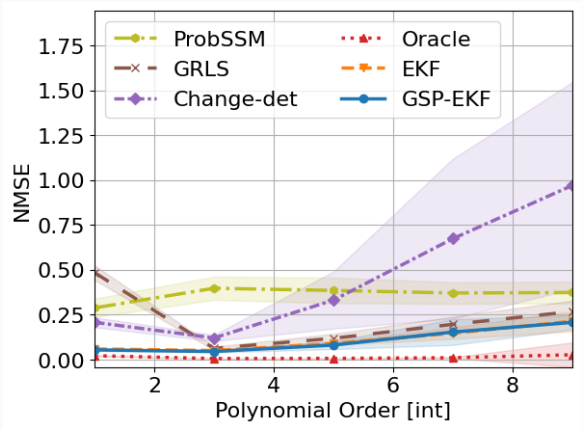


Runtime vs. Filter Order P (Synthetic Data, $N = 10$)

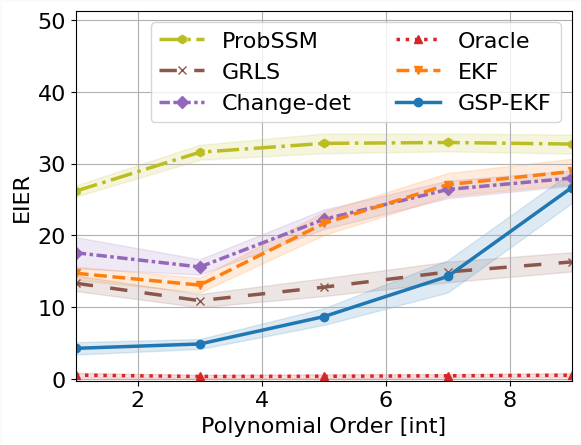
$h(\mathbf{L}_l) = \sum_{p=0}^{P} \frac{1}{2^p} \mathbf{L}_l^p$, where P, the filter order, varies in $[1, N)$



Computation time per step vs. P



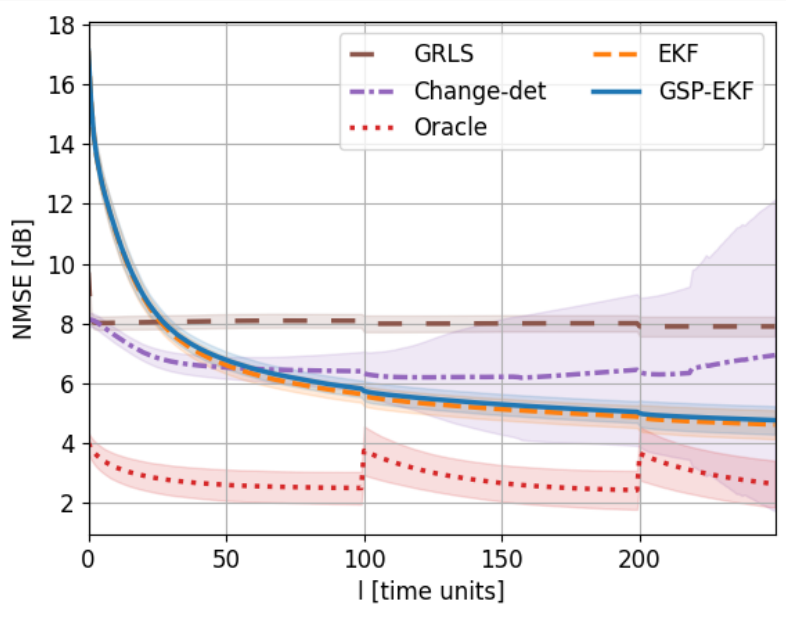
NMSE vs. P



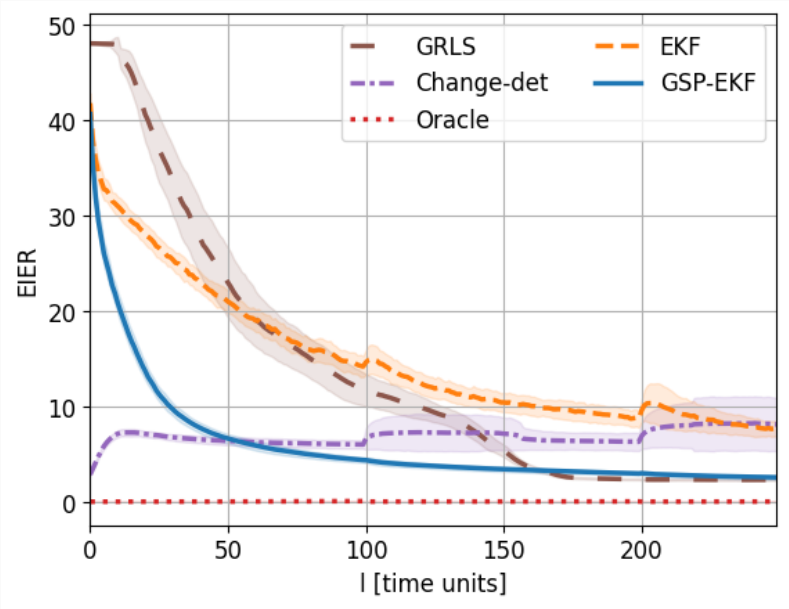
EIER vs. P

IEEE 57-Bus Power System

- $N = 57$ buses, 66 true lines out of 1596 possible edges
- Measurements: **nonlinear AC power flow** (pandapower)
- Estimation: **linear DC approximation** (model mismatch)
- Gradual topology changes every ~ 100 steps



NMSE vs. time



EIER vs. time

Conclusions

1. **Dynamic graph topology tracking** was formulated as a nonlinear graph state-space estimation problem, where the latent state is the sparse edge-weight vector
2. **Sparsity-aware GSP-EKF** combines Bayesian filtering with sparse graph learning, enabling joint support and weight tracking via ℓ_1-regularized EKF update (ISTA)
3. **Theory and computation** were developed through observability analysis and an efficient Jacobian computation, making large-scale tracking feasible.
4. **Empirical evaluation** on synthetic and IEEE 57-bus networks demonstrated improved topology recovery and tracking accuracy over existing methods.

Future directions

- Unknown input $\mathbf{q}_l$ (joint estimation via augmented SSM / EM)
- Directed graphs (alternative Laplacian factorization)
- Learning-aided KF (KalmanNet[5]) for model mismatch
- Distributed implementations for large-scale networks
- Particle filter (ICASSP 2026)
- Observability analysis for nonlinear observation model.

[5] Revach, Shlezinger, Ni, Escoriza, van Sloun, Eldar 2021

Thank you!

Questions?

Link to arxiv

Link to Github

Lital Dabush litaldab@post.bgu.ac.il

Nir Shlezinger nirshl@bgu.ac.il

Tirza Routtenberg tirzar@bgu.ac.il

GSP Refresher: Graph Laplacian & Filters

Graph Laplacian:

$$\mathbf{L} = \mathbf{B}\ \mathrm{diag}(\mathbf{x})\,\mathbf{B}^T \in \mathbb{R}^{N\times N}$$

$\mathbf{B} \in \mathbb{R}^{N\times N(N-1)/2}$ — incidence matrix; $\mathbf{x}$ — non-negative edge weights (zero for absent edges). $\mathbf{L}$ is real PSD with non-positive off-diagonal entries.

Polynomial graph filter:

$$h(\mathbf{L}) = \sum_{p=0}^{P} a_p \mathbf{L}^p = \mathbf{V} h(\mathbf{\Lambda}) \mathbf{V}^T$$

Graph signal: vector $\mathbf{y} \in \mathbb{R}^N$ indexed by nodes.

Filtered output: $\mathbf{y}_{\mathrm{out}} = h(\mathbf{L})\,\mathbf{y}_{\mathrm{in}}$. Each edge contributes across *multiple powers* of $\mathbf{L}$ — this is why first-order SVAR models miss higher-order interactions.

Complexity Analysis Summary

Step	Operation	Complexity
Prediction	$\mathbf{f}_l$, $\mathbf{F}_l$	Standard EKF
Laplacian build	$\mathbf{B}\,\mathrm{diag}(\mathbf{x})\mathbf{B}^T$	$O(N^2)$
Jacobian	Algorithm 2 (DP)	$O(PN^3)$
Covariance	$\hat{\mathbf{\Sigma}}_{l\|l-1}$ (diag $\mathbf{F}$)	$O(N^2)$
Innov. cov.	$\hat{\mathbf{S}}_{l\|l-1}$	$O(N^4)$
Update ($M_{\text{ISTA}} = 1$)	Woodbury identity	$O(N \cdot N(N-1)/2)$
Cov. update	$\hat{\mathbf{\Sigma}}_l$	$O(N^5)$ overall

With diagonal $\mathbf{F}$ and one ISTA step: **total** $O(N^5)$.
Diagonal covariance approximation reduces to $O(N^4)$.

Synthetic setup parameters:

Measurement models:

- *Lin*: $h(\mathbf{L}) = \mathbf{L}$
- *NL4*: $\mathbf{I} + \mathbf{L} + \mathbf{L}^2 + 0.1\mathbf{L}^3 + \mathbf{L}^4$
- *NL5*: $\sum_{p=0}^{5} (0.2p)^{-1}\mathbf{L}^p$ (approximately)
- *NLP*: $\sum_{p=0}^{P} \frac{1}{2^p}\mathbf{L}^p$, varying P

Simulation parameters:

- Sparse graph of N nodes, $3N$ initial edges
- Edge added or removed every $2N$ steps (Theorem 1 constraint)
- State transition: $\mathbf{f}_l(\mathbf{x}) = \mathbf{x}$ (random walk)
- Input $\mathbf{q}_l \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$