GIST: Graph Inference for Structured Time Series

Boya Ma* Maxwell McNeil* Petko Bogdanov*

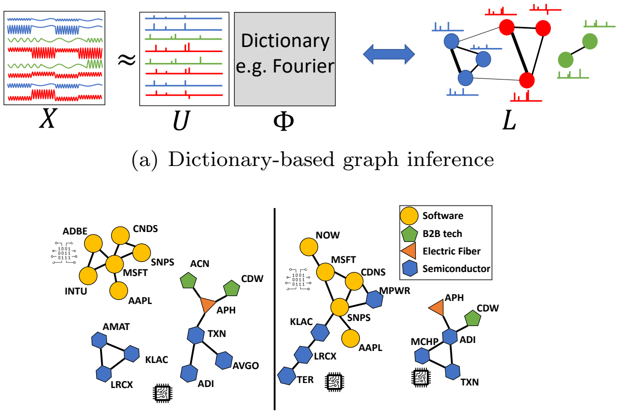
Abstract

Machine learning and data analytics tasks on graphs enjoy a lot of attention from both researchers and practitioners due to the utility that a graph structure among data entities adds for downstream tasks. In many cases, however, a graph structure is not known a priori, and instead has to be inferred from data. Specifically, learning a graph associating time series may elucidate hidden dependencies and also enable improved performance in tasks like classification, forecasting and clustering. While approaches based on pairwise correlation and precision matrix estimation have been employed widely, recent approaches that model observations as signals on graphs have been shown to be more advantageous.

We propose to learn a graph among time series based on similarity of encoding via temporal dictionaries. The key premise is that observed time series have an inherent underlying structure such as periodicity and/or trends and can be succinctly encoded via an appropriate dictionary. Time series with similar encodings are associated via edges in the inferred graph. We formulate the problem as a joint graph Laplacian learning and sparse dictionary-based coding. We consider two alternative solutions for different problem settings: one that associates time series that behave similarly and one that associates them based on shared periodicity. We demonstrate that our solutions enable improved performance over baselines in identifying ground truth edges and ground truth groupings of the time series in 8 real-world datasets from diverse domains.

1 Introduction

Graphs are employed to model relationships among entities and enable improvements compared to non-graph counterparts for tasks like classification and link prediction [34], forecasting [50], filtering [8], and sparse data representation [38]. In typical applications, nodes are endowed with attributes, e.g., interests of social network users or temporal sensor network measurements, and the graph structure among the nodes is known a priori and employed jointly with attributes to improve downstream tasks. In many cases, however, the graph structure is unknown and has to be learned from data. Some prior work on graph inference from node properties focuses on independent attributes while in others nodes are associated with time series [16]. We consider the latter setting, namely our goal is to perform *graph inference for time series data.*



(a) Dictionary-based graph inference

(b) Top inferred edges among S&P 500 tech stocks

Figure 1: (a): Overview of dictionary-based graph inference for structured time series (GIST). We model input time series X as sparsely encodable via an appropriate dictionary Φ (e.g., Fourier, Ramanujan, Spline or other) through an encoding matrix U. The goal is to learn a Laplacian matrix L such that similarly encoded time series are associated with high-weight edges. (b)-left: The top 15 edges among S&P 500 Tech stocks' adjusted daily closing prices time series inferred by our method GIST and (b)-right: the top 15 edges based on correlation.

Graph inference for multivariate time series was previously approached by employing network statistics [6] as well as by learning a sparse estimation of the precision matrix in the Graphical lasso approach [21]. Alternatively, signal processing solutions assume that observed signals should be smooth over the inferred graph [14, 28–30]. Other proposals rely on inferring pairwise causal associations between time series [35, 51]. Graph neural network forecasting approaches also estimate a network among univariate time series while training a non-linear forecasting model [27, 50]. In this work we focus on general (task-agnostic) and unsupervised graph inference for time series.

The key premise in our approach is that related

*Department of Computer Science, University at Albany—SUNY, Emails: {bma,mmcneil2,pbogdanov}@albany.edu

times series share temporal structures such as seasonality or trends. Consider the example in Fig. 1(a) in which groups of time series in X (marked with the same color) have similar temporal behavior. Note that they might not necessarily exhibit similar values in the temporal domain, but they feature similar frequencies. Using an appropriate periodic dictionary Φ, one can view the time series' dominant frequencies in U as sparse encodings for the data X. Beyond periodicity one can consider shared trends via a spline dictionary or other temporal patterns via alternative dictionaries. *Different from other signal processing graph inference methods, we propose to associate time series based on encoding similarity rather temporal domain similarity.*

Example application: S&P 500 stocks. Understanding how stocks of publicly traded companies are connected can provide meaningful insights to investors, where inferred graph edges may imply competition or supply chain links, and inform opportunities for portfolio diversification. The 15 strongest closing price correlation links among S&P 500 stocks are shown in Fig. 1(b)-right where the company sub-sectors are indicated by different shapes. It is clear from the large mixing of connections between software and semiconductor companies that correlation alone does not lead to interpretable networks. In contrast, the edges learned by our proposed approach (Fig. 1(b)-left) capture similarity in global temporal patterns and our inferred graph more accurately draws expected edges within sub-sectors.

We propose GIST, a general framework for joint sparse dictionary coding and graph learning for multivariate time series with temporal structures such as trends and periodicity. We combine the two goals in a single objective which promotes smoothness of time series' codes on the learned graph. We also propose a shared-period alternative that associates lagged- or anti-correlated time series with the same period. Our framework can employ arbitrary temporal dictionaries for sparse coding: both existing analytical dictionaries and data-driven counterparts. We demonstrate GIST's utility on both synthetic and real-world datasets.

Our contributions in this paper are as follows:

- **Novelty:** We propose a dictionary-based graph learning framework for multivariate time series based on the intuitive premise that connected time series share an encoding via an appropriate temporal dictionary.
- **Generality:** Our method is applicable to time series with various temporal structures: seasonality, trends and other temporal patterns.
- **Applicability:** We demonstrate that GIST accurately infers graph edges and communities in multiple application domains and that inferred graphs offer interpretable insights for stocks and COVID data.

2 Related work

Graph inference for time series methods can be categorized in several groups. Statistical models cast the graph structure inference as fitting a joint distribution with a key representative the graphical lasso [20] which estimates a sparse precision matrix (inverse covariance). Since its introduction there have been numerous improvements of both the basic model and associated solvers [10]. A second group includes physically-motivated models that infer a graph by assuming an underlying network process such as network diffusion or information cascades [24]. Graph signal processing models treat inputs as smooth signals over a graph and infer the latter assuming i) global signal smoothness [15, 28, 43]. Our method falls in the signal processing group but different from counterparts, we assume that associated time series are not necessarily similar in the original time domain, but in a dictionary encoding domain, e.g., similar periodicity, trends and combinations of the two.

A different group of methods, neural relational inference models [4, 25, 32], infer a graph structure from a dynamical interacting system while also learning the underlying dynamical model. Some methods optimize a graph structures for specific tasks like classification [19, 48, 53] and forecasting [42]. Others focus on prediction and node representation learning [11, 26] by employing deep latent generative models. Most models in this group require multiple instances of multivariate time series for training, validation and testing. Our setting is different as we seek to learn a graph from a single observation of a multivariate time series.

Graph signal processing (GSP) operates on signals over (known) graphs and comprise a popular research area in signal processing [44]. A central premise is that a graph signal can be represented as a linear combination of graph dictionary bases. The eigenvectors of the graph Laplacian are often adopted as basis in this domain [15]. GSP approaches solve classical signal processing tasks for graph signals, however, they assume that the graph structure is known. Our work seeks to learn the underlying graph structure and is thus complementary to the approaches in this area.

Sparse dictionary modeling represents data via a sparse combination of dictionary bases. It is widely employed in signal processing [41, 52], image analysis [18] and computer vision [49]. In the context of time series, many widely adopted dictionaries have been designed to capture underlying temporal structures, e.g., DFT [40] and the Ramanujan periodic dictionary [45] and the spline dictionaries [22]. Other methods learn temporal dictionaries from data [46]. Our work assumes a given temporal dictionary and jointly learns a graph and a sparse encoding for observed multivariate time series. Thus, it can be viewed as a generalization of sparse modeling and as complementary to dictionary learning approaches since learned dictionaries can be employed within our framework.

3 Preliminaries

Before we define our problem of encoding-based graph inference, we introduce necessary preliminaries and notation. The input to our problem is a real-valued multivariate time series $\mathcal{X} \in \mathbb{R}^{n \times t}$ comprised of n univariate time series of length t. Sparse dictionary coding represents observations X as a linear combination of a few atoms from an appropriate dictionary Φ, where both analytical and data-driven dictionaries can be employed [41]. In its general form sparse coding solves the following problem:

$$\min_{u} f(u) \quad \text{s.t.} \quad x = u\Phi, \tag{3.1}$$

where x is an input signal, u is the encoding of the signal and $f(u)$ is a sparsity promoting function often instantiated as an L1 norm. Tenneti et Al. [45] propose multiple analytical dictionaries for periodic time series based on the framework of nested periodic matrices. Alternative dictionaries to model smooth trends have also been proposed employing on splines [22].

A graph with n nodes is represented by its (combinatorial) Laplacian matrix $L \in \mathcal{R}^{n \times n}$, which in turn is defined as the difference between the degree D and adjacency matrices $L = D - W$. The degree matrix D is a diagonal matrix with elements corresponding to the volume of each node, i.e., the sum of weights on all adjacent edges, while the adjacency matrix W is a symmetric matrix specifying the non-negative weights on edges among node pairs.

4 Problem formulation

The key premise of our graph inference approach is that *time series should be connected by an edge if they have similar sparse encoding via a dictionary* Φ. More specifically, if two time series are encoded as $x_i = u_i\Phi$ and $x_j = u_j\Phi$, we seek to connect them by a high-weight edge w_{ij} if the similarity $s(u_i, u_j)$ of their encoding is high. Depending on the choice of dictionary, similar encoding may correspond to shared trends (e.g., Spline [22]), ii) shared periodicity (e.g., Ramanujan and DFT [45]), iii) similar intervals of on/off states (e.g., Haar wavelets [12]), or other temporal structures.

An illustrative example employing a Ramanujan periodic dictionary Φ is presented in Fig. 2(a). The input X consists of 5 periodic time series x_1 to x_5. Their corresponding sparse encodings (rows of U) are similar for pairs of time series with similar periodicity. For example, x_1 and x_2 which are scaled versions of each other have corresponding scaled encoding in U for the period-2 atom P_2 of Φ. Alternative dictionaries allow us to associate time series based on other structural patterns through their encodings.

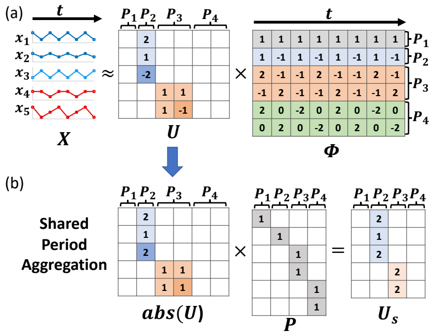


Figure 2: (a): Encoding matrix U for a dataset X of 5 periodic time series employing the Ramanujan periodic dictionary Φ [45]. Different blocks of atoms P_i and correspond to discrete periods. (b): In order to infer edges between time series sharing the same period, though not necessarily correlated, we take the element-wise absolute values of the encodings $abs(U)$ and aggregate them according to the period-specific blocks in Φ to obtain the shared period representation U_s.

The GIST objective. We enforce encoding similarity via a Laplacian quadratic form similar to how smoothness of graph signals is enforced in graph signal processing approaches. A key difference in our case is that our goal is to also simultaneously learn the graph Laplacian L. The GIST objective is as follows:

$$\begin{aligned} &\underset{U,L}{\operatorname{argmin}}\ ||X - U\Phi||_F^2 + \lambda_1 tr(U^T LU) + \lambda_2 ||L||_F^2 + \lambda_3 ||U||_1 \\ &s.t. \quad tr(L) = n \\ &\qquad\ \ L_{ij} = L_{ji} \leq 0, i \neq j \\ &\qquad\ \ L\mathbf{1} = \mathbf{0}, \end{aligned} \tag{4.2}$$

where the first term in the minimization quantifies the quality of fit by a temporal dictionary encoding $U\Phi$ as Frobenius residual loss, the trace term promotes smoothness of the encodings U with respect to the learned graph Laplacian L, and the third term adds a Frobenius norm penalty on the learned L to reduce concentration of weight on individual edges. The last term of the minimization imposes an L_1 norm on U to promote sparse encoding in line with sparse dictionary modeling (Eq. 3.1). The three constrains guarantee a valid Laplacian matrix: the first avoids a trivial solution of all-zeros, the second ensures symmetry while the the last constraint ensures that the Laplacian rows sum to 0. The Laplacian constraints are similar to those employed by earlier graph inference methods [15, 28], with a key difference that we associate time series in the encoding (frequency) domain as opposed to the temporal domain leading to learned graphs of better quality as we demonstrate in Sec. 6.

Shared period objective: GIST-SP. The main GIST objective from Eq. 4.2 penalizes differences in the encodings of connected time series weighted by the learned edge weight among them via the trace term $tr(U^T LU)$. For some applications a more flexible similarity measure might be beneficial to capture similar behavior. Specifically, in the case of periodic time series, we would like to be able to associate pairs that are off-phase or even anti-correlated as such relationships may correspond to causal links, e.g., a change in one time series leads to a delayed change in another. This scenario is present in Fig. 2(a) for pairs x_1 and x_3 which are off-phase by one time step and also the pair x_4 and x_5 which are not correlated, but both have a period of 3. Under the main objective from Eq. 4.2 an edge between x_1 and x_3 and an alternative edge between x_1 and x_4 will incur similar penalties.

To enable associating time series of shared period, we propose a shared period alternative objective by modifying only the encoding smoothness term $tr(U^T LU)$ from Eq. 4.2. The key idea is illustrated in Fig. 2(b) in the context of the Ramanujan dictionary and our example time series. We take the absolute values of encodings $abs(U)$ and aggregate coefficients corresponding to atoms within the same period via an appropriate aggregation matrix P to obtain the shared period encodings U_s. Note that with this transformation the three 2-periodic time series x_1, x_2 and x_3 and the two 3-periodic time series x_4 and x_5 are appropriately grouped in shared-period groups within U_s. Our shared period encoding objective, which we term GIST-SP, imposes smoothness on the learned graph by a similar trace term $tr(U_s^T LU_s)$ involving the shared period encodings U_s as opposed to the raw encodings. Note, that alternative periodic dictionaries, such as the DFT [45], can also be employed with corresponding aggregation matrices P to aggregate loadings of period-specific atoms.

5 Optimization overview: the GIST algorithm

In this section we summarize the optimization solutions for the two objectives GIST and GIST-SP. Since the problems are not convex with respect to both variables L and U, we employ an alternating optimization approach that updates one while keeping the other one fixed. We list the overall optimization algorithm in Alg. 1, while the detailed update derivations are provided in the extended version of the paper [36]. The alternating optimization (Step 6-24) iterates between updating the Laplacian L (Steps 7-10) via the primal-dual method and updating U (Steps 11-19) following an ADMM scheme [9]. To solve for L, we re-formulate the sub-problem with respect to the adjacency matrix W and a pair-wise distance matrix Z between the encodings (Steps 8-9). This reformulation is advantageous as it allows for primal-dual optimization [33]. The ADMM scheme to update U introduces a proxy variable V and Lagrangian multipliers Γ, ρ. The closed-form update for U in the GIST objective is based on the Sylvester equation (Step 13) which we further optimize by pre-computing fixed terms depending on a factorization of the dictionary Φ (Steps 4-5). A closed-form solution for U for the shared period objective GIST-SP is computationally demanding due to an $abs(U)$ term and produces similar results to a simpler gradient descent update in practice, hence we employ the latter (Step 15). Proxy variable V and Lagrangian multipliers Γ, ρ have efficient closed-form updates (Steps 17-19). We track convergence of L and U in (Steps 20-23). Detailed derivation of the updates is available in [36].

Algorithm 1 GIST (and GIST-SP)

```
 1: Input: Time series $X$, dictionary $\Phi$, params: $\lambda_1, \lambda_2, \lambda_3, \eta, iter, \epsilon$
 2: Output: Laplacian $L$, Encodings $U$
 3: Initialize $U = V$, $\Gamma$, $\rho$ randomly
 4: $[P, S, Q] = svd(\Phi)$
 5: $\Lambda = 2SS^T + \rho I$
 6: for $i = 1 \dots iter$ do
 7:     // Update $L$
 8:     Compute $Z$: $Z_{i,j} = ||u_i - u_j||^2, \forall i, j$
 9:     $W = \text{primal-dual}(\frac{\lambda_1}{2} Z, 2\lambda_2)$
10:     $L = D - W$
11:     // Update $U$
12:     if GIST then
13:         $U = sylvester(2\lambda_1 L, P, \Lambda, 2X\Phi^T + \rho V + \Gamma)$
14:     else if GIST-SP then
15:         Perform gradient descent (GD) for $U$
16:     end if
17:     $H = U - \frac{\Gamma}{\rho}$
18:     $V = sign(H) \odot \max(abs(H) - \frac{\lambda_3}{\rho}, 0)$
19:     $\Gamma = \Gamma + \rho(V - U)$
20:     // Check convergence
21:     if $\frac{||U^{(i)} - U^{(i-1)}||_F}{||U^{(i-1)}||_F} \le \epsilon$ & $\frac{||L^{(i)} - L^{(i-1)}||_F}{||L^{(i-1)}||_F} \le \epsilon$ then
22:         break
23:     end if
24: end for
25: return L, U
```

The two steps that dominate the running time of our algorithm are (i) the SVD of the dictionary Φ in Step 4 with complexity $O(min(k^2 T, kT^2))$; and (ii) the Sylvester solution in Step 13 with a worst-case complexity of $O(n^3)$ but faster practical running times due to optimizations [23].

6 Experimental evaluation

We evaluate the ability of our methods to predict ground truth edges among time series and the utility of the

Dataset	Nodes	Edges	Groups	t	Res.	Graph
Synthetic	60-2k	100-4k	SBM	300-10k	/	Random
Bike [2]	142	1723	/	328	1d	Trips
Road [7]	100	128	/	300	5m	Roads
RM [17]	94	795	/	8636	1h	Messages
Crime [13]	77	200	/	795	1h	Spatial
Wiki [39]	128	119	/	792	1h	Co-clicks
TEMP [1]	70	212	Clim. zones	365	1d	Spatial
Covid [31]	74	464	States	678	1d	Spatial
Stock [3]	496	/	Sectors	250	1d	/

Table 1: Summary of datasets used for evaluation.

learned graph structure to discern ground truth groupings of time series on multiple datasets and in comparison to state-of-the-art baselines. An implementation of GIST is available at: `http://www.cs.albany.edu/~petko/lab/code.html`.

6.1 Datasets. We employ synthetic as well as 8 real-world datasets for evaluation. Statistics of all datasets are listed in Tbl. 1. We synthesize ground truth random stochastic block model (SBM) graphs with a fixed number of blocks and varying number of nodes. The real world datasets feature either ground truth edges (Bike, Road, Reality Mining (RM), Crime, Wiki), ground truth groupings (Stock) or both (Covid, TEMP). We normalize all input time series using z-score normalization to reduce effect of varying scales. Details of evaluation data are available in the extended version [36].

6.2 Experimental setup. We employ two dictionaries for experiments: the Ramanujan (R) [45] periodic dictionary and the Spline (S) [22] dictionary for smooth trends. Corresponding GIST variants are denoted as GIST(R) and GIST(S). Selecting an appropriate dictionary for a given dataset can be performed by comparing the quality of time series reconstruction (i.e. GIST without graph learning). Dictionary definitions as well as details on how to select an appropriate dictionary are provided in the extended version [36].

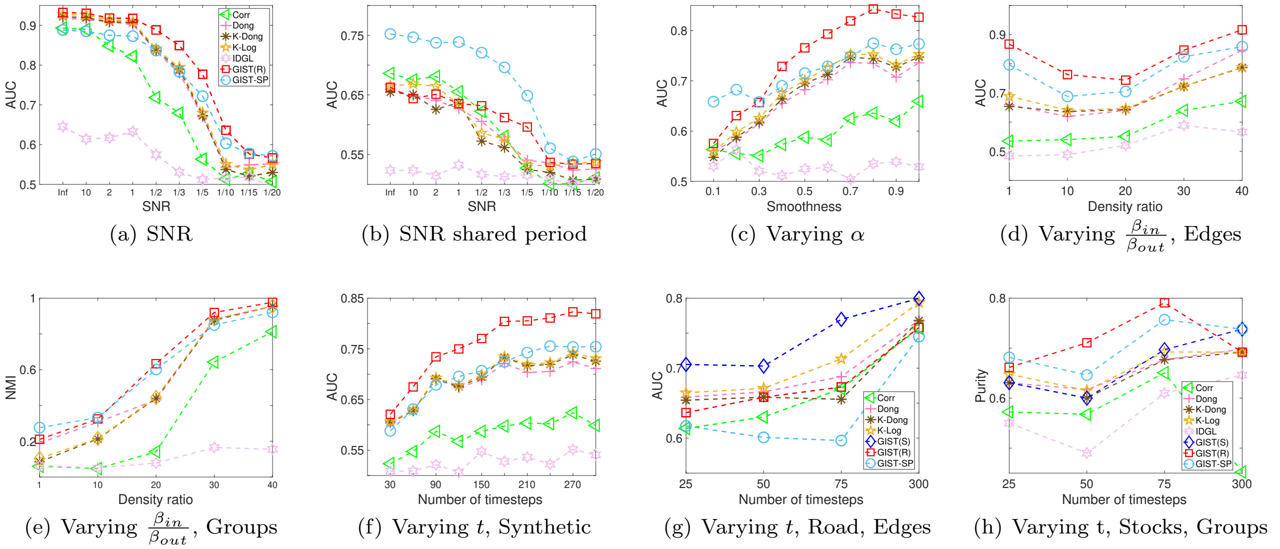
Baselines. We compare our methods to a naive baseline associating times series baseed on their correlation (*Corr*) and four baselines from the literature: *Dong* [15], Kalofolias's Dong (*K-Dong*) and Kalofolias's Log objective (*K-Log*) [28] and Iterative Deep Graph Learning (*IDGL*) [11]. *Dong*, *K-Dong* and *K-Log* are signal processing methods that learn a graph on which the time series X are smooth in the temporal domain. While *IDGL* is not explicitly a time series method (it works with general node feature vectors), it jointly learns a graph structure and node embeddings optimized for node classification within a graph neural network architecture. Thus, we employ it as a baseline for graph structure learning in datasets in which we have ground truth cluster annotations (TEMP, Covid and Stock) which we provide to the method as node labels (note that this information is not provided to any of the other competing techniques). We create 10 random training-testing splits of labeled time series (90%-10%) and report average metric quality for IDGL in all experiments.

Metrics. *Quality of predicted edges* with respect to ground truth edges is measured via the AUC measure. We also measure how well learned graphs reflect known groups (or clusters) of time series. To this end, we quantify *the quality of recovering ground truth groups* by perform spectral clustering using the learned combinatorial Laplacian by baselines [47]. We employ k-means (k is set to the ground truth number of groups in each dataset) as the last step of spectral clustering and run it 10 times to "smooth out" variation due to random initialization. We report average Normalized mutual information (*NMI*) [5] and cluster Purity (*Pur*) [37] from multiple k-means runs and across clusters.

6.3 Effect of data properties. We first compare the performance of our methods and that of baselines as a function of varying data characteristics, including noise level, graph community structure, smoothness of encodings on the graph and length of the time series.

Varying noise. We synthesize an SB graph with 60 nodes and 3 clusters, vary the SNR from infinity (noiseless time series) to 1/20, set the encoding smoothing to $\alpha = 0.9$ (details in [36]), and report the AUC for ground truth edge prediction in Fig. 3(a). GIST with Ramanujan dictionary performs best among baselines at all noise levels. Notably, it maintains better than random quality in very noisy settings. For example, when SNR = 1/5, we can still achieve a score of 0.78, while baselines' AUC values deteriorate to 0.68 and lower. Note that due to the high level of encoding smoothing $\alpha = 0.9$, GIST-SP does not have advantage over the main objective. To evaluate a setting that is advantageous for GIST-SP, we "turn off" smoothing, ensuring that times series within each SBM cluster share periods, but could be anti- or lag-correlated. The results for this experiment are presented in Fig. 3(b). The overall quality of all methods decreases as this a more challenging setting. GIST-SP has a pronounced advantage over alternatives with up to 0.1 AUC improvement in mid-noise regimes. Signal processing alternatives tend to be the next best baselines and IDGL's performance ranks last as it does not explicitly handle time series and optimizes the graph for node classification rather than graph signal smoothness.

Effect of varying α. We next vary the encoding smoothness in the Synthetic dataset via synthesis pa-



(a) SNR

(b) SNR shared period

(c) Varying α

(d) Varying $\frac{\beta_{in}}{\beta_{out}}$, Edges

(e) Varying $\frac{\beta_{in}}{\beta_{out}}$, Groups

(f) Varying t, Synthetic

(g) Varying t, Road, Edges

(h) Varying t, Stocks, Groups

Figure 3: Effect of SNR on edge inference for smoothed encodings ($\alpha = 0.9$) (a), and shared period (b) in synthetic data. Effect of encoding smoothness (c) and cluster density ratio on edge (d) and group (d) recovery (SNR=1/5). Effect of time series length t on edge recovery in Synthetic (SNR=1/5) (f) and Road (g); and on group recovery in Stocks (h).

rameter α (see [36] for details). Large values of α increase the "alignment" of neighbor encodings. GIST's quality dominates alternatives and its advantage widens with α (Fig. 3(c)). A similar trend is evident for alternative methods since within-block time series become more similar in both the encoding and time domains. It is important to note that for low values of smoothness ($\alpha = [0.1, 0.2]$), GIST-SP emerges as the dominant method as its ability to associate time series based on shared periods, but not necessarily similar coefficients is central to its objective.

Varying Cluster Structure. We also study the effect of the underlying ground truth graph structure on the ability of competing methods to recover known clusters. To this end, we vary the density of within and across cluster edges in the SB model. In addition to reporting the AUC for edge inference (Fig. 3(d)), we also quantify the ability to recover ground truth blocks of the SB model in terms of NMI (Fig. 3(e)). In terms of AUC, GIST(R) and GIST-SP are always the best two methods. At low ratios the graph is relatively simple (few edges) to learn. Alternatively, at high density ratios there are more edges to learn, but the clusters are much more "discernable" which helps all baselines get a better AUC. The task is most challenging for mid density ratios of $10 - 20$. In the cluster recovery task (Fig. 3(e)), there is a clear monotonic trend of increasing performance with stronger clusters. GIST(R) is again the dominant method in terms of NMI with a small advantage over GIST-SP.

Varying Time Series Length. Next we evaluate the data complexity of competing techniques. Specifically, we are interested in how the observed time series length affect competitors' inference quality. For synthetic data we vary the number of time steps t in X between 30 and 300, set the SNR to 1/5 and report the AUC for competing techniques in Fig. 3(f). GIST(R)'s advantage is small compared to baselines when a small prefix of the time series is employed ($t = 30$). As the length of observations increases, our method can learn a better encoding matrix U and consequently a more accurate Laplacian. While the AUC of baselines also improves with t, it remains significantly lower than that of GIST(R). We also vary the number of observed timesteps t for two real-world dataset. In Road, we test the AUC for edge inference and report results in Fig. 3(g). While all methods improve with t, GIST(S) retains dominant performance across all values of t. We also evaluate the effect of t on the quality of cluster recovery (Fig. 3(h)). GIST(R) and GIST-SP dominate alternative on this task across all values of t.

6.4 Evaluation on real-word datasets

We next evaluate all techniques on the real-world datasets (including two versions of Stock) and report results for edge prediction, and cluster detection in Tbl. 2.

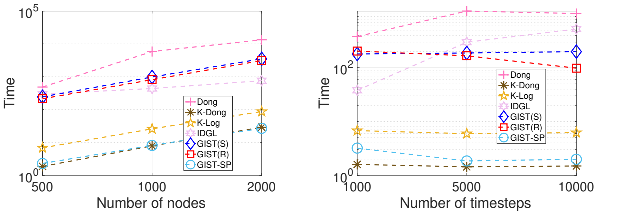
Edge prediction. In all datasets, a variation of our method performs best in terms of edge prediction AUC with the exception of the TEMP dataset where Dong has a small advantage. Despite this disadvantage in

	RM		Wiki		Bike		Crime		Road		TEMP				Covid				Stock-small			Stock S&P 500		
Method	AUC	t(s)	AUC	t(s)	AUC	t(s)	AUC	t(s)	AUC	t(s)	AUC	NMI	Pur	t(s)	AUC	NMI	Pur	t(s)	NMI	Pur	t(s)	NMI	Pur	t(s)
Corr	64.4	.02	50.9	.02	61.0	.02	52.5	.02	75.7	.02	96.3	39.0	65.4	.02	63.2	9.6	47.7	.02	14.5	45.2	.02	13.8	20.7	.05
Dong	71.4	3.7	55.1	7.3	62.9	8.4	52.6	2.9	76.8	3.9	**99.2**	62.2	78.6	4.9	72.3	58.2	73.0	4.5	44.0	69.8	4.5	21.8	33.1	718
K-Dong	71.7	.07	55.4	.07	56.4	.07	49.8	.02	76.8	.04	93.8	68.7	78.6	.03	67.4	52.7	69.1	.02	43.3	69.2	.03	21.6	32.7	2.2
K-Log	73.4	.29	52.5	.25	62.3	.29	49.8	.12	79.4	.17	89.9	62.2	78.6	.15	71.5	52.6	69.0	.12	44.5	69.2	.12	22.9	34.0	8.7
IDGL	/	/	/	/	/	/	/	/	/	/	92.0	72.4	91.4	24	60.7	9.4	47.6	23	44.8	64.6	24	9.7	20.5	45
GIST(S)	73.4	3.8	**57.4**	5.1	**64.0**	6.1	53.8	2.4	**80.0**	1.1	96.1	67.2	80.0	3.0	50.1	20.0	62.2	3.0	46.8	**73.8**	2.7	21.8	33.4	205
GIST(R)	71.6	3.2	52.4	5.4	60.7	10.1	53.8	1.6	75.8	1.6	90.0	**83.4**	**94.3**	3.4	66.4	47.5	69.4	2.4	44.7	69.2	2.3	**24.0**	**36.4**	211
GIST-SP	**74.3**	.38	54.4	.4	61.2	.32	**54.0**	.24	74.5	.22	91.4	80.3	91.7	.2	**73.3**	**58.8**	**74.0**	.14	**53.3**	**73.8**	.18	18.9	31.4	8.8

Table 2: Comparison of all competitors on real-world datasets for (i) edge prediction in terms of Area under the ROC curve (AUC), (ii) cluster recovery via Normalized Mutual Information (NMI) and Purity (Pur), and (iii) running time t(s) in seconds.

AUC, GIST(R) and GIST-SP preform better in terms of cluster recovery (both NMI and purity) which we discuss in the subsequent section. In the Covid, RM and Crime we observe advantage when utilizing GIST-SP. The ground-truth edges in Covid reflect neighboring counties. Throughout the pandemic waves, Covid cases exhibit rises and falls similarly in neighboring counties, however, counties may be temporally lagged in this patterns. Since, GIST-SP only considers shared periods when inferring edges, it is able to capture neighboring lag better than alternatives resulting in better AUC. Advantage in the Reality Mining dataset can be also due to lag in user message responses, resulting in "off-phase" time series. Overall, the strong performance of our methods demonstrate that edges in these datasets reflects global temporal patterns. For example, the connections in Bike rental and Road would be highly reflective of weekly traffic patterns. Note that IDGL was not evaluated on datasets without GT communities which we employ as node labels for IDGL.

Clustering. In the clustering experiments we evaluate how well the graphs inferred by competing techniques reflect ground truth groups. To this end, we perform spectral clustering on the learned graphs and report the NMI and Purity (Pur) scores of the learned clusters compared to the ground truth. Results are also reported in Tbl. 2. In all datasets with ground truth clusters GIST obtains both the highest NMI and Purity. In TEMP, although not the best at edge inference (AUC), GIST(R)' clustering of US cities aligns best with known climate zones. Edges predicted by Dong, K-Log and K-Dong are less reflective of long-term temperature patterns due to enforcing smoothness in the temporal rather encoding domain. In Stock-small GIST-SP produces groupings that best align to sectors, however, GIST(R) performs best in the full Stock data. Stock-small contains only three sectors (Basic Materials, Communication Services, and Energy) with low likelihood of cross-sector shared trends, hence the better results across all methods as compared to the full Stock data.



(a) Varing nodes (T=5000) (b) Varying timesteps (N=500)

Figure 4: Scalability comparison of GIST and competing baselines with increasing number of nodes (a) and number of time steps (b). Running time is in seconds.

6.5 Running time. Beyond quality, Tbl. 2 also lists running times for all methods and datasets. While a simpler baseline like Corr and the optimized signal processing methods from [28] run between 5 and 10 times faster than our method, they produce graphs of lower quality. Our GIST-SP objective is faster than alternatives as it follows a Gradient Descent (GD) approach. GIST(R) and GIST(S) are generally faster than the original Dong method. To further compare the scalability of all methods we vary the number of nodes and number of time steps in a synthetic dataset and report running times in Fig. 4. The running time of all competitors grows super-linearly with the number of nodes (Fig. 4(a)) which is expected as the number of possible edges grows quadratically. Dong, GIST(S), GIST(R) are relatively slower than GIST-SP, K-Log and K-Dong methods, mainly because they use closed-form solutions and/or optimize relatively more complex objectives. All methods can infer a graph among 2000 time series within one hour. Running time of competitors is less affected by the number of time steps (see Fig. 4(b)).

6.6 Case Studies: Tech Stocks and Covid in Vermont We perform a case study of a subset of the Stock dataset corresponding to the technology sector. We compare the top 15 edges and associated companies learned by GIST(R) with $\lambda_i = .0001, i = 1, 2, 3$ and compare them with the those of highest correlation in Fig. 1(b). Each node represents a company listed on the exchange with their industry indicated by the legend and company ticker placed next to the associated node. In Fig 1(b)-left GIST(R) learns edges which correspond to long term shared periodic trends. This can be seen in its ability to create more within-subsector edges compared to correlation (Fig 1(b)-right). This is further supported by a deeper inspection of the formed edges. For example, GIST(R) identifies an edge between Microsoft (MSFT) and Apple (AAPL) which are well known competitors. In contrast, such an edge is not identified by Corr. It is interesting to see that Amphenol (APH) bridges the gap between semiconductor producers and the business-to-business technology companies they supply. A second case study on the Covid dataset [31] is described in [36].

7 Conclusion

We proposed GIST, a dictionary-based method for inferring a graph among time series. It utilizes temporal dictionaries to encode temporal signals while simultaneously inferring a graph that is smooth with respect to encodings. We demonstrated that GIST was able to learn edges that share period, smoothness, or any other pattern that can be readily captured by a dictionary. We also demonstrated the advantage of our technique on a synthetic and eight real-world datasets. Specifically, GIST was able to reconstruct graphs and their ground truth groups more accurately than an array of state-of-the-art baselines. We further demonstrated its applicability through case studies where GIST learned interpretable network structures.

8 Acknowledgements

This research is funded by an academic grant from the National Geospatial-Intelligence Agency (Award No. # HM0476-20-1-0011, Project Title: Optimizing the Temporal Resolution in Dynamic Graph Mining). Approved for public release, NGA-U-2022-00118. The work is also supported by the NSF Smart and Connected Communities (SC&C) grant CMMI-1831547.

References

[1] *Daily Temperature of Major Cities:*. `https://www.kaggle.com/datasets/sudalairajkumar/daily-temperature-of-major-cities`.

[2] *Hubway data:*. `http://hubwaydatachallenge.org`.

[3] *SP 500 historical data from Yahoo! Finance*, Jun 2022.

[4] F. Alet, E. Weng, T. Lozano-Pérez, and L. P. Kaelbling, *Neural relational inference with fast modular meta-learning*, Advances in Neural Information Processing Systems, 32 (2019).

[5] A. Amelio and C. Pizzuti, *Is normalized mutual information a fair measure for comparing community detection methods?*, in 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), 2015, pp. 1584–1585.

[6] H. C. Baggio, A. Abos, B. Segura, A. Campabadal, A. Garcia-Diaz, C. Uribe, Y. Compta, M. J. Marti, F. Valldeoriola, and C. Junque, *Statistical inference in brain graphs using threshold-free network-based statistics*, tech. rep., Wiley Online, 2018.

[7] P. Bogdanov, M. Mongiovi, and A. K. Singh, *Mining heavy subgraphs in time-evolving networks*, in ICDM, 2011.

[8] A. W. Bohannon, B. M. Sadler, and R. V. Balan, *A filtering framework for time-varying graph signals*, in Vertex-Frequency Analysis of Graph Signals, Springer, 2019, pp. 341–376.

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Found. Trends Mach. Learn., 3 (2011), pp. 1–122.

[10] T. T. Cai, W. Liu, and H. H. Zhou, *Estimating sparse precision matrix: Optimal rates of convergence and adaptive estimation*, The Annals of Statistics, 44 (2016), pp. 455–488.

[11] Y. Chen, L. Wu, and M. Zaki, *Iterative deep graph learning for graph neural networks: Better and robust node embeddings*, Advances in neural information processing systems, 33 (2020), pp. 19314–19326.

[12] C. K. Chui, *An introduction to wavelets*, vol. 1, Academic press, 1992.

[13] C. P. Department, *Crimes - 2001 to present: City of chicago: Data portal*, Feb 2022.

[14] W. Dong and A. Pentland, *A network analysis of road traffic with vehicle tracking data.*, in AAAI Spring Symposium: Human Behavior Modeling, 2009.

[15] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, *Learning laplacian matrix in smooth graph signal representations*, 2016.

[16] X. Dong, D. Thanou, M. Rabbat, and P. Frossard, *Learning graphs from data: A signal representation perspective*, IEEE Signal Processing Magazine, 36 (2019), p. 44–63.

[17] N. Eagle and A. S. Pentland, *Reality mining: sensing complex social systems*, Personal and ubiquitous computing, 10 (2006), pp. 255–268.

[18] M. Elad and M. Aharon, *Image denoising via sparse and redundant representations over learned dictionaries*, IEEE Transactions on Image processing, 15 (2006), pp. 3736–3745.

[19] L. Franceschi, M. Niepert, M. Pontil, and X. He, *Learning discrete structures for graph neural networks*, in International conference on machine learning, PMLR, 2019.

[20] J. Friedman, T. Hastie, and R. Tibshirani, *Sparse inverse covariance estimation with the lasso*, 2007.

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

[22] V. Goepp, O. Bouaziz, and G. Nuel, *Spline regression with automatic knot selection*, arXiv preprint arXiv:1808.01770, (2018).

[23] G. Golub, S. Nash, and C. Van Loan, *A hessenberg-schur method for the problem ax + xb= c*, IEEE Transactions on Automatic Control, 24 (1979), pp. 909–913.

[24] M. Gomez-Rodriguez, J. Leskovec, and A. Krause, *Inferring networks of diffusion and influence*, ACM Trans. Knowl. Discov. Data, 5 (2012).

[25] C. Graber and A. Schwing, *Dynamic neural relational inference for forecasting trajectories*, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020.

[26] A. Grover, A. Zweig, and S. Ermon, *Graphite: Iterative generative modeling of graphs*, in International conference on machine learning, PMLR, 2019.

[27] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, *Attention based spatial-temporal graph convolutional networks for traffic flow forecasting*, in Proc. of the AAAI Conf. on Artificial Intelligence, 2019.

[28] V. Kalofolias, *How to learn a graph from smooth signals*, in Artificial Intelligence and Statistics, PMLR, 2016, pp. 920–929.

[29] V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst, *Matrix completion on graphs*, arXiv preprint arXiv:1408.1717, (2014).

[30] V. Kalofolias and N. Perraudin, *Large scale graph learning from smooth signals*, arXiv preprint arXiv:1710.05654, (2017).

[31] S. Kemp, J. W. Howel, and P. C. Lu, *Bing covid-19 tracker*, Apr 2020.

[32] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel, *Neural relational inference for interacting systems*, in International Conference on Machine Learning, PMLR, 2018, pp. 2688–2697.

[33] N. Komodakis and J.-C. Pesquet, *Playing with duality: An overview of recent primal-dual approaches for solving large-scale optimization problems*, 2014.

[34] Z. Liu, T.-K. Nguyen, and Y. Fang, *Tail-gnn: Tail-node graph neural networks*, in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 1109–1119.

[35] S. Löwe, D. Madras, R. Zemel, and M. Welling, *Amortized causal discovery: Learning to infer causal graphs from time-series data*, arXiv preprint arXiv:2006.10833, (2020).

[36] B. Ma, M. McNeil, and P. Bogdanov, *GIST: Graph inference for structured time series — extended version*, Online: `http://www.cs.albany.edu/~petko/lab/publications.html`, (2022).

[37] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*, Cambridge University Press, USA, 2008.

[38] M. J. McNeil, L. Zhang, and P. Bogdanov, *Temporal graph signal decomposition*, in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 1191–1201.

[39] M. Mongiovi, P. Bogdanov, and A. K. Singh, *Mining evolving network processes*, in ICDM, 2013.

[40] K. R. Rao, D. N. Kim, and J.-J. Hwang, *Fast Fourier Transform - Algorithms and Applications*, Springer Publishing Company, 1st ed., 2010.

[41] R. Rubinstein, A. M. Bruckstein, and M. Elad, *Dictionaries for sparse representation modeling*, Proceedings of the IEEE, 98 (2010), pp. 1045–1057.

[42] C. Shang, J. Chen, and J. Bi, *Discrete graph structure learning for forecasting multiple time series*, arXiv preprint arXiv:2101.06861, (2021).

[43] Y. Shen, B. Baingana, and G. B. Giannakis, *Nonlinear structural vector autoregressive models for inferring effective brain network connectivity*, arXiv preprint arXiv:1610.06551, (2016).

[44] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, *The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains*, IEEE signal processing magazine, (2013).

[45] S. V. Tenneti and P. P. Vaidyanathan, *Nested periodic matrices and dictionaries: New signal representations for period estimation*, IEEE Trans. Signal Processing, 63 (2015), pp. 3736–3750.

[46] I. Tošić and P. Frossard, *Dictionary learning*, IEEE Signal Processing Magazine, 28 (2011), pp. 27–38.

[47] U. Von Luxburg, *A tutorial on spectral clustering*, Statistics and computing, 17 (2007), pp. 395–416.

[48] R. Wang, S. Mou, X. Wang, W. Xiao, Q. Ju, C. Shi, and X. Xie, *Graph structure estimation neural networks*, in Proceedings of the Web Conference 2021, 2021, pp. 342–353.

[49] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, *Robust face recognition via sparse representation*, IEEE transactions on pattern analysis and machine intelligence, 31 (2008), pp. 210–227.

[50] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, *Connecting the dots: Multivariate time series forecasting with graph neural networks*, in Proceedings of the 26th ACM SIGKDD Intl. Conference on Knowledge Discovery & Data Mining, 2020.

[51] H. Xu, Y. Huang, Z. Duan, J. Feng, and P. Song, *Multivariate time series forecasting based on causal inference with transfer entropy and graph neural network*, arXiv preprint arXiv:2005.01185, (2020).

[52] Z. Zhang, Y. Xu, J. Yang, X. Li, and D. Zhang, *A survey of sparse representation: algorithms and applications*, IEEE access, 3 (2015), pp. 490–530.

[53] T. Zhao, Y. Liu, L. Neves, O. Woodford, M. Jiang, and N. Shah, *Data augmentation for graph neural networks*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 11015–11023.