ESE 605, Convex Optimization, Spring 2019, Project Report

Alexandre Amice, Eric Dong, Arnab Sarker

Instructor: Mahyar Fazlyab

Network Inference via the Time-Varying Graphical Lasso [1]

Contents

1 Background

Many phenomena in engineering are characterized by long sequences of multivariate, temporally evolving data. Examples include networks of IoT sensing devices [2], neurological activity [3], and financial markets [4]. It is often possible to model these sequences of data as signals being emitted from nodes in a network known as a Markov Random Field (MRF) [5]. Since these signals are often related to each other in some way, edges are introduced into the network in order to model this dependence or influence on each other.

A number of techniques in graph signal processing [6] exist for extracting information from such network-dependent data, however most techniques require full knowledge of the graph structure. Since in practice only the raw time series data is directly observable, estimating the time varying nature of the dependencies in the time-series data becomes an important problem.

In [1], the authors introduce the *time-varying graphical lasso* (TVGL) problem as a method for inferring the underlying graph structure of this multivariate time data under the assumption of Gaussian signals. The method estimates a series of sparse inverse covariance matrices $\Sigma(t)^{-1}$ given time series data from the network nodes. The purpose of estimating $\Sigma(t)^{-1}$ is that $\left[\Sigma(t)^{-1}\right]_{i,j} \neq 0$ if and only if there is an edge between nodes i and j. We would like the network to be sparse as this leads to a more interpretable model, and is often representative of the true underlying data generation process [2], [4]. Using this formulation, the authors are able to estimate the interdependence between stock prices of major technology companies and the interdependence of sensor readings in a network of automobile sensors. Their approach to the solution relies on consensus optimization to reformulate the original problem into a form where a distributed optimization technique known as the alternating direction method of multipliers (ADMM) can be used to efficiently solve the problem.

2 Problem Formulation

In this section we introduce the problem formulated in [1] and discuss the ADMM based-solution of the authors.

2.1 Time Varying Graphical Lasso as a Convex Optimization Problem

Suppose we have a sequence of synchronous, time-stamped signals x_t evolving over time stamps $t = 1, \dots, T$. The signals are sampled from a multivariate, zero mean, Gaussian distribution in $\mathbb{R}^p$, that is $x \sim \mathcal{N}(0, \Sigma(t))$. For each time stamp t, we observe only n_t signals from the K nodes. Let $S_t = \frac{1}{n_t}\sum_{i=1}^{n_t} x_t^{(i)} x_t^{(i)T}$ be the empirical covariance of the observed data at time t. Finally, let $\Theta_t \in \mathcal{S}_{++}^p$ be the symmetric positive definite matrix denoting the inverse covariance matrix of the observed data at time t. The time varying graphical lasso (TVGL) problem is then defined as:

$$\min_{\Theta_1,\dots,\Theta_T \in \mathcal{S}_{++}^p} \sum_{t=1}^{T} -l_t(\Theta_t) + \lambda \|\Theta_t\|_{\text{od},1} + \beta \sum_{t=2}^{T} \psi(\Theta_t - \Theta_{t-1})\,. \tag{1}$$

In the above $l_t(\Theta_t) = n_t(\log \mathbf{det}\,\Theta_t - \mathbf{tr}(S_t\Theta_t))$ is the log likelihood of Θ_t (up to a constant and a scale), $\|\Theta_t\|_{\text{od},1} = \sum_{i\neq j} |[\Theta_t]_{i,j}|$ is a seminorm of Θ_t, and $\psi(\Theta_t - \Theta_{t-1})$ is a convex function minimized at $\psi(0)$ penalizing changes in the inverse covariance matrix over time. Different $\psi(\cdot)$ functions will be analyzed in section 2.2. We remark that when S_t is full rank, then $l(\Theta_t)$ encourages Θ_t to be near (in the PSD ordering) to S_t^{-1}. When S_t is not full rank, then the penalty term $\psi(\Theta_t - \Theta_{t-1})$, enforces structural similarity between Θ_t in a time neighborhood of t, allowing the estimate of Θ_t to partially rely on the information gained from $S_{t'}$ for t' near t. Finally the parameters λ and β are positive regularization parameters controlling different network behaviors. Larger λ will lead to sparser estimations of Θ_t, but may lead to Θ_t to diverge from the true underlying inverse covariance matrix of the network. Meanwhile, larger β will lead to smaller fluctuations in the value of Θ_t over time stamps.

Remark 1 *We note that* (1) *is a convex optimization problem. The negative log-likelihood is known to be convex in its argument [7], all seminorms are known to be convex due to homogeneity and the triangle inequality, and ψ is assumed to be convex. Since the positive sum of convex functions is again convex [7], we have a convex objective. Furthermore, the constraint set is a convex set, as the set $\mathcal{S}^p_{++}$, is known to be convex, and the Cartesian product of convex sets is convex [7].*

2.2 Network Evolution Penalty

The major contribution of the authors in [1] is the time varying nature of the graphical lasso problem. In order to ensure a solution that is not overly sensitive to minor changes, they introduced the term $\psi(\Theta_t - \Theta_{t-1})$ in (1) in order to penalize changes in the network structure. The choice of the ψ function is specific to the task at hand, and should be informed by some prior information about how the network likely will evolve or the objective in estimating the network. The author introduces five different penalties in [1]. Here we explain two different ones and their interpretation in order to help the reader understand the versatility of these functions:

First, we describe the **Group Lasso ℓ_2-penalty**, defined

$$\psi(X) = \sum_j \|[X]_j\|_2 \ , \tag{2}$$

where $[X]_j$ is the j^{th} column of X. This allows the entire graph to change at a select few points in time, while encouraging the graph to remain exactly the same at all other points in time. This is a useful penalty when the objective is to detect significant events that would cause the graph to restructure.

We also discuss the **Elementwise Laplacian Penalty**, defined

$$\psi(X) = \sum_{i,j} X_{i,j}^2 \, . \tag{3}$$

This penalty encourages the graph structure to vary smoothly over time, avoiding a single edge from disappearing or appearing abruptly and instead encouraging a gradual decay or increase in its influence over time. This is useful when the sampling frequency is high and we wish to observe the continuous evolution of a phenomenon.

3 Problem Solution

We now turn to the solution to TVGL proposed by the authors in [1]. As noted in Remark 1, the problem posed in (1) is a convex optimization problem in T variables and therefore it is possible to solve it naively using the interior point method with standard algorithms such as Newton's Method and the subgradient method (note that the lasso penalty is not differentiable). However, due to the large number of decision variables as well as the high dimensionality of the graph, such naive methods prove computationally slow and therefore the authors propose a much faster algorithm that takes advantage of the problem's additive structure.

This additive structure of (1) allows the authors in [1] to leverage a distributed convex optimization method known as the alternating direction method of multipliers (ADMM). This is done by recasting the problem using consensus optimization in order to give the problem a seperable structure. This new, highly seperable objective transforms the joint minimization problem in (1) into one that can be easily distributed and parallelized.

3.1 Alternating Direction Method of Multipliers

ADMM is a robust, distributed method for solving equality constrained, decomposable optimization problems [8]. Formally ADMM is a method for solving convex problems of the form

$$\begin{aligned} &\min\ f(x) + g(z) \\ &\text{subject to } Ax + Bz = c \end{aligned} \tag{4}$$

ADMM introduces an augmented Lagrangian and solves the problem via dual ascent. The augmented Lagrangian is of the form:

$$\begin{aligned} \mathcal{L}_\rho(x, z, \lambda) &= f(x) + g(z) + \lambda^T(Ax + Bz - c) + (\rho/2)(Ax + Bz - c)^T(Ax + Bz - c) \\ &= f(x) + g(z) + (\rho/2)\left\|(Ax + Bz - c) + (1/\rho)\lambda\right\|_2^2 - 1/(2\rho)\left\|\lambda\right\|_2^2 \\ &= f(x) + g(z)(\rho/2)\left\|(Ax + Bz - c) + u\right\|_2^2 - (\rho/2)\left\|u\right\|_2^2 . \end{aligned} \tag{5}$$

where in the above ρ acts as a learning rate to the dual ascent problem and the term $\|Ax + Bz - c\|_2^2$ acts much like a barrier function, driving the problem to a feasible solution at every step even when performing the ADMM split, and u is the *scaled dual variable.* The ADMM algorithm operates by performing dual ascent on each variable separately, while holding all others constant and then passing the result to the next solver. After initialization, the following is repeated until the algorithm converges:

$$x_{k+1} = \operatorname*{argmin}_x f(x) + (\rho/2)\left\|Ax + Bz_k - c + u_k\right\|_2^2 \qquad (x\text{-minimization}) \tag{6}$$

$$z_{k+1} = \operatorname*{argmin}_z g(z) + (\rho/2)\left\|Ax_k + Bz - c + u_k\right\|_2^2 \qquad (z\text{-minimization}) \tag{7}$$

$$u^{k+1} = u^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \qquad (\text{dual update}) \tag{8}$$

Note that in the above updates for x and z we omit any terms that depend only on z or x respectively and the term $-(\rho/2)\|u\|_2^2$ since they are constant with respect to x and z. We note that ADMM is guaranteed to converge to a global optimum in a convex problem due to strong duality [8].

3.2 Consensus Optimization

Consensus is an optimization method for convex functions composed of sums of other convex functions which leverages ADMM in order to distribute the computation of the minimization. Consensus optimization relies on the observation that the two problems below are equivalent:

$$\min \textstyle\sum_{i=1}^N f_i(x) \iff \min \textstyle\sum_{i=1}^N f_i(x_i) \text{ subject to } x_i - z = 0, \ \ \forall\, i = 1, \dots, N\,. \tag{9}$$

As a result, consensus optimization allows for iterative updates to be done in parallel. The interested reader is referred to [8] for a more careful treatment; we note the basics of consensus optimization here because the algorithm proposed in [1] makes use of a similar idea by introducing additional variables with equality constraints to allow the optimization to be distributed.

3.3 TVGL Solution via ADMM and Consensus Optimization

In order to make computation of the TVGL solution efficient, the authors of [1] introduce consensus variable $Z = \{Z_0, Z_1, Z_2\} = \{(Z_{1,0}, \dots, Z_{T,0}), (Z_{1,1}, \dots, Z_{T-1,1}), (Z_{2,2}, \dots, Z_{T,2})\}$. This allows problem (1) to be rewritten as

$$\begin{aligned} &\min_{\Theta \in \mathcal{S}_{++}^p} \sum_{t=1}^T -l_t(\Theta_t) + \lambda \left\|Z_{t,0}\right\|_{\text{od},1} + \beta \sum_{t=2}^T \psi(Z_{t,1} - Z_{t-1,1}) \\ &\text{subject to } \ Z_{t,0} = \Theta_t, \ \ \Theta_t \in \mathcal{S}_{++}^p \qquad\qquad \text{for } i = 1, \dots, T \\ &\qquad\qquad\quad (Z_{t-1,1}, Z_{t,2}) = (\Theta_{t-1}, \Theta_t) \qquad \text{for } i = 2, \dots, T \end{aligned} \tag{10}$$

Using the matrix version of the augmented Lagrangian with scaled dual variable $U = \{U_0, U_1, U_2\}$ introduced in Section 3.1, the authors write the augmented Lagrangian as:

$$\mathcal{L}_\rho(\Theta, Z, U) = \sum_{t=1}^{T} -l_t(\Theta_t) + \lambda \|Z_{t,0}\|_{\text{od},1} + \beta \sum_{t=2}^{T} \psi(Z_{t,2} - Z_{t-1,1}) + (\rho/2) \sum_{t=1}^{T} \left(\|\Theta_t - Z_{t,0} + U_{t,0}\|_F^2 - \|U_{t,0}\|_F^2 \right)$$
$$+ (\rho/2) \sum_{t=2}^{T} \left(\|\Theta_{t-1} - Z_{t-1,1} + U_{t-1,1}\|_F^2 - \|U_{t-1,1}\|_F^2 + \|\Theta_t - Z_{t,2} + U_{t,2}\|_F^2 - \|U_{t,2}\|_F^2 \right). \tag{11}$$

The ADMM algorithm is used to sequentially minimize the variables Θ then Z, and subsequently update the scaled dual variable U.

3.3.1 Θ Update

With the introduction of the consensus variables, the update for each Θ_t is fully seperable from all other $\Theta_{t'}$. If we allow $A = \frac{Z_{t,0}^k + Z_{t,1}^k + Z_{t,2}^k - U_{t,0}^k - U_{t,1}^k - U_{t,2}^k}{3}$ and $\eta = \frac{n_t}{3\rho}$,the authors write

$$\begin{aligned} \Theta_t^{k+1} &= \underset{\Theta_t \in \mathcal{S}_{++}^p}{\operatorname{argmin}} \mathcal{L}(\Theta, Z^k, U^k) = \underset{\Theta_t \in \mathcal{S}_{++}^p}{\operatorname{argmin}} - \log \mathbf{det}(\Theta_t) + \mathbf{tr}(S_t \Theta_t) + \frac{1}{2\eta} \left\| \Theta_t - \frac{A + A^T}{2} \right\|_F^2 \\ &= \mathbf{prox}_{\eta(-\log \mathbf{det}(\cdot) + \mathbf{tr}(S_t \cdot))} \left((A + A^T)/2 \right) = \frac{\eta}{2} Q \left(D + \sqrt{D^2 + 4\eta^{-1} I} \right) Q^T , \end{aligned} \tag{12}$$

where QDQ^T is the eigendecomposition of $\frac{A+A^T}{2\eta} - S_t$. This is the most computationally expensive part of the algorithm requiring $\mathcal{O}(p^3)$ runtime.

3.3.2 Z Update

The update for each $Z_{t,0}$ can be fully separated from the update for (Z_1, Z_2) as well as the other $Z_{t',0}$. The update for $Z_{t,0}$ is easily written in the form of a proximal operator:

$$\begin{aligned} Z_{t,0}^{k+1} &= \underset{Z_{t,0}}{\operatorname{argmin}} \mathcal{L}_\rho(\Theta, Z, U^k) \\ &= \underset{Z_{t,0}}{\operatorname{argmin}} \lambda \|Z_{t,0}\|_{\text{od},1} + \frac{\rho}{2} \left\| \Theta_t^{k+1} + U_t^k - Z_{t,0}^k \right\|_F^2 = \mathbf{prox}_{\frac{\lambda}{\rho} \|\cdot\|_{\text{od},1}} \left(\Theta_t^{k+1} + U_t^k \right) . \end{aligned} \tag{13}$$

The proximal operator is then defined element-wise, yielding a final solution of:

$$\left[Z_{t,0}^{k+1} \right]_{i,j} = \begin{cases} \mathbf{sgn} \left(\left[\Theta_t^{k+1} + U_t^k \right]_{i,j} \right) \left(\left| \left[\Theta_t^{k+1} + U_t^k \right]_{i,j} \right| - \frac{\lambda}{\rho} \right)_+ & (i \neq j) \\ \left[\Theta_t^{k+1} + U_t^k \right]_{i,i} & (i = j) . \end{cases} \tag{14}$$

The most complex part of the update is the joint update of $Z_{t,1}, Z_{t,2}$. Note that for each t, this update can be separated from all other updates $Z_{t',1}, Z_{t',2}$. In order to simplify notation, the authors define $\tilde{\psi} \left(\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} \right) = \psi(Z_2 - Z_1)$. This allows the Z_1, Z_2 update to be written as:

$$\begin{aligned} \begin{bmatrix} Z_{t-1,1}^{k+1} \\ Z_{t,2}^{k+1} \end{bmatrix} &= \underset{Z_{t-1,1}^{k+1}, Z_{t,2}^{k+1}}{\operatorname{argmin}} \beta \psi(Z_{t,2} - Z_{t-1,1}) + \frac{\rho}{2} \left\| \Theta_{t-1}^k + U_{t-1,1}^k - Z_{t-1,1} \right\|_F^2 + \frac{\rho}{2} \left\| \Theta_t^k + U_{t,1}^k - Z_{t,2} \right\|_F^2 \\ &= \underset{Z_{t-1,1}^{k+1}, Z_{t,2}^{k+1}}{\operatorname{argmin}} \beta \tilde{\psi} \left(\begin{bmatrix} Z_{t,2} \\ Z_{t-1,1} \end{bmatrix} \right) + \frac{\rho}{2} \begin{bmatrix} \left\| \Theta_{t-1}^k + U_{t-1,1}^k - Z_{t-1,1} \right\|_F^2 \\ \left\| \Theta_t^k + U_{t,1}^k - Z_{t,2} \right\|_2^2 \end{bmatrix} \\ &= \mathbf{prox}_{\frac{\beta}{\rho} \tilde{\psi(\cdot)}} \begin{bmatrix} \Theta_{t-1}^k + U_{t-1,1}^k \\ \Theta_t^k + U_{t,1}^k \end{bmatrix} . \end{aligned} \tag{15}$$

Efficient solutions to TVGL thus rely on finding efficient, preferably closed form, solutions to the proximal operator of the $\tilde{\psi}$ function. Provided that these operators can be evaluated in faster than $\mathcal{O}(p^3)$ time, then the time complexity of the problem is not increased. The authors of [1] give closed form solutions for each of the ψ penalties introduced in Section 2.2 in their paper.

3.3.3 U Update

The update for the scaled dual variable is the matrix extension of the update introduced in 3.1.

$$U^{k+1} = \begin{bmatrix} U_0^{k+1} \\ U_1^{k+1} \\ U_2^{k+1} \end{bmatrix} = \begin{bmatrix} U_0^{k} \\ U_1^{k} \\ U_2^{k} \end{bmatrix} + \begin{bmatrix} \Theta^{k+1} - Z_0^{k+1} \\ (\Theta_1^{k+1}, \ldots, \Theta_{T-1}^{k+1}) - Z_1^{k+1} \\ (\Theta_2^{k+1}, \ldots, \Theta_{T}^{k+1}) - Z_2^{k+1} \end{bmatrix}.$$

4 Discussion

In this section we discuss the successes presented in the authors' paper [1], as well as some of the limitations of their work. We additionally take note of the applicability of their results as well as how their paper both makes use of and extends the topics of the course.

4.1 Merits of Time Varying Graphical Lasso

Through numeric experiments, the authors of [1] show that the combination of consensus optimization and ADMM allow their method to significantly outperform general purpose solvers for the TVGL problem. For problems on the order of 10^3 unknowns, the solution presented in Section 3 takes milliseconds while a naive implementation using the general purpose interior point method takes over 10^4 seconds or around three hours as it scales as $\mathcal{O}(p^6)$. A naive implemenation of ADMM take 10^3 seconds or 16 minutes, and advanced, general purpose operator splitting solvers still take over 10 seconds. This scalability shows the significant advantage that a targeted solution to the TVGL problem can have. Furthermore, on synthetic data where the true graph structure is known the authors show very high accuracy in predicting graph structure which outperforms both static graphical lasso estimation as well as a kernel based method.

Additionally, the formulation for TVGL is quite satisfying due to its strict emphasis on yielding interpretable results. The restriction that the signals be produced by random Gaussian random variables, as well as the sparsity requirement, allows for extremely interpretable results in real world examples as presented in the paper.

One area where the paper stands to improve is by relaxing the requirement that x have a stationary mean. In a time varying problem where in the structure of the network is itself changing, it is not unreasonable to believe that the mean of the individual signals would also be changing, particularly in event detection problems such as when using group lasso ℓ_2-penalty as events that cause the graph structure to change could also cause the distribution's mean to change. The authors willingly acknowledge this and present this problem as a future work.

Another shortcoming of the paper is its lack of careful treatment on selecting the parameters λ and β. As with most regularization parameters, the authors use empirical methods to determine the best regularization parameters via a grid search. The authors only briefly mention the use of the Akaike Information Criteria as a performance measure on the model parameters yet make no mention as to which parameters they are attempting to measure in order to prevent overfitting.

4.2 Applications

The TVGL provides a general purpose framework for estimating the inverse covariance matrices of a jointly Gaussian distribution which varies over time. Such problems arise naturally in a variety of setting where in networks of agents, sensors, or signals interact. The authors choose to analyze two particular cases in [1].

The first application the authors explore is in estimating the covariance of the stock prices of large companies. Large companies tend to be well established and therefore we expect their influence on the market to vary only slightly over time. However, occasionally events occur which allow a single stock's influence to increase precipitously, reweighing its influence on other companies. The authors demonstrate such a shift can be detected by using TVGL to estimate the inverse covariance matrix of large technology companies. Using TVGL, the authors noted that Apple's stock became significantly more influential to large technology companies such as Intel and Microsoft upon the release of the IPad in January 2010. Furthermore, they note that the estimation of the network was reasonable as Apple's stock was seen to correlate with similar technology companies while note correlating directly with other large companies such as Boeing.

The second application that the authors explore is a network of automobile sensors. Modern cars contain hundreds of sensors each measuring a variety of internal and external condition. Due to the different styles with which people drive, time varying differences in the network of relationships between these sensors can be created in order to understand different driving habits. The authors claim that thorough analysis of these sensor relationships could be used to compare driver ability as well as detect impaired driving. As a case study, the authors attempt to estimate the relationship between the data emitted by a car turning a corner. During the turn, the steering angle of the wheel is shown to occupy a central, highly connected position in the the network. Before and after the turn, the steering angle of the wheel becomes far less connected, migrating to the edge of the network. This can be interpreted as the steering angle being able to explain away most of the variation in the other sensors during a turn, but not before or after.

In our extension discussed in Section 5, we seek to apply the TVGL result to fMRI brain data, U.S. macroeconomic data, and weather data.

4.3 TVGL Connection To ESE 605

As noted in Remark 1, the problem presented in (1) is a convex optimization problem. Therefore, it is directly solvable using only the techniques from this course such as the interior point method. This shows the power of the topics in this course to solve an extremely large class of problems. However, such general purpose techniques for solving convex problems suffer from scaling poorly in the dimension. For the TVGL problem, the interior point method scales at $\mathcal{O}(p^6)$ [9].

In order to reduce the computation time, the authors introduced a framework that leveraged the structure of the problem in order to significantly reduce the time complexity to obtain a solution. By recasting the problem into new variables, the authors were capable of making the problem highly separable, allowing for separate minimizations to be performed in each decision variable rather than jointly. Such separability not only allows for the solution to be parallelized, it additionally reduces the complexity of each minimization step in the algorithm. Lastly, once each of the joint problems has been decomposed into subproblems, the basic optimization techniques developed in this course need to be used to find the solution to these easier subproblems.

The main tool the authors use to solve the actual minimization problem, ADMM, relies on transforming the equality constrained optimization problem present in (10) to a form of the Lagrangian dual and performing dual ascent on the variables in the problem. As we showed in class, this yields the same solution as the primal problem because convex optimization problems with strictly feasible points exhibit no duality gap. This is a fundamental theorem in convex optimization that allows for efficient solutions to constrained convex problems.

Finally, the authors manage to write their solution in closed form by use of the proximal operator. The proximal operator, as defined in the latter part of the course, is a useful tool for rapidly finding the solution to convex problems with separable objective functions where in the first function is convex and differentiable and the second is convex with an inexpensive proximal operator. The penalty function and off diagonal ℓ_1 penalty in TVGL are not always differentiable, yet the use of proximal operators still allows for an efficient solution to be found.

5 Extensions

In this section we propose two extensions to the paper. The first extension we propose is a new convex penalty ψ function. In particular, we examine the effect of putting a nuclear or trace norm penalty on the variation of the inverse covariance matrices. If A is a matrix with singular value decomposition $U \operatorname{diag}(\sigma(A))V^T$, then the nuclear norm of a matrix A is defined as

$$\|A\|_* = \mathbf{tr}\,\sqrt{AA^T} = \textstyle\sum_{i=1}^n \sigma_i(A)\,. \tag{16}$$

The nuclear norm has a known closed form proximal operator[1] [11]

$$\underset{X}{\operatorname{argmin}} \left(\|X\|_* + \frac{1}{2\beta} \|X - A\|_F^2 \right) = U \operatorname{diag}(\max\{\sigma(A) - \beta, 0\})V^T\,. \tag{17}$$

We are motivated to use this penalty due to the relationship of covariance matrices and singular value decompositon in principal component analysis [12]. The use of the nuclear norm encourages the system to maintain some notion of a constant level of randomness. Moreover, due to the relationship between the nuclear norm and rank of a matrix [11], we expect this penalty to cause a few nodes in the system to be strongly connected to almost all other nodes, illustrating that almost all of the randomness in the system can be explained away by only a small number of variables. We note that in Equation (17), an eigenvalue decomposition must occur which runs in $\mathcal{O}(p^3)$ time. While this is prohibitive in some problems, we note that it does not increase the asymptotic time complexity of TVGL.

We then extend the TVGL to three particular applications which are not explored directly in [1].

The first application we analyze is fMRI data of human brains. The human brain is known to exhibit dynamic network activity [13]. This structure is known to change due to a variety of stimuli, and has been directly observed to change when humans learn [14]. fMRI techniques are capable of capturing this connectivity over time by observing the level of activation across different brain regions [15]. We run TVGL on 15 randomly subsampled brain regions from fMRI data taken from healthy humans lying quietly with eyes closed.[2] As no stimuli is presented to the participants over the duration of the fMRI, we hypothesize that brain activity evolves smoothly with a few central regions and therefore experiment with the Laplacian penalty and our own nuclear norm penalty. From Figure 1 and Figure 2 in Appendix A, we can observe using a Laplacian penalty results in gradual updating, representing smoothly evolving brain activity. In particular, Figure 2 charts the temporal deviation (TD) at each time step, t which measures the amount of change in the estimate at t. Temporal deviation is defined as $\mathrm{TD}_t = \|\Theta_t - \Theta_{t-1}\|_F$. From Figure 4 and Figure 5 in Appendix A, we can see that using the nuclear norm penalty enforces only a few brain regions to be strongly connected. This is likely the more representative model of human brain connectivity as it is known that typically only a few regions of the resting brain are active at once [16].

The second example we explored was U.S. macroeconomic data. Patterns among macroeconomic data are closely related but these relationships often shift during national shocks, such as recessions and changes in monetary policy [17]. We apply TVGL on quarterly U.S. macroeconomic data gathered by the St. Louis Federal Reserve. The gathered data includes several metrics measuring the business cycle and overall economic health of the country, such as the consumer price index and gross domestic product. We used the ℓ_2 penalty because as mentioned, we expected changes in the network to come from significant events that affect the entire structure. We also tested the nuclear norm penalty under the assumption that a few macroeconomic indicators provide the sources of variation. From Appendix B, it seems that the estimates are nearly identical between using the group ℓ_2 norm and the nuclear norm as the penalty. What is interesting though is that from Figures 7 and 10 we can clearly see spikes in temporal deviation, corresponding to very large shifts in the estimated graph structure around when actual U.S recessions occurred. For example, TVGL is able to detect global network restructurings at 1973-1975, 2000-2002, and 2007-2009, which align with the recessions caused by stagflation, the dot-com bubble, and the subprime mortgage crisis. Additionally,

[1]Note that this solution uses a soft-threshold function as mentioned in [10] on the singular values, which can be simplified as singular values are defined to be positive.

[2]The preprocessed data can be found at `http://intramural.nimh.nih.gov/chp/articles/matlab.html`

	p	T	ℓ_1	ℓ_2	Laplacian	Nuclear Norm
TVGL [1]	6	20	1.59	1.28	2.56	2.69
	8	50	3.10	3.42	3.12	7.74
	10	100	6.78	14.66	7.11	8.99
	15	200	21.22	33.58	21.54	51.46
CVX	6	20	27.26	15.24	25.12	19.71
	8	50	236.21	66.19	167.29	94.02
	10	100	1607.30	378.31	653.29	413.51
	15	200	>3600	>3600	>3600	>3600

Table 1: Runtime (in seconds) of algorithms for different penalties, algorithms, and data sizes.

Figure 8 highlights the significant graph restructuring from Q4-1987 to Q1-1988 where the previously highly connected node representing gross domestic product loses connectivity entirely. This corresponds to a stock market crash in October 1987 followed by aggressive monetary policy tightening and the Gulf War.

Our last application was using TVGL on hourly and daily weather sensor readings collected this year at the Philadelphia International Aiport, such as temperature and humidity. We hypothesize that day to day and hour by hour changes in weather conditions would alter the underlying relationships between the different sensor readings. Because such changes may evolve slowly (e.g. seasonality) or come as sudden large shocks (e.g. storms) and we expect only a few sensors to be highly connected, we experimented with the ℓ_2 norm, the Laplacian, and the nuclear norm as the penalties. We obtain nearly identical results regardless of the penalty type used. However, from Figure 11 and Figure 12 in Appendix C, we can see that the estimated network changes slowly over the months. Seasonality seems to be the primary driver in this evolution as nodes like maximum temperature gain connectivity while nodes like snowfall lose connectivity over the course of the shift from winter to late spring. On the other hand, we can see from Figure 11 and Figure 12 that the estimated network changes much more frequently, aligning with more granular weather patterns. For example, TVGL is able to detect the shift in structure during storms on April 5 and April 8. Through fMRI brain data, macroeconomic data, and weather data, we show how TVGL can accurately identify the sources of randomness in a system. We further note that the nuclear norm can act as an effective proxy to well selected penalty functions which can be useful in situations where prior knowledge on the evolution of the network is difficult to gather.

6 Implementation

We provide a MATLAB implementation of the algorithm discussed in Section 4 of [1], as well as an implementation of Problem (1) using CVX [18]. The code to implement the algorithm posed in [1] uses some structure from [10] to help with readability and debugging.

Four different convex ψ functions are implemented in this codebase: the element-wise ℓ_1 norm $\psi(X) = \sum_{i,j} |X_{i,j}|$, the induced ℓ_2 norm from Equation (2), the Laplacian norm from Equation (3), and finally the nuclear norm from Equation (16). Each of these functions has a closed form proximal operator.

Table 1 presents the runtime of different algorithms, using different penalties with the two different solution methods. For each experiment, 10 observations were observed at each time step and the data is synthetically created to have a global shift at time $T/2$; moreover, we select $\lambda = 0.4$, $\beta = 1000$, and $\rho = 0.6$ as these parameters consistently led to convergence of both algorithms over all dataset sizes. Our results confirm that the algorithm posed in [1] is orders of magnitude faster than traditional interior point methods such as that used in CVX. Moreover, we note that for our implementation, the nuclear norm takes a longer time to run due to the singular value decomposition step required to compute the proximal operator. However, in the CVX implementation it is interesting to note that the nuclear norm penalty function does not have this drawback, potentially due to the way that the CVX library is written, seeing as NORM_NUC is a built-in function for CVX which computes the nuclear norm.

References

[1] D. Hallac, Y. Park, S. Boyd, and J. Leskovec, "Network inference via the time-varying graphical lasso," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, 2017, pp. 205–213.

[2] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[3] R. P. Monti, P. Hellyer, D. Sharp, R. Leech, C. Anagnostopoulos, and G. Montana, "Estimating time-varying brain connectivity networks from functional mri time series," *NeuroImage*, vol. 103, pp. 427–443, 2014.

[4] D. F. Ahelegbey, "The econometrics of bayesian graphical models: A review with financial application," *University Ca'Foscari of Venice, Dept. of Economics Research Paper Series No*, vol. 13, 2016.

[5] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Foundations and Trends in Machine Learning*, vol. 1, no. 1–2, pp. 1–305, 2008.

[6] A. Ortega, P. Frossard, J. Kovačević, J. M. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

[7] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[8] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[9] K. Mohan, P. London, M. Fazel, D. Witten, and S.-I. Lee, "Node-based learning of multiple gaussian graphical models," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 445–488, 2014.

[10] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Matlab scripts for alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010.

[11] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[12] M. E. Wall, A. Rechtsteiner, and L. Rocha, "Singular value decomposition and principal component analysis," *In A Practical Approach to Microarray Data Analysis*, vol. 5, Sep. 2002.

[13] D. S. Bassett and E. Bullmore, "Small-world brain networks," *The neuroscientist*, vol. 12, no. 6, pp. 512–523, 2006.

[14] D. S. Bassett, N. F. Wymbs, M. A. Porter, P. J. Mucha, J. M. Carlson, and S. T. Grafton, "Dynamic reconfiguration of human brain networks during learning," *Proceedings of the National Academy of Sciences*, vol. 108, no. 18, pp. 7641–7646, 2011.

[15] P. Vertes, A. Alexander-Bloch, N. Gogtay, J. Giedd, J. Rapoport, and E. Bullmore, "Simple models of human brain functional networks," *Proceedings of the National Academy of Sciences*, vol. 109, pp. 5868–5873, 2012.

[16] J. Cabral, M. Kringelbach, and G. Deco, "Exploring the network dynamics underlying brain activity during rest," *Progress in Neurobiology*, vol. 114, pp. 102–131, 2014.

[17] B. McCallum, "Theoretical analysis regarding a zero lower bound on nominal interest rates," *Journal of Money, Credit and Banking*, vol. 32, pp. 870–904, 2000.

[18] M. Grant and S. Boyd, *CVX: Matlab software for disciplined convex programming, version 2.1*, `http://cvxr.com/cvx`, Mar. 2014.

A fMRI Experiment Figures

Note each numerical node labels represents one of the 140 possible brain regions, and that we have subsampled 15 nodes for clarity of presentation.

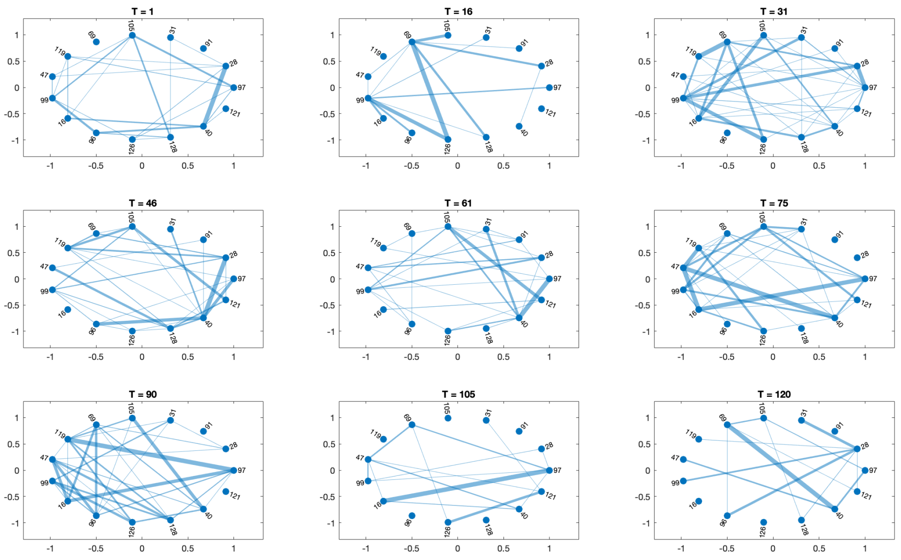

Figure 1: Sample graphs from TVGL with Laplacian penalty on fMRI data

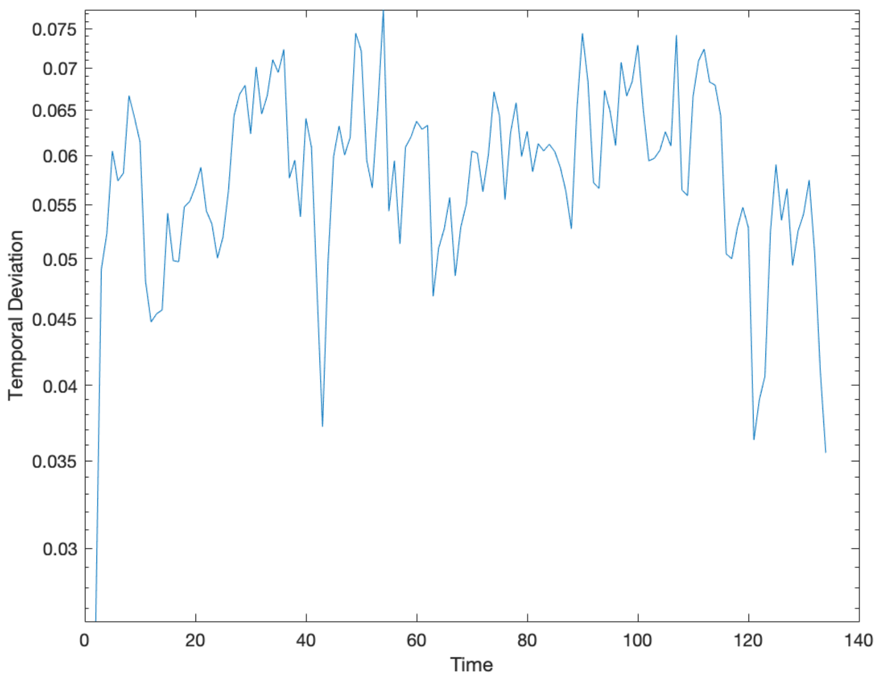

Figure 2: Temporal deviation from TVGL with Laplacian penalty on fMRI data

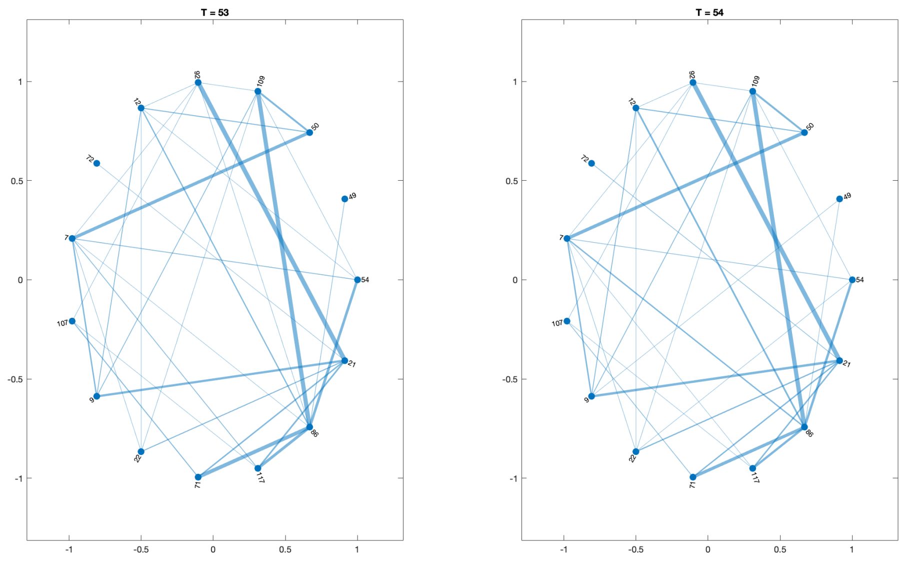

Figure 3: Graphs before and during peak temporal deviation from TVGL with Laplacian penalty on fMRI data

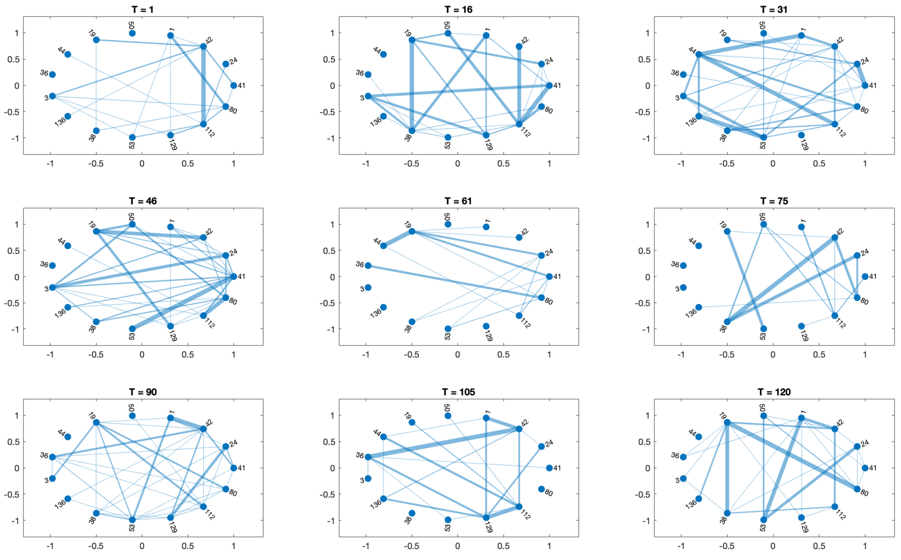

Figure 4: Sample graphs from TVGL with nuclear norm penalty on fMRI data

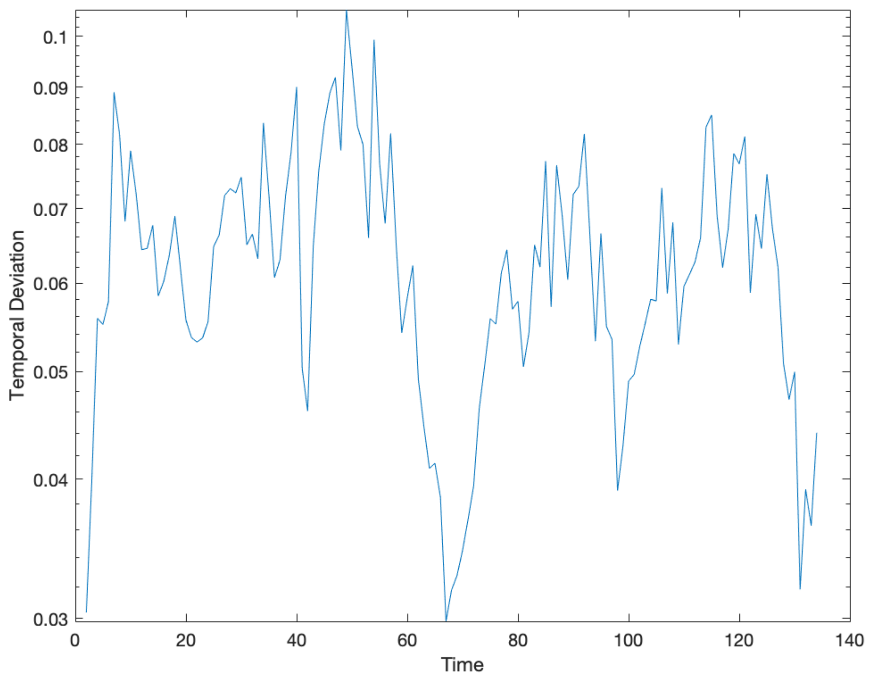

Figure 5: Temporal deviation from TVGL with nuclear norm penalty on fMRI data

B Macroeconomic Experiment Figures

Node label abbreviations are paid compensation of employees in \$ billions (COE), consumer price index (CPIAUCSL), effective federal funds rate (FEDFUNDS), government consumption expenditures and investment in \$ billions (GCE), gross domestic product (GDP), gross domestic product in \$ billions (GDPDEF), gross private domestic investment in \$ billions (GDPI), ten-year treasury bond yield (GS10), non-farm business sector index of hours worked (HOANBS), M1 money supply (M1SL), M2 money supply (M2SL), personal consumption expenditures in \$ billions (PCEC), three-month treasury bill yield (TB3MS), and unemployment rate (UNRATE).

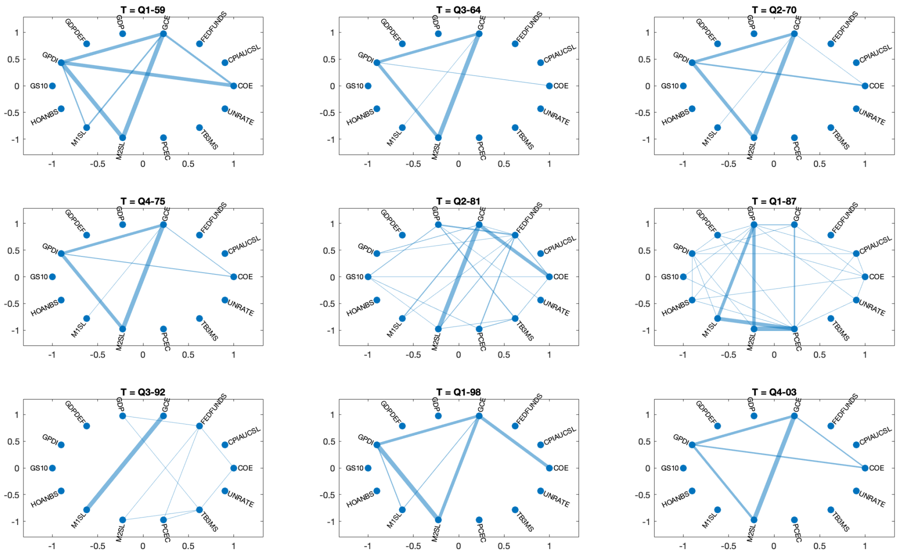

Figure 6: Sample graphs from TVGL with ℓ_2 penalty on macroeconomic data

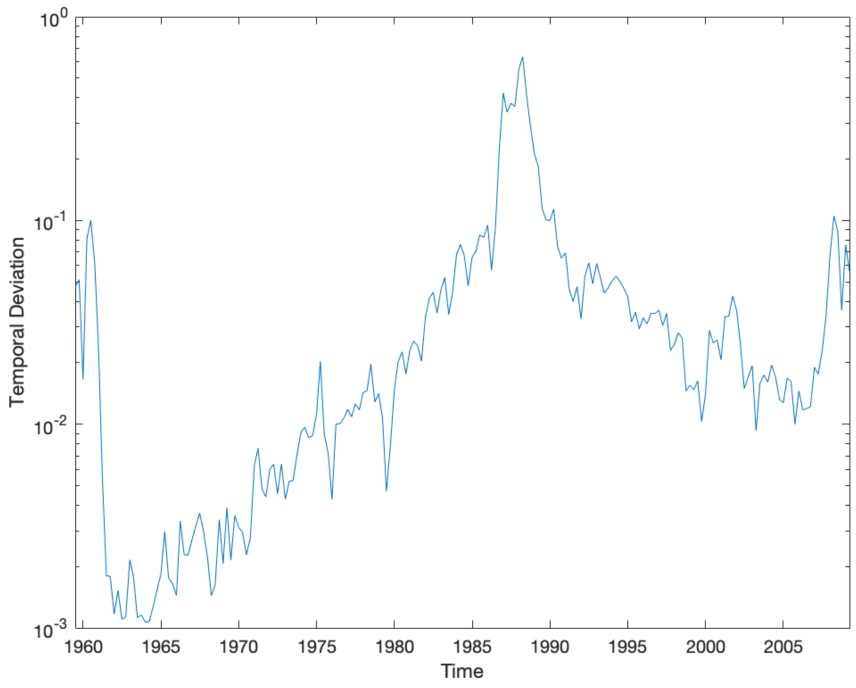

Figure 7: Temporal deviation from TVGL with ℓ_2 penalty on macroeconomic data

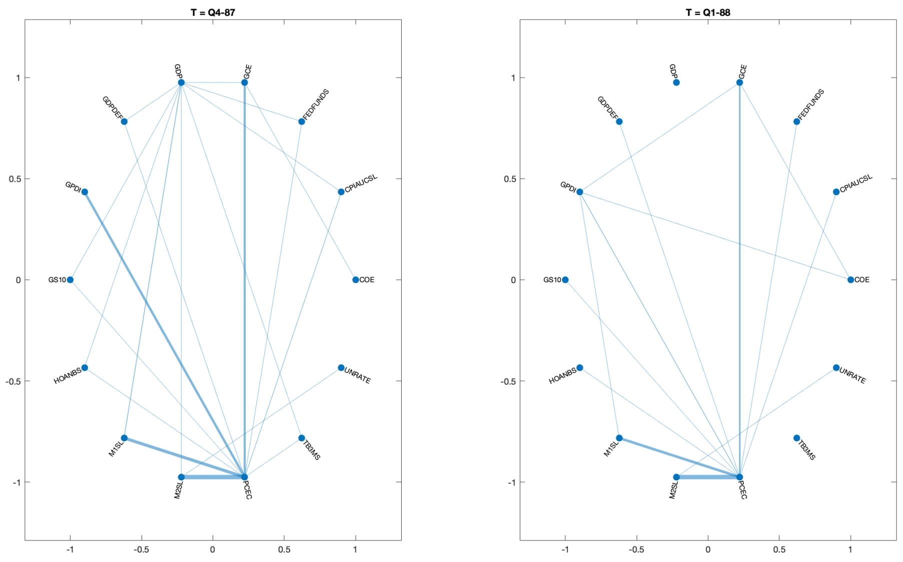

Figure 8: Graphs before and during peak temporal deviation from TVGL with ℓ_2 penalty on macroeconomic data

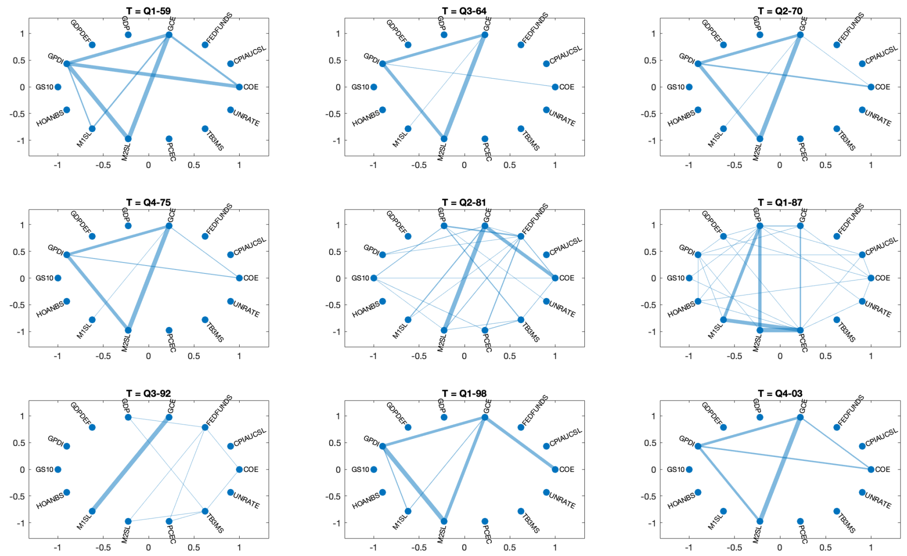

Figure 9: Temporal deviation from TVGL with nuclear norm penalty on macroeconomic data

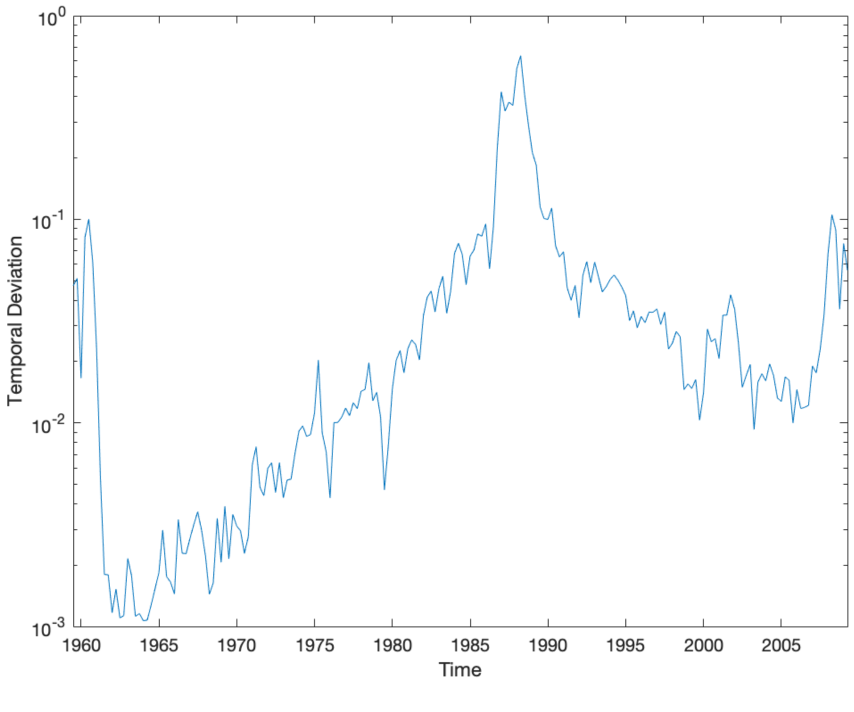

Figure 10: Temporal deviation from TVGL with nuclear norm penalty on macroeconomic data

C Weather Experiment Figures

Node label abbreviations for daily data are temperature (Temp), humidity (Hum), pressure (Pres), wind speed (WS), maximum temperature (Max Temp), minimum temperature (Min Temp), peak wind speed (PWS), precipitation (Prec), snow depth (SD), snow fall (Sf), and sustained wind speed (SWS). Node label abbreviations for hourly data are temperature (Temp), precipitation (Prec), humidity (Hum), pressure (Pres), visibility (Vis), wind gust speed (WGS), and wind speed (WS).

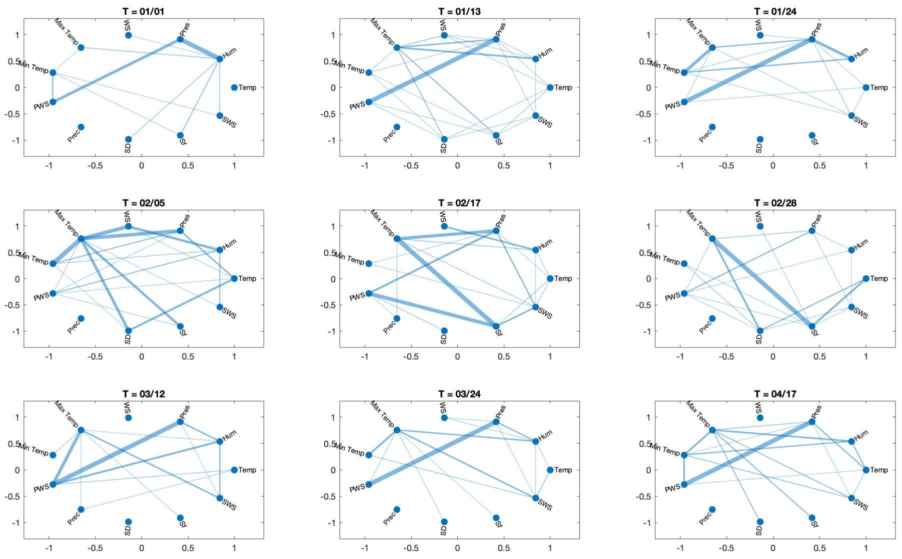

Figure 11: Sample graphs from TVGL with ℓ_2 penalty on daily weather data

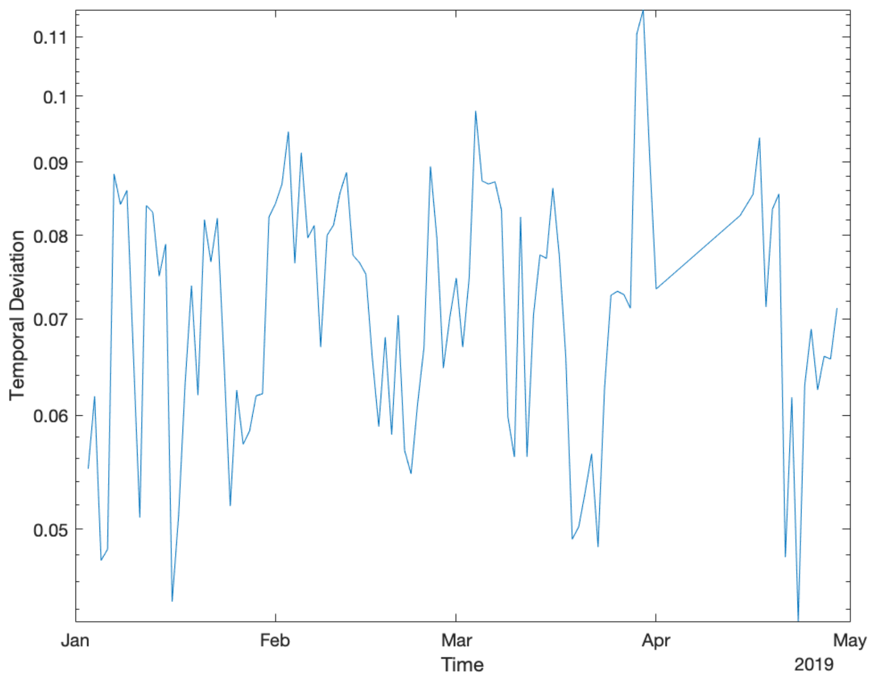

Figure 12: Temporal deviation from TVGL with ℓ_2 penalty on daily weather data

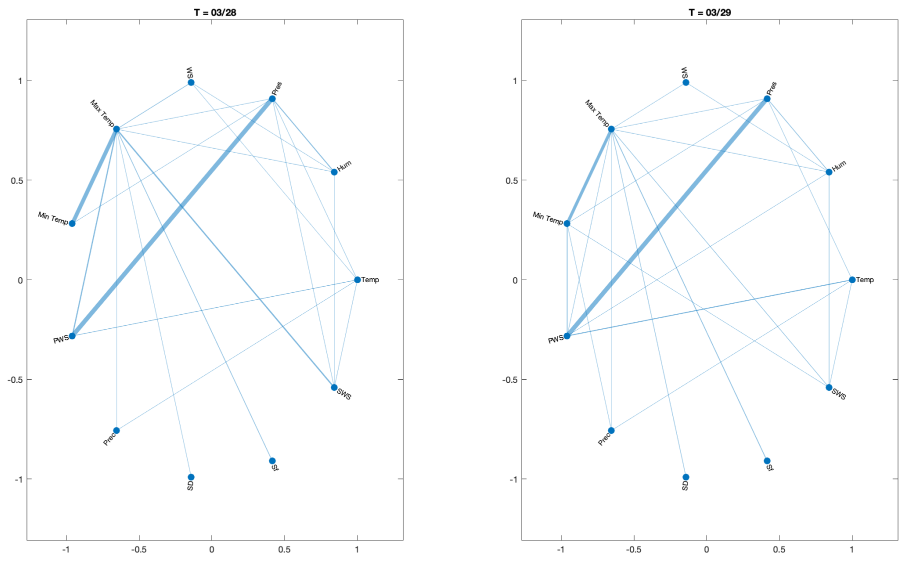

Figure 13: Graphs before and during peak temporal deviation from TVGL with ℓ_2 penalty on daily weather data

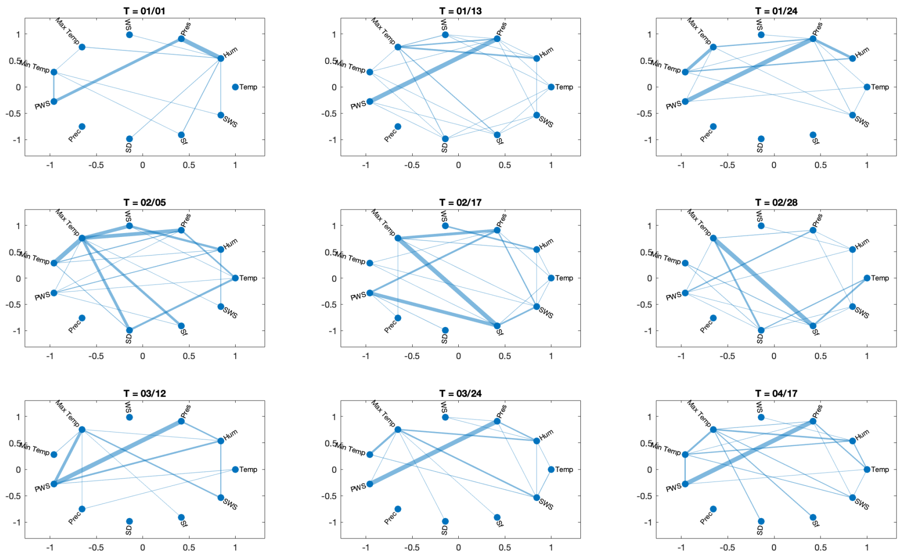

Figure 14: Sample graphs from TVGL with Laplacian penalty on daily weather data

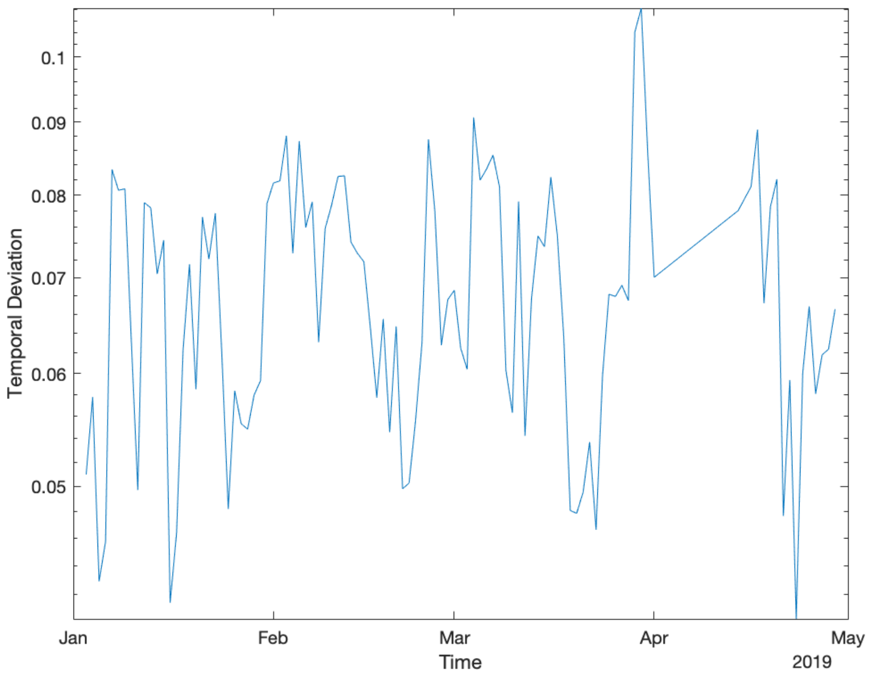

Figure 15: Temporal deviation from TVGL with Laplacian penalty on daily weather data

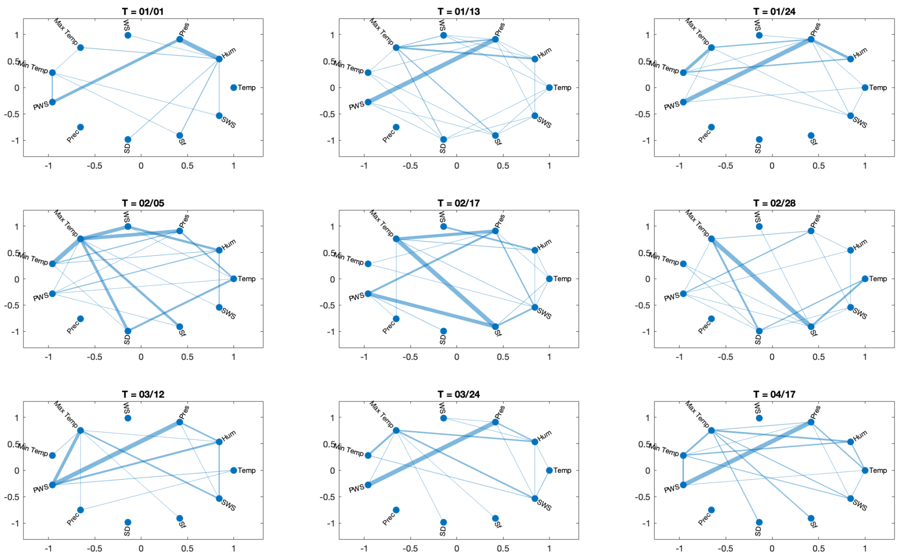

Figure 16: Sample graphs from TVGL with nuclear norm penalty on daily weather data

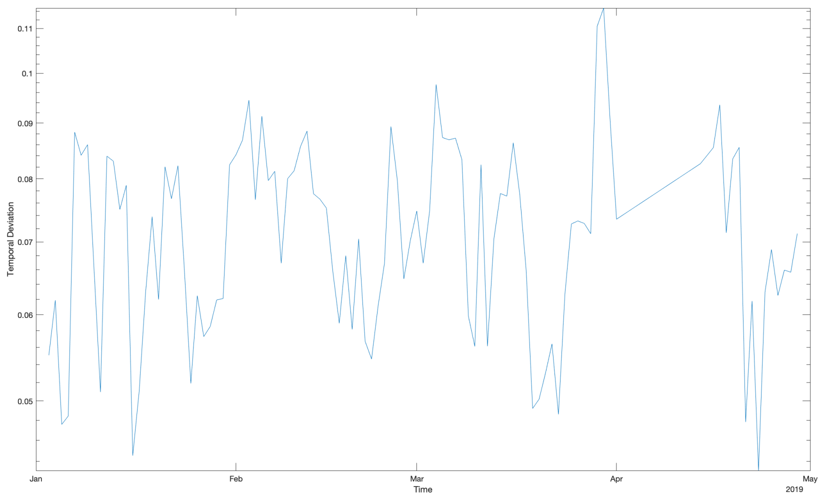

Figure 17: Temporal deviation from TVGL with nuclear norm penalty on daily weather data

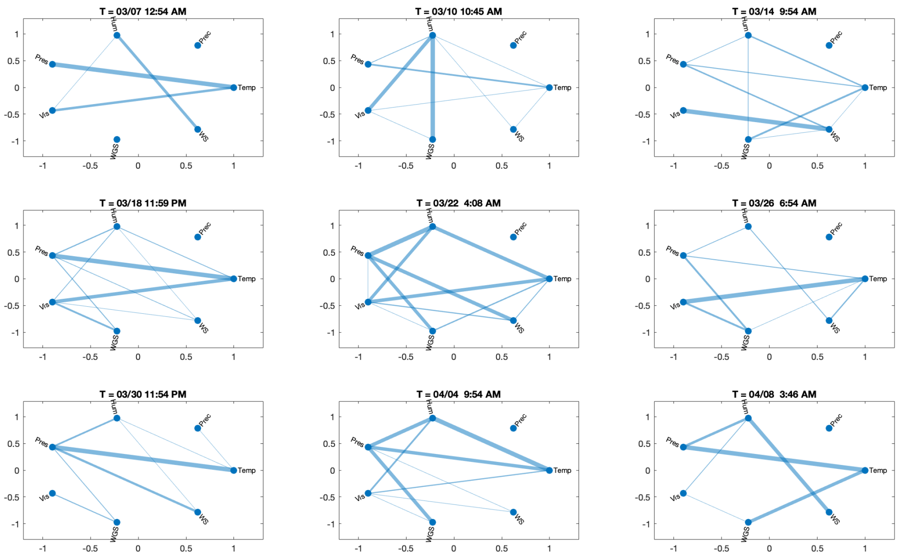

Figure 18: Sample graphs from TVGL with ℓ_2 penalty on hourly weather data

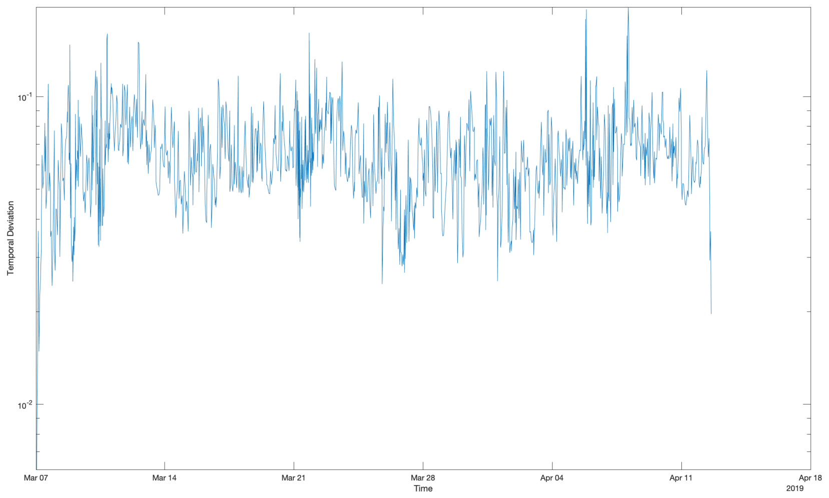

Figure 19: Temporal deviation from TVGL with ℓ_2 penalty on hourly weather data

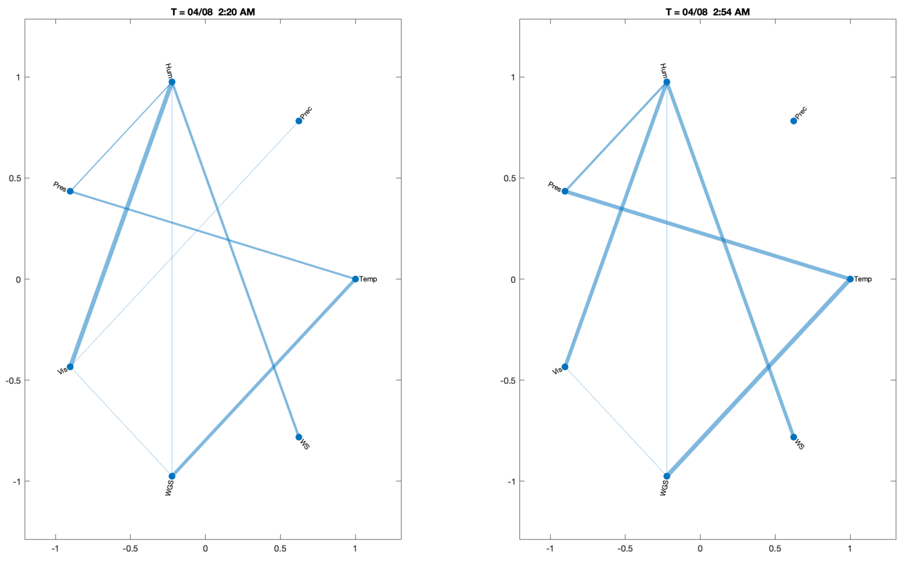

Figure 20: Graphs before and during peak temporal deviation from TVGL with ℓ_2 penalty on hourly weather data

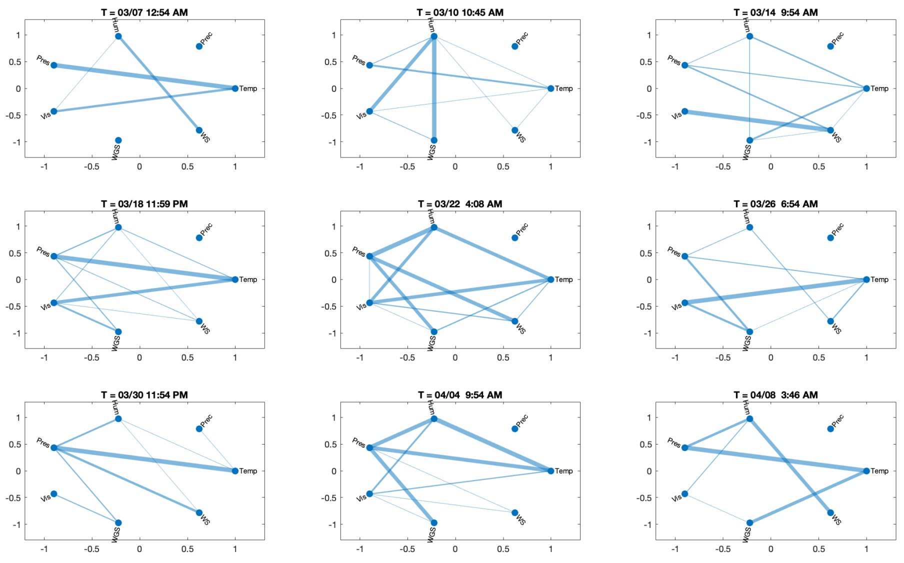

Figure 21: Sample graphs from TVGL with Laplacian penalty on hourly weather data

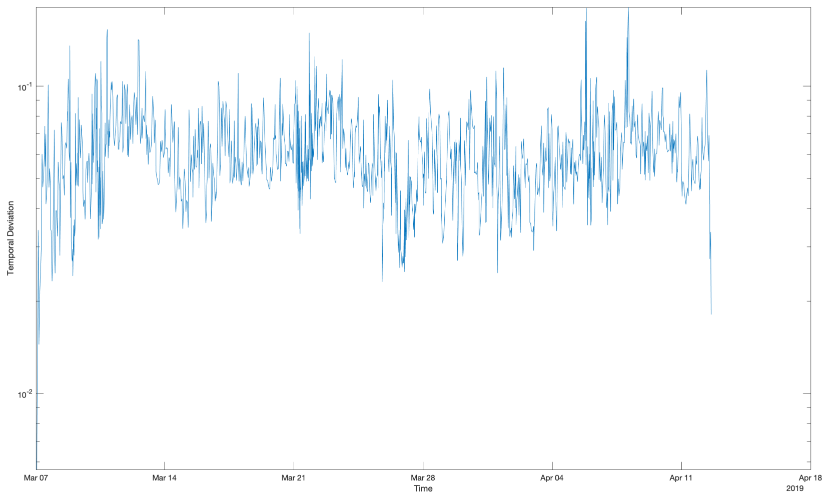

Figure 22: Temporal deviation from TVGL with Laplacian penalty on hourly weather data

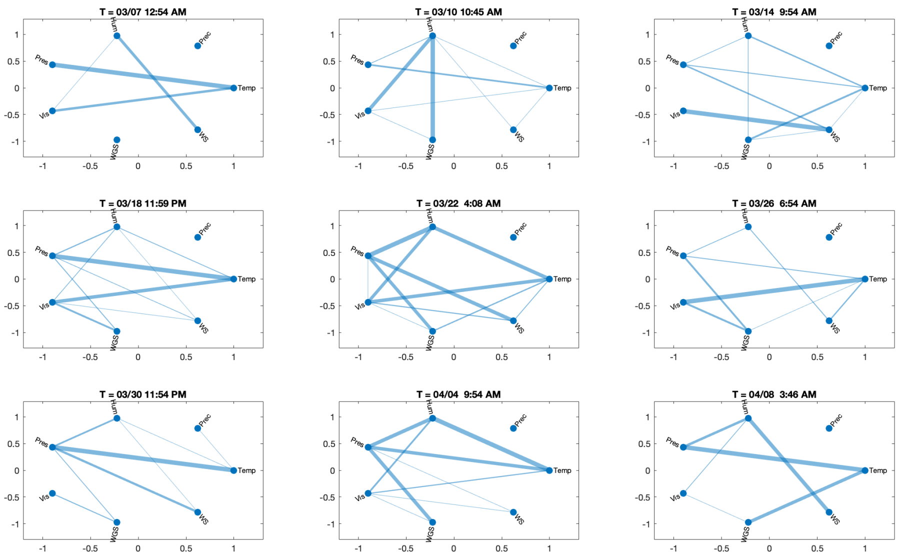

Figure 23: Sample graphs from TVGL with nuclear norm penalty on hourly weather data

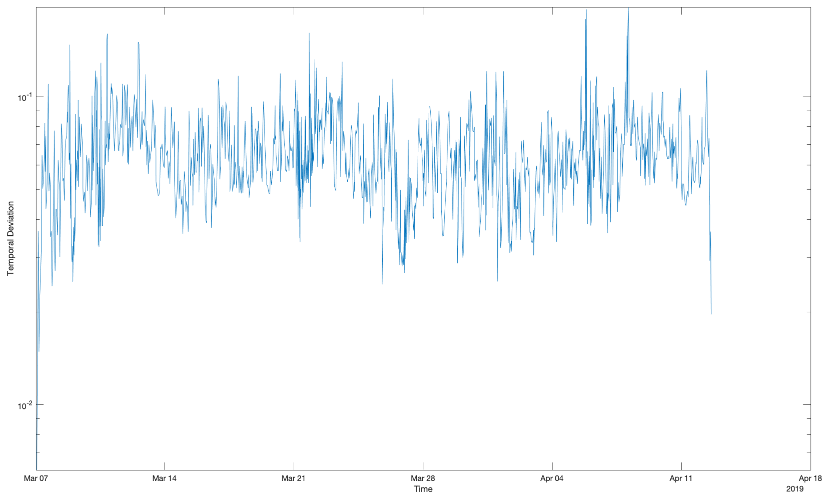

Figure 24: Temporal deviation from TVGL with nuclear norm penalty on hourly weather data