Data-Driven Graph Construction of Power Flow Graphs for Electric Power Transmission Networks

Benjamin Poole[1], Rajan Ratnakumar[2], Dulip Tharaka Madurasinghe[3],
Christian Kümmerle[4], Ganesh Kumar Venayagamoorthy[5], and Minwoo Lee[6]
[146] Department of Computer Science, University of North Carolina at Charlotte, USA
[345] Real-Time Power and Intelligent Systems Laboratory, Clemson University, USA
[5]Department of Electrical, Electronic and Computer Engineering, University of Pretoria, South Africa
bpoole16@charlotte.edu, rajan94@ieee.org, dtmadurasinghe@ieee.org,
kuemmerle@charlotte.edu, gkumar@ieee.org, minwoo.lee@charlotte.edu

***Abstract*—A comprehensive understanding of the topology of the electric power transmission network (EPTN) is essential for reliable and robust control of power systems. While existing research primarily relies on domain-specific methods, it lacks data-driven approaches that have proven effective in modeling the topology of complex systems. To address this gap, this paper explores the potential of data-driven methods for more accurate and adaptive solutions to uncover the true underlying topology of EPTNs. First, this paper examines Gaussian Graphical Models (GGM) to create an EPTN network graph (i.e., undirected simple graph). Second, to further refine and validate this estimated network graph, a physics-based, domain-specific refinement algorithm is proposed to prune false edges and construct the corresponding electric power flow network graph (i.e., directed multi-graph). The proposed method is tested using a synchrophasor dataset collected from a two-area, four-machine power system simulated on the real-time digital simulator (RTDS) platform. Experimental results show both the network and flow graphs can be reconstructed using various operating conditions and topologies with limited failure cases.**

***Index Terms*—critical infrastructure, data-driven modeling, electric power transmission, Gaussian graphical models, graph construction**

I. Introduction

Electric power transmission networks (EPTN) are dynamic complex systems that connect power generating sites to load locations, such as electrical substations, where power is distributed. This critical infrastructure requires real-time control, as power generated must be consumed immediately due to the network's limited storage capability. Thus, reliably monitoring and modeling the EPTN topology is of utmost importance for making optimal power control decisions. Inaccuracies in the topology model can be caused by equipment malfunctions (breakers, relays, isolators), or false data injection attacks [1]. Such inaccuracies can result in catastrophic consequences, including blackouts and brownouts. Therefore, there is a pressing need for additional resilient, reliable, and accurate modeling techniques.

Traditionally, EPTN topology has been modeled using the circuit breaker status information (i.e., connected or not) and other supervisory control and data acquisition (SCADA) sensors [2]. However, the introduction of phaser measurement units (PMUs) has provided an abundance of data related to the underlying physics of the EPTN, such as the magnitude and phase angle for the voltage and current at particular points in the network. Compared to traditional SCADA monitoring, PMUs offer high temporal resolution, allowing for dynamic events to be closely monitored. SCADA information has a time resolution of 0.16-0.5 Hz, compared to PMUs 30-240 Hz [3]. Prior approaches that have attempted to harness PMU data for EPTN topology modeling have been highly domain specific, utilizing rule-based and search-based approaches that exploit the underlying physics of the EPTN [4, 3, 1].

PMU data enables more general data-driven machine learning approaches. However, studies using algorithms such as deep learning or graph signal processing have largely focused on downstream tasks such as fault detection, time-series prediction, predicting optimal power flow, or data interpolation, while often assuming the EPTN topology to be given [5, 6, 7]. As a result, the application of machine learning approaches for modeling the true underlying EPTN topology has largely gone unexplored. Therefore, this paper aims to initiate the exploration of machine learning for modeling the true EPTN topology (i.e., graph structure learning). Additionally, it explores the novel combination of machine learning and domain-specific approaches such that the domain-specific approach acts as a form of physics-based verification of the modeled graph topology. Moreover, the proposed ideas are tested using a time synchronized PMU (i.e., synchrophasor) dataset collected from a real-time EPTN simulation [3].

The proposed approach is delineated into a two-stage process. In Stage 1, graph structure learning employs the Gaussian graphical models (GGMs) [8]. GGMs establish multivariate relationships by learning the conditional dependencies or the "structure" of a network. While extensively used in the fields like genomics and broader -omics fields [9], neuroscience [10], and psychology [11], GGMs have not been used to directly estimate the true EPTN topology. The resulting undirected simple graph estimated by GGMs can be interpreted as a network graph for the EPTN, representing the static physical connectivity of the nodes in the EPTN. Stage 2 consists of graph refinement, using a domain-specific physics-based scoring metrics [3]. Here, the estimated network graph from Stage 1 undergoes refinement by computing edge scores based

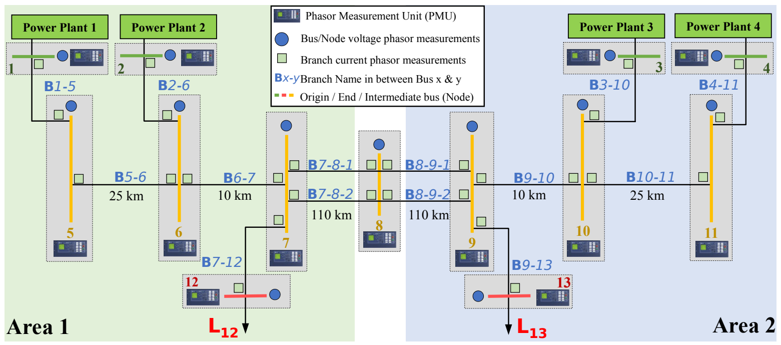


Fig. 1: Kundur's two-area, four-machine test power system model with PMUs installed.

on the connectivity, pruning false edges, and then converting the network graph into a directed multi-graph which is interpreted as a flow graph. This flow graph depicts the power flow in the EPTN, accommodating parallel edges. An alternative perspective of these two stages is that Stage 1 attempts to reduce the search space for Stage 2 by estimating a sparser network graph. In the worst case scenario, Stage 1 produces a fully connected network, similar to the exhaustive search method proposed by Venayagamoorthy et al. [3].

This paper's contributions are summarized as follows:

- Exploration of data-driven GGMs for network graph estimation using a two-area four-machine power system model RTDS synchrophasor dataset.
- Proposal of a domain-specific graph scoring algorithm for refining and validating the modeled network graph by pruning false edges and generating the corresponding flow graph.
- Discussion concerning challenges faced by the GGMs and the proposed graph refinement algorithm, along with those presented by EPTN data.

II. Background

This section briefly introduces the ideas of GGMs using convex and non-convex penalties, along with EPTN concepts and a very basic EPTN topology.

A. Gaussian Graphical Modeling

Graphical modeling is a technique for inferring dependencies between random variables. Graphical models (GMs) are represented via a graph $G = (V, E)$ defined by a set of p nodes or vertices $V = \{1, ..., p\}$ and a set of edges E. The structure or topology of G is typically defined using an adjacency matrix A. GMs are technically derived as a multivariate joint distribution containing certain conditional independencies [8]. As such, vertices are associated with random variables $X = (X_1, ..., X_p)$ while the edges represent the conditional dependencies. Thus, conditional independence is represented by the lack of an edge between any two variables X_i and X_j. In other words, within GMs, the edge $\{i, j\}$ is absent from E if the random variables X_i and X_j associated to the nodes i and j satisfy the pairwise Markov property such that

$$X_i \perp X_j \mid X_{V\setminus\{ij\}}, \tag{1}$$

which means that X_i is conditionally independent of X_j when conditioned on all other random variables $X_{V\setminus\{ij\}}$.

One common approach for estimating an undirected probabilistic graphical model in an unsupervised fashion are Gaussian graphical models (GGMs), where $X \sim \mathcal{N}(\mu, \Sigma)$ is assumed to be a multivariate Gaussian distribution with a mean vector μ and a covariance matrix Σ [8]. Using the precision matrix $\theta = \Sigma^{-1}$, the conditional dependencies can be derived such that an edge between i and j exist if and only if $\theta_{ij} \neq 0$. An adjacency matrix defining the graph connectivity (i.e., dependencies) can then be derived using non-zero entries of θ. Thus, the goal of GGMs is to estimate θ provided an empirical covariance $\hat{\Sigma}$ derived from the data.

A classical approach for estimating a sparse dependency graph precision matrix θ is the graphical lasso (glasso) [12, 13]. Glasso is a penalized maximum likelihood estimator that uses the convex ℓ_1 norm penalty $\|\theta\|_1$ to enforce sparsity on θ by minimizing

$$\min_{\theta \succ 0} -\log\det(\theta) + \text{tr}(\hat{\Sigma}\theta) + \lambda\|\theta\|_1, \tag{2}$$

where λ is the regularization parameter, $\det(\cdot)$ is the determinant, and $\text{tr}(\cdot)$ is the trace.

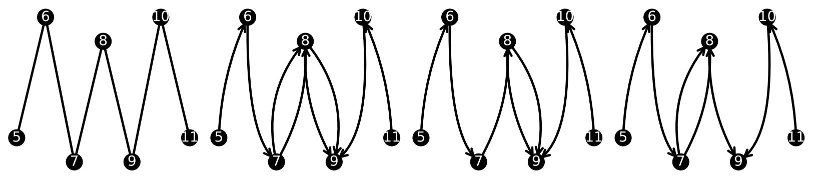
Alternatively, non-convex penalty formulations have also been explored to obtain estimators with more desirable properties, such as the oracle property and unbiasedness, which tend to lead to sparser models that have the same or better prediction accuracy than glasso [14, 15, 11]. The generalized form of a non-convex estimator can be achieved by minimizing a general penalized maximum likelihood estimator

$$\min_{\theta \succ 0} -\log\det(\theta) + \text{tr}(\hat{\Sigma}\theta) + \lambda g(\theta), \tag{3}$$

where $g(\cdot)$ is a non-convex, sparsity-inducing surrogate objective. By simply setting $g(\theta) = \ell_1$ the problem is reduced to glasso. Classical non-convex penalties include smoothly clipped absolute deviation (SCAD) [16], minimax concave penalty (MCP) [17], arctangent penalty (ATAN) [18], and seamless ℓ_0 penalty (SELO) [19].

B. Electric Power Transmission Network

EPTNs are vast, geographically distrusted systems that distribute electrical power over long distance at a high voltage.



(a) Network Graph (b) Complete (c) Partial-Left (d) Partial-Right

Fig. 2: Depiction of the network graph and the corresponding flow graphs for each topology.

Their primary purpose of an EPTN is moving electrical power from energy generating locations (e.g., power plants) to electrical substations with very few connection points along the way. This is in contrast to distribution networks, which are local lines that transfer electrical power from substations to consumers with large numbers of connection points. The combination of the transmission and distribution networks is typically referred to as the electrical grid.

Fig. 1 depicts a very simple EPTN known as Kundur's two-area, four-machine symmetrical power system. This system is divided into two areas which serve as grouping mechanisms for shared control. Each area has two generators (i.e., power plants) and four buses (i.e., nodes) where an additional bus is used to join the two areas so that power might be exchanged. The gray shaded regions represent buses where the color of the bus number resents generators (green), transmission (orange), and distribution (red). The branches (i.e., edges) connecting the buses have their names given above the line and the distance given below. Each bus has a time synchronized PMU for measuring the voltage phasors (blue circle) and branch current phasors (green squares) where a GPS is used for time synchronization. A given bus can have multiple current phasor measurements, as each branch (i.e., edge) connected to a bus is assigned a current phasor measurement. A phasor measurement consists of a magnitude and angle measurement. Additionally, buses $(7, 8)$ and $(8, 9)$ have double transmission branches (i.e., multi-edges). Finally, the distribution buses 12 and 13 represent load consuming nodes (i.e., leaf nodes) which can be interpreted as substations feeding power to the distribution network. The typical power flow then is from the generators to the load consuming nodes.

III. Methodology

This section covers the simulated synchrophasor dataset, network graph modeling using GGMs, and network graph refinement for pruning false edges and constructing the corresponding flow graph. For the latter, a domain specific graph score algorithm is proposed which harnesses the physics of the EPTN contained within the synchrophasor dataset.

A. Simulated Synchrophasor Data

A synchrophasor dataset generated by a real-time digital simulator (RTDS), a multiprocess computer system for power system simulations, is employed to observe power flow over time. This dataset captures both voltage and current phasors in conjunction with the time-domain. This allows for a more realistic and high-fidelity dataset to be generated, unlike other works which rely on computer simulation software (e.g., MATPOWER) where data samples are generated by varying the load of the system and no current or time information can be captured [6, 7]. Further details regarding the data generation are provided by [3] although a brief overview of the data is given below.

The synchrophasor dataset is generated using a two-area four-machine power system model (Fig. 1) on a RTDS. PMUs collect data at a rate of 30 Hz where a total of 1800 samples are collected over 60 seconds. Data is collected only once the system has reached a steady state. There are three different network topologies (Fig. 2) which are denoted as complete, partial-left, and partial-right. The *complete* topology uses the original two-area four-machine network structure (Fig. 2b). The *partial-left* topology uses the same network structure but disconnects one of the dual transmissions branches between buses $(7, 8)$ (Fig. 2c). Likewise, the *partial-right* topology uses the same network structure, but disconnects one of the dual transmissions branches between buses $(8, 9)$ (Fig. 2d). Any "disconnected" branch is still reported in the data, but the current magnitude measurement is reported as a near zero value (i.e., no power flow). These two partial datasets can be used to test the detection of branches that no longer have any power flow (i.e., a topology change).

Additionally, each dataset is collected using three different operating conditions of low, medium, and high loads (exact power flow values are available in [3]). Each operating condition (i.e., load type) indicates power consumption by nodes 12 and 13. The low load signifies a low power demand, prompting generators to reduce power production. Conversely, high load signifies a high power demand, prompting generators to increase power production. The medium load then simply lies in between the low and high load power demand. Thus, the load type affects the entire network's power flow, simulating different power demands and assessing the viability of modeling the EPTN topology across various levels of demand.

A total of nine datasets are collected, representing three network topologies and three operating conditions for each topology. Each dataset consists of 83 features in total, where each bus has two voltage phasor (e.g., magnitude and angle) features and at least two current phasor features, with the total number of current features being determined by the number of branches per bus. Finally, while included in the data, the generators (i.e., buses 1-4) and distribution nodes (i.e., buses 12 and 13) will not be used in favor of focusing on the connectivity between the transmission nodes (i.e., buses 5-11).

B. Stage 1: Network Graph Modeling

In order to estimate the network graph structure from data, GGMs using convex and non-convex penalties are employed. In particular, the convex estimator glasso and estimators using the non-convex penalties SCAD, MCP, ATAN, and SELO[1] are utilized as these are well explored methods for approximating the sparse precision matrix θ [16, 17, 18, 19, 20, 11].

Typically, the input for these estimators is an empirical correlation matrix R with the dimensions $p \times p$ where $p = 7$ (i.e., the number of transmission buses). However, to capture the potential high dimensional interactions between the features, a rational quadratic kernel (RQK) is employed instead. RQK can be seen a mixture of radial basis function kernels and is defined as

$$RQK = \Big(1 + \frac{d(x_i, x_j)^2}{2\alpha l^2}\Big)^{-\alpha} \tag{4}$$

where α controls the scale mixture and l is the length scale of the kernel, and $d(\cdot,\cdot)$ is the Euclidean distance.

To compute the RQK similarity matrix, all current-related features (i.e., edge features) are excluded due to the non-uniform number of current phasor measurements per bus, leading to an inconsistency in the number of features per each node. Thus, only node-specific voltage phasor features are employed for this phase, leaving 14 voltage features in total (two phasor features per node) for the seven nodes (5-11). The data is then standardized across PMU features, meaning all magnitude features across all nodes are standardized together, and the same process is applied to the angle features. Using all 1800 data samples and 14 voltage features, the data is flattened into a matrix with dimensions equal to the number of nodes by voltage phasor measurements times the number data samples (e.g., $(7, 3600)$.[2]

Given the RQK similarity matrix as input, the GGM outputs an estimated precision matrix θ containing the partial dependencies. Transforming θ into the estimated network adjacency matrix $\hat{A}$ is done by setting all the non-zero values to one. Any index containing a zero represents the conditional independence between the respective two variables given all other node variables.

One caveat of the considered estimators is the requirement of hyperparameter selection. In particular, glasso requires the selection of one regularization parameter λ. Meanwhile, the non-convex estimators require the selection of two hyperparameters: the regularization parameter λ and the shaping parameter γ. To select the best model (i.e., hyperparameters), two different metrics are employed, which yield varying results. The first metric is the extended Bayesian information criterion (EBIC) [21]. A lower EBIC score indicates a potentially more "preferred" model, where scores are relative to the problem. The major benefit of EBIC is that it does not require the ground truth for computation. The second metric is the F$_\beta$-score, a generalized version of the F-score, where recall is considered β times as important as precision. However, the F$_\beta$-score requires access to the ground truth. For these controlled experiments, the underlying ground truth is available for both the network and flow graphs, yet this might not always be the case as GGMs are typically formulated as unsupervised methods.

C. Stage 2: Network Graph Refinement

Stage 2 refines the GGM network graph estimation and converts it into its corresponding flow graph. Recall, the output of the GGM estimator is an undirected simple graph, which is interpreted as a network graph. Therefore, the network graph needs to be converted into a directed multi-graph to represent a flow graph. Thus, when converting to a flow graph, direction needs to be determined and edges need to be expanded into multi-edges if they exist. Furthermore, the initial network graph estimation can include false edges, which will need to be pruned. All this can be achieved by using the power flow physics inherently contained within the synchrophasor dataset.

A novel domain specific algorithm, inspired by [3], denoted as the Power Network Graph Score (PNGS) is proposed for Stage 2. PNGS aims to harness the physics of the current phasor measurements, which could not be properly utilized in Stage 1. Current phasor measurements allow for three crucial concepts to be accounted for: 1) The number of in-flows (incoming edges) and out-flows (outgoing edges) for a particular node can be derived, thus determining the power flow direction of an edge; 2) A number of metrics can be computed using current and power differences to generate an edge score, where the higher the score, the more likely the edge is a false edge; 3) Multi-edges can be accounted for as each edge has a current phasor, thus an edge score can be computed for each edge. This can be done as long as the estimated network graph possesses a connection between two nodes.

The core idea of PNGS lies in computing a score for each edge in the estimated network graph. PNGS aims to learn a score matrix S of shape $p \times p$. S functions similarly to an adjacency matrix, but now each element contains scores for the potential edges determined by the estimated network graph adjacency matrix $\hat{A}$. Thus, for each pair of nodes, a set of scores is computed based on the potential edges between them and assigned to the corresponding node-pair index in S. Edges with lower scores are likely to represent true edges, while edges with higher scores are likely to represent false edges. A value of infinity is used to represent impossible edges (i.e., edges that can not exist based on the computed in-flows and out-flows of a node) while a value of NaN (i.e., Not a Number) is used to represent edges that are not in. Similar to an adjacency matrix, the rows of S represent out-flows and columns represent in-flows, with the main diagonal being NaNs to represent the lack of self-connections.

For PNGS to function, partial information about the true graph, denoted as $\bar{G}$, must be extracted using the current

[1]All estimators are executed using the implementations provided by the R package GGMncv [11].

[2]That is, 7 nodes, $2 \times 1800 = 3600$ features as there are 2 phasor measurements per node and 1800 data samples per dataset.

phasor information. Thus, $\bar{G}$ contains information concerning the total number of nodes, the total number of edges, and the number of edges for each node including the number of in-flows and out-flows. Computing the total edges contained within the true graph is equal to half the number of current phasor measurements, assuming no PMUs are taken offline. This is because each branch has a current measure taken at the beginning and end of the branch. Likewise, the number of edges for each node is equal to the number of current phasor measurements reported for a node divided by two[3] In-flows $\bar{G}^{ins}(n)$ and out-flows $\bar{G}^{outs}(n)$ can then be determined for each node by looking at the current angle such that if the current angle is greater than 180 degrees it is an in-flow while any angle less than 180 it is an out-flow[4].

Given $\bar{G}$ and $\hat{A}$, the edge score can be computed by looping over all nodes in $\bar{G}$ where n represents the current node. This is done by first checking the number of out-flows $\bar{G}^{outs}(n)$ for n. If n has no out-flows $\bar{G}^{outs}(n) = \emptyset$, no out-going edges can exist for n and the row indexed at n in S is set to infinity. Given n has out-flows, each connected node $\tilde{n}$ is then checked. If $\tilde{n}$ has no in-flows $\bar{G}^{ins}(\tilde{n}) = \emptyset$, no incoming edge can exist between $(n, \tilde{n})$ and the corresponding value for S is set to infinity. Given $\tilde{n}$ has in-flows, it must be determined if any of the out-flows $\bar{G}^{outs}(n)$ match with the in-flows $\bar{G}^{ins}(\tilde{n})$. Before any computation can be done, all possible pairs C between $\bar{G}^{outs}(n)$ and $\bar{G}^{ins}(\tilde{n})$ must be generated.

For each pair (out, in) in C, three scores proposed by [3] are computed using various aspects of the voltage and current phasor measurements. Absolute current flow difference (ACFD) aims to compute the absolute difference between the out-flow and in-flow current magnitudes:

$$\text{ACFD}(out, in) = \big||I^{out}| - |I^{in}|\big|, \tag{5}$$

where the I represents the current magnitude, out represents the out-flows of n, and in represents the in-flows of $\tilde{n}$. Absolute power flow difference (APFD) aims to compute the absolute different between the out-flow and in-flow power.

$$\begin{aligned}\text{APFD}(out, in) = &\big||V^{out} I^{out} \cos(\delta^{out} - \alpha^{out})| \\ &- |V^{in} I^{in} \cos(\delta^{in} - \alpha^{in})|\big|,\end{aligned} \tag{6}$$

where V represents the voltage magnitude, δ the voltage angle, α the current angle. Finally, the mean power loss (MPL) aims to compute the real power loss for a given power line:

$$\begin{aligned}\text{MPL}(out, in) = &\Big|\text{Re}\Big(\big(V^{out}\angle\delta^{in} - V^{in}\angle\delta^{in}\big) \\ &\cdot \big(\frac{I^{out}\angle\alpha^{out} + I^{in}\angle\alpha^{in}}{2}\big)^*\Big)\Big|,\end{aligned} \tag{7}$$

where Re denotes the real part and $*$ denotes the conjugate. For each equation, the mean is computed over the total number of data samples (i.e., time steps) m.

Once all pairs have been exhausted, Equations (5), (6), and (7) are summed and stored inside a local score matrix $\hat{S}$ with dimensions $|\bar{G}^{outs}(n)| \times |\bar{G}^{ins}(\tilde{n})|$. $\hat{S}$ is then stored with S at the index at $(n, \tilde{n})$ to represent all the scores for the potential connections between the two nodes. Given a completed S, refinement can be naively done by looking at each node's edge scores and taking the top k smallest scores where k is equal to the number of out-flows for a node. As mentioned, the idea is that the k top smallest edge scores should represent the true edges (i.e., smallest values). The result is then a multi-edge directed adjacency matrix. Algorithm 1 summarizes the PNGS process for Stage 2.

Algorithm 1: Power network graph score (PNGS)

```
1  Parameters: partial graph $\bar{G}$, estimated adjacency
     matrix $\hat{A}$
2  Initialize score matrices $S$ with shape $p \times p$
3  foreach $n \in \bar{G}$ do
4      $\hat{A}(n) \leftarrow$ set of nodes connected to $n$ in $\hat{A}$
5      $\bar{G}^{outs}(n) \leftarrow$ set of outgoing edges from $n$ in $G$
6      if $\bar{G}^{outs}(n) = \emptyset$        ▷ No outgoing edges
7       then
8          $S(n, \hat{A}(n)) \leftarrow \infty$
9          continue
10     foreach $\tilde{n} \in \hat{A}(n)$ do
11         $\bar{G}^{ins}(\tilde{n}) \leftarrow$ set of incoming edges for $\tilde{n}$ in $\bar{G}$
12         if $\bar{G}_{ins}(\tilde{n}) = \emptyset$        ▷ No incoming edges
13          then
14             $S(n, \tilde{n}) \leftarrow \infty$
15             continue
16         $C \leftarrow$ pairs between $\bar{G}^{outs}(n)$ and $\bar{G}^{ins}(\tilde{n})$
17         Initialize $\hat{S}$ with shape $|\bar{G}^{outs}(n)| \times |\bar{G}^{ins}(\tilde{n})|$
18         foreach $(out, in) \in C$ do
19             $\hat{S}(out, in) \leftarrow \text{APFD}(out, in) +$
                 $\text{ACFD}(out, in) + \text{MPL}(out, in)$
20         $S(n, \tilde{n}) = \hat{S}$
```

IV. Experiments

The experiments conducted within this section aim to assess the ability of GGMs and PNGS to reproduce the true network and flow graphs under various topology and load changes. As such, the goal is to explore how GGMs might fail and to test if using PNGS for graph refinement can accommodate for the shortcomings of GGMs[5].

For all experiments, the RKQ kernel parameters are set to $\alpha = 1$ and $l = 50$. Likewise, each GGM estimator will search over 100 different λ values, generated using the exponential log scale between 0.001 and 200. Thus, for each λ, a new model is fitted. The selection of γ is done by using each estimator's corresponding recommended value: SCAD $\gamma = 3.7$, MCP $\gamma = 2.0$, while the rest of the non-convex penalties use $\gamma = 0.01$. The default values produced relatively

[3]The division is done to account for each phasor measurement having two features (magnitude and angle).

[4]This method can not be applied to generators as their measurements are typically obfuscated by transformers.

[5]All results and code can be found at https://github.com/RL-BCI-Lab/gc4eptn.

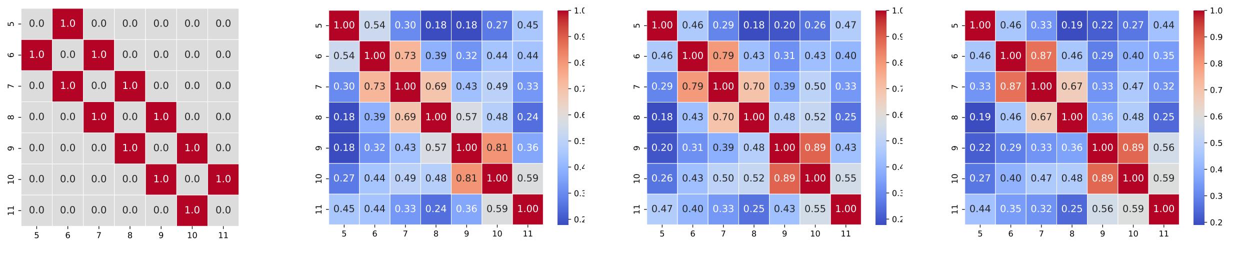

(a) Ground Truth (b) High (c) Medium (d) Low

Fig. 3: (a) The ground truth network graph adjacency matrix. The RQK similarity matrices computed for the complete topology are depicted using the (b) high load, (c) medium load, and (d) low laod. The x-axis and y-axis represent the bus/node number.

sufficient results, negating the need for additional computational resources [11]. The best network graph estimation is selected using the lowest EBIC and highest F_β-score. For F_β-score $\beta = 2$ is selected in order to put emphasis on retaining the underlying ground truth graph. This is crucial as the Stage 2 can not add edges, it can only validate and prune existing edges. Therefore, it is more desirable to have slightly more dense graphs that contain the underlying ground truth than sparse graphs that do not.

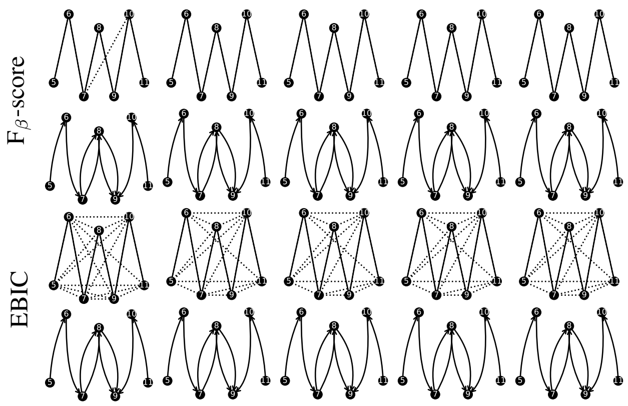
A. *High Loading Condition and Complete Topology*

Baseline experiments are conducted using the high load for the complete topology, allowing for an idealized EPTN setup to be tested. Before running Stage 1, it is useful to first observe the RQK similarity matrix, as it can provide insights into the potential difficulty of the network estimation problem. Fig. 3b depicts the RQK matrix for the high load. When compared with the ground truth adjacency matrix (Fig. 3a), it can be seen that the RQK matrix produces higher similarity values for nodes with edges (e.g., $(6, 5)$) versus those that do not (e.g., $(11, 5)$).

The network graph estimation for all GGM estimators and the corresponding refined flow graphs are depicted in Fig. 4. When selecting using F_β-score, all non-convex penalties result in an F_β-score of 1 while glasso results in an F_β-score of 0.97. Thus, we observe that using non-convex penalties enables estimating the true network graph directly. While glasso still requires refinement, it only estimates a single false edge. However, when selecting using EBIC, all non-convex penalties lead to an F_β-score of 0.71. On the other hand, glasso performs slightly worse with a resulting F_β-score of 0.68. The reasoning for the decreased F_β-score for all estimators is due to denser network graphs estimations. Although the recall scores for EBIC selected network graphs are perfect, the precision scores suffer significantly. This is the first indication that EBIC might not be well suited for selecting the best network graph estimations. Yet, F_β-score selection clearly shows that it is possible for GGMs to directly recover the true network graph.

B. *Topology and Load Changes*

Next, experiments using various combination between different topologies and loads are presented. To simplify the analysis of this section, specific focus is placed on the SELO, as it is one of the top performing penalties. Albeit, all GGM estimators do perform relatively similar to one another, with glasso having a tendency to produce more dense network graphs. In addition to the medium and low loads, a new "all" load type is included, which combines all three loads into one dataset. This is used to simulate the change of loads over time. As a consequence, the all load dataset contains 5400 data samples as compared to the normal 1800.



(a) Glasso (b) SCAD (c) MCP (d) ATAN (e) SELO

Fig. 4: Estimated network graphs (1st and 3rd rows) using each GGM estimator and their corresponding flow graphs (2nd and 4th rows) for the complete topology and high load type. Best network graph estimations are selected using F_β-score (1st and 2nd rows) and EBIC (3rd and 4th rows). The dotted lines for the network graph indicate edges pruned using PNGS.

Fig. 3 depicts the RQK similarity matrices for the complete topology and high, medium, and low loads. It can be observed that the similarity among the ground truth nodes is typically high. However, as the load decreases, some connected nodes start to lose their high similarity (e.g., $(5, 6)$ and $(8, 9)$). This drop in similarity indicates a potential problem as it could become harder to distinguish true connections from false connection as some nodes can have similar values, yet not be connected (e.g., $(5, 11)$ or $(11, 9)$). As such, this highlights the problem of value similarity between unconnected nodes within the synchrophasor dataset, which is likely to arise for other EPTN datasets as well.

Table I reports the performance for both network graph prediction and flow graph refinement using all topologies and load types, where the top SELO models are selected using F_β-score. For the most part, SELO can reconstruct the underlying ground truth graph with the addition of a few false edges, regardless of the load or topology type. This is indicated by the relatively high F_β-scores. When looking at the recall and precision metrics, it can be seen that the recall is almost always perfect, indicating the true network graph is contained within the estimation. Yet, precision tends to suffer as it ranges between 0.60 to 1 (a value of $.60$ translates to approximately 4 false edges). As almost all results report a perfect recall score, refinement using PNGS allows for the recovery of the true flow graph. For the perfect estimation cases which receive an F_β-score of 1 (e.g., high load, complete and partial-left connectives), Stage 2 only needs to add direction and expand edges into multi-edges.

However, there is a point of failure that can be observed. That is, as the load decreases (from high to low) the problem tends to become more difficult (i.e., higher likelihood of false similarities between nodes to arise). It can be seen that for the complete topology and low load that SELO does not properly maintain the underlying ground truth graph indicated by its lower recall score as it fails to predict the edge between $(8, 9)$. This is likely, due to its low RQK similarity of 0.36 (Fig. 3d). As a result, after refinement, this edge remains missing and the reported metrics are even lower. This is because the flow graph metrics are computed using the true directed multi-graph adjacency matrix and edge $(8, 9)$ has multi-edges, yet none were estimated. This highlights the limitation of PNGS for graph refinement, as it can not add edges retrospectively. Thus, placing emphasis on the importance of the initial network estimation.

Additional metrics regarding SELO model selection using EBIC are reported in Table II. It can be seen that the EBIC selected network graph estimations have lower F_β-scores. This is due to EBIC selecting more dense graph estimations, in turn leading to more false edges and a lower precision score. Even with the more dense graph predictions, EBIC too suffers from the same failure case as F_β-score when estimating the graph for the complete topology and low load. Furthermore, EBIC selection fails to ever recover the exact underlying true network graph for any topology or load.

Finally, a contradiction arises when the EBIC values for the same topologies and loads are compared across Table I and Table II. In particular, it can be seen that while Table II reports lower EBIC values, yet it reports worse F_β-scores (specifically due to precision). For example, take the complete topology with a high load, where the EBIC selected network graph estimation reports an EBIC value of 5431.85 while the F_β-score selected network graph estimation reports an EBIC value of 7505.18. The objectively better estimation is the latter, yet it receives a higher EBIC value. This indicates that a lower EBIC value does not represent a "better" network graph estimation. Thus, a more representative metric that does not require knowledge of the underlying ground truth is required.

TABLE I: All topologies and load results using SELO where the best model is selected using F_β-score.

Connec-tivity	Loads	Network Graph				Flow Graph		
		F_β	Precision	Recall	EBIC	F_β	Precision	Recall
Complete	High	1.00	1.00	1.00	7505.18	1.00	1.00	1.00
	Med	0.91	0.67	1.00	4981.20	1.00	1.00	1.00
	Low	0.83	0.83	0.83	5713.06	0.79	1.00	0.75
	All	1.00	1.00	1.00	28999.48	1.00	1.00	1.00
Partial-Left	High	1.00	1.00	1.00	5603.90	1.00	1.00	1.00
	Med	0.97	0.86	1.00	4559.94	1.00	1.00	1.00
	Low	0.88	0.60	1.00	1964.18	1.00	1.00	1.00
	All	1.00	1.00	1.00	26821.58	1.00	1.00	1.00
Partial-Right	High	0.91	0.67	1.00	4806.79	1.00	1.00	1.00
	Med	0.88	0.60	1.00	3462.52	1.00	1.00	1.00
	Low	0.88	0.60	1.00	1472.08	1.00	1.00	1.00
	All	0.91	0.67	1.00	26361.21	1.00	1.00	1.00

TABLE II: All topologies and load results using SELO where the best model is selected using EBIC.

Connec-tivity	Loads	Network Graph				Flow Graph		
		F_β	Precision	Recall	EBIC	F_β	Precision	Recall
Complete	High	0.71	0.33	1.00	5431.85	1.00	1.00	1.00
	Med	0.81	0.46	1.00	4442.88	1.00	1.00	1.00
	Low	0.69	0.42	0.83	3875.21	0.79	1.00	0.75
	All	0.79	0.43	1.00	27666.75	1.00	1.00	1.00
Partial-Left	High	0.79	0.43	1.00	4172.09	1.00	1.00	1.00
	Med	0.81	0.46	1.00	3230.33	1.00	1.00	1.00
	Low	0.83	0.50	1.00	1475.61	1.00	1.00	1.00
	All	0.86	0.55	1.00	25769.53	1.00	1.00	1.00
Partial-Right	High	0.77	0.40	1.00	4105.60	1.00	1.00	1.00
	Med	0.81	0.46	1.00	3204.76	1.00	1.00	1.00
	Low	0.75	0.38	1.00	1327.54	1.00	1.00	1.00
	All	0.83	0.50	1.00	25807.94	1.00	1.00	1.00

V. Limitations

As this work is highly exploratory, there are a handful of limitations that need to be addressed before application in the real-world. While the proposed graph construction and refinement can help recover the true graph from data, it is observed that the metric used to select the "best" graph affects the quality of the chosen estimated network graph. While only GGMs were tested for network graph estimation, any other data-driven approach is vulnerable to this hyperparameter and graph selection challenge. Thus, developing a new metric, specifically one that does not require a ground truth, for EPTN evaluation acts as a vital future point of investigation. For instance, utilizing the PNGS as a metric for model selection, not just for refinement, could act as a starting point. Moreover, due to the simplicity of the simulated EPTN provided here, more complex load settings and network topology experiments are needed to further solidify the robustness of the proposed graph construction and refinement algorithm based on GGMs. Finally, although the estimations do not need to occur at the sampling rate of the PMUs, the bounds at which the proposed algorithm can produce estimations needs to be further explored. In particularly, algorithmic speed needs to be investigated on various larger network topologies and contrasted against prior domain specific approaches.

VI. Conclusion

A data-driven power flow graph construction of a simulated electric power transmission network, the Kundur's two-area four-machine power system, has been investigated in this paper. The two-stage data driven graph construction leverages GGMs using convex and non-convex penalties to achieve an initial sparse estimation of the network graph and further refinement to discover a power flow graph using the novel physics-based PNGS algorithm.

VII. Acknowledgment

This work has been supported by the National Science Foundation under Grant ECCS 2234031 and ECCS 2234032. The opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

References

[1] D. Madurasinghe and G. K. Venayagamoorthy, "An Efficient and Reliable Electric Power Transmission Network Topology Processing," *IEEE Access*, vol. 11, pp. 127 956–127 973, 2023.

[2] M. Kezunovic, "Monitoring of Power System Topology in Real-Time," in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)*, vol. 10, Jan. 2006, pp. 244b–244b.

[3] G. K. Venayagamoorthy, D. Madurasinghe, and R. Ratnakumar, "Graph Models of Electric Power Transmission Networks Using Synchrophasor Data," in *SoutheastCon*, Mar. 2024, pp. 592–597.

[4] M. Farrokhabadi and L. Vanfretti, "State-of-the-art of topology processors for EMS and PMU applications and their limitations," in *38th Annual Conference on IEEE Industrial Electronics Society*, Oct. 2012, pp. 1422–1427.

[5] W. Liao, B. Bak-Jensen, J. R. Pillai, Y. Wang, and Y. Wang, "A Review of Graph Neural Networks and Their Applications in Power Systems," *Journal of Modern Power Systems and Clean Energy*, vol. 10, no. 2, pp. 345–360, Mar. 2022.

[6] C. Dinesh, J. Wang, G. Cheung, and P. Srikantha, "Complex Graph Laplacian Regularizer for Inferencing Grid States," in *IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids*, Oct. 2023, pp. 1–6.

[7] X. Pan, M. Chen, T. Zhao, and S. H. Low, "DeepOPF: A Feasibility-Optimized Deep Neural Network Approach for AC Optimal Power Flow Problems," *IEEE Systems Journal*, vol. 17, no. 1, pp. 673–683, Mar. 2023.

[8] M. Drton and M. H. Maathuis, "Structure learning in graphical modeling," *Annual Review of Statistics and Its Application*, vol. 4, no. 1, pp. 365–393, 2017.

[9] M. Altenbuchinger, A. Weihs, J. Quackenbush, H. J. Grabe, and H. U. Zacharias, "Gaussian and Mixed Graphical Models as (multi-)omics data analysis tools," *Biochimica et Biophysica Acta*, vol. 1863, no. 6, Jun. 2020.

[10] M. Dyrba, R. Mohammadi, M. J. Grothe, T. Kirste, and S. J. Teipel, "Gaussian Graphical Models Reveal Inter-Modal and Inter-Regional Conditional Dependencies of Brain Alterations in Alzheimer's Disease," *Frontiers in Aging Neuroscience*, vol. 12, 2020.

[11] D. R. Williams, "Beyond lasso: A survey of nonconvex regularization in gaussian graphical models," 2020.

[12] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[13] O. Banerjee, L. El Ghaoui, and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data," *The Journal of Machine Learning Research*, vol. 9, pp. 485–516, 2008.

[14] J. Fan, Y. Feng, and Y. Wu, "Network exploration via the adaptive lasso and scad penalties," *The Annals of Applied Statistics*, vol. 3, no. 2, p. 521, 2009.

[15] J. Ying, J. V. de Miranda Cardoso, and D. Palomar, "Nonconvex sparse graph learning under laplacian constrained graphical model," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7101–7113, 2020.

[16] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle properties," *Journal of the American Statistical Association*, vol. 96, no. 456, pp. 1348–1360, 2001.

[17] C.-H. Zhang, "Nearly unbiased variable selection under minimax concave penalty," *The Annals of Statistics*, vol. 38, no. 2, pp. 894–942, 2010.

[18] Y. Wang and L. Zhu, "Variable selection and parameter estimation with the atan regularization method," *Journal of Probability and Statistics*, vol. 2016, 2016.

[19] Z. Li, S. Wang, and X. Lin, "Variable selection and estimation in generalized linear models with the seamless l0 penalty," *The Canadian journal of statistics*, vol. 40, no. 4, pp. 745–769, 2012.

[20] Y. Wang, Q. Fan, and L. Zhu, "Variable selection and estimation using a continuous approximation to the ℓ_1 penalty," *Annals of the Institute of Statistical Mathematics*, vol. 70, no. 1, pp. 191–214, 2018.

[21] R. Foygel and M. Drton, "Extended Bayesian Information Criteria for Gaussian Graphical Models," in *Advances in Neural Information Processing Systems*, vol. 23. Curran Associates, Inc., 2010.