Online Graph Learning From Time-Varying Structural Equation Models

Natali, Aberto; Isufi, Elvin; Coutino, Mario; Leus, Geert

DOI
10.1109/IEEECONF53345.2021.9723163

Publication date
2021

Document Version
Final published version

Published in
55th Asilomar Conference on Signals, Systems and Computers, ACSSC 2021

2021 55th Asilomar Conference on Signals, Systems, and Computers | 978-1-6654-5828-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IEEECONF53345.2021.9723163

ONLINE GRAPH LEARNING FROM TIME-VARYING STRUCTURAL EQUATION MODELS

Alberto Natali[1], *Elvin Isufi*[1], *Mario Coutino*[1,2] *and Geert Leus*[1]

[1] Delft University of Technology, Delft, The Netherlands
[2]Radar Technology, TNO, The Hague, The Netherlands

ABSTRACT

Topology identification is an important problem across many disciplines, since it reveals pairwise interactions among entities and can be used to interpret graph data. In many scenarios, however, this (unknown) topology is time-varying, rendering the problem even harder. In this paper, we focus on a time-varying version of the structural equation modeling (SEM) framework, which is an umbrella of multivariate techniques widely adopted in econometrics, epidemiology and psychology. In particular, we view the linear SEM as a first-order diffusion of a signal over a graph whose topology changes over time. Our goal is to learn such time-varying topology from streaming data. To attain this goal, we propose a real-time algorithm, further accelerated by building on recent advances in time-varying optimization, which updates the time-varying solution as a new sample comes into the system. We augment the implementation steps with theoretical guarantees, and we show performances on synthetic and real datasets.

Index Terms— Dynamic topology identification, graph signal processing, graph learning, time-varying optimization.

1. INTRODUCTION

Graph topology learning is fundamental to identify complex dependencies in irregular datasets and plays a crucial role in the successful deployment of graph signal processing and machine learning tools [1]. Up to date, most efforts are on learning static graph topologies [1, 2], an assumption which may not hold in many real scenarios, e.g. sensor networks and financial markets, which naturally exhibit a temporal variability. This *dynamic* graph learning task has mainly been addressed by a two-step approach: *i)* first, all the samples are collected and split into possibly overlapping windows; and then *ii)* the topology associated to each window is inferred from the data; see [3] for a thorough review on the subject. This "modus-operandi" fails to address the online (data-streaming) setting, recently investigated in [4, 5, 6, 7, 8] under different model assumptions.

Learning a graph from data requires a prior, a model, or an assumption of how the data reflect the underlying graph structure. In this work, we focus on a *time-varying* version of the structural equation modeling (SEM) framework [9], which is a general statistical modeling technique for multivariate data which is widely adopted in several different disciplines, such as in social sciences [10], psychology [11] and genetics [12]. The popularity of SEM is mainly due to its ease of use and the ability to capture the influence of exogenous factors as well as causal effects for the variable of interest [13] (although we should point out that the latter interpretation has been a cause of concern [14]).

Dynamic versions of SEM have been proposed in [15] with the goal of tracking information cascades over (sparse) social networks, in [16] for learning graphs from real stock quotes and in [17] for the analysis of intensive longitudinal data. A dynamic regret study for an online implementation of [15] is given in [18], but a thorough experimental analysis, possibly considering real data, is missing.

Here we focus on a time-varying version of SEM (TV-SEM) which considers data to arrive in an online fashion and without exogenous inputs. While these inputs are of importance in many applications, their role is mainly used to identify the directionality of an edge [19]; in addition, in some scenarios, it is expensive or even impossible to acquire such inputs, as pointed out in [16]. Thus, for inferring only the pairwise relationships between data (i.e., identifying an undirected graph) we can ignore the latter when modeling the data evolution. In turn, this leads to a simpler model akin to a first-order graph signal diffusion [1].

To tackle time and data-constrained SEM scenarios, we propose an on-the-fly algorithm built upon recent advances in time-varying optimization [20, 21], only very recently applied to topology identification for a simple scenario [22]. The proposed approach updates the solution as samples come into the system and relies on light proximal operations [23]. The devised formulation is made adaptive by a simple update rule of empirical moments of the data, meaning that non-stationary topologies can be learned. The proposed algorithm is accompanied with a detailed theoretical analysis and then corroborated with both synthetic and real data.

2. PRELIMINARIES

Topology identification entails learning the structure of a network from data when the former is unavailable from the domain of interest. Formally, consider T data vectors $\{\mathbf{x}_t\}_{t=1}^T \in \mathbb{R}^N$ collected into the matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_T]$. Each data vector $\mathbf{x}_t$ has underlying dependencies between its elements, which can be represented with an unknown graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$, where $\mathcal{V} = \{1, \ldots, N\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ are the vertex and edge set, respectively, while $\mathbf{W}$ is the $N \times N$ (weighted) adjacency matrix [24] with entries $[\mathbf{W}]_{ij} \neq 0$ only if $(j, i) \in \mathcal{E}$, for $i \neq j$. Since at time t each node $i \in \mathcal{V}$ of the graph is related to the ith data value $x_t(i)$, we define as graph signal at time t the vector $\mathbf{x}_t = [x_t(1), \ldots, x_t(N)]^\top$ mapping the node set to the set of real numbers. With this terminology, topology identification aims to infer the *latent* underlying graph $\mathcal{G}$ from the available graph signals, which amounts to estimating the adjacency matrix $\mathbf{W}$.

Structural equation model. With the above formalism, neglecting possible external inputs, and assuming an undirected graph, SEM postulates a linear dependence of the signal value $x_t(i)$ with (some of) the signal values at other nodes $\{x_t(j)\}_{j \neq i}$, representing the *en-*

E-mails: {a.natali; e.isufi-1; m.a.coutinominguez; g.j.t.leus}@tudelft.nl;
This work was sponsored in part by Theory Lab, Central Research Institute, 2012 Labs, Huawei Technologies Co.,Ltd. Mario Coutino is partially supported by CONACYT.

dogenous variables, i.e.,:

$$x_t(i) = \sum_{j \neq i} W(i,j) x_t(j) + e_t(i), \quad t = 1, \ldots, T \tag{1}$$

where $W(i,j)$ weights the influence that node j exerts on node i and $e_t(i)$ captures the unmodeled dynamics. In this view, with $\mathbf{W}$ encoding the graph connectivity, model (1) proposes each node to be influenced by its one-hop neighbors only. In vector form, we can write (1) as:

$$\mathbf{x}_t = \mathbf{W}\mathbf{x}_t + \mathbf{e}_t, \quad t = 1, \ldots, T, \tag{2}$$

with $\mathbf{e}_t \sim \mathcal{N}(\mathbf{0}, \sigma_e^2 \mathbf{I}_N)$, for some noise variance $\sigma_e^2 > 0$. Then, inferring the graph topology under a SEM implies solving the optimization problem:

$$\begin{array}{ll} \underset{\mathbf{W}}{\text{minimize}} & \frac{1}{2T}\|\mathbf{X} - \mathbf{W}\mathbf{X}\|_F^2 + g(\mathbf{W}) \\ \text{subject to} & \mathbf{W} \in \mathcal{W} \end{array} \tag{3}$$

where $\mathcal{W} = \{\mathbf{W} | \operatorname{diag}(\mathbf{W}) = \mathbf{0}, W_{ij} = W_{ji}, i \neq j\}$, i.e., it is the set of hollow symmetric matrices. The first term in (3) imposes a good data-model fit, while the regularizer $g(\mathbf{W})$ enforces the estimated adjacency matrix $\mathbf{W}$ to have specific properties; e.g., sparsity. Note that problem (3) estimates a fixed topology for all realizations columns $\mathbf{x}_t$ of the matrix $\mathbf{X}$.

Dynamic setting. We consider graphs whose topologies change over time. Formally, a *time-varying network* is represented by the sequence of graphs $\{\mathcal{G}_t = (\mathcal{V}, \mathcal{E}_t, \mathbf{W}_t)\}_{t=1}^{\infty}$ with associated adjacency matrices $\{\mathbf{W}_t\}_{t=1}^{\infty}$. Alongside this, we assume that at every time instant t, the graph signal $\mathbf{x}_t$ is supported by the graph $\mathcal{G}_t$. To express the dependency of such time-varying signals with the underlying graph we consider time-varying SEM (TV-SEM) as detailed in the next section.

3. TIME-VARYING STRUCTURAL EQUATION MODEL

Consider again model (2) and define as $\mathbf{X}_t = [\mathbf{x}_1, \ldots, \mathbf{x}_t]$ the matrix collecting all the acquired graph signals up to time t. Solving a time-varying SEM (TV-SEM) problem with a sparsity-promoting regularizer means solving, for $t = 1, 2, \ldots$, the following composite optimization problem:

$$\mathbf{W}_t^{\star} := \underset{\mathbf{W}}{\operatorname{argmin}} \underbrace{\frac{1}{2t}\|\mathbf{X}_t - \mathbf{W}\mathbf{X}_t\|_F^2}_{f(\mathbf{W};t)} + \lambda \underbrace{\|\mathbf{W}\|_1 + \iota_{\mathcal{W}}(\mathbf{W})}_{g(\mathbf{W})} \tag{4}$$

where $\iota_{\mathcal{W}}(\cdot)$ is the indicator function for the convex set $\mathcal{W}$, by definition $\iota_{\mathcal{W}}(\mathbf{W}) = 0$ when $\mathbf{W} \in \mathcal{W}$ and $+\infty$ otherwise; $\lambda \geq 0$ is a parameter which controls the sparsity of the solution and $\|\mathbf{W}\|_1 = \|\operatorname{vec}(\mathbf{W})\|_1$ is the ℓ_1-norm of the vectorization of $\mathbf{W}$. Because $\|\mathbf{W}\|_F^2 = \operatorname{tr}(\mathbf{W}\mathbf{W}^\top)$, the function $f(\cdot)$ in (4) can be rewritten as:

$$f(\mathbf{W};t) = \frac{1}{2}[\operatorname{tr}(\mathbf{W}^2\hat{\mathbf{\Sigma}}_t) - 2\operatorname{tr}(\mathbf{W}\hat{\mathbf{\Sigma}}_t) + \operatorname{tr}(\hat{\mathbf{\Sigma}}_t)] \tag{5}$$

where $\hat{\mathbf{\Sigma}}_t := (1/t)\mathbf{X}_t\mathbf{X}_t^\top$ represents the empirical covariance matrix computed with the signals up to time t. However, instead of using the vanilla update rule for $\hat{\mathbf{\Sigma}}_t$, we make it robust for non-stationary environments by down-weighting the contribution of past data. This is achieved with a weighted moving average of the form $\hat{\mathbf{\Sigma}}_t = \gamma\hat{\mathbf{\Sigma}}_{t-1} + (1-\gamma)\mathbf{x}_t\mathbf{x}_t^\top$, with forgetting factor $\gamma \in (0,1)$. Notice that [cf. (5)] matrix $\hat{\mathbf{\Sigma}}_t$ represents a sufficient statistic of the model: we do not explicitly need to store all the raw observed data, resulting in light storage system requirements and a natural recursive update.

Reduction. Notice that problem (4) is a constrained optimization problem since matrix $\mathbf{W}$ is required to be in the set $\mathcal{W}$. Being symmetric and hollow, the number of independent variables in $\mathbf{W}$ is $l = N(N-1)/2$, thus a lifting of spaces is suggested in order to avoid errors for the gradient computation. For this, we operate on the hollow half-vectorization space (henceforth hh-space) to make the problem unconstrained and reduce the number of independent variables from N^2 to l. Specifically, we represent the matrix $\mathbf{W}$ in its hollow half-vectorization form, i.e., $\mathbf{w} = \operatorname{vechh}(\mathbf{W}) \in \mathbb{R}^l$. This reduction is achieved by using the hollow elimination matrix $\mathbf{E}_h \in \mathbb{R}^{l \times N^2}$ which extracts the variables of the strictly lower (equivalently, upper) triangular part of the matrix, i.e., $\mathbf{w} = \mathbf{E}_h \operatorname{vec}(\mathbf{W})$. Likewise we consider the inverse operation through the hollow duplication matrix $\mathbf{D}_h \in \mathbb{R}^{N^2 \times l}$, which duplicates the values of $\mathbf{w}$ and fills in zeros in the correct positions, i.e., $\operatorname{vec}(\mathbf{W}) = \mathbf{D}_h\mathbf{w}$.

With this notation in place, problem (4) can be written in the hh-space as:

$$\mathbf{w}_t^{\star} := \underset{\mathbf{w}}{\operatorname{argmin}} \underbrace{\frac{1}{2}\mathbf{w}^\top\mathbf{Q}_t\mathbf{w} - 2\mathbf{w}^\top\boldsymbol{\sigma}_t + \frac{1}{2}\sigma_t}_{f(\mathbf{w};t)} + \lambda \underbrace{\|\mathbf{w}\|_1}_{g(\mathbf{w})} \tag{6}$$

where $\mathbf{Q}_t := \mathbf{D}_h^\top(\hat{\mathbf{\Sigma}}_t \otimes \mathbf{I})\mathbf{D}_h$ with $\otimes$ denoting the Kronecker product, $\boldsymbol{\sigma}_t = \operatorname{vechh}(\hat{\mathbf{\Sigma}}_t)$, and $\sigma_t = \operatorname{tr}(\hat{\mathbf{\Sigma}}_t)$. Because $\mathbf{Q}_t \succeq 0$, i.e., it is positive semidefinite, problem (6) is convex. Notice how operating in the hh-space also avoids handling explicitly the symmetry constraint in the optimization.

Solving exactly problem (6) for each t may not be feasible, especially in real-time (streaming) applications. Moreover, the exact solution may not even be needed due to the dynamic environment, in which rather a faster estimate may be preferred. Motivated by this rationale, we propose an online lightweight proximal-based implementation to track the time-varying solution of problem (4), further speeded-up through a *prediction-correction* strategy, which relies on recent advances in time-varying optimization [20], [21].

4. ONLINE IMPLEMENTATION

To solve problem (6) in a time-varying optimization framework [20], first notice that function $f(\cdot)$ is smooth strongly convex and function $g(\cdot)$ is closed, convex and proper. Denote then the composite objective function of problem (6) with $F(\mathbf{w};t) := f(\mathbf{w};t) + \lambda g(\mathbf{w};t)$. The following two-step prediction-correction approach is now put forth.

1) Prediction. At time t, an approximation $\hat{F}(\mathbf{w};t+1)$ of the (yet unobserved) function $F(\mathbf{w};t+1)$ is formed by exploiting only information available up to t. Particularly, from the last graph estimate $\hat{\mathbf{w}}_t$, we employ a Taylor expansion based prediction for function $f(\cdot)$ and the one-step back prediction for $g(\cdot)$ as approximation schemes:

$$\begin{aligned} \hat{f}(\mathbf{w};t+1) = \frac{1}{2}\mathbf{w}^\top\nabla_{\mathbf{ww}}f(\hat{\mathbf{w}}_t;t)\mathbf{w} + \mathbf{w}^\top(\nabla_{\mathbf{w}}f(\hat{\mathbf{w}}_t;t) + \\ - \nabla_{\mathbf{ww}}f(\hat{\mathbf{w}}_t;t)\hat{\mathbf{w}}_t + \nabla_{t\mathbf{w}}f(\hat{\mathbf{w}}_t;t)) \end{aligned} \tag{7a}$$

$$\hat{g}(\mathbf{w};t+1) = g(\mathbf{w};t) \tag{7b}$$

where $\nabla_{\mathbf{w}}f(\cdot)$ and $\nabla_{\mathbf{ww}}f(\cdot)$ are the gradient and the Hessian of $f(\cdot)$ w.r.t. $\mathbf{w}$, respectively, and $\nabla_{t\mathbf{w}}f(\cdot)$ is the time-derivative of the

gradient $\nabla_{\mathbf{w}} f(\cdot)$. The approximated composite function $\hat{F}(\mathbf{w}; t+1)$ is then used to *predict* the topology at time $t+1$ using only the information available up to time t. In other words, the predicted topology is obtained as the solution of the optimization problem:

$$\mathbf{w}^{\star}_{t+1|t} := \underset{\mathbf{w}}{\operatorname{argmin}} \hat{f}(\mathbf{w}; t+1) + \lambda \hat{g}(\mathbf{w}) \tag{8}$$

which is expected to be close to the true topology $\mathbf{w}^{\star}_{t+1}$ at time $t+1$.

Instead of solving problem (8) exactly, which may incur in high computational efforts, and whose solution will need to be updated at the next time step, we find an estimate $\hat{\mathbf{w}}_{t+1|t}$ of $\mathbf{w}^{\star}_{t+1|t}$ through P proximal gradient descent iterations. Specifically, initialize the solution as $\hat{\mathbf{w}}^0 = \hat{\mathbf{w}}_t$ and compute:

$$\hat{\mathbf{w}}^{p+1} = \operatorname{prox}_{\lambda \hat{g}, \alpha_t} \left(\hat{\mathbf{w}}^p - \alpha_t \nabla_{\mathbf{w}} \hat{f}(\hat{\mathbf{w}}^p) \right) \tag{9}$$

for $p = 0, \ldots, P-1$, where we have defined as the proximal operator of function $\lambda \hat{g}(\cdot)$ with penalty parameter (step size) $\alpha_t > 0$, evaluated at some point $\mathbf{v} \in \mathbb{R}^N$, the solution of the problem:

$$\operatorname{prox}_{\lambda g, \alpha_t}(\mathbf{v}) := \underset{\mathbf{z}}{\operatorname{argmin}} \{ \lambda g(\mathbf{z}) + \frac{1}{2\alpha_t} \|\mathbf{z} - \mathbf{v}\|_2^2 \} \tag{10}$$

By defining as $[\,\cdot\,]_+ := \max(\mathbf{0}, \cdot)$, where the maximum operates in an entry-wise fashion and with $\odot$ the element-wise product between two vectors, each iteration of (9) boils down to the two steps:

$$\begin{aligned} \hat{\mathbf{u}}^p &= \hat{\mathbf{w}}^p - \alpha_t [\nabla_{\mathbf{w}} f(\hat{\mathbf{w}}_t; t) + \\ &\quad + \nabla_{\mathbf{ww}} f(\hat{\mathbf{w}}_t; t)(\hat{\mathbf{w}}^p - \hat{\mathbf{w}}_t) + \nabla_{t\mathbf{w}} f(\hat{\mathbf{w}}_t; t)] \end{aligned} \tag{11a}$$

$$\hat{\mathbf{w}}^{p+1} = \operatorname{sign}(\hat{\mathbf{u}}^p) \odot [|\hat{\mathbf{u}}^p| - 2\boldsymbol{\lambda}\boldsymbol{\alpha}_t]_+ \tag{11b}$$

with $\boldsymbol{\lambda} = \lambda \mathbf{1}$ and $\boldsymbol{\alpha}_t = \alpha_t \mathbf{1}$. In other words, equation (11b) is the non-negative soft-thresholding operator which sets to zero all the edge weights of the graph obtained after the (prediction) gradient descent operation in (11a). The prediction $\hat{\mathbf{w}}_{t+1|t}$ is set to $\hat{\mathbf{w}}_{t+1|t} = \hat{\mathbf{w}}^P$, which serves a warm start for the correction step.

2) Correction. At time $t+1$, a new datum $\mathbf{x}_{t+1}$ arrives and the function $F(\mathbf{w}; t+1)$ (depending on the updated covariance matrix $\hat{\boldsymbol{\Sigma}}_{t+1}$) becomes available. But instead of solving exactly problem (6) at time $t+1$, we again estimate an approximate solution $\hat{\mathbf{w}}_{t+1}$ by applying C proximal gradient descent iterations of $F(\cdot)$, i.e.,:

$$\hat{\mathbf{w}}^{c+1} = \operatorname{prox}_{\lambda g, \beta_t} \left(\hat{\mathbf{w}}^c - \beta_t \nabla_{\mathbf{w}} f(\hat{\mathbf{w}}^c) \right) \tag{12}$$

for $c = 0, 1, \ldots, C-1$, where $\beta_t > 0$ is the step-size. We initialize the graph with the predicted topology $\hat{\mathbf{w}}^0 = \hat{\mathbf{w}}_{t+1|t}$. Likewise (9), we implement recursion (12) in two steps:

$$\hat{\mathbf{u}}^c = \hat{\mathbf{w}}^c - \beta_t \nabla_{\mathbf{w}} f(\hat{\mathbf{w}}^c; t+1) \tag{13a}$$

$$\hat{\mathbf{w}}^{c+1} = \operatorname{sign}(\hat{\mathbf{u}}^c) \odot [|\hat{\mathbf{u}}^c| - 2\boldsymbol{\lambda}\boldsymbol{\beta}_t]_+ \tag{13b}$$

where $\boldsymbol{\beta}_t = \beta_t \mathbf{1}$. Once the C steps are performed, the corrected graph $\hat{\mathbf{w}}_{t+1}$ is set to $\hat{\mathbf{w}}_{t+1} = \hat{\mathbf{w}}^C$, which will approximate the solution $\mathbf{w}^{\star}_{t+1}$ of (4). The online TV-SEM algorithm is summarized in Algorithm 1.

Implementation. The derived gradient and Hessian of function $f(\cdot)$ in the hh-space, as well as the time derivative of the gradient $\nabla_{\mathbf{w}} f(\cdot)$[1] can be expressed as:

$$\nabla_{\mathbf{w}} f(\mathbf{w}; t) = \mathbf{Q}_t \mathbf{w} - 2\boldsymbol{\sigma}_t; \quad \nabla_{\mathbf{ww}} f(\mathbf{w}; t) = \mathbf{Q}_t; \tag{14a}$$

$$\nabla_{t\mathbf{w}} f(\mathbf{w}; t) = (\mathbf{Q}_t - \mathbf{Q}_{t-1})\mathbf{w} - 2(\boldsymbol{\sigma}_t - \boldsymbol{\sigma}_{t-1}) \tag{14b}$$

[1]The time derivative of the gradient is given by the partial mixed order derivative [25].

Algorithm 1 Online TV-SEM

Require: Feasible $\hat{\mathbf{w}}_0$, $f(\mathbf{w}; t_0)$, $g(\mathbf{w}; t_0)$ P, C

for $t = 0, 1, \ldots$ **do**
  // Prediction - time t
  Initialize $\hat{\mathbf{w}}^0 = \hat{\mathbf{w}}_t$
  Compute α_t, $\nabla_{\mathbf{w}} f(\hat{\mathbf{w}}_t; t)$, $\nabla_{\mathbf{ww}} f(\hat{\mathbf{w}}_t; t)$, $\nabla_{t\mathbf{w}} f(\hat{\mathbf{w}}_t; t)$
  for $p = 0, 1, \ldots, P-1$ **do**
    Compute $\hat{\mathbf{w}}^{p+1}$ as in (11a)(11b)
  end for
  Set the predicted variable $\hat{\mathbf{w}}_{t+1|t} = \hat{\mathbf{w}}^P$
  // Correction - time $t+1$: new data arrives
  Initialize the corrected variable $\hat{\mathbf{w}}^0 = \hat{\mathbf{w}}_{t+1|t}$
  Compute β_t
  for $c = 0, 1, \ldots, C-1$ **do**
    Compute $\hat{\mathbf{w}}^{c+1}$ as in (13a)(13b)
  end for
  Set the corrected variable $\hat{\mathbf{w}}_{t+1} = \hat{\mathbf{w}}^C$
end for

As final comment, the step-sizes α_t and β_t are chosen in the interval $(0, 2/L_t]$, where L_t is the Lipschitz constant of $\nabla_{\mathbf{w}} f(\cdot)$ at time t. This enables us to ensure the algorithm's convergence and bound the estimation error at each iteration (see Section 5).

5. CONVERGENCE ANALYSIS

We now establish convergence results related to Algorithm 1 by deriving bounds for the error sequence $\{\|\hat{\mathbf{w}}_t - \mathbf{w}^{\star}_t\|_2, t = 1, 2, \ldots\}$, where we recall that $\mathbf{w}^{\star}_t$ is the unique minimizer of problem (6). First, notice that for a fixed t, the TV-SEM function $f(\cdot; t)$ in (6) (and hence its approximated version $\hat{f}(\cdot)$ in (7a)) is m_t-strongly convex and L_t-smooth for:

$$m_t \leq \lambda_{\min}(\hat{\boldsymbol{\Sigma}}_t) \qquad L_t \geq 2\lambda_{\max}(\hat{\boldsymbol{\Sigma}}_t) \tag{15}$$

Next, consider the temporal variability of the optimal solution $\|\mathbf{w}^{\star}_{t+1} - \mathbf{w}^{\star}_t\|$ and the prediction error $\|\mathbf{w}^{\star}_{t+1|t} - \mathbf{w}^{\star}_{t+1}\|$. Then, we have the following non-asymptotic performance guarantee for Algorithm 1, which is an adaptation of [21, Proposition 5.1].

Proposition 1. Consider two scalar bounds $\{d_t, \phi_t\} \in \mathbb{R}_+$ such that:

$$\|\mathbf{w}^{\star}_{t+1} - \mathbf{w}^{\star}_t\| \leq d_t \quad \text{and} \quad \|\mathbf{w}^{\star}_{t+1|t} - \mathbf{w}^{\star}_{t+1}\| \leq \phi_t \tag{16}$$

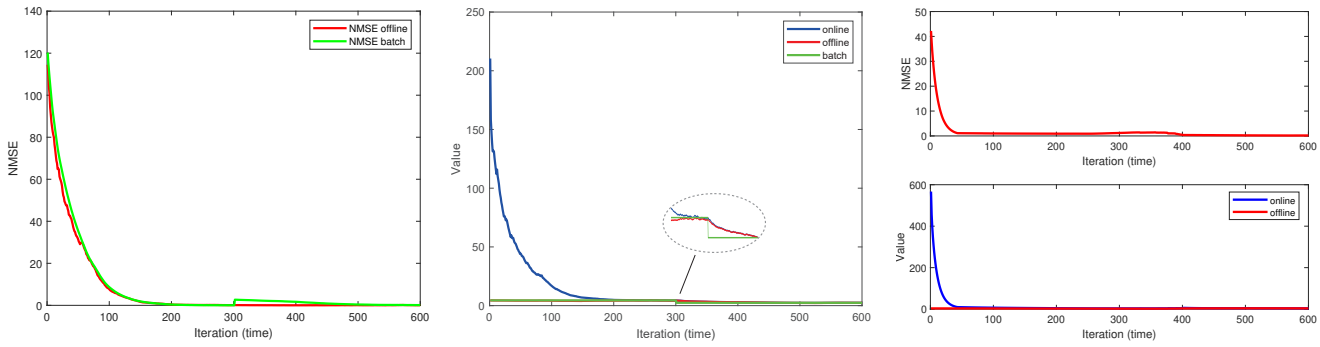
for any $t \in \mathbb{N}_+$. Moreover, assume that the prediction and correction steps use the same step-size $\rho_t = \alpha_t = \beta_t$. Then, by employing P prediction steps in (9) and C correction steps in (12), the sequence of iterates $\{\hat{\mathbf{w}}_t\}$ generated by Algorithm 1 satisfies:

$$\|\hat{\mathbf{w}}_{t+1} - \mathbf{w}^{\star}_{t+1}\|_2 \leq q_t^C (q_t^P \|\hat{\mathbf{w}}_t - \mathbf{w}^{\star}_t\| + q_t^P d_t + (1 + q_t^P)\phi_t)) \tag{17}$$

where $q_t \in (0, 1)$ is the contraction coefficient, which for the proximal gradient method is $q_t = \max\{|1 - \rho_t m_t|, |1 - \rho_t L_t|\}$ [23].

Proof. Follows from [21, Proposition 5.1] and [21, Lemma 2.5], with variables $\lambda = q_t$ and $\chi = \beta = 1$. ∎

Therefore (17) is a contraction (i.e. $q_t^{C+P} < 1$) when $\rho_t < 2/L_t$, which is the reason why we pick $\{\alpha_t, \beta_t\} < 2/\lambda_{\max}(\hat{\boldsymbol{\Sigma}}_t)$ at every iteration t. In turn, this ensures a slightly modified notion of Q-linear convergence for the error sequence $\{\|\hat{\mathbf{w}}_t - \mathbf{w}^{\star}_t\|_2\}$ generated



(a) (b) (c)

Fig. 1: Normalized mean squared error (NMSE) for the synthetic scenarios: (a) NMSE between our online solution $\hat{\mathbf{w}}_t$ with respect to the offline solution $\mathbf{w}_t^\star$ (red) and the batch solution $\mathbf{w}_B^\star$ (green); (b) function value attained by the online $\{\hat{\mathbf{w}}_t\}$ (blue), offline (red) and batch (green) solutions; (c) Smooth scenario, NMSE between the online solution $\hat{\mathbf{w}}_t$ and offline solution $\mathbf{w}_t^\star$ (top) and the associated function value (bottom).

by Algorithm 1 [26]. Notice that choosing an equal step-size ρ_t for the prediction and correction steps is not a limitation, since both the functions $f(\cdot)$ and $\hat{f}(\cdot)$ share the same smooth and strong convexity constants (15).

The bound can then be generalized by taking a constant step size $\rho < 2/\sup_{t\in\mathbb{N}}\{L_t\}$. Let $q := \sup_{t\in\mathbb{N}}\{q_t\}$, $d := \sup_{t\in\mathbb{N}}\{d_t\}$ and $\phi := \sup_{t\in\mathbb{N}}\{\phi_t\}$. Then, an upper bound for the asymptotic error can be established as follows:

$$\limsup_{t\to\infty} \|\hat{\mathbf{w}}_t - \mathbf{w}_t^\star\|_2 = \frac{q^C}{1-q^{C+P}}(q^P d + (1+q^P\phi)) \qquad (18)$$

That is, the output of Algorithm 1 hovers (asymptotically) around the optimal trajectory $\{\mathbf{w}_t^\star\}_{t\geq T_0}$, for some $T_0 \in \mathbb{N}_+$. This distance is bounded by a number depending on the maximum drifting and the maximum prediction error incurred during the algorithm's iterations. In addition, it is important to notice how the initial condition, i.e., the starting condition, does not influence the asymptotic error.

6. NUMERICAL RESULTS

In this section, we demonstrate the capability of Algorithm 1 to closely track the offline counterpart and we show several numerical results obtained by performing several experiments on synthetic and real datasets. For all the experiments carried out, we initialize the empirical covariance matrix $\hat{\mathbf{\Sigma}}_0$ with some samples acquired prior to the analysis, which are then discarded. We fix $P = 1$ prediction steps and $C = 1$ correction steps, i.e., a streaming scenario. The algorithm per-se is however capable of handling different values of P and C, depending on the environment and processing characteristics.

As performance metric, to establish the convergence of Algorithm 1 to the offline counterpart solving (6), we consider the normalized mean squared error (NMSE) between our algorithm's estimate $\hat{\mathbf{w}}_t$ and the optimal (offline) solution $\mathbf{w}_t^\star$[2], computed at iteration t, i.e.,:

$$\text{NMSE}(\mathbf{w}_t^\star, \hat{\mathbf{w}}_t) = \frac{\|\hat{\mathbf{w}}_t - \mathbf{w}_t^\star\|_2^2}{\|\mathbf{w}_t^\star\|_2^2} \qquad (19)$$

Notice that this is a more robust metric than the cost function value attained by $\mathbf{w}_t^\star$ and $\hat{\mathbf{w}}_t$, since there might be solutions "far-away" in space that yield the same function value. For completeness, we show both these metrics. For the real dataset, we consider also other metrics helping us in explaining better the graph behavior.

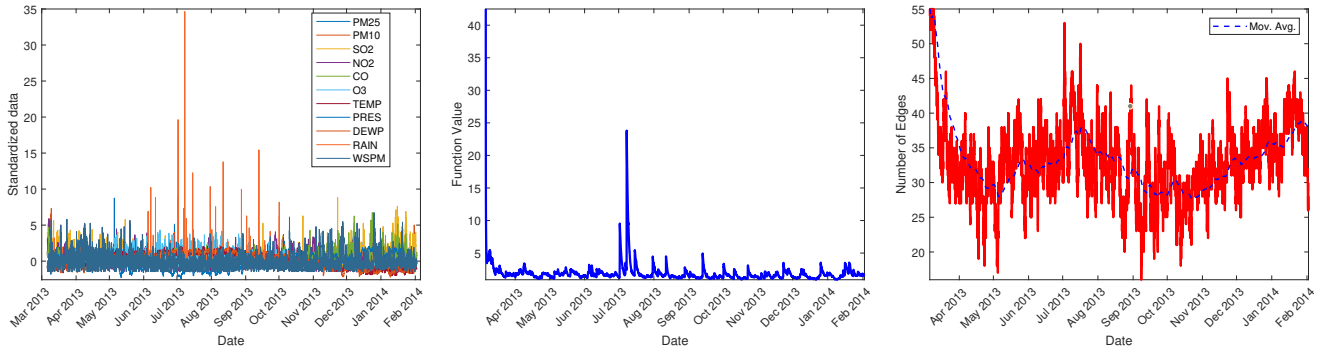
[2]We use CVX [27] as solver for the offline computations of the experiments.

6.1. Tests on synthetic data

To validate the proposed online algorithm, we test it on synthetic data generated under two different scenarios: *abrupt* and *smooth* temporal edge variations. In the first scenario, we generate a random sensor-network of $N = 28$ nodes with a piece-wise constant topology. In particular, the initial graph undergoes a perturbation on $\lceil N/2 \rceil$ nodes of the graph, whose edges undergo a dilation increasing their weights with 50%, after $t = 300$ samples. Then, for $t = \{1, \dots, 600\}$, we generate each graph signal $\mathbf{x}_t$ according to (2), i.e., $\mathbf{x}_t = (\mathbf{I} - \mathbf{W}_t)^{-1}\mathbf{e}_t$, with a noise standard deviation equal to $\sigma_e = 0.5$.

Fig. 1a shows, for $\lambda = 0.05$ and $\gamma = 0.97$, the NMSE of the online solution $\hat{\mathbf{w}}_t$ with respect to: the optimal (offline) solution $\mathbf{w}_t^\star$ (in red) and the optimal batch solutions of the two stationary intervals (in green). The results clearly indicate the convergence of our online algorithm to the optimal offline counterpart after few iterations, and its capability to closely track it. In addition, both solutions converge to the batch solutions of the two stationary intervals. Indeed, after the sudden topology perturbation at $t = 300$, the NMSE with respect to the stationary trajectory increases and then converges again (green), while with respect to the offline solution it remains close to zero (red). This is also visible in Fig. 1b, where the function value attained by the three different solutions over time is shown. Another important metric to consider in time-sensitive applications is the required time per iteration. For the above experiment, considering $N = 15$ and $T = 800$ time instants, each iteration (prediction-correction step) of Algorithm 1 required $0,0017$s ($1,39$s the entire run) in contrast to the $0,4324$s per iteration ($345,94$s the entire run) of the offline (CVX) counterpart solution.

In the second (smooth) scenario, we consider the support of the network generated in the first scenario; then, the edge-evolution pattern follows a sinusoidal behavior, i.e., $S_{ij}(t) = 1 + \sin(2\pi f_s t)$ for $t = \{1, \dots, T\}$ and $f_s = 1/T$, where $T = 600$. Then, we generate again the data $\{\mathbf{x}_t\}$ according to (2). Under this setting, the NMSE between our online solution $\hat{\mathbf{w}}_t$ and the offline counterpart $\mathbf{w}_t^\star$ is shown in Fig. 1c, together with the cost function value. Also in this case we can see how the online solution is able to track the offline one.



(a) Standardized Data (b) Function Value (c) Number of edges

Fig. 2: The figure shows: (a) the standardized hourly time series from March 2013 to February 2014; (b) the objective function value attained by the inferred topology for each date; and (c) the number of edges of the inferred topology for each date (with its moving average in dotted blue).

These examples serve as a sanity check of the algorithm, which is now applied to a real application.

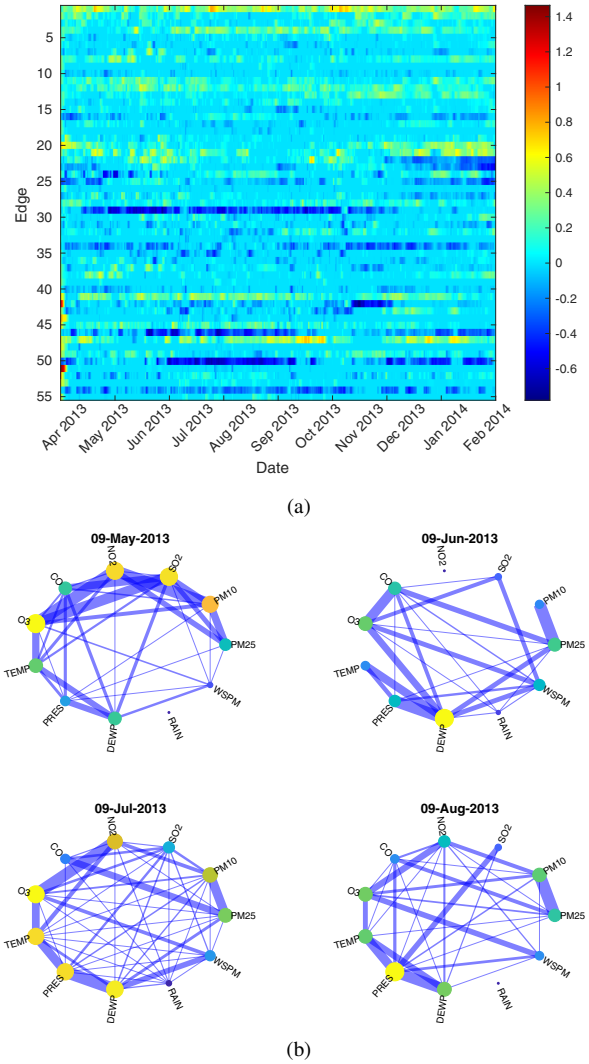
6.2. Tests on Real Data

We now test the proposed algorithm on real data and show the resulting inferred topologies.

Dataset Description: To perform an analysis on real world networks, we consider the publicly available Beijing Multi-Site Air-Quality Data Set [28, 29], which includes hourly measurements from 12 nationally-controlled air-quality monitoring sites in Beijing for the period March 1st, 2013 to February 28th, 2017. Each monitoring site considers: the concentration (μg/m^3) of 6 main air pollutants (PM2.5, PM10, SO2, NO2, CO, O3) and 6 relevant meteorological variables (temperature in $^\circ$C, pressure in hPa, dew point temperature in $^\circ$C, precipitation in mm, wind direction, wind speed in m/s). For the experiments, we analyze all the six pollutants and five meteorological variables (all but wind direction) for the monitoring site of Nongzhanguan, since it contains the lowest number of missing data among the twelve sites. The missing values were substituted with a linear interpolation of their neighboring, non-missing values.

Results. We consider $T = 8000$ hourly measurements (March 2013 to February 2014) as graph signals $\{\mathbf{x}_t\}$ for the eleven quantities of interest, which are further standardized, i.e., each variable is centered and divided for its empirical standard deviation; see Fig. 2a for a plot of the standardized time series. Notice that since RAIN is a sparse variable, its standardization leads to a behavior akin to a change-detection scenario. Based on our previous experiments on the piece-wise constant scenario, we expect to find structural differences during the learning process.

We run Algorithm 1 for different values of the regularization parameter λ and forgetting factor γ. Then, we monitor the evolution over time of some graph properties: among these, the cost function value and the number of edges of the inferred graph. This enables us, by simple inspection, to assess whether there are notable patterns which are consistent with observable phenomena in the data. The values $\lambda = 0.05$ and $\gamma = 0.97$ yielded results most consistent with the data behavior: empirically, a value of λ higher than 1 made the graph almost completely sparse. Notice also how the covariance-based formulation (5) (with the relative update rule) avoids the least squares term in (4) to grow indefinitely, enabling us to confidently select a constant λ over time.



(a)

(b)

Fig. 3: Estimated time-varying graph topologies: (a) Evolution of the edge weights over time, with brighter (darker) colors indicating a positive (negative) weight; (b) Inferred topologies from four different dates of interest.

Fig. 2b shows the function value attained by the inferred topologies $\{\hat{\mathbf{w}}_t\}$ over the considered dates. Predictably, the effects of the variable RAIN around the period of July 2013 are reflected in the cost function value, which exhibits some sharp peaks. This is also evident in Fig. 2c which shows the number of edges of the inferred topology $\hat{\mathbf{w}}_t$ for each date t (where also its moving average with a window size of 30 days is shown - dotted blue - to ease the visualization of the value dynamics). The fact that the cost function does not have smooth peaks indicates that the algorithm can handle to abrupt changes and can immediately go back to its "normal" state (in this case a function value lower than 5). This is visible at the initialization of the algorithm and in the period of July.

A more informative description of the learning behavior of the algorithm over time is given in Fig. 3a, which shows the evolution of the edge weights over time[3], with brighter (darker) colors indicating a positive (negative) weight. Here we can see the sparsity pattern of the graph over the considered temporal horizon, and the importance of the weights at each date. As previously observed, the heavy rainfall happening during the summer period leads to an increasing value of the edge weights. To really enjoy the visualization potential offered by graphs as a tool, we show in Fig. 3b the graphs inferred on the 9th day of May, June, July, and August. Wider edges indicate a larger (absolute) value for their weights, i.e. a higher linear dependence between the variable of interest; the size of a node (also brighter color for bigger sizes) indicates how connected that node is with the rest of the nodes. Notable is the variable RAIN which is disconnected in May and August, while highly connected in July consistent with the previous observations.

From a technical point of view, according to our algorithm, these are the graphs that better explain an instantaneous linear dependence between the observed variables. Are these graphs meaningful representations from a geophysical and meteorological point of view? Unfortunately, our knowledge, does not allow us to answer this question, which requests the feedback of a domain expert.

7. CONCLUSION

In this work, we presented an adaptive algorithm for tracking structural equations models (SEMs) from online data. The proposed approach, which relies on proximal gradient iterations, is further accelerated through a prediction-correction strategy. The strategy builds upon recent advances in time-varying optimization and is accompanied by theoretical performance guarantees to track the optimal time-varying solutions. Numerical experiments on synthetic and real data validate the proposed on-the-fly formulation. Possible algorithmic enhancements include the introduction of an adaptive forgetting factor γ able to automatically adjust itself by recognizing substantial changes of the data distribution (thus discarding old information) as well as a distributed implementation scalable to very large graphs.

[3]We remind that the number of total edges in an undirected graph of N nodes is $N(N-1)/2$.

8. REFERENCES

[1] Gonzalo Mateos, Santiago Segarra, Antonio G Marques, and Alejandro Ribeiro, "Connecting the dots: Identifying network structure via graph signal processing," *IEEE Signal Processing Magazine*, vol. 36, no. 3, pp. 16–43, 2019.

[2] Xiaowen Dong, Dorina Thanou, Michael Rabbat, and Pascal Frossard, "Learning graphs from data: A signal representation perspective," *IEEE Signal Processing Magazine*, vol. 36, no. 3, pp. 44–63, 2019.

[3] Georgios B Giannakis, Yanning Shen, and Georgios Vasileios Karanikolas, "Topology identification and learning over graphs: Accounting for nonlinearities and dynamics," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 787–807, 2018.

[4] S. Vlaski, H. P. Mareti, R. Nassif, P. Frossard, and A. H. Sayed, "Online graph learning from sequential data," in *2018 IEEE Data Science Workshop (DSW)*, 2018, pp. 190–194.

[5] Rasoul Shafipour and Gonzalo Mateos, "Online topology inference from streaming stationary graph signals with partial connectivity information," *Algorithms*, vol. 13, no. 9, 2020.

[6] Mircea Moscu, Ricardo Borsoi, and Cédric Richard, "Online kernel-based graph topology identification with partial-derivative-imposed sparsity," in *2020 28th European Signal Processing Conference (EUSIPCO)*. IEEE, 2021, pp. 2190–2194.

[7] B. Zaman, L. M. L. Ramos, D. Romero, and B. Beferull-Lozano, "Online topology identification from vector autoregressive time series," *IEEE Transactions on Signal Processing*, vol. 69, pp. 210–225, 2021.

[8] Seyed Saman Saboksayr, Gonzalo Mateos, and Mujdat Cetin, "Online discriminative graph learning from multi-class smooth signals," *arXiv preprint arXiv:2101.00184*, 2021.

[9] Jodie B Ullman and Peter M Bentler, "Structural equation modeling," *Handbook of psychology*, pp. 607–634, 2003.

[10] Arthur S Goldberger, "Structural equation methods in the social sciences," *Econometrica: Journal of the Econometric Society*, pp. 979–1001, 1972.

[11] Robert C MacCallum and James T Austin, "Applications of structural equation modeling in psychological research," *Annual review of psychology*, vol. 51, no. 1, pp. 201–226, 2000.

[12] X Cai, JA Bazerque, and GB Giannakis, "Sparse structural equation modeling for inference of gene regulatory networks exploiting genetic perturbations," *PLoS, Computational Biology*, vol. 9, no. 5, 2013.

[13] Judea Pearl, *Causality*, Cambridge university press, 2009.

[14] Norman Cliff, "Some cautions concerning the application of causal modeling methods," *Multivariate behavioral research*, vol. 18, no. 1, pp. 115–126, 1983.

[15] B. Baingana and G. B. Giannakis, "Tracking switched dynamic network topologies from information cascades," *IEEE Transactions on Signal Processing*, vol. 65, no. 4, pp. 985–997, 2017.

[16] Yanning Shen, Brian Baingana, and Georgios B Giannakis, "Tensor decompositions for identifying directed graph topologies and tracking dynamic networks," *IEEE Transactions on Signal Processing*, vol. 65, no. 14, pp. 3675–3687, 2017.

[17] Ellen L Hamaker, Tihomir Asparouhov, Annette Brose, Florian Schmiedek, and Bengt Muthén, "At the frontiers of modeling intensive longitudinal data: Dynamic structural equation models for the affective measurements from the cogito study," *Multivariate Behavioral Research*, vol. 53, no. 6, pp. 820–841, 2018.

[18] Bakht Zaman, Luis Miguel Lopez Ramos, and Baltasar Beferull-Lozano, "Dynamic regret analysis for online tracking of time-varying structural equation model topologies," in *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. IEEE, 2020, pp. 939–944.

[19] Juan Andrés Bazerque, Brian Baingana, and Georgios B Giannakis, "Identifiability of sparse structural equation models for directed and cyclic networks," in *2013 IEEE Global Conference on Signal and Information Processing*. IEEE, 2013, pp. 839–842.

[20] Andrea Simonetto, Emiliano Dall'Anese, Santiago Paternain, Geert Leus, and Georgios B Giannakis, "Time-varying convex optimization: Time-structured algorithms and applications," *Proceedings of the IEEE*, 2020.

[21] Nicola Bastianello, Andrea Simonetto, and Ruggero Carli, "Primal and dual prediction-correction methods for time-varying convex optimization," 2020.

[22] Alberto Natali, Mario Coutino, Elvin Isufi, and Geert Leus, "Online time-varying topology identification via prediction-correction algorithms," *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), arXiv preprint arXiv:2010.11634*, 2021.

[23] Amir Beck, *First-order methods in optimization*, SIAM, 2017.

[24] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 61, no. 7, pp. 1644–1656, April 2013.

[25] Andrea Simonetto, Aryan Mokhtari, Alec Koppel, Geert Leus, and Alejandro Ribeiro, "A class of prediction-correction methods for time-varying convex optimization," *IEEE Transactions on Signal Processing*, vol. 64, no. 17, pp. 4576–4591, 2016.

[26] Andrea Simonetto and Emiliano DallAnese, "Prediction-correction algorithms for time-varying constrained optimization," *IEEE Transactions on Signal Processing*, vol. 65, no. 20, pp. 5481–5494, 2017.

[27] Michael Grant, Stephen Boyd, and Yinyu Ye, "Cvx: Matlab software for disciplined convex programming," 2008.

[28] Shuyi Zhang, Bin Guo, Anlan Dong, Jing He, Ziping Xu, and Song Xi Chen, "Cautionary tales on air-quality improvement in beijing," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 473, no. 2205, pp. 20170457, 2017.

[29] Dheeru Dua and Casey Graff, "UCI machine learning repository," 2017.