Online Structure Learning and Tracking of Power System Graphical Models

Abrar Zahin*, Gautam Dasarathy†, Lalitha Sankar‡ and Oliver Kosut§
School of Electrical, Computer and Energy Engineering
Arizona State University
*Email: *azahin@asu.edu, †gautamd@asu.edu, ‡lsankar@asu.edu, §okosut@asu.edu*

***Abstract*—The abstract goes here.**

1. Introduction & Outline

2. Related Work

This demo file is intended to serve as a "starter file" for IEEE Computer Society conference papers produced under LaTeX using IEEEtran.cls version 1.8b and later. I wish you the best of success.

2.1. Subsection Heading Here

Subsection text here. [2] said that

2.1.1. Subsubsection Heading Here. Subsubsection text here.

3. Introduction

3.1. Motivation

3.2. Related Work

3.3. Contribution

In this paper we propose an algorithm to learn the underlying structure of a power system grid with generated voltage measurements. Our learning algorithm is *online* as it continuously refines it's learning outcome as more samples are coming from the gird. In order to learn the underlying structure our proposed algorithm is a modified version of online algorithm from [2] which can learn an intermediate structure of a power grid by designing the grid as a graphical model. Subsequently, this learned intermediate structure is post-processed to get rid of spurious edges resulting from cycle constraints of the grid.

3.4. Problem Statement

Let us design our power grid by a graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, where $\mathcal{V}$ denotes the set of buses/ nodes, $|\mathcal{V}| = n+1$, $\mathcal{E}$ denotes the set of undirected lines/ edges. Lower case letters, $a, b, c, \dots$ denotes nodes and a pair (ab) denotes a edge between nodes a and b. In the power gird, one bus is considered as a reference bus and voltages at all other buses are measured relative to it. Thus, without loss of generality reference bus is ignored and power flow equations and all further analysis are restricted to n non-reference nodes in the power grid. $v, \theta, p, q \in \mathbb{R}^n$ represents the vector of nodal voltage magnitude, phase, active and reactive injections respectively associated with each bus. The interactions between p, q and v, θ are captured by following LC-PF equation.

$$\left[\begin{array}{c} p \\ \hline q \end{array}\right] = \left[\begin{array}{c|c} H_g & H_\beta \\ \hline H_\beta & -H_g \end{array}\right] \left[\begin{array}{c} v \\ \hline \theta \end{array}\right] \tag{1}$$

$$\left[\begin{array}{c} v \\ \hline \theta \end{array}\right] = \left[\begin{array}{c|c} H_g & H_\beta \\ \hline H_\beta & -H_g \end{array}\right]^{-1} \left[\begin{array}{c} p \\ \hline q \end{array}\right] \tag{2}$$

Here H_g and H_β are the reduced weight Laplacian matrices for the power system graph $\mathcal{G}$ with edge weights given by susceptances β and conductances g, where $g_{ij} + \tilde{i}\beta_{ij} = \frac{1}{r_{ij} - \tilde{i}x_{ij}}$. The sparsity of H_β and H_g encodes the grid structure as follows [3]

$$H_g(i,j) = \begin{cases} \sum_{k:(ik)\in\mathcal{E}} g_{ik} & \text{if } i = j \\ -g_{ij} & \text{if } (ij) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

From here we will consider tuple of voltage magnitude and phase angle $\left[\begin{array}{c} [v_i]_{i\in[n]} \\ \hline [\theta_i]_{i\in[n]} \end{array}\right]$ as a vector $\mathbf{X} \in \mathbb{R}^{2n}$ and each index X_i of $\mathbf{X}$ is a *voltage measurement* in the considered power system grid and there are $2n$ such voltage measurements. We want to learn the underlying edge pattern of the buses generating these measurements. [4] and [1] show that injection powers follows Gaussian distribution. Our algorithm exploits the fact that voltage measurement of a node can be estimated by learning the correct linear combination of voltage measurements from all other nodes, considering the samples are generated from a Gaussian distribution. See Lemma 2 for details.

Lemma 1. *(See Theorem 1 of [3]) The non-zero entries of inverse covariance matrix of voltage measurements, $\Sigma^{-1}_{(v,\theta)}$, denotes edges only at the same bus, neighboring buses, and* *two-hop neighboring buses. We denote this underlying graph and corresponding edge set as $\mathcal{G}^2$ and $\mathcal{E}^2$ respectively.*

Proof. See [3] for the proof. $\square$

Thus goal of our learning algorithm is as follows

> Having access to the voltage measurements (v_i, θ_i) $\forall i = 1, 2, \ldots, 2n$, we want to learn $\left(\mathcal{G}^2, \mathcal{E}^2\right)$ (the non-zero entries of $\Sigma^{-1}_{(v,\theta)}$) in an online manner.

4. Online Learning of Power System Graphical Model

In this section we discuss our proposed online algorithm for learning the two-hop network, $\left(\mathcal{V}^2, \mathcal{E}^2\right)$ of underlying power system graphical model from voltage measurements. In the proposed online setting, learning algorithm is applied to the arrived voltage measurement samples one by one rather than storing all the samples and then learn the graph later.

Given Gaussian zero mean random vector $\mathbf{X} \in \mathbb{R}^{2n}$ of voltage measurements, Covariance matrix $\Sigma_{\mathbf{X}} \in \mathbb{R}^{2n\times 2n}$, Inverse Covariance matrix, $K_{\mathbf{X}} \triangleq \Sigma^{-1}_{\mathbf{X}} \in \mathbb{R}^{2n\times 2n}$; $K = [k_{ij}]_{i,j\in[2n]}$ (where $[2n] = \{v_1, v_2 \ldots v_n, \theta_1, \theta_2 \ldots \theta_n,\}$), we want to recover the underlying graph, $\mathcal{G}^2 = (\mathcal{V}^2, \mathcal{E}^2)$, whose adjacency matrix follows the non-zero entries of K.

Lemma 2. *Expected voltage measurement in one node conditioned on voltage measurements from all other nodes can be represented as a linear combination of voltage measurements from all other nodes.*

Proof. Let us introduce a vector $\mathbf{X}_{-i} \in \mathbb{R}^{2n-1}$, which is constructed by discarding the i^{th} index from $\mathbf{X} \in \mathbb{R}^{2n}$ and $\mathbf{w}_i \in \mathbb{R}^{2n-1}$ represents corresponding weights of these $2n-1$ nodes to estimate X_i. We can express $\mathbb{E}\left[X_i|\mathbf{X}_{-i}\right]$ as follows

$$\mathbb{E}[X_i|\mathbf{X}_{-i}] = \sum_{j\neq i} \frac{-k_{ij}}{k_{ii}} X_j = \mathbf{w}_i \cdot \mathbf{X}_{-i} \qquad (4)$$

$\square$

4.1. Overview of Online Algorithm

As we see in Lemma 2, expected voltage measurement from one node is a linear combination of corresponding weights of all other nodes. Our algorithm exploits this fact to learn the non-zero entries of Inverse Covariance matrix of voltage measurements, K. Our learning algorithm starts with an uniform weight vector and update the weights as the samples are coming based on how well the weight vector predicted a response variable. Algorithm also records all candidate weight vectors and further select the one which pass the minimum empirical risk test. Appying the preceding procedure for all nodes the learning algorithm provides an estimate $\widehat{\mathcal{G}_2}$ or equivalently, an estimate $\widehat{\mathcal{E}_2}$ of the edge set with $m = T + M$ voltage measurements. Where T samples are for outputting T candidate weight vectors so that M samples can choose the optimal one based on minimum empirical risk. Thus the error probability is characterised by following equation

$$\mathbb{P}(\text{error}) = \mathbb{P}\left(\widehat{\mathcal{G}_2} \neq \mathcal{G}_2\right) \qquad (5)$$

Now let us introduce following input parameters of our online learning algorithm: $\lambda_i = \sum_{i\neq j} |\frac{k_{ij}}{k_{ii}}|$ and $\max_i \lambda_i \leq \lambda$. $\kappa = \min_{(i,j)\in\mathcal{E}} |\frac{k_{ij}}{\sqrt{k_{ii}k_{jj}}}|$ denotes minimum normalize edge strength, $\boldsymbol{\rho}^t$ is a distribution vector over $2n$-1 experts at each round t. $k_{\max} = \max_{i,j} |k_{i,j}| = \max_i k_{i,i}$ denotes an upper bound on the maximum value on the entries of inverse covariance matrix, K, $\nu_{\max} = \max_i \text{Var}[V_i]$ denotes an upper bound on the variance of any marginal variable.

Theorem 1. *(Theorem 9 of [2]) In order to learn the non-zero entries of $\Sigma^{-1}_{(v,\theta)} \in \mathbb{R}^{2n\times 2n}$, given normalized edge strength κ, and other input parameters $(\lambda, \nu_{\max}, \theta_{\max})$, algorithm 1 can achieve $\mathbb{P}(\text{error}) \leq \delta$ with a sample complexity of*

$$m = \mathcal{O}\left(\frac{\lambda^4 \nu^2_{\max} \theta^2_{\max}}{\kappa^4} \log^3 \frac{2n}{\delta}\right)$$

Proof. See [2] for proof. $\square$

Algorithm 1 ONLINE LEARNING FOR 2-HOP POWER SYSTEM GRAPHICAL MODEL

```
 1: Input: $T + M$ samples of voltage measurements $X \in \mathbb{R}^{2n}$
 2: Input Parameters: $\eta$, $\lambda_i$, $\kappa$, $k_{\max}$, $\nu_{\max}$
 3: Output: Estimation of $2n-1$ weight vectors, $\mathbf{w}_{v^i}, \forall i \in [2n]$
 4: for $i = 1$ to $2n$ do
 5:     for $i = 1$ to $T$ do
 6:         $\boldsymbol{\rho}^t = \frac{\boldsymbol{\varphi}^{t-1}}{||\boldsymbol{\varphi}^{t-1}||_1}$
 7:         $\boldsymbol{l}^t \in \mathbb{R}^{2n-1}$
 8:         $\boldsymbol{l}^t = (1/2)(\mathbf{1} + (\lambda\rho^t \cdot \mathbf{x}^t_{-i} - x^t_i)\mathbf{x}^t_{-i})$
 9:         Update the weight vector: $\forall i \in [2n-1]$, $\boldsymbol{\varphi}^t_i =$
10:         $\boldsymbol{\varphi}^{t-1}_i \cdot \eta^{l^t_i}$.
11:     end for
12:     $T$ candidates of weight vectors $\lambda\boldsymbol{\rho}^1, \lambda\boldsymbol{\rho}^2, \ldots, \lambda\boldsymbol{\rho}^T$
13:     for $i = 1$ to $M$ do
14:         Compute empirical risk, $\hat{\varepsilon}(\boldsymbol{\lambda}p^t) =$
15:         $\frac{\sum_{j=1}^M (\boldsymbol{\lambda}p^t \cdot \mathbf{a}^j - b^j)^2}{M}$, $\forall t \in \{1, 2, \ldots, T\}$
16:     end for
17:     Return $\lambda\boldsymbol{\rho}^{t^*} = \arg\min_{t\in[T]} \hat{\varepsilon}(\lambda\boldsymbol{\rho}^t)$ as the weight vector.
18: end for
19: $\forall$ pairs $i$ and $j$, identify as an edge $(i,j) \in \mathcal{E}$ if
    corresponding entry $\mathbf{w}_{v^i}(j)$ is greater than a threshold.
20: Return $\mathbf{w}_{v^1}, \mathbf{w}_{v^2}, \ldots, \mathbf{w}_{v^{2n}}$.
```

5. Run-time of the Algorithm

Run time of Algorithm 1 (for learning 2-hop graph) is $\mathcal{O}(mn^2)$, where $m = T + M$, Deka's two-hop learning runtime is $\mathcal{O}(n^3)$ [3].

Theorem 2. *(Theorem 2 of [3]) Rewrite and reproof Deka Theorem 2.*

Theorem 3. *(Theorem 3 of [3]) Rewrite and reproof Deka Theorem 3.*

6. Simulation Results

The efficacy of proposed algorithm on power system graphical model has been showed by simulation results. Our simulation results are from two settings. One with synthesized data from a simple four bus system and another one from Matpower system. In order to show the truly online nature of our algorithm we also generated time varying synthesized data from three instances of the simple four bus system setting.

6.1. Static Graph

6.1.1. Simple Four Bus System Setting. Our processes of preparing synthesised four bus system are as follows

- Used a simple setup of four bus system where active power p and q are independent of each other, this is logical because ultimately we just need to learn the first 4×4 block of an 8×8 adjacency matrix. Other diagonal 4×4 block is a replication and two 4×4 off-diagonal block is just a diagonal entry modification from the diagonal blocks.
- $H_{(g,\beta)}$ matrix were created manually by maintaining corresponding degree and adjacency matrix as in fig. 6a fig. 6b fig. 6c, and also making the matrix a little bit diagonally dominant in order to calculate the inverse.
- Thus $\Sigma_{(p,q)}$ as an Identity matrix $I \in \mathbb{R}^{8\times8}$ and computed $\Sigma_{(v,\theta)} = H^{-1}_{(g,\beta)}\, \Sigma_{(p,q)}\, H^{-1}_{(g,\beta)}$.
- $\mathcal{A}_T$ = True Adjacency Matrix, $\mathcal{A}_O$ = Output Adjacency Matrix.
- Error in each run is computed as the ratio of the sum of number of false links identified and true links missed, to the total number of links in the underlying network.
- Error = $\frac{\sum_{i=1}^{2n}\sum_{j=1}^{2n} |\mathcal{A}_T - \mathcal{A}_O|}{2n\times2n}$

6.1.2. Matpower Graph. In this section we discuss the simulation setup and corresponding results for three Matpower power system cases, i.e. Case 5, Case 9 and Case 14. Power injections are generated as zero mean Gaussian random vector and $H_{(g,\beta)}$ matrix has been created from the branch data from a power system graph (case) with the corresponding conductance and susceptance of each lines. Subsequently these voltage measurements were used as the input of Algorithm 1 and Algorithm **??**. For all simulation results, slack buses were ignored while creating the block laplacian matrices H_g and H_β. Error performance in learning 2-hop network for Case 5, Case 9 and Case 14 are in fig. 3, fig. 4 and fig. 5 respectively.

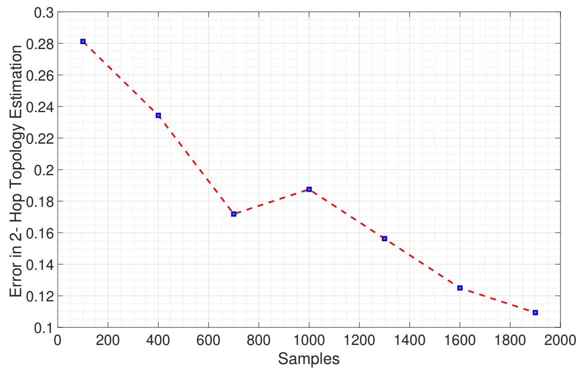


Figure 1: Graph Learning Performance for a 4-bus toy problem

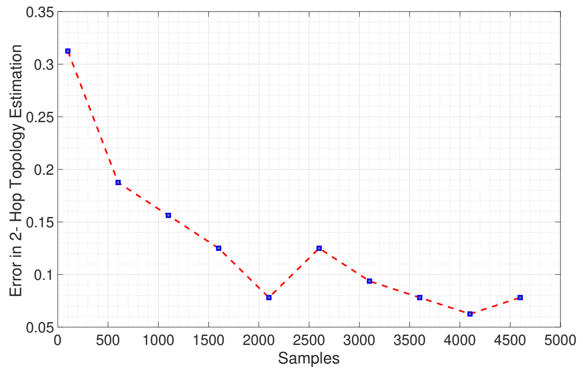


Figure 2: Graph Learning Performance for another 4-bus toy problem

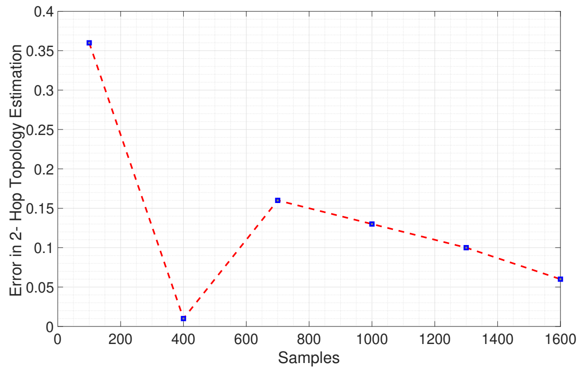


Figure 3: Graph Learning Performance for Case 5

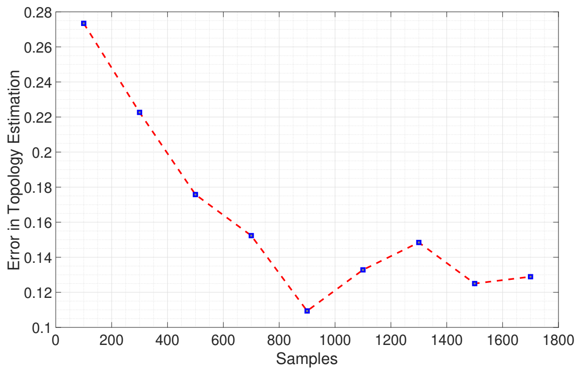


Figure 4: Graph Learning Performance for Case 9

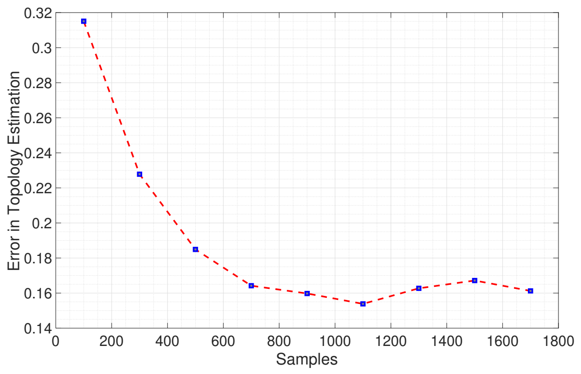


Figure 5: Graph Learning Performance for Case 14

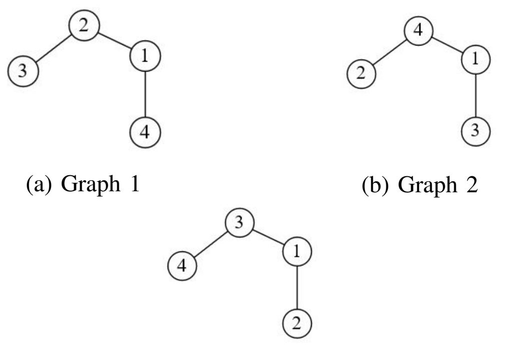
6.2. Time Varying Network

In order to show the efficacy of our algorithm for time-varying network, we considered two situations with three simply synthesized power system graph of 4 buses in fig. 6, where we considered two situations. In first situation-underlying power system graph changes once: (6a to 6b) and in second situation underlying power system graph changes twice: (6a to 6b and 6b to 6c). In both situations we assumed that a graph remains static for a considerable amount of time. [1] We can see from the simulation results in fig. 7 and in fig. 8 that the online learning algorithm learns the underlying graph, i.e. error is decreasing until the samples are generated from different graph from the current one. As the underlying graph changes error increases instantaneously and continue to decrease as the graph remains static for a some time. This same phenomenon is observed for second situation (underlying power system graph changes twice).

7. Change Detection

Proposed online algorithm can also detect changes in underlying power system topology in an online manner.

1. We assume that underlying power system graph is static for enough time for online algorithm to learn the 2-hop structure with decent certainty.



(c) Graph 3

Figure 6: 3 instances of a 4 bus power system Graphical Model

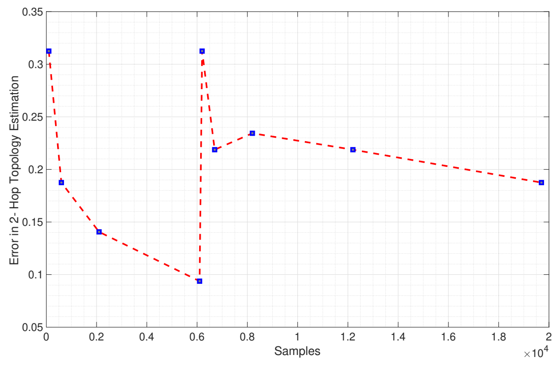


Figure 7: Graph Learning Performance for Time Varying Graph

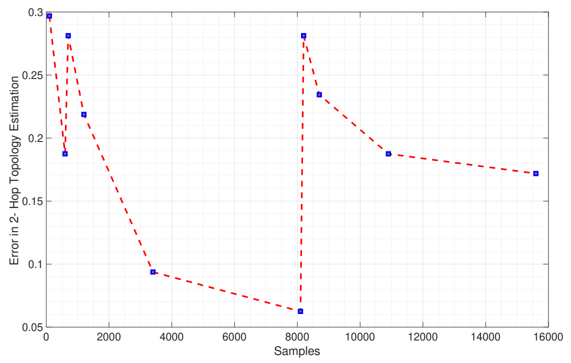


Figure 8: Graph Learning Performance for Time Varying Graph Twice

Notice that in fig. 7 and in fig. 8 underlying power system graph changes after 6000 and 8000 samples respectively and as a result error goes up instantaneously. This event indicates a change in topology and if the power system remains in that changed topology for a considerable amount of time, the algorithm learns the new graph and as a result error drops gradually. in this cases we can detect the change in topology by comparing the learned adjacency matrix corresponding to two different graphs [2].

8. Conclusion

The conclusion goes here.

Acknowledgments

The authors would like to thank...

References

[1] Guido Cavraro, Vassilis Kekatos, and Sriharsha Veeramachaneni. “Voltage analytics for power distribution network topology verification”. In: *IEEE Transactions on Smart Grid* 10.1 (2017), pp. 1058–1067.

[2] Anamay Chaturvedi and Jonathan Scarlett. “Learning Gaussian Graphical Models via Multiplicative Weights”. In: *arXiv preprint arXiv:2002.08663* (2020).

[3] Deepjyoti Deka et al. “Graphical Models in Meshed Distribution Grids: Topology estimation, change detection & limitations”. In: *IEEE Transactions on Smart Grid* (2020).

[4] Yizheng Liao et al. “Urban mv and lv distribution grid topology estimation via group lasso”. In: *IEEE Transactions on Power Systems* 34.1 (2018), pp. 12–27.

2. We assume that underlying power system graph is static for enough time for online algorithm to learn the 2-hop structure with decent certainty.