Learning the Leader-Follower Structure from Data Streams

Mariana Galrinho
mariana.galrinho@tecnico.ulisboa.pt
Instituto Superior Técnico

Abstract—A common objective of modern applications is to make sense of large streams of information. Such data is usually constituted by signals that evolve according to some underlying structure that, once known, allows not only to understand the data but also to carry inference tasks. This work focuses on a leader-follower structure, where a small subset of signals (leaders) drives the remaining ones (followers). The main goal is to identify the leaders and the parameters representing the influence of leaders on followers, extracting the underlying graph of relationships. Two restrictions incorporating prior information are considered: constraints on the parameters and an upper-bound on the number of leaders. We start by solving the problem where each follower has its own set of leaders. We formulate it as a nonconvex optimization problem and propose heuristics based on a convex relaxation to solve it. We then approach the problem of finding a shared set of leaders for all signals. Two strategies are proposed: one solves the first problem for each signal and performs a voting to select the global leaders, the other extends the proposed heuristics to perform the inference with all network signals. For both problems, experiments show improved performance over methods adapted from the literature. Moreover, the proposed heuristics can be efficiently applied using Frank-Wolfe algorithms, rendering them attractive for larger networks. Finally, this work is extended in different ways, dealing with missing data and applying our methods in a wider range of setups, thus, showing their flexibility.

Index Terms—Leader-Follower Relationships, Causal Graph Identification, Sparse Networks, Leadership Identification, Convex Optimization Problems, Frank-Wolfe Algorithm

I. INTRODUCTION

Several fields, among them biology, finances, and social networks, strive to make sense of large amounts of data. The signals that compose the data often arise from a common structure that, once exposed, helps us elucidating ongoing behaviors and carrying signal processing tasks. This structure can often be represented by a directed graph, in which a node corresponds to a signal and an arc corresponds to some kind of causal relationship.

In this work, we deal with the problem of identifying a graph of interest, the leader-follower graph. This graph models setups in which a small subset of the signals, the leaders, triggers the behavior of the remaining ones, the followers. Therefore the leader-follower graph is a bipartite graph, where all arcs go from leaders to followers, as illustrated in Fig. 1.

This kind of relationship is the fabric of a wealth of practical settings, from social interactions, where individuals respond to actions of others, to financial markets, where stock prices react to shifts of others either by copying their trend or opposing it.

Exposing the leader-follower graph from a given set of signals, helps us to know which signals are influencing the others and how the web of influence plays out.

Leadership identification has been mostly approached as the identification of the entities with a higher influence in social networks to increase the diffusion of information. However, the relations between entities are often assumed to be known [1].

The inference of causal relationships has attracted significant attention. An early notion of causality is that of Granger causality [2], which usually represents the relationships with Vector Autoregressive (VAR) models. However, this framework only considers time-lagged influences. Structural Vector Autoregressive models (SVAR), proposed in [3], remove that restriction and also account for instantaneous dependencies.

Many approaches regularize the combinatorial problem of finding the most plausible structure by searching for a sparsely connected graph. This problem differs from the usual sparsity \( \ell_1 \) norm regularization since the goal is to impose block-sparisty, jointly penalizing the edge coefficients. Some approaches solve an optimization with a group Lasso penalty, constraining a combination of the \( \ell_1 \) and \( \ell_2 \) norms of the coefficients [4]. However, this does not always result in the simplest network, as shown in [5], where they introduced a reweighted reformulation with the group Lasso in the objective. Based on an extension of the atomic norm framework, an alternative formulation that constrains a relaxation of the \( \ell_0 \) norm of the edge vectors was proposed in [6]. They use a Frank-Wolfe type algorithm to improve computational efficiency. A different approach based on the connection with
compressed sensing was explored in [7], where they solve the problem with Block OMP (BOMP), an adapted Orthogonal Matching Pursuit (OMP) algorithm.

The previous approaches decouple the problem in several subproblems, one for each node, and use a VAR model, disregarding instantaneous relationships. Methods that are able to learn SVAR models, such as [8], are often restricted to nonlinear models or nonlinear perturbations.

Moreover, all of the mentioned approaches allow relationships between all signals, as illustrated in Fig. 2, so they do not obtain a bipartite graph and do not identify the leader signals. Most of them also offer no possibility of fixing a maximum number of leaders, loosely controlling sparsity. Finally, an important feature that is missing is the inclusion of constraints for the coefficients’ values. In real scenarios, this may lead to disregarding prior information or may result in values that are inconsistent with the physical processes being described.

In this work, we address the leader-follower problem with the above constraints. As a starting point, we approach the problem where each follower has its own set of leaders. In that case, each signal can be handled individually. Capitalizing on our solutions, we then move to the problem of identifying a single set of leaders for the whole network, uncovering the underlying leader-follower graph.

We formulate both problems as nonconvex optimizations and introduce heuristics based on convex relaxations to solve them. We also show that the convex relaxations can be solved with variants of the Frank-Wolfe algorithm that considerably reduce the computing time in relation to a standard solver (CVX), thus improving the efficiency of the proposed methods.

Although not presented here due to space constraints, this work is extended in the thesis [9], where we consider a recently proposed approach that deals with missing data, and improve it using an alternating minimization technique. In the context of that work, we also propose an alternative to the Frank-Wolfe algorithms: a Douglas-Rachford splitting approach. In addition, we explore other uses of our methods beyond network signals, thereby showing their flexibility.

The remainder of this extended abstract is organized as follows. In Section II, we state the leader-follower problem. Section III approaches the problem where each follower has its own leaders. Section IV extends the solutions to the case where all followers share the same set of leaders. In both sections, numerical results are provided and Frank-Wolfe algorithms are introduced. Finally, we state our conclusions in Section V.

II. PROBLEM STATEMENT

We observe a set of $N$ signals, where the $n$th signal is the set of samples $\{y_n(t), y_n(2), \ldots, y_n(T)\}$, with $y_n(t) \in \mathbb{R}$ for $1 \leq t \leq T$. We assume that a small subset of the signals drives the others. Specifically, letting $\mathcal{N} = \{1, 2, \ldots, N\}$ index the set of available signals, we assume $\mathcal{N}$ can be partitioned as $\mathcal{N} = \mathcal{L} \cup \mathcal{F}$, where $\mathcal{L}$ indexes the leaders and has small cardinality, and $\mathcal{F}$ indexes the followers.

We consider a linear model, where each follower signal is a noisy superposition of filtered versions of the leaders’ signals:

$$y_f(t) = \sum_{l \in \mathcal{L}} \sum_{d=0}^{D} x_{lf}(d)y_l(t-d) + v_f(t), \quad t = D + 1, \ldots, T,$$

for $f \in \mathcal{F}$. The vector $x_{lf} = (x_{lf}(0), x_{lf}(1), \ldots, x_{lf}(D)) \in \mathbb{R}^{D+1}$ contains the coefficients of the finite-impulse response (FIR) that models the influence of $l$th leader on the $f$th follower. The vector $v_f = (v_f(D+1), v_f(D+2), \ldots, v_f(T)) \in \mathbb{R}^{T-D}$ models additive noise. Defining $y_f = (y_f(D+1), y_f(D+2), \ldots, y_f(T)) \in \mathbb{R}^{T-D}$ and the Toeplitz matrix with delayed versions of the signal $y_l$,

$$A_l = \begin{bmatrix}
    y_l(D+1) & \cdots & y_l(1) \\
    y_l(D+2) & \cdots & y_l(2) \\
    \vdots & \ddots & \vdots \\
    y_l(T) & \cdots & y_l(T-D)
\end{bmatrix} \in \mathbb{R}^{(T-D) \times (D+1)},$$

we can rewrite (1) in matrix form:

$$y_f = \sum_{l \in \mathcal{L}} A_l x_{lf} + v_f, \quad f \in \mathcal{F}. \quad (2)$$

Our problem is two-fold. Given the set of $N$ observed signals, we wish to identify which signals are the leaders (i.e., the set $\mathcal{L}$) and how the leaders drive the followers (i.e., the filters $x_{lf}$ for $l \in \mathcal{L}$ and $f \in \mathcal{F}$). In short, we want to infer the leader-follower structure underlying the signals, as in Fig 1.

We adopt the following key restrictions for the problem. Constraints for the filters: each $x_{lf}$ lies in a compact, convex set $X_{lf} \subset \mathbb{R}^{D+1}$ known a priori. Cardinality upper-bound on the set of leaders: we know $L$ such that $L \leq L$, where $L = |\mathcal{L}|$. We also assume that the order $D$ of the filters is known.

III. EACH FOLLOWER WITH ITS OWN SET OF LEADERS

We first handle the problem where each signal has its own set of leaders. Since the signals do not need to share the same set of leaders, the coupling present in model (1) does not hold and, thus, it is possible to handle each signal independently. Focusing on a given signal $y_n \in \mathbb{R}^{T-D}$, we can drop the dependence on $n$ and work with the model

$$y = \begin{bmatrix}
    A_1 & \cdots & A_K
\end{bmatrix} \begin{bmatrix}
    x_1 \\
    \vdots \\
    x_K
\end{bmatrix} + v,$$

where...
where we define, for this section, \( A_k = \{ A_m : m \neq n \} \) and \( \{ x_{kn} : m \neq n \} \). Here, \( K = N - 1 \) and \( x_k \in X_k \) if \( k \in \mathcal{L} \), otherwise, \( x_k = 0 \).

A. Problem formulation

We propose the following formulation for the problem:

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \| y - \sum_{k=1}^{K} A_k x_k \|^2 \\
\text{subject to} & \quad \sum_{k=1}^{K} I_k (x_k, s_k) \leq T, \\
& \quad 0 \leq s_k \leq 1, \quad k \in K
\end{align*}
\]

(3)

where the extended-valued function \( I_k : \mathbb{R}^{D+1} \times \mathbb{R} \rightarrow \mathbb{R} \cup \{ +\infty \} \) is defined, for \( 1 \leq k \leq K \), as

\[
I_k (x_k, s_k) = \begin{cases} 
0, & \text{if } (x_k, s_k) = (0, 0) \\
1, & \text{if } (x_k, s_k) \in (X_k - \{ 0 \}) \times \{ 1 \} \\
+\infty, & \text{otherwise}.
\end{cases}
\]

(4)

The variables \( s_k \in \{ 0, 1 \} \) act as switches. When \( s_k = 0 \), the constraint forces \( x_k = 0 \), and thus signal \( y_k \) is not used to explain \( y \). When \( s_k = 1 \), the constraint forces \( x_k \) to be taken from the desired convex domain \( X_k \) (except the origin). Thus, the constraint counts how many signals are used to form \( y \) and restricts this value to be at most \( \bar{T} \). Note that problem (3) is difficult to solve as each indicator function \( I_k \) is nonconvex.

B. Convex relaxation

We attempt to solve the nonconvex problem (3) by replacing it with an approximate convex optimization. We note that the closed, convex hull of the indicator function \( I_k \) is

\[
\mathcal{I}^*_k (x_k, s_k) = \begin{cases} 
s_k, & \text{if } 0 \leq s_k \leq 1 \text{ and } x_k \in s_k X_k \\
+\infty, & \text{otherwise},
\end{cases}
\]

for \( 1 \leq k \leq K \). The proof is in [10]. Since the function \( \sum_{k=1}^{K} I_k \) in the constraint of (3) is separable, we have that its closed, convex hull is given by \( \sum_{k=1}^{K} \mathcal{I}^*_k \). Therefore, by replacing it in (3), we obtain the convex relaxation

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \| y - \sum_{k=1}^{K} A_k x_k \|^2 \\
\text{subject to} & \quad \sum_{k=1}^{K} s_k \leq T, \\
& \quad 0 \leq s_k \leq 1, \quad 1 \leq k \leq K \\
& \quad x_k \in s_k X_k, \quad 1 \leq k \leq K
\end{align*}
\]

(5)

In this formulation, we have \( s_k \in [0, 1] \). Therefore all signals can participate in the formation of \( y \) and the value \( s_k \) can be interpreted as the level of participation of signal \( y_k \).

The last constraint in (5) can usually be made explicit. Some examples are as follows:

- **Box domains**: \( X = B_\infty (R) = \{ x : \| x \|_\infty \leq R \} \), for which the constraint is equivalent to \( \| x_k \| \leq s_k R \);
- **Ellipsoid domains**: \( X = E(c, Q) = \{ x : (x - c)^T Q^{-1} (x - c) \leq 1 \} \), with \( Q \succ 0 \), for which the constraint is the condition \( \| x_k - s_k c \| \leq s_k \). A special case is a ball domain \( B_2 (c, R) = \{ x : \| x - c \| \leq R \} \), corresponding to \( Q = R^2 I \), for which the constraint turns into \( \| x_k - s_k c \| \leq s_k R \).

C. Heuristics

Because problem (3) is a convex relaxation of (2), a solution for (3) may not be feasible for (2). We devise heuristics that find a suboptimal (hopefully good) solution for problem (3) by resorting to the convex relaxation (5).

1) **One-shot heuristic**: Solve (5) to obtain a solution \( (x_k^*, s_k^*) \) for \( 1 \leq k \leq K \). Find \( \mathcal{L} \) by sorting \( \{ s_k^* \} \) in descending order and picking the top \( \bar{T} \) values as the leaders. We then fit \( y \) using only the leader signals and constraining the corresponding filters to their domains (filters’ polishing step):

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \| y - \sum_{k \in \mathcal{L}} A_k x_k \|^2 \\
\text{subject to} & \quad x_k \in X_k, \quad k \in \mathcal{L}
\end{align*}
\]

(6)

The filters \( x_k \), for \( k \notin \mathcal{L} \), are set to zero.

This heuristic constructs the set of leaders based on a single solution and on the idea that the convex relaxation is close to problem (3), by trusting that signals which result in the highest level of participation on (5) correspond to leaders.

In this heuristic, problem (5) and (6) are both solved once.

2) **Iterative Rounding heuristic**: Builds \( \mathcal{L} \) by adding one leader at a time in an iterative process. We start by solving (5) to get a solution \( (x_k^*, s_k^*) \) for \( 1 \leq k \leq K \). Let \( k_1 \) be the index of \( s_k^* \) giving the top value: \( s_k^* = \max \{ s_k^* : k \in \{ 1, \ldots, K \} \} \). We now round \( s_{k_1} \) to 1, that is, we add the constraint \( s_{k_1} = 1 \) to (5) and re-solve it. For simplicity, denote also this next solution as \( (x_k^*, s_k^*) \). Let \( k_2 \) be the index of \( s_k^* \) giving the top value (excluding \( k_1 \)): \( s_{k_2} = \max \{ s_k^* : k \in \{ 1, \ldots, K \} - \{ k_1 \} \} \). We further add \( s_{k_2} = 1 \) to (5) and re-solve it (so, this time, we have two constraints \( s_{k_1} = 1 \) and \( s_{k_2} = 1 \)). This process is continued until \( \bar{T} \) values of \( s_k \) have been rounded up to 1, which then provides the estimate of the set of leaders \( \mathcal{L} = \{ k_1, \ldots, k_\bar{T} \} \). With \( \mathcal{L} \) in hand, we polish the filters by solving (6).

This iterative process of identifying and fixing the leaders, one at a time, is done with the prospect that fixing the leaders guessed so far helps estimate the remaining ones. The hope is that fixing a given leader will remove possible dependencies of \( y \) on other signals of the network that are also formed by that leader.

In this heuristic, problem (5) is solved \( \bar{T} \) times and problem (6) is solved once.

3) **Rollout heuristic**: This strategy is based on the Rollout algorithm [11]. It constructs a solution sequentially by repeatedly applying a base heuristic that gives a feasible solution for the problem as a one-step lookahead. We use as base heuristic the One-shot heuristic since it is the heuristic requiring less computational efforts.

First, we solve \( K \) variations of the One-shot heuristic, where the variation \( k \) contains an added constraint \( s_k = 1 \). As a result, we obtain the optimal values, \( p_k^* \), for variations \( k = 1, \ldots, K \). We then let \( k_1 \) be the index of the smallest \( p_k^* \): \( p_{k_1}^* = \min \{ p_k^* : k \in \{ 1, \ldots, K \} \} \). We now add permanently the constraint \( s_{k_1} = 1 \) to the problem. Next, we solve \( K - 1 \) variations of the One-shot heuristic, one per variation \( k \in \{ 1, \ldots, K \} - \{ k_1 \} \), with the variation \( k \)
containing the constraint $s_k = 1$ (besides $s_{k_1} = 1$). Let the new $K - 1$ optimal values still be denoted by $p_k^*$. We now let $k_2$ be the index of the smallest one, that is, $p_{k_2}^* = \min\{p_k^* : k \in \{1, \ldots, K\} - \{k_1\}\}$, and add permanently the constraint $s_{k_2} = 1$ to the problem (which, at this point, contains two constraints, $s_{k_1}$ and $s_{k_2} = 1$). This process is continued until we have a set of $\mathcal{L}$ leaders, $\hat{\mathcal{L}} = \{k_1, \ldots, k_{\mathcal{L}}\}$.

Because it performs a larger search over the set of possible solutions, the Rollout heuristic improves (or, at worst, matches) the solution obtained by a simple application of the One-shot heuristic. This incurs in added computational complexity, because problem (5) is solved $K + (K - 1) + \ldots + (K - L + 2) + 2$ times and problem (6) is solved $K + (K - 1) + \ldots + (K - L + 1)$ times.

Since the feasible set of (5) is larger than that of (3), the optimal value of (5), say $q^*$, lower bounds the optimal value of (3), say $p^*$. The heuristics search for a possible suboptimal solution of (5). Therefore, their objective value, say $p_H$, is lower bounded as follows: $q^* \leq p^* \leq p_H$. The optimality gap between problems is defined as $p^* - q^*$. As we do not know $p^*$ we can try to evaluate this gap by looking instead at $p_H - q^*$. When $p_H - q^*$ is small, we can be sure that the heuristic is nearly optimal.

D. Adaption of block-sparsity approaches

There are also some available approaches that although not designed to solve problem (3), can be slightly modified to address it and used as a term of comparison to the previously proposed heuristics. We consider the following methods:

- Relaxed Norm (RXN) formulation from (6). It fails to account for the domain constraints when selecting the leaders, so we first apply it to extract a set of leaders with cardinality $\mathcal{L}$ and then solve the polishing step (9).
- Subspace Matching Pursuit (SMP) from [12]. It is a block-sparse extension of OMP. It does not account for the filters’ domains, so we solve (9) with the estimated set of leaders obtained after applying this method.
- Constrained Subspace Matching Pursuit (CSMP). An extension of SMP introduced in the thesis [9] to accommodate the filters’ domains when selecting the leaders.

A more detailed description of the methods and of their adaptation to our problem is provided in [9].

E. Numerical experiments

Next, we compare the performance of the heuristics and the adapted RXN, SMP and CSMP using synthetic tests. We also solve (6) using the true set of leaders. We resort to the following metrics:

- **Objective function value**: objective value of (3). We also show the objective value of the convex relaxation (5).
- **Jaccard Index**: $JI = |\mathcal{L} \cap \hat{\mathcal{L}}|/|\mathcal{L} \cup \hat{\mathcal{L}}|$, where $\mathcal{L}$ and $\hat{\mathcal{L}}$ are the set of true and estimated leaders. If all leaders are correctly determined, $JI$ evaluates to 1.

We perform tests where the filters are restricted to ball domains. We generate the leader filters as $x_k = c_k + u_k \rho_k$, where $c_k$ is randomly sampled from the unit sphere, $u_k$ is an isotropic direction and $\rho_k$ is drawn uniformly from [0, $R$]. We assume the same $R$ for all signals. We generate a network with $K + 1 = 31$ signals, and select one of the followers to be the output $y$. First, $\mathcal{L} = 4$ leader signals are drawn from a standard normal distribution of length $T = 30$. The follower signals are generated following (1) and all signals are then corrupted with additive zero-mean Gaussian noise, to achieve a Signal-to-Noise Ratio (SNR) of 20dB. For the heuristics, we set $\mathcal{L}$ as the true value. In RXN, the parameter controlling the block-sparsity in the solution is set to $\mathcal{L}$. The CSMP methods are also stopped after $\mathcal{L}$ iterations, extracting the corresponding number of leaders. All optimizations are solved with the package CVX from MATLAB in a computer with: Intel(R) Xeon(R) CPU E5-2650 0 @ 2.00GHz processor, 125 GB of RAM and Ubuntu 12.10. The presented values are the average over 100 independent tests.

Fig. 3 and 4 show the objective value and the Jaccard Index, respectively, for different domain sizes, $R$. We do not show the objective values for SMP and RXN as they were much larger than those of the other methods. This was expected, since SMP and RXN do not constrain the filters to their domains when selecting the leaders. Observe, in Fig. 3, that the gap between the convex relaxation and the True Leaders (and the heuristics) objective value is increasing, meaning that we are likely solving a worse approximation of the problem. In fact, the $JI$ of the methods including constraints (heuristics and CSMP), decreases with $R$. However, they largely outperform SMP and RXN, whose $JI$ always stays below 0.25. The Rollout heuristic achieves both the lowest reconstruction error and the largest $JI$, always outperforming CSMP. Comparing with the other heuristics, it is followed by Iterative Rounding and by One-shot. Although CSMP obtains a larger $JI$ than the One-shot and Iterative Rounding heuristics for a larger $R$, its performance is worse for small values.

Note that, although the $JI$ of the Rollout heuristic tends to decrease with $R$, the objective value continues to almost coincide with that of the True Leaders. Seeing that the same
F. Algorithms for the convex relaxation

As seen in the previous section, the Rollout heuristic requires a long computing time. However, we want the proposed heuristics to be competitive not only in terms of signal reconstruction but also in terms of computational requirements. Therefore, we focus on the application of algorithms to solve problem (5) more efficiently than the package CVX.

1) Frank-Wolfe: The Frank-Wolfe (FW) [13] is an iterative method that solves the problem: \( \min_{x \in C} F(x) \), where \( F \) is a convex and continuously differentiable function and \( C \) is a compact convex set. The method, in Algorithm 1 obtains, at each iteration \( t \), a search direction by minimizing a linearization of the function around the previous iterate, \( x^{(t-1)} \). The current solution is then determined by doing a line-search between the previous iterate and the new search atom, \( \hat{x}^{(t)} \).

When solving problem (5), the line-search in Step 4 can be efficiently solved due to the quadratic objective \( F(x) = \frac{1}{2} \| y - Ax \|^2 \). An explicit solution is given by \( \gamma^{(t)} = \max(\min(\gamma_0, 1), 0) \), with

\[
\gamma_0 = \frac{(A(\hat{x}^{(t)} - x^{(t-1)}))^T(y - Ax^{(t-1)})}{\| A(\hat{x}^{(t)} - x^{(t-1)}) \|^2}
\]

The performance of this algorithm depends highly on the capability to efficiently solve Step 3. An example where this is possible is with ellipsoid domains (proved in [9]), for which this step amounts to choose as \( K^* \) the indices of the lowest \( L \) values of

\[
\psi_k = \nabla F_k^T c_k - \frac{\nabla F_k^T Q_k \nabla F_k}{\sqrt{\nabla F_k^T Q_k \nabla F_k}}, \tag{7}
\]

where \( \nabla F_k = (\partial F(x_k))/\partial x_k \), and setting

\[
x_k^* = c_k - \frac{Q_k \nabla F_k}{\sqrt{\nabla F_k^T Q_k \nabla F_k}}, \quad \text{if } k \in K^* \text{ and } \psi_k \leq 0, \tag{8}
\]

\[
x_k^* = 0, \quad \text{otherwise.}
\]

A closed-form solution of Step 3 for box domains is also derived in [9].

2) Fully Corrective Frank-Wolfe: We consider a variant of the FW, the Fully Corrective Frank-Wolfe (FCFW), which was shown to have a faster convergence [14]. This variant is described in Algorithm 2. Instead of performing a line-search, it re-optimizes the objective over the convex hull formed by the current search atom and the previously selected atoms, stored in \( S^{(t)} \) (Step 5). Although this step adds more complexity to

Algorithm 1 FW: Frank-Wolfe

1: Initialize: \( x^{(0)} \in C \)
2: for \( t = 1, 2, \ldots \) do
3: \( \hat{x}^{(t)} \in \text{argmin}_{x \in C} \nabla F(x^{(t-1)})^T x \)
4: \( S^{(t)} \leftarrow \left[ x^{(t-1)} \quad \hat{x}^{(t)} \right] \)
5: \( \alpha^{(t)} \in \text{argmin}_{\alpha} F(S^{(t)} \alpha) \) s.t. \( 1^T \alpha = 1, \alpha \geq 0 \)
6: \( x^{(t)} = \hat{x}^{(t)} \alpha^{(t)} \)
7: end for

Algorithm 2 FCFW: Fully Corrective Frank-Wolfe

1: Initialize: \( x^{(0)} \in C, S^{(0)} = [x^{(0)}] \)
2: for \( t = 1, 2, \ldots \) do
3: \( \hat{x}^{(t)} \in \text{argmin}_{x \in C} \nabla F(x^{(t-1)})^T x \)
4: \( S^{(t)} \leftarrow \left[ S^{(t-1)} \quad \hat{x}^{(t)} \right] \)
5: \( \alpha^{(t)} \in \text{argmin}_{\alpha} F(S^{(t)} \alpha) \) s.t. \( 1^T \alpha = 1, \alpha \geq 0 \)
6: \( x^{(t)} = S^{(t)} \alpha^{(t)} \)
7: end for

each iteration, since it does not have a closed-form solution, the intention is that it will boost the quality of the iterates.

We obtain an estimate for Step 5 using two accelerated projected gradient methods, the Nesterov Accelerated Projected Gradient (NAPG), with the momentum coefficient discussed in [15], and the Spectral Projected Gradient (SPG), with the SPG2 Algorithm from [16]. Both strategies perform a step
where the solution is projected onto the feasible set, which, for Step 5, corresponds to the probability simplex. To alleviate the complexity of the projection, we note that it is instead possible to solve the following problem with some $e > 0$:

$$\begin{align*}
\text{minimize} & \quad \frac{1}{2} \| y^T - A S^{(t)} \|_2^2 + e (1^T \alpha - 1)^2 \\
\text{subject to} & \quad \alpha \geq 0, \quad \alpha^* \text{ by scaling the solution of (9), } \alpha^*, \text{ as } \alpha^* = \frac{\alpha}{2}. \end{align*}$$

and obtain the solution to Step 5, $\alpha^*$, by scaling the solution of (9), $\alpha$, as $\alpha^* = \frac{\alpha}{2}$ (proved in [9]). Now the feasible set amounts simply to the nonnegative orthant.

3) Fully Corrective Frank-Wolfe with Truncation: We analyze another variant of the Frank-Wolfe, denoted by Fully Corrective Frank-Wolfe with Truncation (FCFWT), which introduces a truncation step in FCFW to promote sparsity in the solution. This variant, described in Algorithm 3, is based on the CoGenT method introduced in the context of atomic norm regularization problems in [17]. The algorithm

\begin{algorithm}
\caption{FCFWT: Fully Corrective FW with Truncation}
\begin{algorithmic}[1]
\State Initialize: $x^{(0)} \in \mathcal{C}$, $S^{(0)} = \left[ x^{(0)} \right]$
\For{$t = 1, 2, ...$}
\State $\hat{x}^{(t)} \in \text{argmin}_{x \in \mathcal{C}} \nabla F \left( x^{(t-1)} \right)^T x$
\State $S^{(t)} \leftarrow \left[ S^{(t-1)} \right]$ \quad (proving in $\alpha$)
\State $\tilde{\alpha}^{(t)} \in \text{argmin}_{\alpha} \left\{ F \left( \tilde{S}^{(t)} \right) \alpha \right\}$ \quad (proved in [9])
\State $\tilde{x}^{(t)} = \tilde{S}^{(t)} \tilde{\alpha}^{(t)}$
\State $T_F \leftarrow \mu \cdot \left( x^{(t-1)} \right) + (1 - \mu) F \left( \tilde{x}^{(t)} \right)$
\State $S^{(t)}, \alpha^{(t)}, x^{(t)} = \text{Truncate} \left( S^{(t)}, \alpha^{(t)}, \tilde{x}^{(t)}, T_F \right)$
\EndFor
\end{algorithmic}
\end{algorithm}

starts exactly as the FCFW. The difference is in Step 7 and 8, which corresponds to the Truncation step. For Step 8, we apply the first procedure proposed in [17]. This procedure finds a reduced basis by sequentially selecting the search atom whose removal minimizes the variation in the objective value, removing it from the basis $S^{(t)}$ (and the corresponding coefficient from $\tilde{\alpha}^{(t)}$) and recomputing the solution $\tilde{x}^{(t)}$ as in Step 6. We remove one search atom at a time, while the objective value does not increase beyond the threshold defined by $T_F$. The variation in the objective function at a time, the variation of the objective function value is expressed as $F(\tilde{x}^{(t)} - \alpha \bar{x}) = F(\tilde{x}^{(t)}) - \alpha \bar{x}\nabla F(\tilde{x}^{(t)})^T \bar{x} + \frac{1}{2} \alpha^2 ||A \bar{x}||^2$.

4) Numerical comparison: Next, we present a numerical comparison between FW, FCFW-NAPG, FCFW-SPG, FCFWT-NAPG and FCFWT-SPG (-NAPG and -SPG denote the methods solving the correction step) and the package CVX when solving problem [5]. We wish to assess if the algorithms are able to reduce the computing time while converging to a sufficiently good solution. We first solve the problem with CVX to obtain a benchmark, $F_{CVX}$, and then run the algorithms until they reach a value within a tolerance, following: $(F(\tilde{x}^{(t)}) - F_{CVX})/F_{CVX} \leq \text{tol.}$ We also specify a maximum number of iterations for each algorithm. It is set to $10^5$ for FW and to $1000$ for the remaining algorithms. For FCFWT, we set $\mu = 0.5$ [17]. NAPG and SPG are stopped using a maximum number of iterations, set to 500, and a tolerance on the relative error of the objective value between iterations, set to $10^{-7}$.

The parameters used for these algorithms are detailed in [9]. In all the experiments, the initialization is the zero vector. The code was developed in MATLAB and the tests were carried out in a computer with: Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz processor, 6.0 GB of RAM and Ubuntu 16.04.5 LTS. The performance is evaluated in terms of the running time, in seconds, and the NMSE($\%$) = $\| x_{CVX} - x_{FW} \|^2 / \| x_{CVX} \|^2 \times 100$, where $x_{CVX}$ is the CVX solution and $x_{FW}$ the solution estimated by the FW algorithms. We perform experiments as the size of the network, $K$, increases. We generate the data as described in Section III-E, with $K$ signals of length $T = K$. The number of leaders $L$ is defined as the closest integer to $0.05K$. The delay is set to $D = 2$ and the signals are corrupted with noise using a SNR=20dB. We consider ellipsoid domains for the filters. We start by uniformly generating a point inside the euclidean ball, as described in Section III-E. The filter is the obtained by mapping the previous point to the domain of the ellipsoid defined by $Q_k$ and $c_k$, where $Q_k$ is randomly generated and scaled so that its semi-major axis has a length of 0.5.

Fig. 5 presents the running times for $\text{tol} = 0.05$, averaged over 10 tests. We first observe that FW is slower than CVX. In fact, we verified that for $K \geq 250$, FW was not able to converge in the maximum number of iterations. The FW variants are essential to achieve a sufficiently low run time. Observe that FCFWT is always faster than FCFW, showing the benefit of the truncation step. For FCFWT, SPG is slightly faster than NAPG. For FCFW, the opposite is observed.

Table IV presents the NMSE for different network sizes. FCFW and FCFWT achieve similar results, with a NMSE between 0.98% and 1.58%. Although FW is slower, its NMSE is slightly smaller, being always inferior to 0.51%.

IV. ALL FOLLOWERS WITH THE SAME SET OF LEADERS

In this section, we address the general leader-follower problem. We wish to estimate a unique set of leaders for all signals and the coefficients representing the leader-follower relationships. The problem is now coupled since the signals have to jointly agree on who are the leaders.
Table II: NMSE(%) as a function of the network size, K, for the FW algorithms, with tol = 0.05. FW always has the smallest NMSE

<table>
<thead>
<tr>
<th>K</th>
<th>50</th>
<th>150</th>
<th>250</th>
<th>350</th>
</tr>
</thead>
<tbody>
<tr>
<td>FW</td>
<td>0.23</td>
<td>0.13</td>
<td>0.25</td>
<td>0.51</td>
</tr>
<tr>
<td>FCFW-NAPG</td>
<td>1.21</td>
<td>1.35</td>
<td>0.98</td>
<td>0.77</td>
</tr>
<tr>
<td>FCFW-SPG</td>
<td>1.39</td>
<td>1.30</td>
<td>1.03</td>
<td>0.98</td>
</tr>
<tr>
<td>FCFWT-NAPG</td>
<td>1.58</td>
<td>1.30</td>
<td>1.09</td>
<td>1.00</td>
</tr>
<tr>
<td>FCFWT-SPG</td>
<td>1.13</td>
<td>1.39</td>
<td>1.08</td>
<td>0.98</td>
</tr>
</tbody>
</table>

A. Problem formulation

We propose the following formulation for limiting the cardinality of the shared set of leaders by \( L \):

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \sum_{n=1}^{N} \| y_n - \sum_{m=1}^{N} A_{m} x_{m,n} \|^2 \\
\text{subject to} & \quad \sum_{m=1}^{N} \max(n_{mn} (x_{m,n}, s_{m,n})) \leq L,
\end{align*}
\]

with \( n_{mn} \) defined in (3). Similarly to the formulation (3), the integer variable \( s_{m,n} \) controls the value of \( x_{m,n} \), restraining it to \( X_{m,n} \), if \( y_m \) leads \( y_n \), or to 0, otherwise. If at least one signal is led by \( y_m \), the maximum in the constraint is 1 and \( m \) is a leader. Therefore, the constraint limits the number of leaders to \( L \). To provide an explanation for the leaders in terms of themselves, the domains \( X_{m,n} \), for \( m = n \), should be set as the singleton \( [1 0]^T \). Note also that the objective of (10) can be rewritten in matrix form as \( F(X) = \frac{1}{2} \| Y - AX \|^2_F \), with \( Y = [y_1 \ldots y_N], A = [A_1 \ldots A_N] \) and

\[
X = \begin{bmatrix} x_{11} & \cdots & x_{1N} \\
\vdots & \ddots & \vdots \\
x_{N1} & \cdots & x_{NN} \end{bmatrix}
\]

Due to the nonconvex indicator functions, problem (10) is hard to solve. Therefore, we devise methods that estimate a single set of leaders, \( \hat{L} \), by solving easier, approximate problems. For this, we introduce two types of strategies: decoupled methods, which start by solving a separate problem for each signal and then join their individual information, and coupled methods, which handle a single problem that includes the whole network. With \( \hat{L} \) in hand, a feasible solution to (10) is obtained by polishing the filters, as follows:

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \sum_{n=1}^{N} \| y_n - \sum_{m \in \hat{L}} A_m x_{m,n} \|^2 \\
\text{subject to} & \quad x_{m,n} \in X_{m,n}, \ m \in \hat{L}, \ n \in \hat{F},
\end{align*}
\]

where \( \hat{F} = N - \hat{L} \). The filters \( x_{m,n} \), for \( m \in \hat{F} \), are set to zero. This two-step process is illustrated in Fig. 6.

B. Decoupled approaches

The decoupled approaches start by independently estimating an individual set of leaders, \( \hat{L}_n \), with \( |\hat{L}_n| = L \), for each signal \( y_n \). This is the problem studied in Section III and hence we can capitalize on any of the methods proposed there.

To fuse all individual sets of leaders and obtain a single set for the whole network, we propose a voting scheme. The final leaders are selected as the \( \hat{L} \) signals that were chosen more times as individual leaders.

Therefore, we start by relying on the individual information of each signal, merging it only in the voting process with the expectation that the true leaders will predominate. Note that the individual sets of leaders extracted for the true leaders, which do not have any meaning, also enter in the voting process, as we do not know beforehand which signals correspond to leaders and followers. However, the number of leaders is much smaller than the number of followers. Thus, we expect the votes from the true leaders to not have any noticeable influence in the results.

C. Coupled approaches

1) Convex relaxation: We introduce methods to find a sub-optimal (if not optimal) solution to problem (10) by resorting to a coupled convex optimization solving an approximate problem. We propose the following relaxation:

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \sum_{n=1}^{N} \| y_n - \sum_{m=1}^{N} A_{m} x_{m,n} \|^2 \\
\text{subject to} & \quad \sum_{m=1}^{N} \max(s_{mn}) \leq \hat{L} \\
& \quad 0 \leq s_{mn} \leq 1, \ 1 \leq m \leq N, \ 1 \leq n \leq N \\
& \quad x_{mn} \in X_{m,n}, \ 1 \leq m \leq N, \ 1 \leq n \leq N, \ (12)
\end{align*}
\]

Since the variables \( s_{mn} \) are now continuous, any signal can participate in the explanation of any other signal. However, the value of \( s_{mn} \) can be interpreted as the level of influence signal \( y_m \) has on signal \( y_n \), thus providing valuable information.

2) Coupled heuristics: Although the solution of (12) is, in general, not feasible for (10), it can be used to estimate the set of leaders. To do this, we extend the heuristics introduced in Section III-C to the coupled case. They follow the same procedure as before but the selection of the leaders is now based on the maximum level of participation of the vector \( s_m = [s_{m1} \cdots s_{mN}]^T \). We append the prefix “Coupled” to the next heuristics to indicate that they jointly approach the problem with all network signals.

a) Coupled One-shot: Obtains a solution \( \{x_{mn}^*, s_{mn}^*\} \) by solving (12). The set \( \hat{L} \) is determined by choosing as leaders the indices of the top \( \hat{L} \) values of \( \{\max_m(s_{mn})\}_{m=1}^N \).

b) Coupled Iterative Rounding: Solves problem (12) \( \hat{L} \) times, while sequentially choosing the leaders, one at a time, and fixing them. In each iteration \( l \), we obtain a solution \( \{x_{mn}^*, s_{mn}^*\} \) to (12). The \( l \)th leader is chosen as the index, \( m_l \), corresponding to the top value: \( m_l = \max \{\max_m(s_{mn}^*) : m \in \{1, \ldots, N\} - \{m_1, \ldots, m_{l-1}\} \} \). In the next step, we solve (12) while rounding the variables corresponding to the index \( m_l \) to 1, that is, with the added constraints \( s_{mn} = 1, n \notin \{m_1, \ldots, m_{l-1}\} \) the variables for the current leaders are not rounded since they should not be represented in terms of other leaders). The constraints imposed in the previous iterations for the newly selected leader, \( s_{mmn} = 1 \) for \( m = \{m_1, \ldots, m_{l-1}\} \), are also released. The final set of leaders is \( \hat{L} = \{m_1, \ldots, m_{\hat{L}}\} \).

c) Coupled Rollout: Proceeds by at each iteration \( l \), solving \( N + 1 - l \) variations of the problem with the Coupled One-shot heuristic, one per variation \( m \in \{1, \ldots, N\} - \{m_1, \ldots, m_{l-1}\} \). The variation \( m \), which tests \( m \) as the
D. Numerical experiments

We analyze the performance of the previous methods using synthetic tests. We denote the decoupled strategies by the method being applied to extract the individual sets of leaders. For each decoupled method, there is a coupled counterpart extending it to solve the problem with all signals. Therefore, we use D- and C- to distinguish between the decoupled and coupled versions, respectively. The heuristics are compared with the RXN and CSMP methods. We do not present the results for the SMP methods since we concluded in Section III-E that CSMP always had a superior performance. Although the decoupled heuristics could have been solved with the faster Frank-Wolfe algorithms introduced in Section III-F, we opted to solve all problems using CVX so that a fair comparison could be made without the interference of stopping criteria. We run 100 Monte Carlos and average the results. We consider a network with $N = 30$ signals. We first generate $L = 4$ leader signals from a standard normal distribution, with $T = 30$. For the follower signals we apply model (1), with a delay $D = 2$. All signals are corrupted with additive Gaussian noise to achieve a SNR=20dB and each matrix $A_m$ is built with the noisy signal $y_m$. We consider ball domains for the filters, generating them as described in Section III-E, except for $m = n$, where we impose the domain $\mathcal{X}_{nm} = \{[1 0_D^T]^T\}$. The methods’ parameters are set as in Section III-E.

Fig. 7 displays the objective function value of (11) as a function of the domain size. The True Leaders curve corresponds to the objective value obtained by solving (11) with the true set of leaders. We do not show the values for the RXN methods, since they obtained a reconstruction error much larger than the remaining methods. This was anticipated since they disregard the domain constraints when selecting the leaders.

We observe that all methods have a similar performance until $R = 0.5$. For larger sizes, both CSMP methods present a smaller objective value than most of the heuristics. However, they are always outperformed by the Coupled Rollout heuristic, which achieves the smallest reconstruction error. Comparing the coupled and decoupled approaches, the CSMP, Rollout and Iterative Rounding perform better in the coupled version. On the other hand, the One-shot and RXN reconstruct better the signals when applied in a decoupled fashion.

Fig. 8 presents the Jaccard Index. The results are considerably improved over the single output case (see Fig. 4), with the JI holding larger values for larger domain sizes. This exhibits the advantage in identifying a set of leaders using...
all network signals. Note also that we no longer observe the phenomenon verified when explaining a single signal (in Section III-E), where although the JI decreased for some methods, the objective value remained as good as that of the True Leaders. In this case, the evolution of the JI and of the reconstruction error match, indicating that performing the inference using several output signals tends to remove the ambiguity in the solution.

E. Algorithms for solving the convex relaxation

The efficiency of the decoupled heuristics can be improved by solving 5 with the Frank-Wolfe algorithms from Section III-F. However, that is not the case for the coupled methods. To make the coupled heuristics competitive in terms of time, we extend the FW algorithms to solve the coupled convex relaxation 12, thereby dropping the package CVX. Although the algorithms can be almost directly applied to this case (with matrices as variables), there are some differences that need to be considered and which we explain next.

1) Frank-Wolfe: The Frank-Wolfe (Algorithm 1) can be extended to the matrix case by solving the minimization of the linear approximation of the objective (Step 3) using the trace inner product, as follows: $\min_{X \in C} \text{tr} \left( \nabla F(X^{(t-1)})^T X \right)$.

In this case, an explicit expression for the line-search in Step 4 is given by $\gamma^{(t)} = \max(\min(\gamma_0, 1), 0)$, with

$$\gamma_0 = \frac{\text{tr}( (A(\hat{X}^{(t)} - X^{(t-1)})^T Y - A X^{(t-1)}) )}{\|A(\hat{X}^{(t)} - X^{(t-1)})\|^2_F}.$$ 

When considering ellipsoid domains for the filters $x_{mn}$, with $m \neq n$, and the singleton domain $x_{mn} = \{ x_0 \} = \{ [1 0]^T \}$, for $m = n$, Step 3 can be efficiently solved (as proved in 9). This is done by choosing for $\mathcal{M}^*$ the indices of the lowest $\mathcal{L}$ values of $\{ \sum_{n \neq m} (\min(\psi_{mn}, 0)) + \min(\nabla F_{mm} x_0, 0) \}_{m=1}^N$, with $\psi_{mn}$ defined in 7. The optimal values of the filters for $m \neq n$ are given by $\mathcal{K}^*$, with $\mathcal{K}^* = \mathcal{M}^*$. For $m = n$, we have $x_{mm} = x_0$, if $m \in \mathcal{M}^*$ and $\nabla F_{mm} x_0 \leq 0$, or $x_{mm} = 0$, otherwise.

2) Fully Corrective Frank-Wolfe: When applying the FCFWT to the coupled problem, we have that the minimization over the convex hull of the previously selected atoms $\{ \hat{X}^{(j)} \}_{j=0}^t$ corresponds to

$$\text{minimize}_{\{ \alpha_j : 0 \leq j \leq t \}} \frac{1}{2} \| Y - A \sum_{j=0}^t \hat{X}^{(j)} \alpha_j \|^2_F,$$

subject to $\alpha_j \geq 0$, $0 \leq j \leq t$.

The objective of (13) is equivalent to $f(\alpha_0, \ldots, \alpha_t) = \frac{1}{2} \| \text{vec}(Y) - \sum_{j=0}^t \text{vec}(A \hat{X}^{(j)}) \alpha_j \|^2$, where $\text{vec}(Z)$ vectorizes matrix $Z$. Thus, problem (13) can be rewritten as the problem in Step 5 of Algorithm 2 with $y = \text{vec}(Y)$ and $A = [\text{vec}(A \hat{X}^{(1)}) \cdots \text{vec}(A \hat{X}^{(t)})]$, and we can apply the NAPG and SPG methods to solve it, as discussed in Section III-F.

3) Fully Corrective Frank-Wolfe with Truncation: The variation of the objective after removing an atom in the Truncation step (Step 8 of Algorithm 3) now amounts to $F(\hat{X}^{(t)} - \alpha \hat{X}) = F(\hat{X}^{(t)}) - \alpha \hat{X} \text{tr}(\nabla F(\hat{X}^{(t)}))^T \hat{X} + \frac{1}{2} \alpha^2 \hat{X}^T F(\hat{X}^{(t)})^T \hat{X}$. 

4) Numerical comparison: In this section we report numerical experiments to analyze the performance of the FW algorithms when solving problem 12. We follow the same procedure as in Section III-F4 where we run the algorithms until their objective value falls below a tolerance of the objective value of CVX, $(F(X^{(t)}) - F_{\text{CVX}})/F_{\text{CVX}} \leq \text{tol}$. We compare the methods as the size of the problem grows, by varying the network size $N$. The values we report for all experiments are the average over 10 independent trials. In each trial, a network with the following parameters is used: $\mathcal{L} = \text{round}(0.05 N)$, $T = N$, $D = 2$ and SNR=20dB. We consider ellipsoid domains, generating the groundtruth filters as described in Section III-F4, except for the filters $x_{mn}$, with $m = n$, for which we impose the singleton $x_{mn} = \{ [1 0]^T \}$.

The maximum number of iterations is set as $10^6$ for FW and 5000 for the remaining algorithms. For SPG and NAPG, the maximum number of iterations is set to 100 and the threshold on the relative error of the objective value to $10^{-7}$. For FCFWT, we set $\mu = 0.5$. All algorithms are initialized with the zero matrix. The experiments were carried out in a computer with: Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz processor, 6.0 GB of RAM and Ubuntu 16.04.5 LTS.
The methods are analyzed in terms of the running time, in seconds, and the NMSE(%) = ||X_{Cvx} − X_{FW}||_F^2 / ||X_{Cvx}||_F^2 × 100. The results are respectively displayed in Table III and IV for tol = 0.05. We start by observing that the algorithms are always two orders of magnitude faster than CVX. The running time of FCFW is always smaller than that of FW, but in this case, there is not an advantage in applying the Truncation step. Using SPG tends to be faster than using NAPG, with FCFW-SPG consistently providing the best results. Although FW tends to be slower, it always obtains the smallest NMSE. Nevertheless, both FCFW and FCFWT obtain accurate solutions, with a NMSE always inferior to 1%.

### References