Proximal-Gradient Algorithms for Tracking Cascades Over Social Networks

Brian Baingana, *Student Member, IEEE*, Gonzalo Mateos, *Member, IEEE*, and Georgios B. Giannakis, *Fellow, IEEE*

***Abstract*—Many real-world processes evolve in cascades over complex networks, whose topologies are often unobservable and change over time. However, the so-termed adoption times when blogs mention popular news items, individuals in a community catch an infectious disease, or consumers adopt a trendy electronics product are typically known, and are implicitly dependent on the underlying network. To infer the network topology, a *dynamic* structural equation model is adopted to capture the relationship between observed adoption times and the unknown edge weights. Assuming a slowly time-varying topology and leveraging the sparse connectivity inherent to social networks, edge weights are estimated by minimizing a sparsity-regularized exponentially-weighted least-squares criterion. To this end, solvers with complementary strengths are developed by leveraging (pseudo) real-time sparsity-promoting proximal gradient iterations, the improved convergence rate of accelerated variants, or reduced computational complexity of stochastic gradient descent. Numerical tests with both synthetic and real data demonstrate the effectiveness of the novel algorithms in unveiling sparse dynamically-evolving topologies, while accounting for external influences in the adoption times. Key events in the political leadership in North Korea and the initial public offering of *LinkedIn* explain connectivity changes observed in the associated networks inferred from global cascades of online media.**

***Index Terms*—Structural equation model, dynamic network, social network, contagion, sparsity.**

I. Introduction

NETWORKS arising in natural and man-made settings provide the backbone for the propagation of *contagions* such as the spread of popular news stories, the adoption of buying trends among consumers, and the spread of infectious diseases [11], [36]. For example, a terrorist attack may be reported within minutes on mainstream news websites. An information cascade emerges because these websites' readership typically includes bloggers who write about the attack as well, influencing their own readers in turn to do the same. Although the times when "nodes" get infected are often observable, the underlying network topologies over which cascades propagate are typically unknown and dynamic. Knowledge of the topology plays a crucial role for several reasons e.g., when social media advertisers select a small set of initiators so that an online campaign can go viral, or when healthcare initiatives wish to infer hidden needle-sharing networks of injecting drug users. As a general principle, network structural information can be used to predict the behavior of complex systems [16], such as the evolution and spread of information pathways in online media underlying e.g., major social movements and uprisings due to political conflicts [35].

Inference of networks using temporal traces of infection events has recently become an active area of research. According to the taxonomy in [16, Ch. 7], this can be viewed as a problem involving inference of *association* networks. Two other broad classes of network topology identification problems entail (individual) link prediction, or, tomographic inference. Several prior approaches postulate probabilistic models and rely on maximum likelihood estimation (MLE) to infer edge weights as pairwise transmission rates between nodes [34], [27]. However, these methods assume that the network does not change over time. A dynamic algorithm has been recently proposed to infer time-varying diffusion networks by solving an MLE problem via stochastic gradient descent iterations [35]. Although successful experiments on large-scale web data reliably uncover information pathways, the estimator in [35] does not explicitly account for edge sparsity prevalent in social and information networks. Moreover, most prior approaches only attribute node infection events to the network topology, and do not account for the influence of external sources such as a ground crew for a mainstream media website.

Structural equation models (SEMs) provide a general statistical modeling technique to estimate linear relationships among both endogenous and exogenous traits; see e.g., [15]. SEMs are attractive because of their simplicity and ability to capture edge directionalities. These directional effects are often not revealed by standard linear models that leverage symmetric associations between random variables, such as those represented by covariances or correlations, [26], [12], [17], [2]. They have been widely adopted in many fields, such as economics, psychometrics [28], social sciences [13], and genetics [6], [22]. In particular, SEMs have recently been proposed for *static* gene regulatory network inference from gene expression data; see e.g., [6], [23] and references therein. However, SEMs have not been utilized to track the (possibly) time-varying topology of dynamic and directed networks.

Manuscript received September 23, 2013; revised February 02, 2014, April 08, 2014; accepted April 08, 2014. Date of publication April 14, 2014; date of current version July 16, 2014. This work was supported by the NSF ECCS Grants 1202135 and 1343248, and the NSF AST Grant 1247885. This paper appeared in part at the Fifth International Workshop on Computational Advances in Multi-Sensor Adaptive Processing, Saint Martin, December 15-18, 2013, and the Neural Information Processing Systems (NIPS) workshop on Frontiers of Network Analysis, Lake Tahoe, NV, USA, December 9, 2013. The guest editor coordinating the review of this manuscript and approving it for publication was Prof. Patrick Wolfe.

The authors are with the Department of Electrical and Computer Engineering and the Digital Technology Center, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: baing011@umn.edu; mate0058@umn.edu; georgios@umn.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSTSP.2014.2317284

In this context, the present paper proposes a *dynamic* SEM to account for directed networks over which contagions propagate, and describes how node infection times depend on both topological (endogenous) and external (exogenous) influences. Topological influences are modeled in Section II as linear combinations of infection times of other nodes in the network, whose weights correspond to entries in the time-varying asymmetric adjacency matrix. Accounting for external influences is well motivated by drawing upon examples from online media, where established news websites depend more on on-site reporting than blog references. External influence data is also useful for model identifiability, since it has been shown necessary to resolve directional ambiguities [4]. Supposing the network varies slowly with time, parameters in the proposed dynamic SEM are estimated adaptively by minimizing a sparsity-promoting exponentially-weighted least-squares (LS) criterion (Section III-A). To account for the inherently sparse connectivity of social networks, an ℓ_1-norm regularization term that promotes sparsity on the entries of the network adjacency matrix is incorporated in the cost function; see also [1], [2], [7], [18].

A novel algorithm to jointly track the network's adjacency matrix and the weights capturing the level of external influences is developed in Section III-B, which minimizes the resulting non-differentiable cost function via a proximal-gradient (PG) solver; see e.g., [5], [10], [31]. The resulting dynamic iterative shrinkage-thresholding algorithm (ISTA) is provably convergent, and offers parallel, closed-form, and sparsity-promoting updates per iteration. Proximal-splitting algorithms such as ISTA have been successfully adopted for various signal processing tasks [9], and for parallel optimization [8]. Further algorithmic improvements are outlined in Section IV. These include enhancing the algorithms' rate of convergence through Nesterov's acceleration techniques [5], [29], [30] (Section IV-A), and also adapting it for real-time operation (Section IV-B). When minimal computational complexity is at a premium, a stochastic gradient descent (SGD) algorithm is developed in Section IV-C, which adaptively minimizes an instantaneous (noisy) approximation of the ensemble LS cost. Throughout, insightful and useful extensions to the proposed algorithms that are not fully developed due to space limitations are highlighted as remarks.

Numerical tests with synthetically generated data demonstrate the effectiveness of the novel algorithms in unveiling sparse dynamically-evolving topologies (Section V-A). Experiments in Section V-B involve real temporal traces of popular global events that propagated on news websites and blogs in 2011 [21]. For instance, topologies inferred from cascades associated to the meme "Kim Jong-un" exhibit an abrupt increase in the number of edges following the appointment of the new North Korean ruler.

Notation. Bold uppercase (lowercase) letters will denote matrices (column vectors), while operators $(\cdot)^\top$, $\lambda_{\max}(\cdot)$, and $\mathrm{diag}(\cdot)$ will stand for matrix transposition, maximum eigenvalue, and diagonal matrix, respectively. The $N \times N$ identity matrix will be represented by $\mathbf{I}_N$, while $\mathbf{0}_N$ will denote the $N \times 1$ vector of all zeros, and $\mathbf{0}_{N\times P} := \mathbf{0}_N\mathbf{0}_P^\top$. The ℓ_p and Frobenius norms will be denoted by $\|\cdot\|_p$, and $\|\cdot\|_F$, respectively.

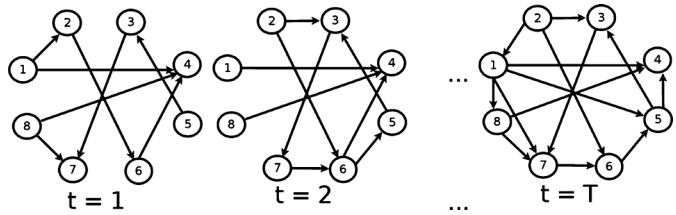

Fig. 1. Dynamic network observed across several time intervals. Note that few edges are added/removed in the transition from $t = 1$ to $t = 2$ (slowly time-varying network).

II. Network Model and Problem Statement

Consider a dynamic network with N nodes observed over time intervals $t = 1, \ldots, T$, whose abstraction is a graph with topology described by an unknown, time-varying, and weighted adjacency matrix $\mathbf{A}^t \in \mathbb{R}^{N\times N}$. Entry (i, j) of $\mathbf{A}^t$ (henceforth denoted by a_{ij}^t) is nonzero only if a directed edge connects nodes i and j (pointing from j to i) during the time interval t, as illustrated in the 8-node network in Fig. 1. As a result, one in general has $a_{ij}^t \neq a_{ji}^t$, i.e., matrix $\mathbf{A}^t$ is generally non-symmetric, which is suitable to model directed networks that explicitly encode the level of directional influence. For instance, if i denotes a news blog maintained by a journalism student, whereas j represents the web portal of a mainstream newspaper, then it is likely that $a_{ij}^t \gg a_{ji}^t \approx 0$ for those t where $a_{ij}^t \neq 0$, since the student is more likely to mention the news portal on her blog. Probably, the aforementioned directionality would have been reversed during Nov.–Dec. 2010, if i instead represents the Wikileaks blog. Note that the model tacitly assumes that the network topology remains fixed during any given time interval t, but can change across time intervals.

Suppose C contagions are sampled out of all contagions that propagate over the network during the observation interval, and the difference between infection time of node i by contagion c and the earliest observation time is denoted by $y_{ic}^t \geq 0$. In online media, y_{ic}^t can be obtained by recording the time when website i mentions news item c. For uninfected nodes at interval t, y_{ic}^t is infinite and is set to a large positive value for practical considerations. Assume that the susceptibility x_{ic} of node i to external (non-topological) infection by contagion c is known and time invariant over the observation interval. In the web context, x_{ic} can be set to the search engine rank of website i with respect to (w.r.t.) keywords associated with c.

In order to track the unknown network topology, this paper postulates that y_{ic}^t is linearly related to x_{ic} and the infection times of its single-hop neighbors. Events that adhere to this model of network-facilitated propagation abound on the web where mention of e.g., a major baseball event by a blog will not only depend on the times when similar blogs first reported the event, but also the level of interest of the blogger in baseball as a sport. In epidemiological studies, an individual's infection time by an infectious disease depends on the infection times of her immediate contacts as well as her level of immunity to the disease. Consequently, y_{ic}^t is modeled according to the following linear *dynamic* structural equation model (SEM)

$$y_{ic}^t = \sum_{j\neq i} a_{ij}^t y_{jc}^t + b_{ii}^t x_{ic} + e_{ic}^t \tag{1}$$

where b_{ii}^t captures the time-varying level of influence of external sources, and e_{ic}^t accounts for measurement errors and unmodeled dynamics. It follows from (1) that if $a_{ij}^t \neq 0$, then y_{ic}^t is affected by the value of y_{jc}^t. Rewriting (1) for the entire network leads to the vector model

$$\mathbf{y}_c^t = \mathbf{A}^t \mathbf{y}_c^t + \mathbf{B}^t \mathbf{x}_c + \mathbf{e}_c^t \tag{2}$$

where the $N \times 1$ vector $\mathbf{y}_c^t := [y_{1c}^t, \ldots, y_{Nc}^t]^\top$ collects the node infection times by contagion c during interval t, and $\mathbf{B}^t := \mathrm{diag}(b_{11}^t, \ldots, b_{NN}^t)$. Similarly, $\mathbf{x}_c := [x_{1c}, \ldots, x_{Nc}]^\top$ and $\mathbf{e}_c^t := [e_{1c}^t, \ldots, e_{Nc}^t]^\top$. Collecting observations for all C[1] contagions yields the dynamic matrix SEM

$$\mathbf{Y}^t = \mathbf{A}^t \mathbf{Y}^t + \mathbf{B}^t \mathbf{X} + \mathbf{E}^t \tag{3}$$

where $\mathbf{Y}^t := [\mathbf{y}_1^t, \ldots, \mathbf{y}_C^t]$, $\mathbf{X} := [\mathbf{x}_1, \ldots, \mathbf{x}_C]$, and $\mathbf{E}^t := [\mathbf{e}_1^t, \ldots, \mathbf{e}_C^t]$ are all $N \times C$ matrices. Note that the same network topology $\mathbf{A}^t$ is adopted for all contagions, which is suitable e.g., when different information cascades are formed around a common meme or trending (news) topic over the web; see also the real data tests in Section V-B. For this same reason, it is natural to assume that all conditional error variances are equal.

Given $\{\mathbf{Y}^t\}_{t=1}^T$ and $\mathbf{X}$, the goal is to track the underlying network topology $\{\mathbf{A}^t\}_{t=1}^T$ and the effect of external influences $\{\mathbf{B}^t\}_{t=1}^T$. To this end, the novel algorithm developed in the next section assumes slow time variation of the network topology and leverages the inherent sparsity of edges that is typical of social networks.

III. Topology Tracking Algorithm

This section deals with a regularized LS approach to estimating $\{\mathbf{A}^t, \mathbf{B}^t\}$ in (3). In a *static* setting with all measurements $\{\mathbf{Y}^t\}_{t=1}^T$ available, one solves the batch problem

$$\{\hat{\mathbf{A}}, \hat{\mathbf{B}}\} = \arg\min_{\mathbf{A},\mathbf{B}} \frac{1}{2}\sum_{t=1}^{T} \|\mathbf{Y}^t - \mathbf{A}\mathbf{Y}^t - \mathbf{B}\mathbf{X}\|_F^2 + \lambda\|\mathbf{A}\|_1 \quad \text{s. to} \quad a_{ii} = 0,\ b_{ij} = 0,\ \forall i \neq j \tag{4}$$

where $\|\mathbf{A}\|_1 := \sum_{i,j} |a_{ij}|$ is a sparsity-promoting regularization, and $\lambda > 0$ controls the sparsity level of $\hat{\mathbf{A}}$. Absence of a self-loop at node i is enforced by the constraint $a_{ii} = 0$, while having $b_{ij} = 0$, $\forall i \neq j$, ensures that $\hat{\mathbf{B}}$ is diagonal as in (2). Note that the estimator (4) reasonably assumes equal residual variances since infection times per cascade result from the same contagion over the entire network.

A. Exponentially-Weighted LS Estimator

In practice, measurements are typically acquired in a sequential manner over large social networks with thousands or even millions of nodes, calling for estimation algorithms with minimal storage requirements. Recursive solvers enabling sequential inference of the underlying network topology are thus preferred. Moreover, introducing a "forgetting factor" that assigns more weight to the most recent residuals makes it possible to track slow temporal variations of the topology. Note that the batch estimator (4) yields single estimates $\{\hat{\mathbf{A}}, \hat{\mathbf{B}}\}$ that best fit the data $\{\mathbf{Y}^t\}_{t=1}^T$ and $\mathbf{X}$ over the whole measurement horizon $t = 1, \ldots, T$, and as such (4) neglects potential network variations across time intervals.

For $t = 1, \ldots, T$, the sparsity-regularized exponentially-weighted LS estimator (EWLSE)

$$\{\hat{\mathbf{A}}^t, \hat{\mathbf{B}}^t\} = \arg\min_{\mathbf{A},\mathbf{B}} \frac{1}{2}\sum_{\tau=1}^{t} \beta^{t-\tau}\|\mathbf{Y}^\tau - \mathbf{A}\mathbf{Y}^\tau - \mathbf{B}\mathbf{X}\|_F^2 + \lambda_t\|\mathbf{A}\|_1 \quad \text{s. to} \quad a_{ii} = 0,\ b_{ij} = 0,\ \forall i \neq j \tag{5}$$

where $\beta \in (0, 1]$ is the forgetting factor that forms estimates $\{\hat{\mathbf{A}}^t, \hat{\mathbf{B}}^t\}$ using all measurements acquired until time t. Whenever $\beta < 1$, past data are exponentially discarded thus enabling tracking of dynamic network topologies. The first summand in the cost corresponds to an exponentially-weighted moving average (EWMA) of the squared model residuals norms. The EWMA can be seen as an average modulated by a sliding window of equivalent length $1/(1-\beta)$, which clearly grows as $\beta \to 1$. In the so-termed infinite-memory setting whereby $\beta = 1$, (5) boils down to the batch estimator (4). Notice that λ_t is allowed to vary with time in order to capture the generally changing edge sparsity level. In a linear regression context, a related EWLSE was put forth in [1] for adaptive estimation of sparse signals; see also [18] for a projection-based adaptive algorithm.

Before moving on to algorithms, a couple of remarks are in order.

Remark 1 (Modeling Slow Network Variations via Sparsity): To explicitly model slow topological variations across time intervals, a viable approach is to include an additional regularization term $\mu_t\|\mathbf{A} - \hat{\mathbf{A}}^{t-1}\|_1$ in the cost of (5). This way, the estimator penalizes deviations of the current topology estimate relative to its immediate predecessor $\hat{\mathbf{A}}^{t-1}$. Through the tuning parameter μ_t, one can adjust how smooth are the admissible topology variations from interval to interval. With a similar goal but enforcing temporal smoothness via kernels with adjustable bandwidth, an ℓ_1-norm-regularized logistic regression approach was put forth in [17].

Remark 2 (Selection of λ_t): Selection of the (possibly time-varying) tuning parameter λ_t is an important aspect of regularization methods such as (5), because λ_t controls the sparsity level of the inferred network and how its structure may change over time. For sufficiently large values of λ_t one obtains the trivial solution $\hat{\mathbf{A}}^t = \mathbf{O}_{N\times N}$, while increasingly more dense graphs are obtained as $\lambda_t \to 0$. An increasing λ_t will be required for accurate estimation over extended time-horizons, since for $\beta \approx 1$ the norm of the LS term in (5) grows due to noise accumulation. This way the effect of the regularization term will be downweighted unless one increases λ_t at a suitable rate, for instance proportional to $\sqrt{\sigma^2 t}$ as suggested by large deviation tail bounds when the errors are assumed $e_{ic}^t \sim \mathcal{N}(0, \sigma^2)$, and the problem dimensions N, C, T are sufficiently large [1], [25], [26]. In the topology tracking experiments of Section V, a time-invariant value of λ is adopted and typically chosen via trial and error to optimize the performance. This is justified since smaller values of β are selected for tracking network variations, which also implies that past data (and noise) are discarded faster, and the norm of the LS term in (5) remains almost invariant. As

[1]The general case where the number of contagions C_t possibly varies across time intervals can be accommodated without extra effort.

future research it would be interesting to delve further into the choice of λ_t using model selection techniques such as cross-validation [6], Bayesian information criterion (BIC) scores [17], or the minimum description length (MDL) principle [33], and investigate how this choice relates to statistical model consistency in a dynamic setting.

B. Proximal Gradient Algorithm

Exploiting the problem structure in (5), a proximal gradient (PG) algorithm is developed in this section to track the network topology; see [31] for a comprehensive tutorial treatment on proximal methods. PG methods have been popularized for ℓ_1-norm regularized linear regression problems, through the class of iterative shrinkage-thresholding algorithms (ISTA); see e.g., [10], [39]. The main advantage of ISTA over off-the-shelf interior point methods is its computational simplicity. Iterations boil down to matrix-vector multiplications involving the regression matrix, followed by a soft-thresholding operation [14, p. 93].

In the sequel, an ISTA algorithm is developed for the sparsity regularized dynamic SEM formulation (5) at time t. Based on this module, a (pseudo) real-time algorithm for tracking the dynamically-evolving network topology over the horizon $t = 1, \ldots, T$ is obtained as well. The resulting algorithm's memory storage requirement and computational cost per data sample $\{\mathbf{Y}^t, \mathbf{X}\}$ does not grow with t.

Solving (5) for a single time interval *t*. Introducing the optimization variable $\mathbf{V} := [\mathbf{A}\,\mathbf{B}]$, observe that the gradient of $f(\mathbf{V}) := \frac{1}{2}\sum_{\tau=1}^{t}\beta^{t-\tau}\|\mathbf{Y}^\tau - \mathbf{A}\mathbf{Y}^\tau - \mathbf{B}\mathbf{X}\|_F^2$ is Lipschitz continuous with a (minimum) Lipschitz constant $L_f = \lambda_{\max}(\sum_{\tau=1}^{t}\beta^{t-\tau}[(\mathbf{Y}^\tau)^\top\ (\mathbf{X})^\top]^\top[(\mathbf{Y}^\tau)^\top\ (\mathbf{X})^\top])$, i.e., $\|\nabla f(\mathbf{V}_1) - \nabla f(\mathbf{V}_2)\| \leq L_f\|\mathbf{V}_1 - \mathbf{V}_2\|$, $\forall\, \mathbf{V}_1, \mathbf{V}_2$ in the domain of f. The Lipschitz constant is time varying, but the dependency on t is kept implicit for notational convenience. On the other hand, the regularizer $g(\mathbf{V}) := \lambda_t\|\mathbf{A}\|_1$ is non-smooth. Instead of directly optimizing the cost in (5), PG algorithms minimize a sequence of overestimators evaluated at judiciously chosen points $\mathbf{U}$ (typically the current iterate, or a linear combination of the two previous iterates as discussed in Section IV-A). From the Lipschitz continuity of ∇f, for any $\mathbf{V}$ and $\mathbf{U}$ in the domain of f, it holds that $f(\mathbf{V}) \leq Q_f(\mathbf{U}, \mathbf{V}) := f(\mathbf{U}) + \langle\nabla f(\mathbf{U}), \mathbf{V} - \mathbf{U}\rangle + (L_f/2)\|\mathbf{V} - \mathbf{U}\|_F^2$. Next, form the quadratic approximation of the cost $f(\mathbf{V}) + g(\mathbf{V})$ [cf. (5)] at a given point $\mathbf{U}$

$$\begin{aligned} Q(\mathbf{V}, \mathbf{U}) :&= Q_f(\mathbf{V}, \mathbf{U}) + g(\mathbf{V}) \\ &= \frac{L_f}{2}\|\mathbf{V} - \mathbf{G}(\mathbf{U})\|_F^2 + g(\mathbf{V}) \\ &\quad + f(\mathbf{U}) - \frac{\|\nabla f(\mathbf{U})\|_F^2}{2L_f} \end{aligned} \tag{6}$$

where $\mathbf{G}(\mathbf{U}) := \mathbf{U} - (1/L_f)\nabla f(\mathbf{U})$, and clearly $f(\mathbf{V}) + g(\mathbf{V}) \leq Q(\mathbf{V}, \mathbf{U})$ for any $\mathbf{V}$ and $\mathbf{U}$. Note that $\mathbf{G}(\mathbf{U})$ corresponds to a gradient-descent step taken from $\mathbf{U}$, with step-size equal to $1/L_f$.

With $k = 1, 2, \ldots$ denoting iterations, PG algorithms set $\mathbf{U} := \mathbf{V}[k-1]$ and generate the following sequence of iterates

$$\begin{aligned} \mathbf{V}[k] :&= \arg\min_{\mathbf{V}} Q(\mathbf{V}, \mathbf{V}[k-1]) \\ &= \arg\min_{\mathbf{V}} \left\{ \frac{L_f}{2}\|\mathbf{V} - \mathbf{G}(\mathbf{V}[k-1])\|_F^2 + g(\mathbf{V}) \right\} \end{aligned} \tag{7}$$

where the second equality follows from the fact that the last two summands in (6) do not depend on $\mathbf{V}$. The optimization problem (7) is known as the *proximal operator* of the function g/L_f evaluated at $\mathbf{G}(\mathbf{V}[k-1])$, and is denoted as $\text{prox}_{g/L_f}(\mathbf{G}(\mathbf{V}[k-1]))$. Henceforth adopting the notation $\mathbf{G}[k-1] := \mathbf{G}(\mathbf{V}[k-1])$ for convenience, the PG iterations can be compactly rewritten as

$$\mathbf{V}[k] = \text{prox}_{g/L_f}(\mathbf{G}[k-1]). \tag{8}$$

A key element to the success of PG algorithms stems from the possibility of efficiently solving the sequence of subproblems (7), i.e., evaluating the proximal operator.

Let $(\mathbf{y}_i^\tau)^\top$ and $\mathbf{x}_i^\top$ denote row i of $\mathbf{Y}^\tau$ and $\mathbf{X}$, respectively; while $\mathbf{a}_{-i}^\top$ denotes the $1 \times (N-1)$ vector obtained by removing entry i from row i of $\mathbf{A}$, and likewise $\mathbf{Y}_{-i}^\tau$ is the $(N-1) \times C$ matrix obtained by removing row i from $\mathbf{Y}^\tau$. At time interval t, consider the data-related EWMAs $\mathbf{\Sigma}^t := \sum_{\tau=1}^{t}\beta^{t-\tau}\mathbf{Y}^\tau(\mathbf{Y}^\tau)^\top$, $\boldsymbol{\sigma}_i^t := \sum_{\tau=1}^{t}\beta^{t-\tau}\mathbf{Y}^\tau\mathbf{y}_i^\tau$, and $\bar{\mathbf{Y}}^t := \sum_{\tau=1}^{t}\beta^{t-\tau}\mathbf{Y}^\tau$. In addition, let $(\bar{\mathbf{y}}_i^t)^\top$ denote the i-th row of $\bar{\mathbf{Y}}^t$, and $\bar{\mathbf{Y}}_{-i}^t$ the $(N-1) \times C$ matrix obtained by removing row i from $\bar{\mathbf{Y}}^t$. Further, let $\mathbf{\Sigma}_{-i}^t$ denote the $(N-1) \times (N-1)$ matrix obtained by removing the i-th row and i-th column from $\mathbf{\Sigma}^t$. Upon computing the gradients (see Appendix A for the detailed derivation)

$$\nabla_{\mathbf{a}_{-i}} f[k] = \mathbf{\Sigma}_{-i}^t\mathbf{a}_{-i}[k] + \bar{\mathbf{Y}}_{-i}^t\mathbf{x}_i b_{ii}[k] - \boldsymbol{\sigma}_{-i}^t \tag{9}$$

$$\nabla_{b_{ii}} f[k] = \mathbf{a}_{-i}^\top[k]\bar{\mathbf{Y}}_{-i}^t\mathbf{x}_i + \frac{(1-\beta^t)}{1-\beta}b_{ii}[k]\|\mathbf{x}_i\|_2^2 - (\bar{\mathbf{y}}_i^t)^\top\mathbf{x}_i \tag{10}$$

it turns out that the parallel ISTA iterations are

$$\mathbf{a}_{-i}[k+1] = \mathcal{S}_{\lambda_t/L_f}\left(\mathbf{a}_{-i}[k] - (1/L_f)\nabla_{\mathbf{a}_{-i}} f[k]\right) \tag{11}$$

$$b_{ii}[k+1] = b_{ii}[k] - (1/L_f)\nabla_{b_{ii}} f[k] \tag{12}$$

where $\mathcal{S}_\mu(\mathbf{M})$ with entry (i,j) given by $\text{sign}(m_{ij})\max(|m_{ij}| - \mu, 0)$ denotes the soft-thresholding operator. The iterations are provably convergent to the globally optimal solution $\{\hat{\mathbf{A}}^t, \hat{\mathbf{B}}^t\}$ of (5), as per the general convergence results available for PG methods and ISTA in particular [10], [31].

Computation of the gradients in (9)–(10) requires one matrix-vector multiplication by $\mathbf{\Sigma}_{-i}^t$ and one by $\bar{\mathbf{Y}}_{-i}^t$, in addition to three vector inner-products, plus a few (negligibly complex) scalar and vector additions. Both the update of $b_{ii}[k+1]$ as well as the soft-thresholding operation in (11) entail negligible computational complexity. All in all, the simplicity of the resulting iterations should be apparent. Per iteration, the actual rows of

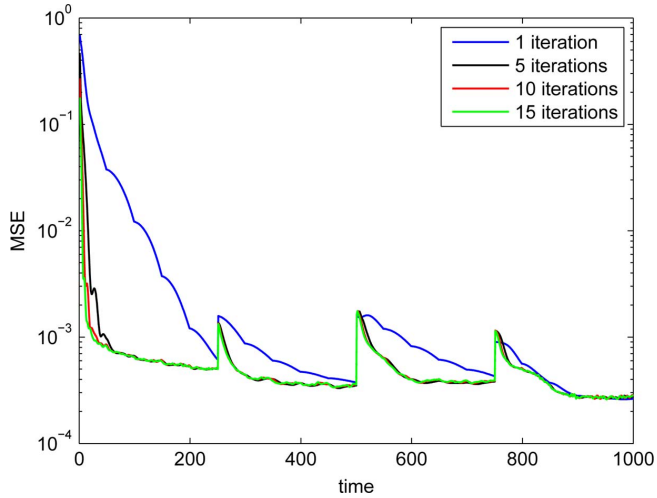

Fig. 2. MSE (i.e., $\sum_{i,j}(\hat{a}_{ij}^t - a_{ij}^t)^2/N^2$) performance of Algorithm 2 versus time. For each t, problem (5) is solved "inexactly" for $k = 1$ (Algorithm 3), 5, 10, and 15 inner iterations. It is apparent that $k = 5$ iterations suffice to attain convergence to the minimizer of (5) per t, especially after a short transient where the warm-restarts offer increasingly better initializations.

the adjacency matrix are obtained by zero-padding the updated $\mathbf{a}_{-i}[k]$, namely setting

$$\mathbf{a}_i^\top[k] = [a_{-i,i1}[k] \dots a_{-i,ii-1}[k]\, 0\, a_{-i,ii}[k] \dots a_{-i,iN}[k]]. \tag{13}$$

This way, the desired SEM parameter estimates at time t are given by $\hat{\mathbf{A}}^t = [\mathbf{a}_1^\top[k], \dots, \mathbf{a}_N^\top[k]]^\top$ and $\hat{\mathbf{B}}^t = \mathrm{diag}(b_{11}[k], \dots, b_{NN}[k])$, for k large enough so that convergence has been attained.

Remark 3 (General Sparsity-Promoting Regularization): Beyond $g(\mathbf{A}) = \lambda_t\|\mathbf{A}\|_1$, the algorithmic framework here can accommodate more general *structured sparsity*-promoting regularizers $\gamma(\mathbf{A})$ as long as the resulting proximal operator $\mathrm{prox}_{\gamma/L_f}(\cdot)$ is given in terms of scalar or (and) vector soft-thresholding operators. In addition to the ℓ_1-norm (Lasso penalty), this holds e.g., for the sum of the ℓ_2-norms of vectors with groups of non-overlapping entries of $\mathbf{A}$ (group Lasso penalty [40]), or, a linear combination of the aforementioned two—the so-termed hierarchical Lasso penalty that encourages sparsity across and within the groups defined over $\mathbf{A}$ [38]. These types of regularization could be useful if one e.g., has a priori knowledge that some clusters of nodes are more likely to be jointly (in)active [35].

Solving (5) over the entire time horizon $t = 1, \dots, T$. To track the dynamically-evolving network topology, one can go ahead and solve (5) sequentially for each $t = 1, \dots, T$ as data arrive, using (9)–(12). (The procedure can also be adopted in a batch setting, when all $\{\mathbf{Y}^t\}_{t=1}^T$ are available in memory.) Because the network is assumed to vary slowly across time intervals, it is convenient to warm-restart the ISTA iterations, that is, at time t initialize $\{\mathbf{A}[0], \mathbf{B}[0]\}$ with the previous solution $\{\hat{\mathbf{A}}^{t-1}, \hat{\mathbf{B}}^{t-1}\}$. Since the sought estimates are expected to be close to the initial points, one expects convergence to be attained after few iterations.

To obtain the new SEM parameter estimates via (9)–(12), it suffices to update (possibly) λ_t and the Lipschitz constant L_f,

Algorithm 1 Pseudo real-time ISTA for topology tracking

Require: $\{\mathbf{Y}^t\}_{t=1}^T$, $\mathbf{X}$, β.

Initialize $\hat{\mathbf{A}}^0 = \mathbf{0}_{N\times N}, \hat{\mathbf{B}}^0 = \mathbf{\Sigma}^0 = \mathbf{I}_N, \bar{\mathbf{Y}}^0 = \mathbf{0}_{N\times C}, \lambda_0$.
for $t = 1, \dots, T$ **do**
  Update λ_t, L_f and $\mathbf{\Sigma}^t$, $\bar{\mathbf{Y}}^t$ via (14)–(15).
  Initialize $\mathbf{A}[0] = \hat{\mathbf{A}}^{t-1}$, $\mathbf{B}[0] = \hat{\mathbf{B}}^{t-1}$, and set $k = 0$.
  while not converged **do**
    for $i = 1 \dots N$ (in parallel) **do**
      Compute $\mathbf{\Sigma}_{-i}^t$ and $\bar{\mathbf{Y}}_{-i}^t$.
      Form gradients at $\mathbf{a}_{-i}[k]$ and $b_{ii}[k]$ via (9)–(10).
      Update $\mathbf{a}_{-i}[k+1]$ via (11).
      Update $b_{ii}[k+1]$ via (12).
      Update $\mathbf{a}_i[k+1]$ via (13).
    end for
    $k = k + 1$.
  end while
  return $\hat{\mathbf{A}}^t = \mathbf{A}[k]$, $\hat{\mathbf{B}}^t = \mathbf{B}[k]$.
end for

as well as the data-dependent EWMAs $\mathbf{\Sigma}^t$ ($\boldsymbol{\sigma}_i^t$ is the i-th column of $\mathbf{\Sigma}^t$), and $\bar{\mathbf{Y}}^t$. Interestingly, the potential growing-memory problem in storing the entire history of data $\{\mathbf{Y}^t\}_{t=1}^T$ can be avoided by performing the recursive updates

$$\mathbf{\Sigma}^t = \beta\mathbf{\Sigma}^{t-1} + \mathbf{Y}^t(\mathbf{Y}^t)^\top \tag{14}$$

$$\bar{\mathbf{Y}}^t = \beta\bar{\mathbf{Y}}^{t-1} + \mathbf{Y}^t. \tag{15}$$

Note that the complexity in evaluating the Gram matrix $\mathbf{Y}^t(\mathbf{Y}^t)^\top$ dominates the per-iteration computational cost of the algorithm. To circumvent the need of recomputing the Lipschitz constant per time interval (that in this case entails finding the spectral radius of a data-dependent matrix), the step-size $1/L_f$ in (11)–(12) can be selected by a line search [31]. One possible choice is the backtracking step-size rule in [5], under which convergence of (33)–(12) to $\{\hat{\mathbf{A}}^t, \hat{\mathbf{B}}^t\}$ can be established as well.

Algorithm 1 summarizes the steps outlined in this section for tracking the dynamic network topology, given temporal traces of infection events $\{\mathbf{Y}^t\}_{t=1}^T$ and susceptibilities $\mathbf{X}$. It is termed *pseudo real-time* ISTA, since in principle one needs to run multiple (inner) ISTA iterations till convergence per time interval $t = 1, \dots, T$. This will in turn incur an associated delay, that may (or may not) be tolerable depending on the specific network inference problem at hand. Nevertheless, numerical tests indicate that in practice 5–10 inner iterations suffice for convergence; see also Fig. 2 and the discussion in Section IV-B.

Remark 4 (Comparison with the ADMM in [3]): In a conference precursor to this paper [3], an alternating-direction method of multipliers (ADMM) algorithm was put forth to estimate the dynamic SEM model parameters. While the basic global structure of the algorithm in [3] is similar to Algorithm 1, ADMM is adopted (instead of ISTA) to solve (5) per time $t = 1, \dots, T$. To

update $\mathbf{a}_{-i}[k+1]$, ADMM iterations require inverting the matrix $\mathbf{\Sigma}^t_{-i} + \mathbf{I}_{N-1}$, that could be computationally demanding for very large networks. On the other hand, Algorithm 1 is markedly simpler and more appealing for larger-scale problems.

IV. Algorithmic Enhancements and Variants

This section deals with various improvements to Algorithm 1, that pertain to accelerating its rate of convergence and also adapting it for real-time operation in time-sensitive applications. In addition, a stochastic-gradient algorithm useful when minimal computational complexity is at a premium is also outlined.

A. Accelerated Proximal Gradient Method and Fast ISTA

In the context of sparsity-regularized inverse problems and general non-smooth optimization, there have been several recent efforts towards improving the sublinear global rate of convergence exhibited by PG algorithms such as ISTA; see e.g., [5], [29], [30] and references therein. Since for large-scale problems first-order (gradient) methods are in many cases the only admissible alternative, the goal of these works has been to retain the computational simplicity of ISTA while markedly enhancing its global rate of convergence.

Remarkable results in [30] assert that convergence speedups can be obtained through the so-termed *accelerated* (A)PG algorithm. Going back to the derivations in the beginning of Section IV-A, APG algorithms generate the following sequence of iterates [cf. (7) and (8)]

$$\mathbf{V}[k] = \arg\min_{\mathbf{V}} Q(\mathbf{V}, \mathbf{U}[k-1]) = \text{prox}_{g/L_f}(\mathbf{G}(\mathbf{U}[k-1]))$$

where

$$\mathbf{U}[k] := \mathbf{V}[k-1] + \left(\frac{c[k-1]-1}{c[k]}\right)(\mathbf{V}[k-1] - \mathbf{V}[k-2]) \tag{16}$$

$$c[k] = \frac{1+\sqrt{4c^2[k-1]+1}}{2}. \tag{17}$$

In words, instead of minimizing a quadratic approximation to the cost evaluated at $\mathbf{V}[k-1]$ as in ISTA [cf. (7)], the accelerated PG algorithm [a.k.a. fast (F)ISTA] utilizes a linear combination of the previous two iterates $\{\mathbf{V}[k-1], \mathbf{V}[k-2]\}$. The iteration-dependent combination weights are a function of the scalar sequence (17). FISTA offers quantifiable iteration complexity, namely a (worst-case) convergence rate guarantee of $\mathcal{O}(1/\sqrt{\epsilon})$ iterations to return an ϵ-optimal solution measured by its objective value (ISTA instead offers $\mathcal{O}(1/\epsilon)$) [5], [30]. Even for general (non-)smooth optimization, APG algorithms have been shown to be optimal within the class of first-order (gradient) methods, in the sense that the aforementioned worst-case convergence rate cannot be improved [29], [30].

The FISTA solver for (5) entails the following steps [cf. (9)–(12)]

$$\tilde{\mathbf{a}}_{-i}[k] := \mathbf{a}_{-i}[k] + \left(\frac{c[k-1]-1}{c[k]}\right)(\mathbf{a}_{-i}[k] - \mathbf{a}_{-i}[k-1]) \tag{18}$$

Algorithm 2 Pseudo real-time FISTA for topology tracking

Require: $\{\mathbf{Y}^t\}_{t=1}^T$, $\mathbf{X}$, β.

Initialize $\hat{\mathbf{A}}^0 = \mathbf{0}_{N\times N}, \hat{\mathbf{B}}^0 = \mathbf{\Sigma}^0 = \mathbf{I}_N, \bar{\mathbf{Y}}^0 = \mathbf{0}_{N\times C}, \lambda_0$.
for $t = 1, \ldots, T$ **do**
  Update λ_t, L_f and $\mathbf{\Sigma}^t$, $\bar{\mathbf{Y}}^t$ via (14)–(15).
  Initialize $\mathbf{A}[0] = \mathbf{A}[-1] = \hat{\mathbf{A}}^{t-1}, \mathbf{B}[0] = \mathbf{B}[-1] = \hat{\mathbf{B}}^{t-1}$, $c[0] = c[-1] = 1$, and set $k = 0$.
  while not converged **do**
    for $i = 1 \ldots N$ (in parallel) **do**
      Compute $\mathbf{\Sigma}^t_{-i}$ and $\bar{\mathbf{Y}}^t_{-i}$.
      Update $\tilde{\mathbf{a}}_{-i}[k]$ and $\tilde{b}_{ii}[k]$ via (18)–(19).
      Form gradients at $\tilde{\mathbf{a}}_{-i}[k]$ and $\tilde{b}_{ii}[k]$ via (20)–(21).
      Update $\mathbf{a}_{-i}[k+1]$ via (22).
      Update $b_{ii}[k+1]$ via (23).
      Update $\mathbf{a}_i[k+1]$ via (13).
    end for
    $k = k+1$.
    Update $c[k]$ via (17).
  end while
  return $\hat{\mathbf{A}}^t = \mathbf{A}[k]$, $\hat{\mathbf{B}}^t = \mathbf{B}[k]$.
end for

$$\tilde{b}_{ii}[k] := b_{ii}[k] + \left(\frac{c[k-1]-1}{c[k]}\right)(b_{ii}[k] - b_{ii}[k-1]) \tag{19}$$

$$\nabla_{\mathbf{a}_{-i}} f[k] = \mathbf{\Sigma}^t_{-i}\tilde{\mathbf{a}}_{-i}[k] + \bar{\mathbf{Y}}^t_{-i}\mathbf{x}_i\tilde{b}_{ii}[k] - \boldsymbol{\sigma}^t_{-i} \tag{20}$$

$$\nabla_{b_{ii}} f[k] = \tilde{\mathbf{a}}^\top_{-i}[k]\bar{\mathbf{Y}}^t_{-i}\mathbf{x}_i + \frac{(1-\beta^t)}{1-\beta}\tilde{b}_{ii}[k]\|\mathbf{x}_i\|_2^2 - (\bar{\mathbf{y}}^t_i)^\top\mathbf{x}_i \tag{21}$$

$$\mathbf{a}_{-i}[k+1] = \mathcal{S}_{\lambda_t/L_f}\left(\tilde{\mathbf{a}}_{-i}[k] - (1/L_f)\nabla_{\tilde{\mathbf{a}}_{-i}} f[k]\right) \tag{22}$$

$$b_{ii}[k+1] = \tilde{b}_{ii}[k] - (1/L_f)\nabla_{\tilde{b}_{ii}} f[k] \tag{23}$$

where $c[k]$ is updated as in (17). The overall (pseudo) real-time FISTA for tracking the network topology is tabulated under Algorithm 2. As desired, the computational complexity of Algorithms 1 and 2 is roughly the same. Relative to Algorithm 1, the memory requirements are essentially doubled since one now has to store the two prior estimates of $\mathbf{A}$ and $\mathbf{B}$, which are nevertheless sparse and diagonal matrices, respectively. Numerical tests in Section V suggest that Algorithm 2 exhibits the best performance when compared to Algorithm 1 and the ADMM solver of [3], especially when modified to accommodate real-time processing requirements—the subject dealt with next.

B. Inexact (F)Ista for Time-Sensitive Operation

Additional challenges arise with real-time data collection, where analytics must often be performed "on-the-fly" as well as without an opportunity to revisit past entries. Online operation in delay-sensitive applications may not tolerate running multiple inner (F)ISTA iterations per time interval, so that convergence is attained for each t as required by Algorithms 1 and 2. This section touches upon an interesting tradeoff that emerges with time-constrained data-intensive problems, where

Algorithm 3 Real-time inexact FISTA for topology tracking

Require: $\{\mathbf{Y}^t\}_{t=1}^T$, $\mathbf{X}$, β.

Initialize $\mathbf{A}[1]=\mathbf{A}[0]=\mathbf{0}_{N\times N}, \mathbf{B}[1]=\mathbf{B}[0]=\mathbf{\Sigma}^0=\mathbf{I}_N, \bar{\mathbf{Y}}^0=\mathbf{0}_{N\times C}, c[1]=c[0]=1, \lambda_0$.
for $t=1,\ldots,T$ **do**
  Update λ_t, L_f and $\mathbf{\Sigma}^t$, $\bar{\mathbf{Y}}^t$ via (14)–(15).
  for $i=1\ldots N$ (in parallel) **do**
    Compute $\mathbf{\Sigma}^t_{-i}$ and $\bar{\mathbf{Y}}^t_{-i}$.
    Update $\tilde{\mathbf{a}}_{-i}[t]$ and $\tilde{b}_{ii}[t]$ via (18)–(19).
    Form gradients at $\tilde{\mathbf{a}}_{-i}[t]$ and $\tilde{b}_{ii}[t]$ via (20)–(21).
    Update $\mathbf{a}_{-i}[t+1]$ via (22).
    Update $b_{ii}[t+1]$ via (23).
    Update $\mathbf{a}_i[t+1]$ via (13).
  end for
  Update $c[t+1]$ via (17).
  return $\hat{\mathbf{A}}^t=\mathbf{A}[t+1], \hat{\mathbf{B}}^t=\mathbf{B}[t+1]$.
end for

a high-quality answer that is obtained slowly can be less useful than a medium-quality answer that is obtained quickly.

Consider for the sake of exposition a scenario where the underlying network processes are stationary, or just piecewise stationary with sufficiently long coherence time for that matter. The rationale behind the proposed real-time algorithm hinges upon the fact that the solution of (5) for each $t=1,\ldots,T$ does not need to be super accurate in the aforementioned stationary setting, since it is just an intermediate step in the outer loop matched to the time-instants of data acquisition. This motivates stopping earlier the inner iteration which solves (5) (cf. the **while** loop in Algorithms 1 and 2), possibly even after a single soft-thresholding step, as detailed in the real-time Algorithm 3. Note that in this case the inner-iteration index k coincides with the time index t. A similar adjustment can be made to the ISTA variant (Algorithm 1), and one can in general adopt a less aggressive approach by allowing a few (not just one) inner-iterations per t.

A convergence proof of Algorithm 3 in a stationary network setting will not be provided here, and is left as a future research direction. For the infinite-memory case [cf. $\beta=1$ in (5)] and the simpler ISTA counterpart of Algorithm 3 obtained when $c[t]=1$, $\forall t$, it appears possible to adapt the arguments in [24], [25] to establish that the resulting iterations converge to a minimizer of the batch problem (4). In the dynamic setting where the network is time-varying, then convergence is not expected to occur because of the continuous network fluctuations. Still, as with adaptive signal processing algorithms [37] one would like to establish that the tracking error attains a bounded steady-state. These interesting and challenging problems are the subject of ongoing investigation and will be reported elsewhere.

For synthetically-generated data according to the setup described in Section V-A, Fig. 2 shows the time evolution of Algorithm 2's mean-square error (MSE) estimation performance. For each time interval t, (5) is solved "inexactly" after running only $k=1,5,10$ and 15 inner iterations. Note that the case $k=1$ corresponds to Algorithm 3. Certainly $k=10$ iterations

Algorithm 4 SGD algorithm for topology tracking

Require: $\{\mathbf{Y}^t\}_{t=1}^T$, $\mathbf{X}$, η.

Initialize $\mathbf{A}[1]=\mathbf{0}_{N\times N}, \mathbf{B}[1]=\mathbf{I}_N, \lambda_1$.
for $t=1,\ldots,T$ **do**
  Update λ_t.
  for $i=1\ldots N$ (in parallel) **do**
    Form gradients at $\mathbf{a}_{-i}[t]$ and $b_{ii}[t]$ via (24)–(25).
    Update $\mathbf{a}_{-i}[t+1]$ via (26).
    Update $b_{ii}[t+1]$ via (27).
    Update $\mathbf{a}_i[t+1]$ via (13).
  end for
  return $\hat{\mathbf{A}}^t=\mathbf{A}[t+1], \hat{\mathbf{B}}^t=\mathbf{B}[t+1]$.
end for

suffice for the FISTA algorithm to converge to the minimizer of (5); the curve for $k=15$ is identical. Even with $k=5$ the obtained performance is satisfactory for all practical purposes, especially after a short transient where the warm-restarts offer increasingly better initializations. While Algorithm 3 shows a desirable convergent behavior, it seems that this example's network coherence time of $t=250$ time intervals is too short to be tracked effectively. Still, if the network changes are sufficiently smooth as it occurs at $t=750$, then the real-time algorithm is able to estimate the network reliably.

C. Stochastic-Gradient Descent Algorithm

A stochastic gradient descent (SGD) algorithm is developed in this section, which operates in real time and can track the (slowly-varying) underlying network topology. Among all algorithms developed so far, the SGD iterations incur the least computational cost.

Towards obtaining the SGD algorithm, consider $\beta=0$ in (5). The resulting cost function can be expressed as $f_t(\mathbf{V})+g(\mathbf{V})$, where $\mathbf{V}:=[\mathbf{A}\,\mathbf{B}]$ and $f_t(\mathbf{V}):=(1/2)\|\mathbf{Y}^t-\mathbf{A}\mathbf{Y}^t-\mathbf{B}\mathbf{X}\|_F^2$ only accounts for the data acquired at time interval t. Motivated by computational simplicity, the "inexact" gradient descent plus soft-thresholding ISTA iterations yield the following updates

$$\nabla_{\mathbf{a}_{-i}} f_t[t]=\mathbf{Y}^t_{-i}\left((\mathbf{Y}^t_{-i})^\top \mathbf{a}_{-i}[t]+\mathbf{x}_i b_{ii}[t]-\mathbf{y}_i^t\right) \quad (24)$$

$$\nabla_{b_{ii}} f_t[t]=\mathbf{a}_{-i}^\top[t]\mathbf{Y}^t_{-i}\mathbf{x}_i+b_{ii}[t]\|\mathbf{x}_i\|^2-\left(\mathbf{y}_i^t\right)^\top\mathbf{x}_i \quad (25)$$

$$\mathbf{a}_{-i}[t+1]=\mathcal{S}_{\lambda_t/\eta}\left(\mathbf{a}_{-i}[t]-\eta\nabla_{\mathbf{a}_{-i}} f_t[t]\right) \quad (26)$$

$$b_{ii}[t+1]=b_{ii}[t]-\eta\nabla_{b_{ii}} f_t[t]. \quad (27)$$

Compared to the parallel ISTA iterations in Algorithm 1 [cf. (9)–(11)], three main differences are noteworthy: (i) iterations k are merged with the time intervals t of data acquisition; (ii) the stochastic gradients $\nabla_{\mathbf{a}_{-i}} f_t[t]$ and $\nabla_{b_{ii}} f_t[t]$ involve the (noisy) data $\{\mathbf{Y}^t(\mathbf{Y}^t)^\top, \mathbf{Y}^t\}$ instead of their time-averaged counterparts $\{\mathbf{\Sigma}^t, \bar{\mathbf{Y}}^t\}$; and (iii) a generic constant step-size η is utilized for the gradient descent steps.

The overall SGD algorithm is tabulated under Algorithm 4. Forming the gradients in (24)–(25) requires one matrix-vector

multiplication by $(\mathbf{Y}^t_{-i})^\top$ and two by $\mathbf{Y}^t_{-i}$. These multiplications dominate the per-iteration computational complexity of Algorithm 4, justifying its promised simplicity. Accelerated versions could be developed as well, at the expense of marginal increase in computational complexity and doubling the memory requirements.

To gain further intuition on the SGD algorithm developed, consider the online learning paradigm under which the network topology inference problem is to minimize the expected cost $E[f_t(\mathbf{V}) + g(\mathbf{V})]$ (subject to the usual constraints on $\mathbf{V} = [\mathbf{A}\ \mathbf{B}]$). The expectation is taken w.r.t. the *unknown* probability distribution of the data. In lieu of the expectation, the approach taken throughout this paper is to minimize the empirical cost $C^T(\mathbf{V}) := (1/T)[\sum_{t=1}^T f_t(\mathbf{V}) + g(\mathbf{V})]$. Note that for $\beta = 1$, the minimizers of $C^T(\mathbf{V})$ coincide with (4) since the scaling by $1/T$ does not affect the optimal solution. For $\beta < 1$, the cost $C^T_\beta(\mathbf{V}) := \sum_{t=1}^T \beta^{T-t} f_t(\mathbf{V}) + g(\mathbf{V})$ implements an EWMA which "forgets" past data and allows tracking. In all cases, the rationale is that by virtue of the law of large numbers, if data $\{\mathbf{Y}^t\}_{t=1}^T$ are stationary, solving $\lim_{T\to\infty} \min_{\mathbf{V}} C^T(\mathbf{V})$ yields the desired solution to the expected cost.

A different approach to achieve this same goal—typically with reduced computational complexity—is to drop the expectation (or the sample averaging operator for that matter), and update the estimates via a stochastic (sub)gradient iteration $\mathbf{V}(t) = \mathbf{V}(t-1) - \eta \partial\{f_t(\mathbf{V}) + g(\mathbf{V})\}|_{\mathbf{V}=\mathbf{V}[t-1]}$. The subgradients with respect to $\mathbf{a}_{-i}$ are

$$\partial_{\mathbf{a}_{-i}} f_t[t] = \mathbf{Y}^t_{-i}\left((\mathbf{Y}^t_{-i})^\top \mathbf{a}_{-i}[t] + \mathbf{x}_i b_{ii}[t] - \mathbf{y}^t_i\right) + \lambda_t \text{sign}(\mathbf{a}_{-i}[t]) \tag{28}$$

so the resulting algorithm has the drawback of (in general) not providing sparse solutions per iteration; see also [7] for a sparse least-mean squares (LMS) algorithm. For that reason, the approach here is to adopt the proximal gradient (ISTA) formalism to tackle the minimization of the instantaneous costs $f_t(\mathbf{V}) + g(\mathbf{V})$, and yield sparsity-inducing soft-thresholded updates (26). Also acknowledging the limitation of subgradient methods to yield sparse solutions, related "truncated gradient" updates were advocated for sparse online learning in [19].

D. Choice of Algorithm

In order to track the topology of a network from cascade data, one is faced with making a choice among the developed algorithms. Algorithm 2 enjoys a theoretically-proven faster convergence than Algorithm 1, and it is recommended in situations where real-time operation and computational cost are not critical e.g., when the number of nodes in the network are in the range of hundreds or thousands. Algorithm 3 is a more practical alternative to Algorithm 2 in real-time topology tracking applications. Algorithm 4 is the most lightweight and scalable option for real-time settings, provided a slightly degraded error performance is affordable. Matlab implementation of the algorithms tested in the following section assumes that the network data can be stored in main memory, and does not exploit the parallel structure of the update recursions.

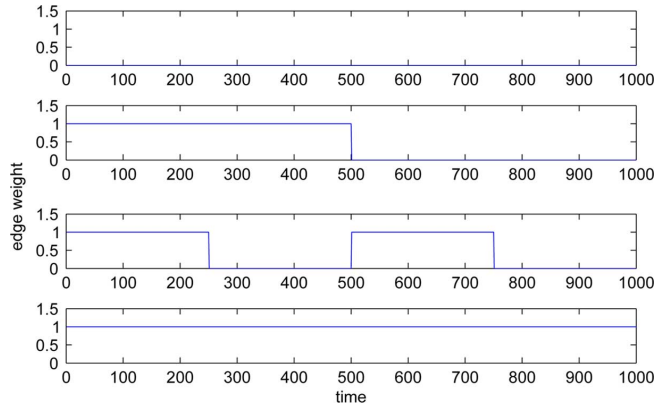

Fig. 3. Nonsmooth variation of synthetically-generated edge weights of the time-varying network. For each edge, one of the four depicted profiles is chosen uniformly at random.

V. Numerical Tests

Performance of the proposed algorithms is assessed in this section via computer simulations using both synthetically-generated network data, and real traces of information cascades collected from the web [21].

A. Synthetic Data

Data generation. Numerical tests on synthetic network data are conducted here to evaluate the tracking ability and compare Algorithms 1–4. From a "seed graph" with adjacency matrix

$$\mathbf{M} = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

a Kronecker graph of size $N = 64$ nodes was generated as described in [20].[2] The resulting nonzero edge weights of $\mathbf{A}^t$ were allowed to vary over $T = 1000$ intervals under 3 settings: i) i.i.d. Bernoulli(0.5) random variables; ii) random selection of the edge-evolution pattern uniformly from a set of four smooth functions: $a_{ij}(t) = 0.5 + 0.5\sin(0.1t)$, $a_{ij}(t) = 0.5 + 0.5\cos(0.1t)$, $a_{ij}(t) = e^{-0.01t}$, and $a_{ij}(t) = 0$; and iii) random selection of the edge-evolution pattern uniformly from a set of four nonsmooth functions shown in Fig. 3.

The number of contagions was set to $C = 80$, and $\mathbf{X}$ was formed with i.i.d. entries uniformly distributed over $[0, 3]$. Matrix $\mathbf{B}^t$ was set to $\text{diag}(\mathbf{b}^t)$, where $\mathbf{b}^t \in \mathbb{R}^N$ is a standard Gaussian random vector. During time interval t, infection times were generated synthetically as $\mathbf{Y}^t = (\mathbf{I}_N - \mathbf{A}^t)^{-1}(\mathbf{B}^t\mathbf{X} + \mathbf{E}^t)$, where $\mathbf{E}^t$ is a standard Gaussian random matrix.

Performance evaluation. With $\beta = 0.98$, Algorithm 1 was run after initializing the relevant variables as described in the algorithm table (cf. Section III-B). Cross validation was used to initialize λ_0 over the grid $[0, 100]$ by batch estimation of $\{\hat{\mathbf{A}}^1, \hat{\mathbf{B}}^1\}$ for $t = 1$. Assigning a single cascade per fold, it turned out that the best initialization was $\lambda_0 = 25$. In addition, $\lambda_t = \lambda_0$ for $t = 1, \ldots, T$ as discussed in Remark 2. Fig. 4 shows the evolution of the mean-square error (MSE), $\sum_{i,j}(\hat{a}^t_{ij} - a^t_{ij})^2/N^2$. As

[2]The Matlab implementation of Algorithms 1–4 used here can handle networks of several thousand nodes. Still a smaller network is analyzed since results are still representative of the general behavior, and offers better visualization of the results in e.g., the adjacency matrices in Figs. 5 and 6.

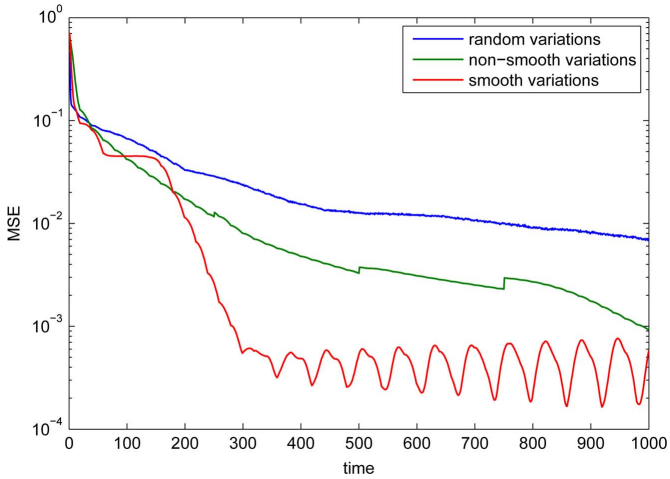

Fig. 4. MSE versus time obtained using pseudo real-time ISTA (Algorithm 1), for different edge evolution patterns.

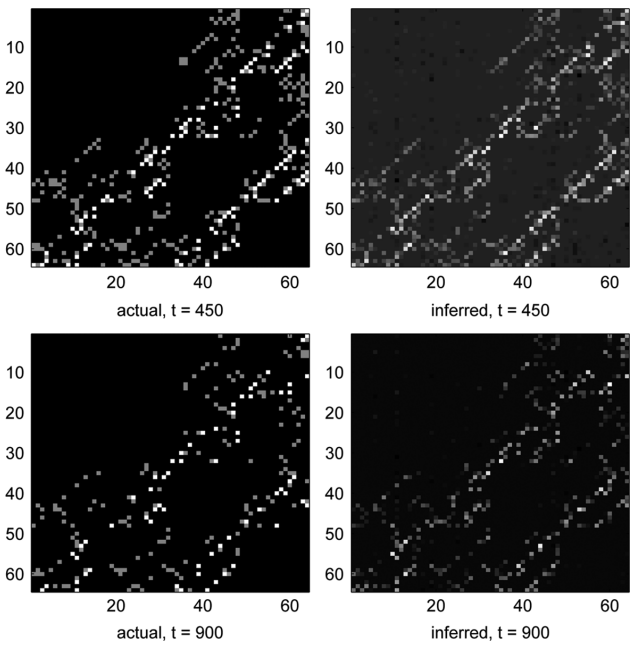

Fig. 5. Actual adjacency matrix $\hat{\mathbf{A}}^t$ and corresponding estimate $\hat{\mathbf{A}}^t$ obtained using pseudo real-time ISTA (Algorithm 1), at time intervals $t = 450$ and $t = 900$.

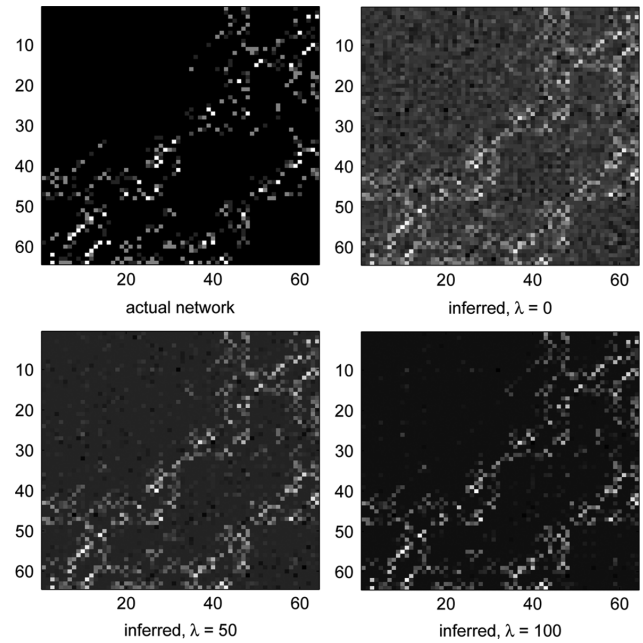

Fig. 6. Actual adjacency matrix at $t = 900$ compared with the inferred adjacency matrices using pseudo real-time FISTA (Algorithm 2), with $\lambda_t = \lambda$ for all t and $\lambda = 0$, $\lambda = 50$, and $\lambda = 100$. While $\lambda = 0$ and $\lambda = 50$ markedly overestimate the support set associated with the true network edges, the value $\lambda = 100$ in this case appears to be just about right.

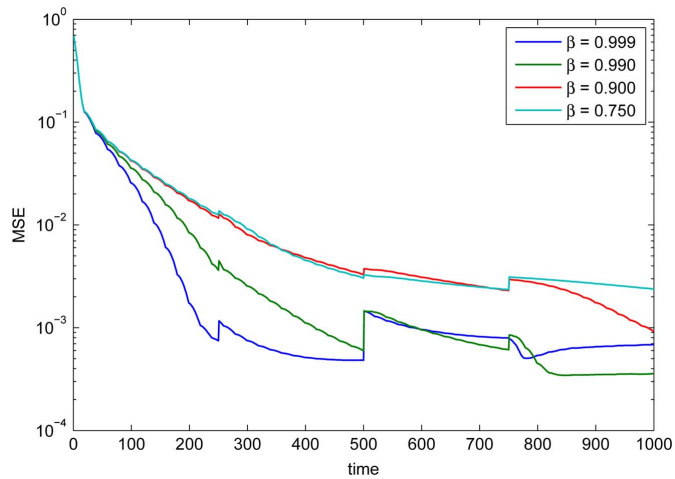

Fig. 7. MSE performance of the pseudo real-time FISTA (Algorithm 2) versus time, for different values of the forgetting factor β.

expected, the best performance was obtained when the temporal evolution of edges followed smooth functions. Even though the Bernoulli evolution of edges resulted in the highest MSE, Algorithm 1 still tracked the underlying topology with reasonable accuracy as depicted in the heat maps of the inferred adjacency matrices; see Fig. 5.

Selection of a number of parameters is critical to the performance of the developed algorithms. In order to evaluate the effect of each parameter on the network estimates, several tests were conducted by tracking the non-smooth network evolution using Algorithm 2 with varying parameter values. To illustrate the importance of leveraging sparsity of the edge weights, Fig. 6 depicts heatmaps of the adjacency matrices inferred at $t = 900$, with λ set to 0, 50, and 100 for all time intervals. Comparisons with the actual adjacency matrix reveal that increasing λ progressively refines the network estimates by driving erroneously detected nonzero edge weights to 0. Indeed, the value $\lambda = 100$ in this case appears to be just about right, while smaller values markedly overestimate the support set associated with the edges present in the actual network.

Fig. 7 compares the MSE performance of Algorithm 2 for $\beta \in \{0.999, 0.990, 0.900, 0.750\}$. As expected, the MSE associated with values of β approaching 1 degrades more dramatically when changes occur within the network (at time intervals $t = 250$, $t = 500$, and $t = 750$ in this case; see Fig. 3). The MSE spikes observed when $\beta \in \{0.999, 0.990\}$ are a manifestation of the slower rate of adaptation of the algorithm for these values of the forgetting factor. In this experiment, $\beta = 0.990$ outperformed the rest for $t > 500$. In addition, comparisons of the MSE performance in the presence of increasing noise variance are depicted in Figure 8. Although the MSE values are comparable during the initial stages of the topology inference process,

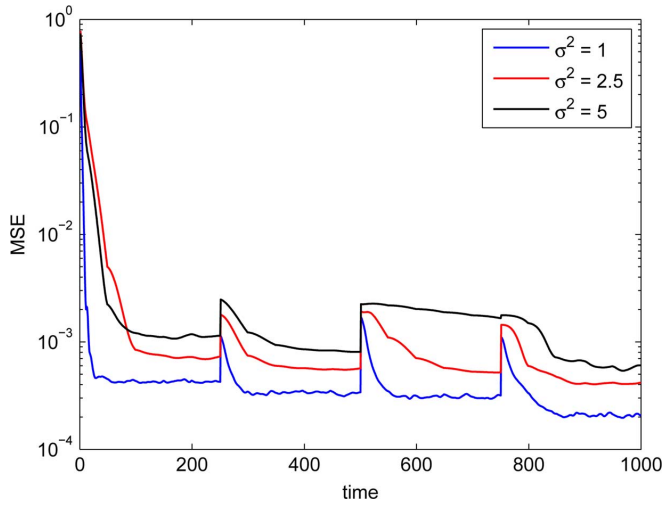

Fig. 8. MSE performance of the pseudo real-time FISTA (Algorithm 2) versus time, for different values of the noise variance σ^2.

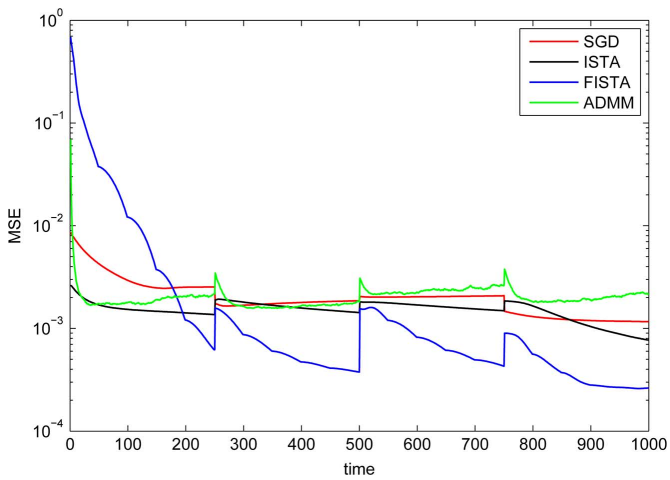

Fig. 9. MSE performance of the real-time algorithms versus time. Algorithms 3 (real-time FISTA) and 4 (SGD), as well as inexact versions of Algorithm 1 (ISTA) and the ADMM solver in [3] are compared.

as expected higher noise levels lead to MSE performance degradation in the long run.

Finally, a comparison of the real-time version of the different algorithms was carried out when tracking the synthetic time-varying network with non-smooth edge variations. Specifically, the real-time (inexact) counterparts of ISTA, FISTA (cf. Algorithm 3), SGD (cf. Algorithm 4), and a suitably modified version of the ADMM algorithm developed in [3] were run as suggested in Section IV-B, i.e., eliminating the inner **while** loop in Algorithms 1 and 2 so that a single iteration is run per time interval. Fig. 9 compares the resulting MSE curves as the error evolves with time, showing that the inexact online FISTA algorithm achieves the best error performance. The MSE performance degradation of Algorithm 3 relative to its (exact) counterpart Algorithm 2 is depicted in Fig. 2, as a function of the number of inner iterations k.

B. Real Data

Dataset description. The real data used was collected during a prior study by monitoring blog posts and news articles for memes (popular textual phrases) appearing within a set of over 3.3 million websites [35]. Traces of information cascades were recorded over a period of one year, from March 2011 till February 2012; the data is publicly available from [21]. The time when each website mentioned a specific news item was recorded as a Unix timestamp in hours (i.e., the number of hours since midnight on January 1, 1970). Specific globally-popular topics during this period were identified and cascade data for the top 5 000 websites that mentioned memes associated with them were retained.

The real-data tests that follow focus on two keywords: i) "Kim Jong-un" the current leader of North Korea whose popularity rose after the death of his father and predecessor, during the observation period; and ii) "Reid Hoffman" the founder of the professional online social network *LinkedIn*, that went public during the observation period. Each keyword is associated with several phrases mentioned on the web by blogs and news websites that covered the two individuals. Each phrase is assigned a list of tuples in the form (website id, timestamp) capturing the time when a website mentioned the phrase. Defining a cascade as any list with at least 7 tuples, the dataset was significantly reduced to the 360 websites over which 466 cascades related to "Kim Jong-un" propagated during a 45 week period. Similarly, 125 websites were retained for propagation of 85 cascades related to "Reid Hoffman" over 41 weeks. Each time interval was set to one week and the observation time-scale was adjusted to start at the beginning of the earliest cascades.

In both cases, matrix $\mathbf{Y}^t$ was constructed by setting y_{ic}^t to the time when website i mentioned phrase c if this occurred during the span of week t. Otherwise y_{ic}^t was set to a large number, $100t_{\max}$, where $t_{\max}$ denotes the largest timestamp in the dataset. Typically, the entries of matrix $\mathbf{X}$ capture prior knowledge about the susceptibility of each node to each contagion. For instance, the entry $\mathbf{x}_{ic}$ could denote the online search rank of website i for a search keyword associated with contagion c. In the absence of such real data, the entries of $\mathbf{B}^t$ were set to zero. Although the caveat here is non-uniqueness of $\{\mathbf{A}^t\}$ [4], experimental results presented next interestingly demonstrate a strong consistency with well-known events.

Experimental results. Algorithm 2 was run on real data with $\beta = 0.9$ and $\lambda_t = 100$. The choice of Algorithm 2 was based on its faster convergence properties when compared to the other alternatives, and that this is not a delay-sensitive application. Significantly more challenging for real data, λ_t was fixed for all intervals, and its value was heuristically guided by a uniform grid search over $[25, 150]$. Observing the edge evolution over time, $\lambda_t = 100$ yielded results most consistent with real-world events as will be described shortly. Fig. 10 depicts visualizations of the inferred network at $t = 10$ and $t = 40$ weeks. Little was known about Kim Jong-un during the first 10 weeks of the observation period. However, speculation about the possible successor of the dying North Korean ruler, Kim Jong-il, rose until his death on December 17, 2011 (week 38). He was succeeded by Kim Jong-un on December 30, 2011 (week 40). The network visualizations show an increasing number of edges over the 45 weeks, illustrating the growing interest of international news websites and blogs in the new ruler. Unfortunately,

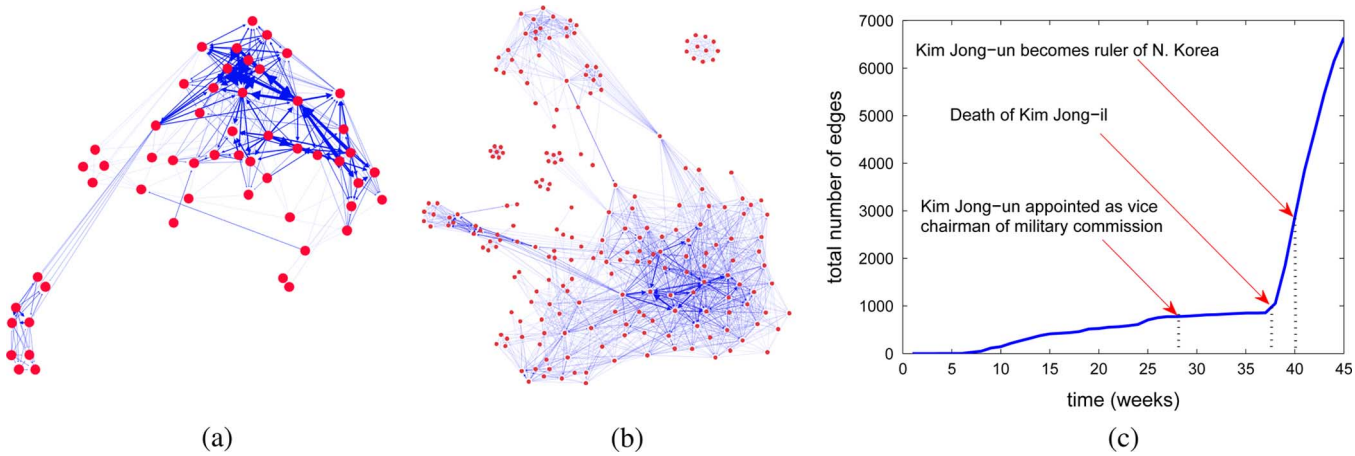

Fig. 10. Visualization of the estimated networks obtained by tracking those information cascades related to the topic "Kim Jong-un". Isolated nodes (without edges) were filtered out of the network visualization. The abrupt increase in network connectivity can be explained by three key events: i) Kim Jong-un was appointed as the vice chairman of the North Korean military commission ($t = 28$); ii) Kim Jong-il died ($t = 38$); and iii) Kim Jong-un became the ruler of North Korea ($t = 40$). (a) $t = 10$, (b) $t = 40$, (c) Inferred edges per week.

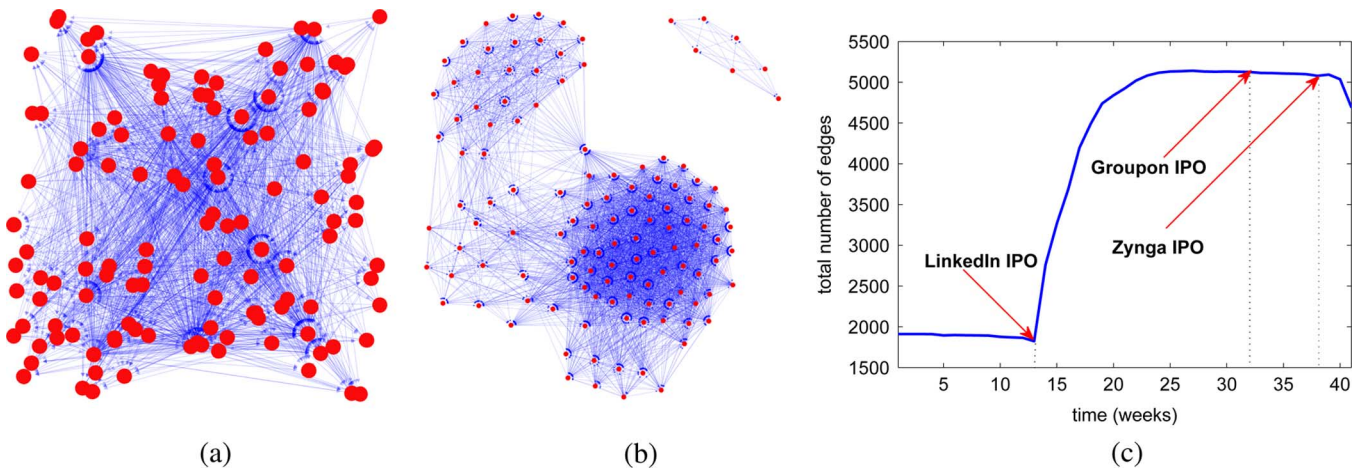

Fig. 11. Visualization of estimated networks obtained by tracking "Reid Hoffman" cascades at $t = 5$ and $t = 30$ weeks. (a) $t = 5$, (b) $t = 30$, (c) Inferred edges per week.

the observation horizon does not go beyond $T = 45$ weeks. A longer span of data would have been useful to investigate at what rate did the global news coverage on the topic eventually subside.

Fig. 10 c) depicts the time evolution of the total number of edges in the inferred dynamic network. Of particular interest are the weeks during which: i) Kim Jong-un was appointed as the vice chairman of the North Korean military commission; ii) Kim Jong-il died; and iii) Kim Jong-un became the ruler of North Korea. These events were the topics of many online news articles and political blogs, an observation that is reinforced by the experimental results shown in the plot.

The results of running Algorithm 2 on the second dataset are shown in Fig. 11. Although Reid Hoffman was already popular in technology media coverage, his visibility in popular news and blogs increased tremendously following the highly successful initial public offering (IPO) of LinkedIn on May 19, 2011. Towards the end of 2011, a number of other successful technology companies like Groupon and Zynga went public, possibly stabilizing the amount of media coverage on Reid Hoffman. In fact, the drop in the number of edges towards week 41 could be attributed to the captivation of media attention by the IPOs that occurred later in the year.

VI. Concluding Summary

A dynamic SEM was proposed in this paper for network topology inference, using timestamp data for propagation of contagions typically observed in social networks. The model explicitly captures both topological influences and external sources of information diffusion over the unknown network. Exploiting the inherent edge sparsity typical of large networks, a computationally-efficient proximal gradient algorithm with well-appreciated convergence properties was developed to minimize a suitable sparsity-regularized exponentially-weighted LS estimator. Algorithmic enhancements were proposed, that pertain to accelerating convergence and performing the network topology inference task in real time. In addition, reduced-complexity stochastic-gradient iterations were outlined and showed to attain worthwhile performance.

A number of experiments conducted on synthetically-generated data demonstrated the effectiveness of the proposed algorithms in tracking dynamic and sparse networks. Ex-

perimental results on real datasets focused on two popular personalities that made headlines during the observation period successfully showed changes in edge connectivity between media websites corresponding to increased media frenzy following specific events centered on them.

The present work opens up multiple directions for exciting follow-up work. Future and ongoing research includes: i) investigating the conditions for identifiability of sparse and dynamic SEMs, as well as their statistical consistency properties tied to the selection of λ_t; ii) consider also measures of causal influence, and formalizing links between causal effects and model parameters [32]; iii) formally establishing the convergence of the (inexact) real-time algorithms in a stationary network setting, and tracking their MSE performance under simple models capturing the network variation; iv) devising algorithms for MLE of dynamic SEMs and comparing the performance of the LS alternative of this paper; v) generalizing the SEM using kernels or suitable graph similarity measures to enable network topology forecasting; and vi) exploiting the parallel structure of the algorithms to devise disk-based MapReduce/Hadoop implementations scalable to million-node graphs.

Appendix A
Derivation of ISTA Iterations

This Appendix gives a more detailed derivation of the ISTA iterations in Algorithm 1. Specializing to (5), note that (7) decomposes into

$$\begin{aligned}\mathbf{A}[k] :&= \arg\min_{\mathbf{A}} \left\{ \frac{L_f}{2}\|\mathbf{A}-\mathbf{G}_A[k-1]\|_F^2 + \lambda_t\|\mathbf{A}\|_1 \right\} \\ &= \mathrm{prox}_{\lambda_t\|\cdot\|_1/L_f}(\mathbf{G}_A[k-1]) \qquad (29)\\ \mathbf{B}[k] :&= \arg\min_{\mathbf{B}} \left\{ \|\mathbf{B}-\mathbf{G}_B[k-1]\|_F^2 \right\} = \mathbf{G}_B[k-1] \qquad (30)\end{aligned}$$

subject to the constraints in (5) which so far have been left implicit, and $\mathbf{G} := [\mathbf{G}_A\, \mathbf{G}_B]$. Because there is no regularization on the matrix $\mathbf{B}$, the corresponding update (30) boils-down to a simple gradient-descent step. It follows that $\mathrm{prox}_{\lambda_t\|\cdot\|_1/L_f}(\cdot) = \mathcal{S}_{\lambda_t/L_f}(\cdot)$, e.g., [10], [14]; so that

$$\mathbf{A}[k] = \mathcal{S}_{\lambda_t/L_f}(\mathbf{G}_A[k-1]). \qquad (31)$$

What remains now is to obtain expressions for the gradient of $f(\mathbf{V})$ with respect to $\mathbf{A}$ and $\mathbf{B}$, which are required to form the matrices $\mathbf{G}_A$ and $\mathbf{G}_B$. To this end, note that by incorporating the constraints $a_{ii}=0$ and $b_{ij}=0$, $\forall j\neq i$, $i=1,\ldots N$, one can simplify the expression of $f(\mathbf{V})$ as

$$f(\mathbf{V}) := \frac{1}{2}\sum_{\tau=1}^{t}\sum_{i=1}^{N}\beta^{t-\tau}\|(\mathbf{y}_i^\tau)^\top - \mathbf{a}_{-i}^\top\mathbf{Y}_{-i}^\tau - b_{ii}\mathbf{x}_i^\top\|_F^2 \qquad (32)$$

where $(\mathbf{y}_i^\tau)^\top$, $\mathbf{a}_{-i}^\top$, and $\mathbf{Y}_{-i}$ have been previously defined. It is apparent from (32) that $f(\mathbf{V})$ is separable across the trimmed row vectors $\mathbf{a}_{-i}^\top$, and the scalar diagonal entries b_{ii}, $i=1,\ldots,N$. The sought gradients are readily obtained as

$$\begin{aligned}\nabla_{\mathbf{a}_{-i}} f(\mathbf{V}) &= -\sum_{\tau=1}^{t}\beta^{t-\tau}\mathbf{Y}_{-i}^\tau(\mathbf{y}_i^\tau - (\mathbf{Y}_{-i}^\tau)^\top\mathbf{a}_{-i} - \mathbf{x}_i b_{ii})\\ \nabla_{b_{ii}} f(\mathbf{V}) &= -\sum_{\tau=1}^{t}\beta^{t-\tau}((\mathbf{y}_i^\tau)^\top - \mathbf{a}_{-i}^\top\mathbf{Y}_{-i}^\tau - b_{ii}\mathbf{x}_i^\top)\mathbf{x}_i.\end{aligned}$$

Using the definitions for $\mathbf{\Sigma}^t$, $\mathbf{\Sigma}_{-i}^t$, $\boldsymbol{\sigma}_i^t$, $\boldsymbol{\sigma}_{-i}^t$, $\bar{\mathbf{Y}}^t$, and $\bar{\mathbf{Y}}_{-i}^t$, the gradient expressions for $i=1,\ldots,N$ can be compactly expressed as

$$\nabla_{\mathbf{a}_{-i}} f(\mathbf{V}) = \mathbf{\Sigma}_{-i}^t\mathbf{a}_{-i} + \bar{\mathbf{Y}}_{-i}^t\mathbf{x}_i b_{ii} - \boldsymbol{\sigma}_{-i}^t \qquad (33)$$

$$\nabla_{b_{ii}} f(\mathbf{V}) = \mathbf{a}_{-i}^\top\bar{\mathbf{Y}}_{-i}^t\mathbf{x}_i + \frac{1-\beta^t}{1-\beta}b_{ii}\|\mathbf{x}_i\|_2^2 - (\bar{\mathbf{y}}_i^\tau)^\top\mathbf{x}_i. \qquad (34)$$

From (30)–(31) and (33)–(34), the resulting parallel ISTA iterations are

$$\nabla_{\mathbf{a}_{-i}} f[k] = \mathbf{\Sigma}_{-i}^t\mathbf{a}_{-i}[k] + \bar{\mathbf{Y}}_{-i}^t\mathbf{x}_i b_{ii}[k] - \boldsymbol{\sigma}_{-i}^t \qquad (35)$$

$$\nabla_{b_{ii}} f[k] = \mathbf{a}_{-i}^\top[k]\bar{\mathbf{Y}}_{-i}^t\mathbf{x}_i + \frac{(1-\beta^t)}{1-\beta}b_{ii}[k]\|\mathbf{x}_i\|_2^2 - (\bar{\mathbf{y}}_i^t)^\top\mathbf{x}_i \qquad (36)$$

$$\mathbf{a}_{-i}[k+1] = \mathcal{S}_{\lambda_t/L_f}\left(\mathbf{a}_{-i}[k] - (1/L_f)\nabla_{\mathbf{a}_{-i}} f[k]\right) \qquad (37)$$

$$b_{ii}[k+1] = b_{ii}[k] - (1/L_f)\nabla_{b_{ii}} f[k]. \qquad (38)$$

References

[1] D. Angelosante, J. A. Bazerque, and G. B. Giannakis, "Online adaptive estimation of sparse signals: Where RLS meets the ℓ_1-norm," *IEEE Trans. Signal Process.*, vol. 58, pp. 3436–3447, Jul. 2010.

[2] D. Angelosante and G. B. Giannakis, "Sparse graphical modeling of piecewise-stationary time series," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Prague, Czech Republic, May 2011, pp. 1960–1963.

[3] B. Baingana, G. Mateos, and G. B. Giannakis, "Dynamic structural equation models for tracking topologies of social networks," in *Proc. 5th Int. Workshop Comput. Adv. Multi-Sensor Adaptive Process.*, Saint Martin, Dec. 2013.

[4] J. A. Bazerque, B. Baingana, and G. B. Giannakis, "Identifiability of sparse structural equation models for directed and cyclic networks," in *Proc. Global Conf. Signal Inf. Process.*, Austin, TX, USA, Dec. 2013.

[5] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci.*, vol. 2, pp. 183–202, Jan. 2009.

[6] X. Cai, J. A. Bazerque, and G. B. Giannakis, "Gene network inference via sparse structural equation modeling with genetic perturbations," *PLoS Comp. Biol.*, vol. 9, May 2013, e1003068 doi:10.1371/journal.pcbi.1003068.

[7] Y. Chen, Y. Gu, and A. O. Hero, III, "Sparse LMS for system identification," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Taipei, Taiwan, Apr. 2009, pp. 3125–3128.

[8] P. L. Combettes and J.-C. Pesquet, "A proximal decomposition method for solving convex variational inverse problems," *Inverse Problems*, vol. 24, no. 6, pp. 1–27, 2008.

[9] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, ser. ser. Springer Optimization and its Applications, H. H. Bauschke, R. S. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, Eds. New York, NY, USA: Springer, 2011, pp. 185–212.

[10] I. Daubechies, M. Defrise, and C. D. Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Comm. Pure Appl. Math.*, vol. 57, pp. 1413–1457, Aug. 2004.
[11] D. Easley and J. Kleinberg, *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. New York, NY, USA: Cambridge Univ. Press, 2010.
[12] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, pp. 432–441, Dec. 2007.
[13] A. S. Goldberger, "Structural equation methods in the social sciences," *Econometrica*, vol. 40, pp. 979–1001, Nov. 1972.
[14] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed. New York, NY, USA: Springer, 2009.
[15] D. Kaplan, *Structural Equation Modeling: Foundations and Extensions*, 2nd ed. Thousand Oaks, CA, USA: Sage, 2009.
[16] E. D. Kolaczyk, *Statistical Analysis of Network Data: Methods and Models*. New York, NY, USA: Springer, 2009.
[17] M. Kolar, L. Song, A. Ahmed, and E. P. Xing, "Estimating time-varying networks," *Ann. Appl. Statist.*, vol. 4, pp. 94–123, 2010.
[18] Y. Kopsinis, K. Slavakis, and S. Theodoridis, "Online sparse system identification and signal reconstruction using projections onto weighted ℓ_1 balls," *IEEE Trans. Signal Process.*, vol. 59, no. 3, pp. 936–952, Mar. 2011.
[19] J. Langford, L. Li, and T. Zhang, "Sparse online learning via truncated gradient," *J. Mach. Learn. Res.*, vol. 10, pp. 719–743, Mar. 2009.
[20] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani, "Kronecker graphs: An approach to modeling networks," *J. Mach. Learn. Res.*, vol. 11, pp. 985–1042, Mar. 2010.
[21] J. Leskovec, "Web and blog datasets," *Stanford Network Analysis Project* 2011 [Online]. Available: http://snap.stanford.edu/infopath/data.html
[22] B. Liu, A. de la Fuente, and I. Hoeschele, "Gene network inference via structural equation modeling in genetical genomics experiments," *Genetics*, vol. 178, pp. 1763–1776, Mar. 2008.
[23] B. A. Logsdon and J. Mezey, "Gene expression network reconstruction by convex feature selection when incorporating genetic perturbations," *PLoS Comp. Biol.*, vol. 6, Dec. 2010, e1001014. doi:10.1371/journal.pcbi.1001014.
[24] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *J. Mach. Learn. Res.*, vol. 11, pp. 19–60, Jan. 2010.
[25] M. Mardani, G. Mateos, and G. B. Giannakis, "Dynamic anomalography: Tracking network anomalies via sparsity and low rank," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 1, pp. 50–66, Feb. 2013.
[26] N. Meinshausen and P. Buhlmann, "High-dimensional graphs and variable selection with the lasso," *Ann. Statist.*, vol. 34, pp. 1436–1462, 2006.
[27] S. Meyers and J. Leskovec, "On the convexity of latent social network inference," in *Proc. Neural Inf. Process. Syst. Conf.*, Vancouver, BC, Canada, Feb. 2013.
[28] B. Muthén, "A general structural equation model with dichotomous, ordered categorical, and continuous latent variable indicators," *Pyschometrika*, vol. 49, pp. 115–132, Mar. 1984.
[29] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $O(1/k^2)$," *Soviet Math. Doklady*, vol. 27, pp. 372–376, 1983.
[30] Y. Nesterov, "Smooth minimization of nonsmooth functions," *Math. Prog.*, vol. 103, pp. 127–152, 2005.
[31] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optimiz.*, vol. 1, pp. 123–231, 2013.
[32] J. Pearl, *Causality: Models, Reasoning, and Inference*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2009.
[33] I. Ramirez and G. Sapiro, "An MDL framework for sparse coding and dictionary learning," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 2913–2927, Jun. 2012.
[34] M. G. Rodriguez, D. Balduzzi, and B. Scholkopf, "Uncovering the temporal dynamics of diffusion networks," in *Proc. 28th Int. Conf. Mach. Learn.*, Bellevue, WA, USA, Jul. 2011.
[35] M. G. Rodriguez, J. Leskovec, and B. Scholkopf, "Structure and dynamics of information pathways in online media," in *Proc. 6th ACM Int. Conf. Web Search and Data Mining*, Rome, Italy, Dec. 2010.
[36] E. M. Rogers, *Diffusion of Innovations*, 4th ed. Washington, DC: Free Press, 1995.
[37] V. Solo and X. Kong, *Adaptive Signal Processing Algorithms: Stability and Performance*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1995.
[38] P. Sprechmann, I. Ramirez, G. Sapiro, and Y. Eldar, "C-HiLasso: A collaborative hierarchical sparse modeling framework," *IEEE Trans. Signal Process.*, vol. 59, no. 9, pp. 4183–4198, Sep. 2011.
[39] S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo, "Sparse reconstruction by separable approximation," *IEEE Trans. Signal Process.*, vol. 57, no. 7, pp. 2479–2493, Jul. 2009.
[40] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *J. R. Statist. Soc. B*, vol. 68, pp. 49–67, 2006.

Brian Baingana (S'11) received the B.Sc. degree in electrical engineering from Makerere University, Kampala, Uganda in 2007 and the M.Sc. degree in electrical engineering from the University of Minnesota (UofM), Minneapolis, in 2013. Currently, he is working towards the Ph.D. degree with the Department of Electrical and Computer Engineering at the UofM. From 2007 to 2009, he worked as a software systems engineer with MTN Uganda, a telecommunications operator.

His broader research interests include statistical learning, network science, and statistical signal processing. Specifically, he is currently interested in models and algorithms for low-cost, real-time analytics in dynamic social networks and big data. He was a recipient of the prestigious university-wide Graduate School Fellowship for the academic years 2009–2012 at the UofM.

Gonzalo Mateos (M'12) received his B.Sc. degree in electrical engineering from Universidad de la República (UdelaR), Montevideo, Uruguay in 2005 and the M.Sc. and Ph.D. degrees in electrical engineering from the University of Minnesota (UofM), Minneapolis, in 2009 and 2011.

Currently, he is a visiting scholar with the Computer Science Department at Carnegie Mellon University. He also holds an appointment as a post-doctoral research associate with the Department of Electrical and Computer Engineering and the Digital Technology Center, UofM. From 2003 to 2006, he was an assistant with the Department of Electrical Engineering, UdelaR. From 2004 to 2006, he worked as a Systems Engineer at Asea Brown Boveri (ABB), Uruguay. His research interests lie in the areas of statistical learning from Big Data, network science, wireless communications, and signal processing. His current research focuses on algorithms, analysis, and application of statistical signal processing tools to dynamic network health monitoring, social, power grid, and Big Data analytics. His doctoral work has been recognized with the 2013 UofM's Best Dissertation Award (Honorable Mention) across all Physical Sciences and Engineering areas.

Georgios B. Giannakis (F'97) received his Diploma in electrical engr. from the Ntl. Tech. Univ. of Athens, Greece, 1981. From 1982 to 1986 he was with the Univ. of Southern California (USC), where he received his M.Sc. in electrical engineering, 1983, M.Sc. in mathematics, 1986, and Ph.D. in electrical engr., 1986. Since 1999 he has been a professor with the Univ. of Minnesota, where he now holds an ADC Chair in Wireless Telecommunications in the ECE Department, and serves as director of the Digital Technology Center.

His general interests span the areas of communications, networking and statistical signal processing—subjects on which he has published more than 360 journal papers, 600 conference papers, 20 book chapters, two edited books and two research monographs (h-index 107). Current research focuses on sparsity and big data analytics, wireless cognitive radios, mobile ad hoc networks, renewable energy, power grid, gene-regulatory, and social networks. He is the (co-) inventor of 21 patents issued, and the (co-) recipient of 8 best paper awards from the IEEE Signal Processing (SP) and Communications Societies, including the G. Marconi Prize Paper Award in Wireless Communications. He also received Technical Achievement Awards from the SP Society (2000), from EURASIP (2005), a Young Faculty Teaching Award, and the G. W. Taylor Award for Distinguished Research from the University of Minnesota. He is a Fellow of EURASIP, and has served the IEEE in a number of posts, including that of a Distinguished Lecturer for the IEEE-SP Society.