2024 IEEE 63rd Conference on Decision and Control (CDC)
December 16-19, 2024. MiCo, Milan, Italy

Resilient Infrastructure Network: Sparse Edge Change Identification via ℓ_1-Regularized Least Squares

Rajasekhar Anguluri

***Abstract*— Adversarial actions and a rapid climate change are disrupting operations of infrastructure networks (e.g., energy, water, and transportation systems). Unaddressed disruptions lead to system-wide shutdowns, emphasizing the need for quick and robust identification methods. One significant disruption arises from edge changes (addition or deletion) in networks.**

We present an ℓ_1-norm regularized least-squares framework to identify multiple but sparse edge changes using noisy data. We focus only on networks that obey equilibrium equations, as commonly observed in the above sectors. The presence or lack of edges in these networks is captured by the sparsity pattern of the weighted, symmetric Laplacian matrix, while noisy data are node injections and potentials. Our proposed framework systematically leverages the inherent structure within the Laplacian matrix, effectively avoiding overparameterization.

We demonstrate the robustness and efficacy of the proposed approach through a series of representative examples, with a primary emphasis on power networks.

I. Introduction

Critical infrastructure network systems are those that are important to our societies [1]. But for us, any such system is a graph having nodes and edges linking those nodes. At the basic level, many infrastructure networks satisfy conservation laws[2] (a more appropriate word is equilibrium equations). These equations state: flows entering a network are neither created nor destroyed. Flows may be power, traffic, or fluid. The laws we study apply in the steady-state regime and are linear (governed by algebraic relations between edge flows and flows injected by the nodes) [2].

Identifying edge changes—additions, removals, or both—in networks means pinpointing node indices of the changed edges from noisy measurements. For finite-dimensional networks obeying equilibrium equations, these indices can be deduced from the sparsity pattern (zero and non-terms) of the symmetric Laplacian matrix. Specifically, the indices of the nodes of a removed edge is indicated by an off-diagonal element in the Laplacian that changes from non-zero to zero after the removal. A related but very different problem that does not concern is the topology estimation: estimate edges present between all pairs of nodes (see [3], [4], [5]).

The author works with the Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County, MD 85281, USA (e-mail: rajangul@umbc.edu). The author dearly thanks ChatGPT for enhancing the clarity and conciseness of the sentences.

[1]United States of Department of Homeland Security lists sixteen sectors, including energy, water, manufacturing, as critical infrastructure systems [1]. Other countries might have an equivalent or even expanded list.

[2]Our usage of conservation laws is morally correct but is not rigorous as perceived by a theoretical physicist or a pure mathematician.

We restrict our attention to the undirected networks. Thus, the Laplacian is a symmetric, weighted matrix. Suppose that a disruptive event happens and certain edges are changed in an otherwise known reference network (we chiefly focus on edge removal exclusively and briefly comment on additions). Then, our goal is to identify those edge changes using noisy measurements of node potentials and injected flows.

A naive approach involves estimating the entire network and comparing it with a reference, but this is inefficient when changes are sparse compared to the total number of edges. A more effective approach is to estimate these sparse edge changes directly. To achieve this, we develop a sparsity-based estimation framework, with the following contributions:

1) Using the equilibrium equations, we develop a sparse error-in-variable type linear regression model in which the unknown sparse regression vector contains information about the edge changes. We take complete account of the structure implicit in the Laplacian matrix, thereby avoiding over-parameterization of the unknown vector.
2) We develop two estimators to estimate the sparse vector. The first is the ℓ_1-norm regularized total least squares (TLS) estimator. To overcome the difficulties inherent in implementing TLS type estimators, in Theorem 4.1, we develop an equivalent reformulation, which allows for fast heuristics like proximal-gradient in [6].
3) The second is the ℓ_1-norm regularized least squares (the LASSO estimator). This estimator is applicable when the error in variables is not high (which happens when the noise in injected flows is not too large).

We validate LASSO's performance on a synthetic network and IEEE benchmark power systems. Despite our preliminary theoretical characterization of the sparse TLS estimator, we did not validate its performance empirically. This is due to the absence of standard solvers for implementing the TLS estimator, unlike the readily available ones for the LASSO estimator. Nonetheless, we identify this as an area for future exploration and development.

Related Research: The authors in [7] use the eigenspectrum of the Laplacian matrix to discern topological variations in the network. In [8], [9], the authors provide an analytical characterization of the Laplacian matrix under edge additions and removals. Instead, [10] uses sparse covariance matrix techniques to learn edge changes. Hypothesis testing-based identification methods are considered in [11], [12]. Finally, edge identification in linear dynamical networks is considered in [13], [14], [15]. Studies in references [16], [17] are close to our approach. They suggest using LASSO and sparse

979-8-3503-1632-2/24/$31.00 ©2024 IEEE

TLS-type estimation for the topology estimation, not for the edge change identification problem.

II. Model and Definitions

A. Steady-state Model for Infrastructure Networks

We abstract any infrastructure network as a graph, having n nodes connected (or not) by m edges. The graph has no multiple edges between nodes and no self-loops. Specify an orientation for each edge; that is, one node is the "starting" node, and the other is the "end" node. The incidence matrix $A \in \{-1, 0, 1\}^{m\times n}$ indicates which nodes belong to which edge. Suppose the i-th edge's starting node is j and the ending node is k. Then the i-th row of A has -1 in column j and $+1$ in column k. Thus, $A\mathbb{1}_n = 0$.

Associate numbers $u_1, u_2, \ldots, u_n$ (called potentials) for nodes, numbers $w_1, w_2, \ldots, w_m$ (called flows) and positive weights $c_1, c_2, \ldots, c_m$ for edges in the graph. The potentials could represent the heights of nodes (in geodetic leveling networks), the pressures (in hydraulic networks), the voltages (in electrical networks), or the mass positions (in spring-mass networks). The flows could represent branch currents in electrical networks or the flow of commodities in hydraulic, traffic, or manufacturing networks. The inverse of the weights measures the resistance offered by edges to the flows (for e.g., in electrical networks, the weights are conductance).

Let $u = (u_1, u_2, \ldots, u_n)'$ be the vector of node potentials; $e = (e_1, e_2, \ldots, e_m)'$ the vector of edge potentials (potential difference on edges); $w = (w_1, w_2, \ldots, w_m)'$ be the vector of flows going through the edges; and $f = (f_1, f_2, \ldots, f_n)$ be the vector of injected flows. Define the diagonal matrix $C \in \mathbb{R}^{m\times m}$ whose diagonal entries specify weights given to the edges. Then, several infrastructure networks satisfy:

- *Kirchhoff's law for edge flows:* $A^{\mathsf{T}}w = f$.
- *Kirchhoff's law for potentials:* $e = -Au$.
- *Constitutive relations (Ohm's or Hooke's law):* $w = Ce$.

These equations hold for the vectors w, u, and f at any time instant. Let these vectors to be time-varying and combine the laws into one equilibrium (or balanced) equation:

$$f(t) = A^{\mathsf{T}}CAu(t) = Lu(t). \tag{1}$$

Here, $L = A^{\mathsf{T}}CA$ is the $n\times n$ weighted symmetric Laplacian matrix. The equation in (1) holds for complex-valued vectors and matrices (for e.g., AC power networks).

The incidence matrix A tells us edges incident to the nodes and the diagonal C tells us the weights associated with the edges in the graph. By construction, the Laplacian L specifies both the incident edges and the weights linked to these edges. Thus, the sparsity pattern of L (indicating the positions of zero and non-zero entries) directly reveals the presence or absence of edges between nodes. Indeed, let l_{ij} be the weight of the edge linking nodes i and j (this weight can be read off from the edge weight diagonal matrix C). Then,

$$L_{i,j} := \begin{cases} -l_{i,j} & \text{if } i \neq j \\ \sum_{k\neq i} l_{i,k} & \text{if } i = j \end{cases}. \tag{2}$$

If the nodes i and j have no edge, then $l_{i,j} = 0$. The sparsity pattern (indices of zero and non-zero entries) of L gives the edges in the network and calls this pattern the topology.

B. Edge Changes Identification Problem

The edge changes identification problem differs from the topology identification, focusing on determining the presence or absence of edges in the operating network compared to a reference. This reference network is a predetermined configuration known to the practitioner or the network configuration from the past, perhaps a few hours or days ago. We focus on the edge removal scenario because this is the most common concern in infrastructure networks. Examples include: line loss in power networks or leaky pipes in hydraulic networks. However, it is easy to extend our results to the edge addition and the mixed case, which involves additions and removals.

Let L_0 and L_1 be Laplacians of the infrastructure network before and after a change[3]. Let $\Delta L = L_1 - L_0$. This sparsity pattern of Δ_L denotes edges removed after the change (see 1 for an illustration and [9] for an algebraic characterization of Δ_L). After the change, the balanced equation obeys

$$f(t) = L_1u(t) = (L_0 + \Delta L)u(t). \tag{3}$$

For any t, let $\tilde{u}(t) = u(t) - \Delta u(t)$ and $\tilde{f}(t) = f(t) - \Delta f(t)$ be the noisy measurements of the potentials $u(t)$ and injected flows $f(t)$ obeying the model in (3). Here, $\Delta u(t)$ and $\Delta f(t)$ are measurement errors. Use of the negative symbols in front of the errors will be clear in Section III.

Assumption 2.1: The pre-change Laplacian matrix L_0 and the noisy data after the change $\{\tilde{u}(t), \tilde{f}(t)\}$ are known.

Assuming 2.1, the edge changes identification problem is to estimate the true sparsity pattern of ΔL using $\{\tilde{u}(t), \tilde{f}(t)\}$ collected over a finite time horizon after the change. Fig. 1 states the problem pictorially for edges removed or missing from the reference network with Laplacian L_0.

III. A Sparse Linear Model Formulation for Edge Changes Identification

This section introduces a sparse total least squares (hereafter, sparse TLS) framework to estimate the sparsity pattern of ΔL. We take account of the structure implicit in the Laplacian, avoiding over-parameterization in the TLS framework.

Let $t = 1, \ldots, T$, and define the matrices of node potentials and injected flows after the change:

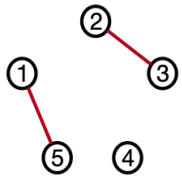
$$\begin{aligned} U_T &= [u(1), u(2), \ldots, u(T)] \in \mathbb{R}^{n\times T} \\ F_T &= [f(1), f(2), \ldots, f(T)] \in \mathbb{R}^{n\times T}. \end{aligned} \tag{4}$$

Then, the equilibrium equation in (3) over the discrete time horizon can be succinctly expressed as

$$F_T = (L_0 + \Delta L)U_T \tag{5}$$

[3]We assume that there is an inbuilt detection mechanism that alarms us if there is an edge change or (are changes) in the network, but what is unknown a priori is how many or which edges were changed.

(a) pre-change (L_0) (b) after-change (L_1)



(c) difference ($\Delta_L = L_1 - L_0$)

Fig. 1: **problem statement (illustrative)**: The graph underlying the Laplacian L_0 before the change with the network system is in (a). The Laplacian after edge changes is L_1, and is in (b). We want to directly estimate the graph underlying $\mathbf{\Delta}_{\mathbb{L}}$ in (c) (equivalently its sparsity pattern) using node potentials and injected flows.

Similar to the matrices U_T and F_T defined earlier, construct the $n \times T$ matrices $\tilde{U}_T$, $\tilde{F}_T$, ΔU_T, and ΔF_T using the vectors $\tilde{u}(t)$, $\tilde{f}(t)$, $\Delta u(t)$, and $\Delta f(t)$. By definition $\tilde{F}_T = F_T - \Delta F_T$ and $\tilde{U}_T = U_T - \Delta U_T$. After some manipulations (5) becomes

$$\tilde{F}_T + \Delta F_T = (L_0 + \Delta L)(\tilde{U}_T + \Delta U_T), \tag{6}$$

which is the so-called error-in-variable (EIV) model. In this model, $\tilde{F}_T$, $\tilde{U}_T$, and L_0 are known while the data errors ΔU_T and ΔF_T, the difference matrix ΔL are unknown.

Assumption 3.1: The number of edge changes are sparse; that is, $\|\Delta L\|_0 \ll n^2$, where $\|\cdot\|_0$ is the counting norm.

Assumption 3.1 is reasonable for infrastructure systems as these are built to withstand extreme conditions. However, due to adversaries or unexpected operating conditions, a few edges might undergo disruptions.

Under Assumptions in 2.1 and 3.1, a straightforward way to estimate Δ_L in the presence of data errors is to solve the optimization problem:

$$\begin{aligned} \underset{\Delta L = \Delta L^{\mathsf{T}},\, \Delta F_T,\, \Delta U_T}{\arg\min} \quad & \|\Delta F_T\|_F^2 + \|\Delta U_T\|_F^2 + \lambda \|\Delta L\|_0 \\ \text{subject to} \quad & \text{EIV model in (6).} \end{aligned} \tag{7}$$

Here $\|A\|_F = \sqrt{\mathrm{tr}(AA^{\mathsf{T}})}$ is the Frobenius norm and $\lambda > 0$ is the regularization parameter. However, the problem in (7) is a non-convex, combinatorial optimization, which is hard to solve in practice. A computationally superior alternative is to consider a convex relaxation to (7) by replacing $\|\cdot\|_0$-norm with the ℓ_1 norm; that is $\|\Delta_L\|_1 = \sum_{i,j} |\Delta_L(i,j)|$.

While the ℓ_1-relaxed problem is computationally simpler to solve, there are still problems with the naive formulation in (7). First, the optimization in (7) accounts for symmetry and sparsity of Δ_L but fails to account the structured sparsity of Δ_L (that is, zeros of ΔL correspond to presence or absence of edges common to L_0 and L_1). To systematically exploit these properties, we convert the matrix-valued EIV in (6) to the matrix-vector form in which ΔL is a vector.

A. Vectorization of the EIV model

To reduce the notation clutter, we rewrite $L_0 + \Delta_L$ as L_1 in (6). Recall that $A \otimes B$ is the Kroneckor product of matrices A and B. Applying $\mathrm{Vec}(\cdot)$ on both sides of (6) gives

$$\begin{aligned} \mathrm{Vec}(\tilde{F}_T + \Delta F_T) &= \mathrm{Vec}(L_1(\tilde{U}_T + \Delta U_T)) \\ &= ((\tilde{U}_T + \Delta U_T)^{\mathsf{T}} \otimes \mathbb{I}_n)\mathrm{Vec}(L_1) \\ &= [(\tilde{U}_T^{\mathsf{T}} \otimes \mathbb{I}_n) + ((\Delta U_T)^{\mathsf{T}} \otimes \mathbb{I}_n)]\mathrm{Vec}(L_1). \end{aligned} \tag{8}$$

The $n \times n$ symmetric matrix L_1 in (2), including diagonal, has $n(n+1)/2$ free parameters and $n(n-1)/2$ repeats. We assume these repeats are from the lower triangular portion of L_1. Thus, $\mathrm{Vec}(L_1)$ is n^2-dimensional vector with $n(n-1)/2$ repeats. Let $\mathrm{Vech}(L_1)$ be the $n(n+1)/2$-dimensional vector obtained by deleting the repeats in $\mathrm{Vec}(L_1)$. Formally, depending on n, there exists a unique duplication matrix D with $\mathrm{Vec}(L_1) = D\mathrm{Vech}(L_1)$ [18]. The elimination matrix E reverses the duplication operation: $\mathrm{Vech}(L_1) = E\mathrm{Vec}(L_1)$. Algebraic properties of D and E are documented in [18].

We express $\mathrm{Vech}(L_1))$ in L_0 and ΔL using the linearity of the vectorization operator. Thus, $\mathrm{Vech}(L_1) = \mathrm{Vech}(L_0) + \mathrm{Vech}(\Delta L)$. Further, owing to the linearity, the left-hand side of (8) can also be expanded as $\mathrm{Vec}(\tilde{F}_T + \Delta F_T) = \mathrm{Vec}(\tilde{F}_T) + \mathrm{Vec}(\Delta F_T)$. Putting the pieces together, write (8) as

$$\begin{aligned} &\mathrm{Vec}(\tilde{F}_T) + \mathrm{Vec}(\Delta F_T) = \\ &\quad [\tilde{U}_T^{\mathsf{T}} \otimes \mathbb{I}_n + (\Delta U_T)^{\mathsf{T}} \otimes \mathbb{I}_n] D [\mathrm{Vech}(L_0) + \mathrm{Vech}(\Delta L)]. \end{aligned}$$

We drop the subscript T and express the above equality using the compact notation (defined in the table below):

$$y + \Delta y = (X + \Delta X)(\beta_0 + \beta), \tag{9}$$

where X, y, and β_0 are known; ΔX and Δy are unknown data errors; and β is the unknown sparse vector.

TABLE I: Summary of matrices and vector notations

Matrices	Vectors
	$\beta_0 = \mathrm{Vech}(L_0)$
$X = (\tilde{U}_T^{\mathsf{T}} \otimes \mathbb{I}_n)D$	$\beta = \mathrm{Vech}(\Delta L)$
$\Delta X = ((\Delta U_T)^{\mathsf{T}} \otimes \mathbb{I}_n)D$	$y = \mathrm{Vec}(\tilde{F}_T)$
	$\Delta y = \mathrm{Vec}(\Delta F_T)$

In the passage preceding Section III-A, we mentioned that ΔL is structured sparse: the non-zeros of Δ_L correspond to the edges either absent or present in the network before the change. To exploit this sparsity, without loss of generality partition[4] β_0 as $\beta_0 = [\beta_{0,s}^{\mathsf{T}}\, 0^{\mathsf{T}}]^{\mathsf{T}}$, where β_s contains non-zero terms in the lower triangular part of L_0 (including diagonals). Then β can be partitioned as $[\beta_s^{\mathsf{T}}\, 0^{\mathsf{T}}]^{\mathsf{T}}$. But unlike the dense[5]

[4]This can be done by renumbering the nodes such that β_s contain non-zero terms in L_0. Alternatively, we can introduce a permutation matrix Γ such that $\beta_0 = \Gamma[\beta_{0,s}^{\mathsf{T}}\, 0^{\mathsf{T}}]^{\mathsf{T}}$.

[5]By construction, $\beta_{s,0}$ will not even have one zero entry; see Fig. 2.

$\beta_{s,0}$, the vector β_s is sparse. Thus, (9) becomes

$$y + \Delta y = \left(\begin{bmatrix} X_s & X_{s'} \end{bmatrix} + \begin{bmatrix} \Delta X_s & \Delta X_{s'} \end{bmatrix}\right) \begin{bmatrix} \beta_{s,0} + \beta_s \\ 0 \end{bmatrix}, \\ = (X_s + \Delta X_s)(\beta_{s,0} + \beta_s) \quad (10)$$

The model in 10 is minimal in that there are no redundant parameters in β_s or edges not present in the network. For example, in the IEEE 118 bus system, $n = 118$; but there are only 358 edges. Thus, dimension of β_0 is $n(n+1)/2 = 7021$. Instead, the dimension of β_s (including diagonals) is 476, which is small. (See Fig. 2 for a visual description). The savings in dimensionality become evident in large-scale systems with sparse connections and fewer edge changes.

IV. Solution Strategies

The techniques in this section hold for models in (9) and (10). For ease of notation, we stick with the model in (9).

Suppose that data errors Δy and ΔX are non-negligible. Then, a sparse estimate of β can be obtained by solving the sparse variant of the TLS problem:

$$\textbf{(P.1)} \quad \hat{\beta} = \underset{\Delta y,\, \Delta X,\, \beta}{\arg\min} \quad \|\Delta y\|_2^2 + \|\Delta X\|_F^2 + \lambda \|\beta\|_1 \\ \text{subject to} \quad y + \Delta y = (X + \Delta X)(\beta_0 + \beta).$$

The regularizer is $\|\beta\|_1 = \sum_i |\beta_i|$. The changed edges can be identified from the support of $\hat{\beta}$ returned by solving (P.1). Note that (P.1) is not equivalent to the naive optimization in (7) because we accounted for the structured sparsity of ΔL in the former problem via the vector β or β_s.

The TLS problem without the ℓ_1-norm regularizer is non-convex due to $\Delta X \beta_0$ in the equality constraint. Nonetheless, the unique solution is computed in a closed form using the singular value decomposition (SVD) of the matrix $[\Delta X \ \Delta y]$ [19]. Unfortunately, such SVD-type solutions do not exist in the presence of the ℓ_1-regularizer term. Recently, [20] suggested a reformulation for a problem akin to (P.1), allowing the use of greedy methods. We generalize [20, Lemma 2.1].

***Theorem* 4.1:** The constrained optimization problem in **(P.1)** is equivalent to solving the unconstrained optimization problem involving only the variable β:

$$\hat{\beta} = \underset{\beta}{\arg\min} \frac{\|X(\beta_0 + \beta) - y\|_2^2}{1 + \|(\beta_0 + \beta)\|_2^2} + \lambda \|\beta\|_1 \quad (11)$$

Proof: We follow the technique in [20, Lemma 2.1]. Let $v = \mathrm{Vec}([\Delta X \ \Delta y])$ and write the first two terms in the cost function of (P.1) as $\|\Delta y\|_2^2 + \|\Delta X\|_F^2 = [\Delta X \ \Delta y]_F^2 = \|v\|_2^2$. Next, rewrite the constraint in (P.1):

$$y - X(\beta_0 + \beta) = \Delta X(\beta_0 + \beta) - \Delta y \\ / \begin{bmatrix} \Delta X & \Delta y \end{bmatrix} \begin{bmatrix} (\beta_0 + \beta) \\ -1 \end{bmatrix}.$$

Using the crucial fact that $Ax = \mathrm{Vec}(Ax) = (x^{\mathsf{T}} \otimes I)\mathrm{Vec}(A)$ for some matrix A and x, we have

$$y - X(\beta_0 + \beta) = \underbrace{\left([((\beta_0 + \beta))^{\mathsf{T}} \ -1] \otimes I\right)}_{G(\beta)} v. \quad (12)$$

For any fixed β, the ℓ_1-norm constrained can be dropped in (P.1), and the resulting optimization problem then becomes $\min_v \|v\|_2^2$ subject to the constraint in (12)—the least norm problem. The solution to this problem is $v(\beta) = G(\beta)^{\dagger}[y - X(\beta_0 + \beta)]$. Using the special structure of $G(\beta)$ we further simplify the pseudoinverse $G^{\dagger}(\beta)$ as

$$G^{\dagger}(\beta) = G^{\mathsf{T}}(\beta)[G(\beta)G^{\mathsf{T}}(\beta)]^{-1} \\ = G^{\mathsf{T}}(\beta)[(1 + \|(\beta_0 + \beta)\|_2^2) \otimes I]^{-1} \\ = G^{\mathsf{T}}(\beta)(1 + \|(\beta_0 + \beta)\|_2^2)^{-1}.$$

Thus, $v(\beta) = (1 + \|(\beta_0 + \beta)\|_2^2)^{-1} G^{\mathsf{T}}(\beta)[y - X(\beta_0 + \beta)]$ for fixed β. Plugging this expression of $v(\beta)$ back in $\|v\|_2^2$, and simplifying it, gives the equivalent form in (11). ■

Compared to (P.1), the optimization in (11) is computationally simpler since it involves only one vector variable β. Nevertheless, the problem remains non-convex and necessitates heuristic methods for its resolution. We leave this task to future work as it falls outside the scope of this paper.

A. ℓ_1-regularized least squares: LASSO estimator

Suppose that the error ΔX is small compared to Δy, then we may ignore it from (6) to get

$$y - X\beta_0 = X\beta - \Delta y. \quad (13)$$

Since X and β_0 are known, the left-hand side is completely determined. Thus, the model in (13) is a simple linear model with the additive noise term Δy. Then, the sparse estimate of β can obtained by solving the ℓ_1-regularized least squares problem (or familiarly called the LASSO estimator):

$$\textbf{(P.2)} \quad \hat{\beta} = \underset{\beta}{\arg\min} \ \frac{1}{2}\|y - X\beta_0 - X\beta\|_2^2 + \lambda \|\beta\|_1 \quad (14)$$

Unlike the optimization problems we have seen thus far, (P.2) is a convex problem, and several numerical methods are proposed in the literature to compute $\hat{\beta}$ [21]. The most reliable and fast method is the coordinate descent method (available in standard software packages such as MATLAB).

V. Simulations

We analyze the performance of the LASSO estimator (14) on a synthetic network and on three real-world power system networks (IEEE-57, IEEE-118, IEEE-145; the number here denotes the total nodes). We set $T = 30$ (the measurement window) and generate error terms $\Delta u(t)$ and $\Delta f(t)$ in (5) using zero-mean Gaussian distribution with variance 0.1.

We rely on the indexes below to quantify the performance of the estimator. The symbol $\wedge$ is the AND operator.

- TP: proportion of the removed edges correctly identified ($\widehat{\beta}(i) \neq 0 \wedge \beta(i) \neq 0$)
- TN: proportion of the non-removed edges correctly identified ($\widehat{\beta}(i) = 0 \wedge \beta(i) = 0$)
- FN: proportion of the removed edges incorrectly identified ($\widehat{\beta}(i) \neq 0 \wedge \beta(i) = 0$)
- FP: proportion of the non-removed edges incorrectly identified ($\widehat{\beta}(i) = 0 \wedge \beta(i) \neq 0$).

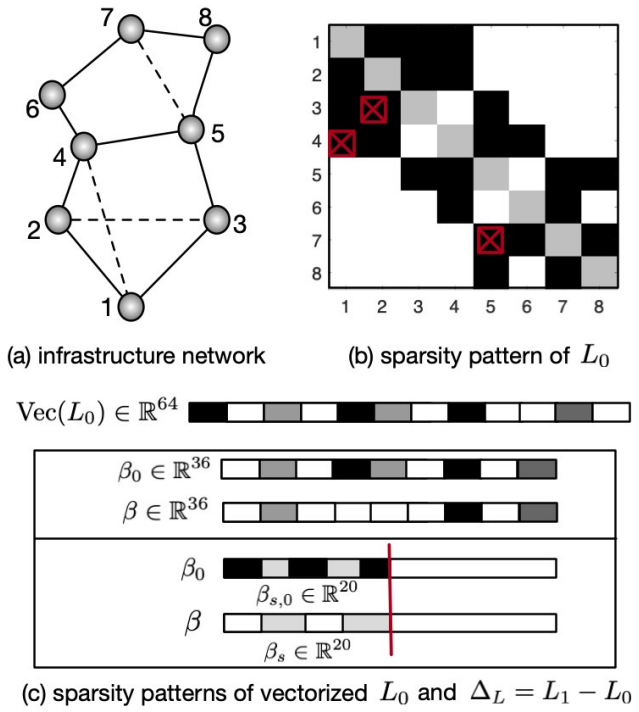


Fig. 2: **Visualizing synthetic network** (8 **nodes and** 12 **edges)**: (a) The edges shown in dashed lines are removed after the change. (b) White spaces correspond to nodes with no edges; making red check boxes white (including a copy on the upper triangular portion) gives the sparsity pattern for L_1. Sparsity patterns in (c) are representative only and should not be confused with the no. of boxes in the vectors and the dimensions of β's or $\mathrm{Vec}(L_0)$ denoting these vectors; see second paragraph in V-A for additional details.

We report $\mathrm{acc} = \mathrm{TP} + \mathrm{TN}\,/(\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN})$ measure with $0 \leq \mathrm{acc} \leq 1$ to summarize the indexes. The higher the acc measure, the better the performance. We averaged our results over twenty independent runs (standard deviation is not reported since it is not informative). Our results suggest that for a suitable $\lambda > 0$, sparsity patterns of $\hat{\beta}$ and β match.

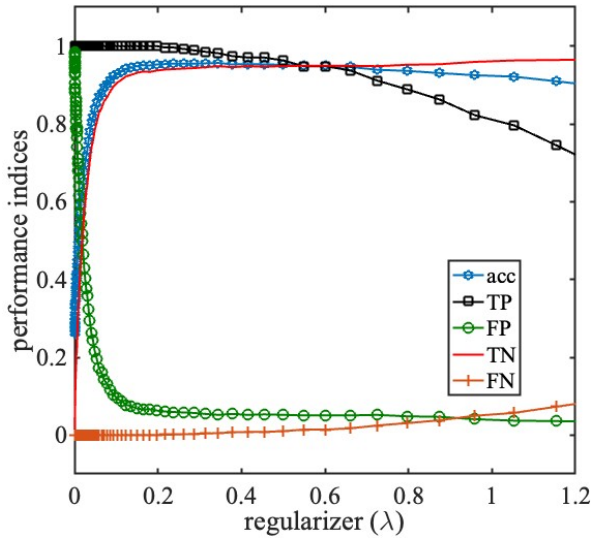


Fig. 3: LASSO performance on the synthetic network in Fig. 2. The optimal λ is for which acc, TP, and TN are closer to one and FP and FN are closer to zero. Thus, the optimal range is $0.15 \leq \lambda \leq 0.6$.

TABLE II: Number of non-zeros in $\mathrm{Vec}(\Delta L)$ and β

vector	dimension	# of non-zeros	model
$\mathrm{Vec}(\Delta L)$	64	6	Eq (8)
β	36	6	Eq (9)
β_s	20	6	Eq (10)

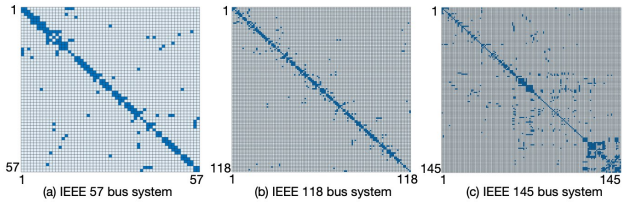


Fig. 4: **Sparsity patterns of power networks' Laplacian matrices** (L_0) **before change**: The dimensions of matrices are (57×57), (118×118), and (145×145), from left to right. The dark-colored boxes denote non-zeros, while the light-shaded boxes denote zeros.

A. *Synthetic network*

Consider the network shown in Fig. 2 (a), consisting of eight nodes and twelve edges—three removed after a change ($\{(2,3),(4,1),(7,5)\}$). Sparsity patterns of L_0 (pre-change) and L_1 (after-change) are in Fig. 2 (b). And Fig. 2 (c) shows $\mathrm{Vec}(L_0)$, its reduced vector β_0, and the difference vector β. The bottom box shows β_0 and β after a suitable permutation of nodes; and the white space to the right of the vertical line are zeros. By construction, $\beta_{s,0}$ has twenty non-zero terms (twelve edges plus eight diagonals in L_0), and β_s have six non-zeros (three terms below and on the diagonal of ΔL); see Section III-A for details.

From Table II, the computationally efficient model is (10) because the vector to be estimated has a lesser dimension. But we work with the model in (9) for simplicity and solve (P.2) in (14) using MATLAB's built-in command LASSO for $\hat{\beta}$. Fig.3 shows performance indices as a function of λ.

B. *Power system networks*

The sparsity patterns of Laplacian matrices (L_0) of power system networks before edge change changes are shown in 4. We use MATPOWER (open-free software) to obtain L_0. For each network, we randomly remove ten edges and compute L_1 such that (2) holds. Thus, true β has at most 20 zeros (ten from off-diagonals and another ten from diagonals of ΔL). Fig.5 shows the performance of the LASSO estimator in (14) subject to the model in (9). For all networks, the optimal range of λ values (based on the acc measure) that result in better performance indices is between 0 and 0.15.

VI. Conclusions

This work presented two ℓ_1-norm regularized least-squares estimators to identify edge changes in critical infrastructure networks obeying equilibrium equations. The first one was a sparse total least squares (TLS) estimator, applicable when node potentials and injected flows were corrupted by noises. Theorem 4.1 provided a simple reformulation for the sparse TLS estimator. The second one was the well-known LASSO estimator, applicable if the noise of the injected flow was

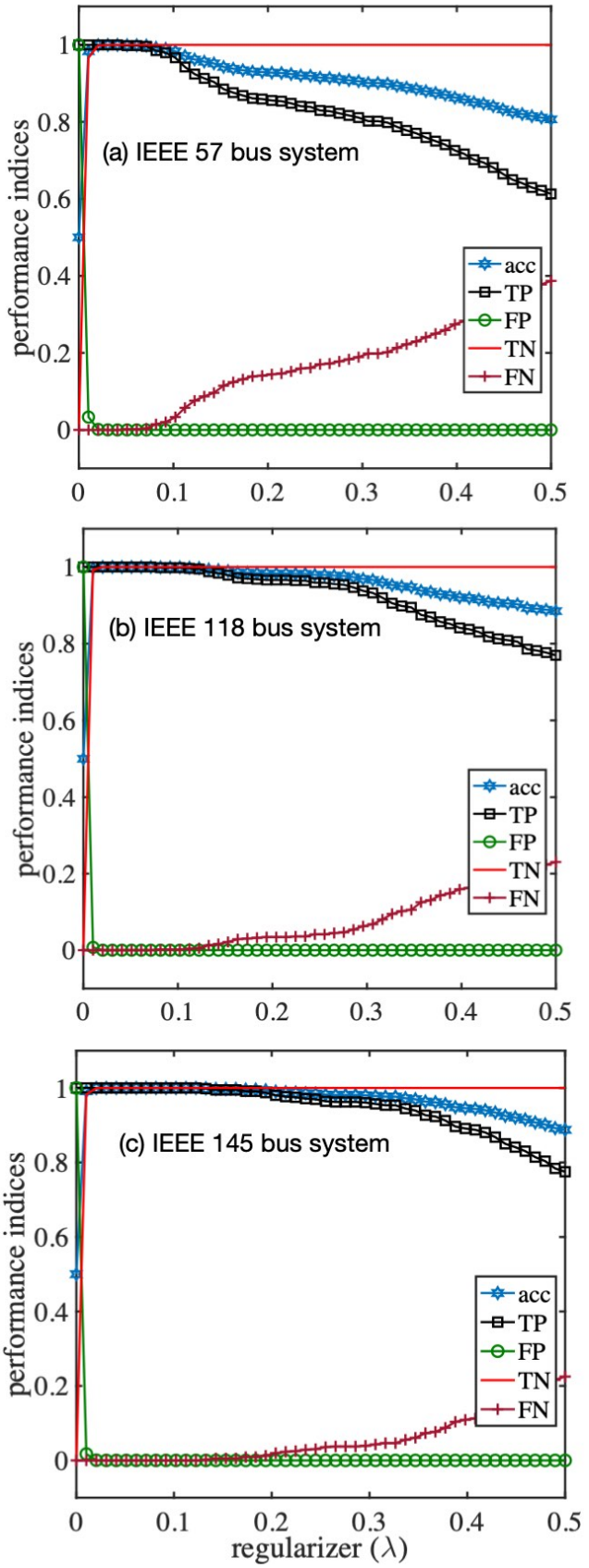


Fig. 5: LASSO performance on power system networks in Fig. 4. The optimal range of λ are the ones for which acc, TP, and TN are closer to one, and FP and FN are closer to zero. Thus, the optimal range for all three systems is $0.1 \leq \lambda \leq 1$.

not high (in magnitude). The LASSO estimator was convex and computationally efficient; instead, the TLS estimator was non-convex and needs heuristic methods, which will address in our future research.

REFERENCES

[1] Cybersecurity and Infrastructure Security Agency (CISA). Critical infrastructure sectors, 2024. Accessed: 2024-09-10.

[2] Gilbert Strang. A framework for equilibrium equations. *SIAM Review*, 30(2):283–297, 1988.

[3] Anirudh Rayas, Rajasekhar Anguluri, and Gautam Dasarathy. Learning the structure of networked systems obeying conservation laws. *Advances in Neural Information Process. Sys.*, 35:14637–14650, 2022.

[4] Francesco Seccamonte, Ambuj K Singh, and Francesco Bullo. Inference of infrastructure network flows via physics-inspired implicit neural networks. In *2023 IEEE Conference on Control Technology and Applications (CCTA)*, pages 1040–1045. IEEE, 2023.

[5] Deepjyoti Deka, Vassilis Kekatos, and Guido Cavraro. Learning distribution grid topologies. *IEEE Transactions on Smart Grid*, 2023.

[6] Reza Arablouei. Fast reconstruction algorithm for perturbed compressive sensing based on total least-squares and proximal splitting. *Signal Processing*, 130:57–63, 2017.

[7] Santiago Segarra, Antonio G Marques, Gonzalo Mateos, and Alejandro Ribeiro. Network topology inference from spectral templates. *IEEE Transactions on Signal and Information Processing over Networks*, 3(3):467–483, 2017.

[8] Deepjyoti Deka, Saurav Talukdar, Michael Chertkov, and Murti V Salapaka. Graphical models in meshed distribution grids: Topology estimation, change detection & limitations. *IEEE Transactions on Smart Grid*, 11(5):4299–4310, 2020.

[9] Rajasekhar Anguluri, Oliver Kosut, and Lalitha Sankar. Identifying edge changes in networks from input and output covariance data. *IEEE Control Systems Letters*, 6:1244–1249, 2024. in press.

[10] Anirudh Rayas, Rajasekhar Anguluri, Jiajun Cheng, and Gautam Dasarathy. Differential analysis for networks obeying conservation laws. In *2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[11] Arsh Bawa, Nirav Bhatt, et al. Verification and rectification of error in topology of conserved networks. *IFAC-Papers*, 55(30):43–48, 2022.

[12] Saverio Bolognani. Grid topology identification via distributed statistical hypothesis testing. In *Big Data Application in Power Systems*, pages 281–301. Elsevier, 2018.

[13] Yuqing Hao, Qingyun Wang, Zhisheng Duan, and Guanrong Chen. Discernibility of topological variations for networked LTI systems. *IEEE Transactions on Automatic Control*, 68(1):377–384, 2021.

[14] R. Dhal, J. Abad Torres, and S. Roy. Detecting link failures in complex network processes using remote monitoring. *Physica A: Statistical Mechanics and its Applications*, 437:36 – 54, 2015.

[15] Gianfranco Parlangeli and Maria Elena Valcher. On the detection and identification of edge disconnections in a multi-agent consensus network. *arXiv preprint arXiv:2101.06728*, 2021.

[16] Omid Ardakanian, Vincent WS Wong, Roel Dobbe, Steven H Low, Alexandra von Meier, Claire J Tomlin, and Ye Yuan. On identification of distribution grids. *IEEE Transactions on Control of Network Systems*, 6(3):950–960, 2019.

[17] Jean-Sébastien Brouillon, Emanuele Fabbiani, Pulkit Nahata, Keith Moffat, Florian Dörfler, and Giancarlo Ferrari-Trecate. Bayesian error-in-variables models for the identification of power networks. *arXiv preprint arXiv:2107.04480*, 2021.

[18] Jan R Magnus and Heinz Neudecker. The elimination matrix: some lemmas and applications. *SIAM Journal on Algebraic Discrete Methods*, 1(4):422–449, 1980.

[19] Ivan Markovsky and Sabine Van Huffel. Overview of total least-squares methods. *Signal processing*, 87(10):2283–2302, 2007.

[20] Hao Zhu, Geert Leus, and Georgios B Giannakis. Sparsity-cognizant total least-squares for perturbed compressive sampling. *IEEE Transactions on Signal Processing*, 59(5):2002–2016, 2011.

[21] Yujie Zhao and Xiaoming Huo. A survey of numerical algorithms that can solve the Lasso problems. *Wiley Interdisciplinary Reviews: Computational Statistics*, 15(4):e1602, 2023.