Fast Graph Filters for Decentralized Subspace Projection

Daniel Romero, *Member, IEEE,* Siavash Mollaebrahim, *Student Member, IEEE,*
Baltasar Beferull-Lozano, *Senior Member, IEEE*, and César Asensio-Marco, *Member, IEEE.*

***Abstract*—A number of inference problems with sensor networks involve projecting a measured signal onto a given subspace. In existing decentralized approaches, sensors communicate with their local neighbors to obtain a sequence of iterates that asymptotically converges to the desired projection. In contrast, the present paper develops methods that produce these projections in a finite and approximately minimal number of iterations. Building upon tools from graph signal processing, the problem is cast as the design of a graph filter which, in turn, is reduced to the design of a suitable graph shift operator. Exploiting the eigenstructure of the projection and shift matrices leads to an objective whose minimization yields approximately minimum-order graph filters. To cope with the fact that this problem is not convex, the present work introduces a novel convex relaxation of the number of distinct eigenvalues of a matrix based on the nuclear norm of a Kronecker difference. To tackle the case where there exists no graph filter capable of implementing a certain subspace projection with a given network topology, a second optimization criterion is presented to approximate the desired projection while trading the number of iterations for approximation error. Two algorithms are proposed to optimize the aforementioned criteria based on the alternating-direction method of multipliers. An exhaustive simulation study demonstrates that the obtained filters can effectively obtain subspace projections markedly faster than existing algorithms.**

***Keywords*—Subspace projection, graph filters, graph signal processing, decentralized signal processing, wireless sensor networks.**

I. Introduction

A frequent inference problem in signal processing involves the estimation of a spatial field using measurements collected by a (possibly wireless) sensor network [2]–[6]. The field of interest may quantify magnitudes such as temperature, electromagnetic radiation, concentration of airborne or liquid pollutants, flows of gas or liquids in porous soils and rocks such as oil reservoirs, acoustic pressure, and radioactivity to name a few. Instead of spatial fields, one may be alternatively interested in fields defined on the nodes or edges of a network; see e.g. [7]. In either case, a number of common inference tasks such as least squares estimation, denoising, (weighted) consensus, and decentralized detection can be cast as projecting the observations onto a given signal subspace; see e.g. [8]–[10]. Such a fundamental task can be implemented in a centralized fashion, where a fusion center gathers and processes the measurements collected by all sensors. Unfortunately, this approach gives rise to i) communication bottlenecks, since those nodes near the fusion center are required to forward data packets from many sensors, ii) computational challenges, since the load is concentrated in the fusion center, and iii) vulnerability to attacks or failure of the fusion center. For these reasons, the decentralized paradigm, where there is no central processor and all nodes share the computational load, is oftentimes preferred [11]. These implementations are therefore scalable, robust, and balance the communication and processing requirements across nodes. The present paper capitalizes on the notion of graph filter [12], [13] to develop algorithms for computing projections in a decentralized fashion with an approximately minimal number of iterations.

To tackle problems involving decentralized processing, it is common to define a *communication graph* where each node represents a sensor and there exists an edge between two nodes if the corresponding sensors can communicate, e.g. via a radio link. One could therefore feel inclined to address the problem at hand using standard inference tools for data defined on graphs; see e.g. [14]–[19]. However, these methods are based on exploiting a certain relation between the data and the graph; e.g. smoothness [16]. Therefore, this framework fundamentally differs from the one at hand since here the graph only provides information about how the sensors communicate, i.e., it does not generally provide information about the spatial field of interest.[1] To obtain a decentralized subspace projection algorithm, one could instead adopt a decentralized optimization standpoint, e.g. via the *distributed least mean squares* (DLMS) method in [20], based on the *alternating direction method of multipliers* (ADMM) [21], [22], or the *decentralized gradient descent* (DGD) method in [23], which builds upon gradient descent. Although these algorithms can accommodate general objective functions, their convergence is only asymptotic and can be significantly improved by exploiting the structure of the subspace projection problem; see Sec. V. For this reason,

Daniel Romero is with the Department of Information and Communication Technology, University of Agder, Norway. Siavash Mollaebrahim and Baltasar Beferull-Lozano are with the Intelligent Signal Processing and Wireless Networks (WISENET) Center, University of Agder, Norway. Cesar Asensio is with the AINIA Technology Center, Spain. e-mail:{daniel.romero, siavash.mollaebrahim, baltasar.beferull}@uia.no, ceamar@gmail.com

This work was supported in part by the PETROMAKS Smart-Rig grant 244205/E30 and the SFI Offshore Mechatronics grant 237896/O30 from the Research Council of Norway.

Part of this work was presented in the Int. Conf. on Acoustics, Speech, and Signal Processing, Calgary, Canada, 2018 [1].

[1] If the field is smooth over space and the graph is also smooth over space, meaning that nodes that are spatially close have a low geodesic distance in the graph, then both notions may approximately coincide. However, even in this case, exploiting spatial information would still be more accurate. Fortunately, as described in this paper, graph signal processing tools can still be applied to exploit spatial smoothness rather than graph smoothness.

a method tailored to computing projections in a decentralized fashion was proposed in [24], later extended in [25] and [26], where every node obtains each iterate by linearly combining its previous iterate with the previous iterate of its neighbors. The combination weights are adjusted to achieve a fast asymptotic convergence. The main strength of this approach is its simplicity, since each node simply repeats the same operation over and over. The price to be paid is that convergence is asymptotic, which means that a large number of data packages need to be exchanged to attain a prescribed projection accuracy. In addition, these algorithms can only accommodate a limited set of topologies [27]. A special case of the subspace projection problem is average consensus, where the signal subspace comprises the vectors whose entries are all equal. For this special case, the algorithms in [28], [29] produce projections with a finite number of communication rounds. A graph signal processing [12], [13] perspective to tackle this special case was adopted in [30]. Unfortunately, these schemes cannot be applied to the general subspace projection problem. A more general framework is proposed in [31], which would allow implementation of a projection with a graph filter if one were given a *shift matrix* such that a subset of its eigenvectors spans the subspace of interest. Unfortunately, this framework does not include any method to find such a shift matrix in a general case unless the target subspace is of dimension 1. An even more general setup is presented in [32], which approximates an arbitrary linear transformation at the expense of more complex node computations through the notion of "edge-variant" graph filters. Unfortunately, the non-convex [33] nature of the optimization problem involved therein yields no guarantees that a projection filter can be found even if it exists and may lead to unpredictable behavior if the network topology is modified and the filter weights need to be updated. Besides, even in the unlikely event that the optimization algorithm finds a global optimum, the resulting filter is not necessarily implementable in a small number of iterations. Further graph-filter design schemes abound, but they typically seek implementing a given frequency response [34]–[36] relative to a given shift matrix.

To sum up, there is no decentralized algorithm for computing general subspace projections in a finite number of iterations. The present paper fills this gap by suitably designing a shift matrix and the graph filter coefficients. The sought filter is of approximately minimal order, which implies that the number of data exchanges among nodes is approximately minimized. The minimal order is seen to depend on the multiplicity of the eigenvalues of the shift matrix. Since maximizing this multiplicity would lead to a non-convex problem, a novel convex relaxation technique is developed relying on a nuclear norm functional of the shift matrix. To solve the resulting optimization problem, a solver based on ADMM is also developed. For those scenarios where there exists no graph filter that can implement the desired projection on the given topology, a method is proposed to approximate such a projection while trading approximation error for filter order. Another ADMM solver is developed for this case.

The conference precursor [1] of this work contains Theorem 1 and the key steps leading to (P1-R). Most of the analysis, simulations, the approximate projection method, and the ADMM solvers are presented here anew. We also published a related subgradient method in [37] but it is not contained in the present manuscript.

The paper is structured as follows. Sec. II formulates the problem, reviews common applications, and lies some background on graph filters. Secs. III and IV respectively propose methods for exact and approximate projection implementation. Finally, Sec. V presents the simulations and Sec. VI summarizes the main conclusions and provides a discussion. One proof and the derivations of the ADMM methods are provided in the supplementary material.

Notation: Symbol $:=$ denotes equality by definition. For sets $\mathcal{A}$ and $\mathcal{B}$, the cardinality of $\mathcal{A}$ is denoted as $|\mathcal{A}|$ whereas $\mathcal{A} \subsetneq \mathcal{B}$ indicates that $\mathcal{A}$ is a proper subset of $\mathcal{B}$. Boldface lowercase (uppercase) letters represent column vectors (matrices). The ℓ_n norm of vector $\boldsymbol{v}$ is denoted as $\|\boldsymbol{v}\|_n$. With $\boldsymbol{A}$ and $\boldsymbol{B}$ matrices of appropriate dimensions, $[\boldsymbol{A};\boldsymbol{B}]$ and $[\boldsymbol{A},\boldsymbol{B}]$ respectively denote their vertical and horizontal concatenation, $\boldsymbol{A}^\top$ the transpose of $\boldsymbol{A}$, $\mathsf{cols}(\boldsymbol{A})$ the set of the columns of $\boldsymbol{A}$, $\mathrm{diag}(\boldsymbol{A})$ a vector comprising the diagonal entries of $\boldsymbol{A}$, $\mathcal{R}(\boldsymbol{A})$ the span of the columns of $\boldsymbol{A}$, $\boldsymbol{A}\otimes\boldsymbol{B}$ the Kronecker product of $\boldsymbol{A}$ and $\boldsymbol{B}$, $\mathsf{evals}(\boldsymbol{A})$ the set of eigenvalues of $\boldsymbol{A}$, $\lambda_i(\boldsymbol{A})$ the i-th largest eigenvalue of $\boldsymbol{A}$, $\sigma_i(\boldsymbol{A})$ the i-th largest singular value of $\boldsymbol{A}$, $||\boldsymbol{A}||_2 := \sigma_1(\boldsymbol{A})$ the 2-norm of $\boldsymbol{A}$, and $||\boldsymbol{A}||_\star := \sum_i \sigma_i(\boldsymbol{A})$ the nuclear norm of $\boldsymbol{A}$. For a subspace $\mathcal{A}$, notation $\mathcal{A}^\perp$ represents the orthogonal complement. Finally, $\mathbb{E}$ denotes expectation and $\mathcal{N}$ the normal distribution.

II. Preliminaries

A. The Subspace Projection Problem

Let $\mathcal{G}(\mathcal{V},\mathcal{E})$ denote a graph with vertex set $\mathcal{V}=\{1,\ldots,N\}$, where each vertex corresponds to a sensor or *node*, and edge set $\mathcal{E}\subset\mathcal{V}^2$. Let there be an edge (n,n') in $\mathcal{E}$ if and only if (iff) the nodes n and n' can communicate directly, e.g. through their radio interface. Thus, it is natural to assume (i) that $\mathcal{E}$ contains all self loops, i.e., $(n,n)\in\mathcal{E}\ \forall n\in\mathcal{V}$, and (ii) that $\mathcal{G}$ is undirected, which means that $(n,n')\in\mathcal{E}$ implies that $(n',n)\in\mathcal{E}$. The neighborhood of the n-th node is defined as $\mathcal{N}_n=\{n' \mid (n,n')\in\mathcal{E}\}$.

Given $\boldsymbol{z}=[z_1,\ldots,z_N]^\top$, where $z_n\in\mathbb{R}$ denotes the observation or measurement acquired by the n-th node, the goal is to estimate the signal vector $\boldsymbol{\xi}\in\mathbb{R}^N$, which quantifies the phenomenon of interest (e.g. temperature field). The latter is related to $\boldsymbol{z}$ via

$$\boldsymbol{z}=\boldsymbol{\xi}+\boldsymbol{v}, \tag{1}$$

where $\boldsymbol{v}\in\mathbb{R}^N$ stands for additive noise. Vector $\boldsymbol{\xi}$ is known to lie in a given subspace $\mathcal{R}\{\boldsymbol{U}_\parallel\}$ of dimension $r<N$, where the columns of $\boldsymbol{U}_\parallel\in\mathbb{R}^{N\times r}$ are assumed orthonormal without loss of generality (w.l.o.g.). Hence, $\boldsymbol{\xi}$ can be expressed as $\boldsymbol{\xi}=\boldsymbol{U}_\parallel\boldsymbol{\alpha}$ for some $\boldsymbol{\alpha}\in\mathbb{R}^r$.

The orthogonal projection of $\boldsymbol{z}$ onto $\mathcal{R}\{\boldsymbol{U}_\parallel\}$, also known as the least-squares estimate[2] of $\boldsymbol{\xi}$, is given by:

$$\hat{\boldsymbol{\xi}} := \boldsymbol{U}_\parallel\boldsymbol{U}_\parallel^\top\boldsymbol{z} \triangleq \boldsymbol{P}\boldsymbol{z}\,, \tag{2}$$

[2]Also the *best linear unbiased estimator*, *minimum variance unbiased estimator*, and *maximum likelihood estimator* [8] under appropriate assumptions.

where $\boldsymbol{P} \in \mathbb{R}^{N\times N}$ is the projection matrix onto $\mathcal{R}\{\boldsymbol{U}_\parallel\}$. The subspace projection problem is to find $\hat{\boldsymbol{\xi}}$ given $\boldsymbol{z}$ and $\boldsymbol{U}_\parallel$. Vector $\hat{\boldsymbol{\xi}}$ is expected to be a better estimate of $\boldsymbol{\xi}$ than $\boldsymbol{z}$ since the noise is annihilated along $N-r$ dimensions.

B. The Choice of the Basis

This section discusses specific choices of the basis $\mathcal{U} := \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r\}$ formed by the columns of $\boldsymbol{U}_\parallel$ in different application scenarios where a spatial field needs to be monitored. To this end, let $\boldsymbol{x}_n \in \mathbb{R}^d$ denote the spatial location of the n-th sensor, where $d = 2$ or 3. Similarly, let $\boldsymbol{x} \in \mathbb{R}^d$ denote an arbitrary location in the area of interest. Suppose that the goal is to estimate a spatial field $\xi : \mathbb{R}^d \to \mathbb{R}$ given the measurements in (1), where $\boldsymbol{\xi} := [\xi(\boldsymbol{x}_1), \ldots, \xi(\boldsymbol{x}_N)]^\top$.

Oftentimes, the physics of the problem directly provides a linear parametric expansion for ξ. For example, in the case of a diffusion field, such as a temperature field, one has

$$\xi(\boldsymbol{x}) = \sum_{i=1}^{r} \frac{\exp\left\{-\|\boldsymbol{x} - \boldsymbol{x}_{\mathrm{s},i}\|_2^2/(2\sigma_i^2)\right\}}{2\pi\sigma_i^2}\tilde{\alpha}_i \tag{3}$$

for some coefficients $\tilde{\alpha}_i$, where $\boldsymbol{x}_{\mathrm{s},i}$ is the location of the i-th source and the parameters $\{\sigma_i^2\}_{i=1}^r$ are related to the diffusivity of the medium. In some cases governed by a wave equation, as occurs in wireless communications (see e.g. [38]), ξ may admit an expansion in terms of Cauchy bells [24]:

$$\xi(\boldsymbol{x}) = \sum_{i=1}^{r} \frac{1}{1 + \|\boldsymbol{x} - \boldsymbol{x}_{\mathrm{s},i}\|_2^2/\sigma_i^2}\tilde{\alpha}_i. \tag{4}$$

To obtain $\mathcal{U}$, evaluate (3) or (4) at the sensor locations and collect the coefficients that multiply each $\tilde{\alpha}_i$ to form the vector $\tilde{\boldsymbol{u}}_i \in \mathbb{R}^N$, $i = 1, \ldots, r$. This yields the expansion $\boldsymbol{\xi} = \sum_{i=1}^r \tilde{\boldsymbol{u}}_i\tilde{\alpha}_i$. Finally, orthonormalize $\{\tilde{\boldsymbol{u}}_i\}_{i=1}^r$.

This approach applies when ξ satisfies a parametric expansion as in (3) or (4) and this expansion is known. However, it is often the case that the form of the expansion is known but it contains unknown parameters, the form of the expansion is not even known, or the field does not even admit a linear expansion but it approximately does. In these situations, one may still pursue a linear inference approach by capitalizing on some form of smoothness that the target field exhibits across space. For instance, ξ can be approximately bandlimited, which means that ξ can be reasonably approximated by a reduced number r of Fourier or *discrete cosine transform* (DCT) basis functions. In the latter case, upon letting $\boldsymbol{x} := [x_1, x_2]^\top$, one can write

$$\xi(\boldsymbol{x}) \approx \sum_{i_1=0}^{r_1-1}\sum_{i_2=0}^{r_2-1} \alpha_{i_1,i_2} \times \cos\left(\frac{\pi}{X_1}i_1\left(x_1 + \frac{1}{2}\right)\right)\cos\left(\frac{\pi}{X_2}i_2\left(x_2 + \frac{1}{2}\right)\right). \tag{5}$$

Here, X_1 and X_2 denote the length along the 1st and 2nd dimensions of the region where ξ is defined. The vector $\boldsymbol{\alpha}$ defined in Sec. II-A can be recovered by stacking the $r_1 r_2$ coefficients $\{\alpha_{i_1,i_2}\}$, whereas $\mathcal{U}$ can be found as described earlier in this section.

Besides Fourier or DCT bases, one may pursue approximations based on any other collection of basis functions such as conventional polynomials, discrete prolate spheroidal functions, and wavelets. Note that in any approximation of this kind there is a fundamental variance-bias trade-off; see e.g. [39, Ch. 3.4]. To see this, note that the *signal-to-noise ratio* after projection in the model (1) is given by $\|\boldsymbol{P}\boldsymbol{\xi}\|_2^2/\mathbb{E}[\|\boldsymbol{P}\boldsymbol{v}\|_2^2]$. If $\boldsymbol{v}$ has zero mean and covariance matrix $\mathbb{E}[\boldsymbol{v}\boldsymbol{v}^\top] = \sigma^2\boldsymbol{I}_N$, then $\mathbb{E}[\|\boldsymbol{P}\boldsymbol{v}\|_2^2] = \mathbb{E}[\|\boldsymbol{U}_\parallel\boldsymbol{U}_\parallel^\top\boldsymbol{v}\|_2^2] = \sigma^2\,\mathrm{Tr}[\boldsymbol{P}] = \sigma^2 r$. Thus, although a basis with a larger r may capture more signal energy $\|\boldsymbol{P}\boldsymbol{\xi}\|_2^2$, the power of the noise component in $\hat{\boldsymbol{\xi}}$ is also increased.

C. Graph Filters

This section briefly reviews the notion of graph filters [12], [13], which constitute a central part of the proposed algorithms. In this context, vector $\boldsymbol{z} := [z_1, \ldots, z_N]^\top$ is referred to as a *graph signal*, which emphasizes the fact that the entry z_n is stored at the n-th node.

A graph filter involves two steps, as described next. In the first step, a finite sequence of graph signals $\{\boldsymbol{z}^{(l)}\}_{l=0}^L$, where $\boldsymbol{z}^{(l)} := [z_1^{(l)}, \ldots, z_N^{(l)}]^\top$, is collaboratively obtained by the network through a sequence of L *local data exchange rounds*, or just *local exchanges* for short, where $\boldsymbol{z}^{(0)} := \boldsymbol{z}$ is the graph signal to filter. At the l-th round, each node sends its $z_n^{(l-1)}$ to its neighbors and computes a linear combination of the entries $\{z_{n'}^{(l-1)}\}_{n'\in\mathcal{N}_n}$ that it receives from them. Specifically, the next graph signal is obtained as $z_n^{(l)} = \sum_{n'\in\mathcal{N}_n} s_{n,n'}z_{n'}^{(l-1)}$, $n = 1, \ldots, N$, where $s_{n,n'}$ is the coefficient corresponding to the linear aggregation that takes place between nodes n and n'. By letting $s_{n,n'} = 0$ whenever $n' \notin \mathcal{N}_n$, one can equivalently write $z_n^{(l)} = \sum_{n'=1}^N s_{n,n'}z_{n'}^{(l-1)}$ or, in matrix form, $\boldsymbol{z}^{(l)} = \boldsymbol{S}\boldsymbol{z}^{(l-1)}$, where $\boldsymbol{S} \in \mathbb{R}^{N\times N}$ is given by $(\boldsymbol{S})_{n,n'} = s_{n,n'}$, $n, n' = 1, \ldots, N$. In the graph signal processing literature, the matrix $\boldsymbol{S}$ is usually referred to as *shift matrix* [31]. More generally, an $N \times N$ matrix $\boldsymbol{S}$ is said to be a shift matrix over the graph $\mathcal{G} := (\mathcal{V}, \mathcal{E})$ if $(\boldsymbol{S})_{n,n'} = 0$ for all $(n, n') \notin \mathcal{E}$. The set of all possible shift matrices over $\mathcal{G}$ will be denoted as $\mathcal{S}_\mathcal{G}$. Examples of matrices in $\mathcal{S}_\mathcal{G}$ include the adjacency and Laplacian matrices of $\mathcal{G}$ [31]. Associated with the shift matrix is the *shift operator*, defined as the function $\boldsymbol{z} \mapsto \boldsymbol{S}\boldsymbol{z}$. Notice that, upon recursively applying $\boldsymbol{z}^{(l)} = \boldsymbol{S}\boldsymbol{z}^{(l-1)}$, one can write[3] $\boldsymbol{z}^{(l)} = \boldsymbol{S}^l\boldsymbol{z}$, $l = 0, \ldots, L$.

In the second step of the graph filter, all nodes linearly combine the iterates in the first step. Specifically, the following graph signal is computed:

$$\boldsymbol{y} = \sum_{l=0}^{L} c_l\boldsymbol{z}^{(l)} = \sum_{l=0}^{L} c_l\boldsymbol{S}^l\boldsymbol{z} \tag{6}$$

where $c_l \in \mathbb{R}$, $l = 0, \ldots, L$, are the so-called filter coefficients.

[3]Throughout the paper, $\boldsymbol{A}^0$ for a square matrix $\boldsymbol{A}$ denotes the identity matrix of the same size as $\boldsymbol{A}$, regardless of whether $\boldsymbol{A}$ is invertible.

The operation in (6) can be generically expressed as $\boldsymbol{z} \mapsto \boldsymbol{H}\boldsymbol{z}$, where

$$\boldsymbol{H} := \sum_{l=0}^{L} c_l \boldsymbol{S}^l, \tag{7}$$

and is commonly referred to as an order-L *graph filter*. An important implication of the Cayley-Hamilton Theorem [40] is that for any order-L graph filter $\boldsymbol{H}$ with $L \geq N$, there exists an order-$(N-1)$ graph filter $\boldsymbol{H}'$ with shift matrix $\boldsymbol{S}$ and coefficients c_l' such that $\boldsymbol{H} = \boldsymbol{H}'$. This establishes an upper bound on the order and, therefore, the number of local exchanges required to apply a graph filter. Thus, one can assume w.l.o.g. that $L \leq N-1$.

D. Asymptotic Decentralized Projections

A decentralized scheme for subspace projection was proposed in [24]. There, a matrix $\boldsymbol{S}$ is found such that (i) $\boldsymbol{S} \in \mathcal{S}_{\mathcal{G}}$ and (ii) $\lim_{l\to\infty} \boldsymbol{S}^l = \boldsymbol{P}$. Then, the nodes compute the sequence $\{\boldsymbol{z}^{(l)}, l = 0, 1, \ldots\}$, where $\boldsymbol{z}^{(l)} = \boldsymbol{S}^l\boldsymbol{z}$. This constitutes the infinite counterpart of the first step in a graph filter; cf. Sec. II-C. Due to (ii), it follows that $\lim_{l\to\infty} \boldsymbol{z}^{(l)} = \boldsymbol{P}\boldsymbol{z}$, as desired. The main strength of this method is its simplicity, since each node just needs to store one coefficient for each neighbor and the same operation is repeated over and over. A limitation is that the number of local exchanges required to attain a target error $||\boldsymbol{z}^{(l)} - \boldsymbol{P}\boldsymbol{z}||$ is generally high since this approach only provides asymptotic convergence. Furthermore, the set of graphs for which (i) and (ii) can be simultaneously satisfied is considerably limited; see Sec. V and [27].

III. Exact Projection Filters

This section proposes an algorithm to find a graph filter that yields a subspace projection in an approximately minimal number of iterations. To this end, Secs. III-A and III-B formalize the problem and characterize the set of feasible shift matrices for a given $\mathcal{E}$ and $\boldsymbol{U}_{\parallel}$. Subsequent sections introduce an optimization methodology to approximately minimize the order of the filter, i.e. the number of communication steps needed to obtain the projection via graph filtering.

A. Minimum-order Projection Filters

To solve the subspace projection problem formulated in Sec. II-A with a graph filter, one could think of finding a shift matrix $\boldsymbol{S} \in \mathcal{S}_{\mathcal{G}}$ and a set of coefficients $\{c_l\}_{l=0}^{L}$ such that $\boldsymbol{P}\boldsymbol{z} = \sum_{l=0}^{L} c_l \boldsymbol{S}^l\boldsymbol{z}$ for all $\boldsymbol{z} \in \mathbb{R}^N$ or, equivalently, such that $\boldsymbol{P} = \sum_{l=0}^{L} c_l \boldsymbol{S}^l$. Since $\boldsymbol{P}$ is symmetric, it will be assumed that $\boldsymbol{S}$ is also symmetric. To assist in this quest, consider the following definition:

Definition 1: Let $\boldsymbol{Q} \in \mathbb{R}^{N\times N}$ be an arbitrary (not necessarily a projection) matrix. A symmetric matrix (not necessarily a shift matrix) $\boldsymbol{S} \in \mathbb{R}^{N\times N}$ is polynomially feasible to implement the operator $\boldsymbol{z} \mapsto \boldsymbol{Q}\boldsymbol{z}$ if there exist L and $\boldsymbol{c} := [c_0, \ldots, c_L]^\top$ such that $\sum_{l=0}^{L} c_l \boldsymbol{S}^l = \boldsymbol{Q}$.

For a given $\boldsymbol{Q}$, the set of all polynomially feasible matrices $\boldsymbol{S} \in \mathbb{R}^{N\times N}$ will be denoted as $\mathcal{F}_{\boldsymbol{Q}}$. Except for the tip, this set is a cone since $\boldsymbol{S} \in \mathcal{F}_{\boldsymbol{Q}}$ implies $\kappa\boldsymbol{S} \in \mathcal{F}_{\boldsymbol{Q}}$ for all $\kappa \neq 0$.

Note that given a matrix in $\mathcal{F}_{\boldsymbol{Q}}$, it is straightforward to obtain $\boldsymbol{c}$ such that $\sum_{l=0}^{L} c_l \boldsymbol{S}^l = \boldsymbol{Q}$; see e.g. [31] and Sec. III-B. One may then consider the following *feasibility problem*: Given a graph $\mathcal{G} := (\mathcal{V}, \mathcal{E})$ and a projection matrix $\boldsymbol{P}$, find a polynomially feasible shift matrix, i.e., find $\boldsymbol{S} \in \mathcal{S}_{\mathcal{G}} \cap \mathcal{F}_{\boldsymbol{P}}$. When this problem admits a solution, we say that *there exists an exact projection filter* for implementing $\boldsymbol{P}$ on $\mathcal{G}$. Whether this is the case depends on $\boldsymbol{P}$ and $\mathcal{G}$. For example, if $\mathcal{G}$ is too sparse, then $\boldsymbol{P}$ will not be computable as a graph filter. In the extreme case where $\mathcal{G}$ is fully disconnected, then the only computable projection is $\boldsymbol{P} = \boldsymbol{I}_N$. Conversely, when $\mathcal{G}$ is fully connected, then all projections can be computed as a graph filter. Furthermore, given that $\mathcal{S}_{\mathcal{G}}$ is a subspace and that $\boldsymbol{S} \in \mathcal{F}_{\boldsymbol{Q}}$ implies $\kappa\boldsymbol{S} \in \mathcal{F}_{\boldsymbol{Q}}$ for all $\kappa \neq 0$, it is easy to see that such a feasibility problem either has no solution or has infinitely many.

When feasible shifts exist, it is reasonable to seek the $\boldsymbol{S}$ that minimizes the number of local exchanges L. For arbitrary matrices $\boldsymbol{S}$ and $\boldsymbol{Q}$, define $\mathcal{O}_{\boldsymbol{Q}}(\boldsymbol{S})$ as the minimum L such that $\boldsymbol{Q} = \sum_{l=0}^{L} c_l \boldsymbol{S}^l$ for some $\{c_l\}_{l=0}^{L}$. In view of the bound dictated by the Cayley-Hamilton Theorem (Sec. II-C), $\mathcal{O}_{\boldsymbol{Q}}(\boldsymbol{S})$ can be viewed as a function $\mathcal{O}_{\boldsymbol{Q}} : \mathcal{F}_{\boldsymbol{Q}} \to \{0, \ldots, N-1\}$. Given a projection matrix $\boldsymbol{P}$, the problem of finding the shift matrix associated with the minimum-order filter can therefore be formulated as:

$$\text{(P1)} \qquad \underset{\boldsymbol{S}}{\text{minimize}} \quad \mathcal{O}_{\boldsymbol{P}}(\boldsymbol{S}) \qquad \text{s.t.} \quad \boldsymbol{S} \in \mathcal{S}_{\mathcal{G}} \cap \mathcal{F}_{\boldsymbol{P}}.$$

Conversely, when $\mathcal{S}_{\mathcal{G}} \cap \mathcal{F}_{\boldsymbol{P}} = \varnothing$, there exists no graph filter capable of implementing $\boldsymbol{P}$. For those cases, Sec. IV describes how to find a graph filter that approximates $\boldsymbol{P}$.

B. Polynomially Feasible Matrices

To assist in solving (P1), this section presents an algebraic characterization of the set $\mathcal{F}_{\boldsymbol{P}}$ of polynomially feasible matrices. Recall that the matrices in this set need not be shift matrices, that is, they need not satisfy the topology constraints determined by the edge set $\mathcal{E}$.

Lemma 1: Let $\boldsymbol{U}_{\parallel} \in \mathbb{R}^{N\times r}$ with orthonormal columns be given and let $\boldsymbol{P} = \boldsymbol{U}_{\parallel}\boldsymbol{U}_{\parallel}^\top$. Let also $\boldsymbol{U}_{\perp} \in \mathbb{R}^{N\times N-r}$ with orthonormal columns satisfy $\mathcal{R}(\boldsymbol{U}_{\perp}) = \mathcal{R}^{\perp}(\boldsymbol{U}_{\parallel})$. If $\boldsymbol{S} \in \mathcal{F}_{\boldsymbol{P}}$, then there exist symmetric matrices $\boldsymbol{F}_{\parallel} \in \mathbb{R}^{r\times r}$ and $\boldsymbol{F}_{\perp} \in \mathbb{R}^{N-r\times N-r}$ such that:

$$\boldsymbol{S} = \begin{bmatrix}\boldsymbol{U}_{\parallel} & \boldsymbol{U}_{\perp}\end{bmatrix} \begin{bmatrix}\boldsymbol{F}_{\parallel} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{F}_{\perp}\end{bmatrix} \begin{bmatrix}\boldsymbol{U}_{\parallel}^\top \\ \boldsymbol{U}_{\perp}^\top\end{bmatrix}. \tag{8}$$

Proof: See Appendix A. ∎

Note that matrices $\boldsymbol{F}_{\parallel}$ and $\boldsymbol{F}_{\perp}$ satisfying (8) exist regardless[4] of the choice of $\boldsymbol{U}_{\parallel}$ and $\boldsymbol{U}_{\perp}$ as long as the columns of

[4]Note that the algorithms in this paper produce the same filters for all matrices $\boldsymbol{U}_{\parallel}$ that span a given signal subspace $\mathcal{R}\{\boldsymbol{U}_{\parallel}\}$; likewise for $\boldsymbol{U}_{\perp}$. Thus, the obtained filters only depend on the signal subspace and not on the specific choice of the basis. This property is what bypasses the difficulty encountered in [31, eq. (20)], which will typically be infeasible for graphs with more than N missing edges.

$\boldsymbol{U}_\parallel$ and $\boldsymbol{U}_\perp$ respectively constitute an orthonormal basis for the signal subspace $\mathcal{R}\{\boldsymbol{U}_\parallel\}$ and its orthogonal complement $\mathcal{R}^\perp(\boldsymbol{U}_\parallel)$. As seen later, the converse of Lemma 1 does not hold.

To understand the implications of Lemma 1, rewrite (8) as

$$\boldsymbol{S} = \boldsymbol{U}_\parallel \boldsymbol{F}_\parallel \boldsymbol{U}_\parallel^\top + \boldsymbol{U}_\perp \boldsymbol{F}_\perp \boldsymbol{U}_\perp^\top = \boldsymbol{S}_\parallel + \boldsymbol{S}_\perp, \tag{9}$$

where $\boldsymbol{S}_\parallel := \boldsymbol{U}_\parallel \boldsymbol{F}_\parallel \boldsymbol{U}_\parallel^\top$ and $\boldsymbol{S}_\perp := \boldsymbol{U}_\perp \boldsymbol{F}_\perp \boldsymbol{U}_\perp^\top$ are symmetric matrices whose column spans are respectively contained in the signal subspace and its orthogonal complement. Thus, they clearly satisfy $\boldsymbol{S}_\parallel^\top \boldsymbol{S}_\perp = \boldsymbol{S}_\perp^\top \boldsymbol{S}_\parallel = \boldsymbol{0}$. Note that even if $\boldsymbol{S}$ is a shift matrix, i.e. $\boldsymbol{S} \in \mathcal{S}_\mathcal{G}$, matrices $\boldsymbol{S}_\parallel$ and $\boldsymbol{S}_\perp$ may not be shift matrices. Consider now the eigendecompositions $\boldsymbol{F}_\parallel = \boldsymbol{Q}_\parallel \boldsymbol{\Lambda}_\parallel \boldsymbol{Q}_\parallel^\top, \boldsymbol{F}_\perp = \boldsymbol{Q}_\perp \boldsymbol{\Lambda}_\perp \boldsymbol{Q}_\perp^\top$ for orthogonal $\boldsymbol{Q}_\parallel \in \mathbb{R}^{r\times r}, \boldsymbol{Q}_\perp \in \mathbb{R}^{N-r\times N-r}$ and diagonal $\boldsymbol{\Lambda}_\parallel \in \mathbb{R}^{r\times r}, \boldsymbol{\Lambda}_\perp \in \mathbb{R}^{N-r\times N-r}$. Then, (8) can be rewritten as:

$$\boldsymbol{S} = \begin{bmatrix}\boldsymbol{U}_\parallel \boldsymbol{Q}_\parallel & \boldsymbol{U}_\perp \boldsymbol{Q}_\perp\end{bmatrix} \begin{bmatrix}\boldsymbol{\Lambda}_\parallel & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Lambda}_\perp\end{bmatrix} \begin{bmatrix}\boldsymbol{Q}_\parallel^\top \boldsymbol{U}_\parallel^\top \\ \boldsymbol{Q}_\perp^\top \boldsymbol{U}_\perp^\top\end{bmatrix}. \tag{10}$$

This is clearly an eigenvalue decomposition of $\boldsymbol{S}$. It further shows that $\text{evals}(\boldsymbol{S}) = \text{evals}(\boldsymbol{F}_\parallel) \cup \text{evals}(\boldsymbol{F}_\perp)$. In view of (10), Lemma 1 establishes that any $\boldsymbol{S} \in \mathcal{F}_\boldsymbol{P}$ has exactly r orthogonal eigenvectors (the columns of $\boldsymbol{U}_\parallel \boldsymbol{Q}_\parallel$) in the signal subspace and $N-r$ (the columns of $\boldsymbol{U}_\perp \boldsymbol{Q}_\perp$) in its orthogonal complement.

The following definition builds upon Lemma 1 to introduce a necessary condition for feasibility of a shift matrix that will prove instrumental in subsequent sections.

Definition 2: Let $\boldsymbol{U}_\parallel$ and $\boldsymbol{U}_\perp$ be given. If $\boldsymbol{S} \in \mathbb{R}^{N\times N}$ is such that it satisfies (8) for some symmetric $\boldsymbol{F}_\parallel \in \mathbb{R}^{r\times r}$ and $\boldsymbol{F}_\perp \in \mathbb{R}^{N-r\times N-r}$, then $\boldsymbol{S}$ is said to be pre-feasible.

Note again that this definition is independent of the choice of $\boldsymbol{U}_\parallel$ and $\boldsymbol{U}_\perp$ so long as their columns respectively form a basis for the signal subspace and its orthogonal complement. Given $\boldsymbol{P} \in \mathbb{R}^{N\times N}$, the set of all pre-feasible matrices in $\mathbb{R}^{N\times N}$ will be denoted as $\tilde{\mathcal{F}}_\boldsymbol{P}$.

Observe that all matrices in $\mathcal{F}_\boldsymbol{P}$ are also in $\tilde{\mathcal{F}}_\boldsymbol{P}$. However, not all matrices in $\tilde{\mathcal{F}}_\boldsymbol{P}$ are in $\mathcal{F}_\boldsymbol{P}$. Trivial examples include $\boldsymbol{S} = \boldsymbol{I}_N$ (recall that $r < N$) and $\boldsymbol{S} = \boldsymbol{0}$. The rest of this section will characterize the matrices in $\tilde{\mathcal{F}}_\boldsymbol{P}$ that are in $\mathcal{F}_\boldsymbol{P}$. In particular, it will be seen that any pre-feasible matrix $\boldsymbol{S}$ where $\boldsymbol{\Lambda}_\parallel$ and $\boldsymbol{\Lambda}_\perp$ share at least an eigenvalue is not polynomially feasible. To this end, note from Definition 1 and (10) that any pre-feasible matrix must satisfy

$$\boldsymbol{P} = \boldsymbol{U}_\parallel \boldsymbol{Q}_\parallel \Big[\sum_{l=0}^{L} c_l \boldsymbol{\Lambda}_\parallel^l\Big] \boldsymbol{Q}_\parallel^\top \boldsymbol{U}_\parallel^\top + \boldsymbol{U}_\perp \boldsymbol{Q}_\perp \Big[\sum_{l=0}^{L} c_l \boldsymbol{\Lambda}_\perp^l\Big] \boldsymbol{Q}_\perp^\top \boldsymbol{U}_\perp^\top \tag{11}$$

for some $\{c_l\}_{l=0}^L$ to be polynomially feasible. Multiplying both sides of (11) on the left by $\boldsymbol{U}_\parallel^\top$ and on the right by $\boldsymbol{U}_\parallel$, it follows that $\boldsymbol{Q}_\parallel \big[\sum_{l=0}^L c_l\ \boldsymbol{\Lambda}_\parallel^l\big] \boldsymbol{Q}_\parallel^\top = \boldsymbol{I}_r$ or, equivalently, $\sum_{l=0}^L c_l\ \boldsymbol{\Lambda}_\parallel^l = \boldsymbol{I}_r$. Likewise, multiplying both sides of (11) on the left by $\boldsymbol{U}_\perp^\top$ and on the right by $\boldsymbol{U}_\perp$, it follows that $\sum_{l=0}^L c_l\ \boldsymbol{\Lambda}_\perp^l = \boldsymbol{0}$. Arranging these two conditions in matrix form yields:

$$\begin{bmatrix}\boldsymbol{1}_r \\ \boldsymbol{0}_{N-r}\end{bmatrix} = \begin{bmatrix}1 & \lambda_1 & \dots & \lambda_1^L \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_r & \dots & \lambda_r^L \\ 1 & \lambda_{r+1} & \dots & \lambda_{r+1}^L \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_N & \dots & \lambda_N^L\end{bmatrix} \begin{bmatrix}c_0 \\ c_1 \\ \vdots \\ c_L\end{bmatrix}, \tag{12}$$

where $\lambda_1, \dots, \lambda_N$ are such that $\boldsymbol{\Lambda}_\parallel \triangleq \text{diag}\{\lambda_1, \dots, \lambda_r\}$ and $\boldsymbol{\Lambda}_\perp \triangleq \text{diag}\{\lambda_{r+1}, \dots, \lambda_N\}$. Vandermonde systems such as (12) frequently arise when designing graph filters; see e.g. [30], [31]. With the appropriate definitions, it can be expressed in matrix form as:

$$\boldsymbol{\lambda}_\boldsymbol{P} = \boldsymbol{\Psi} \boldsymbol{c}, \tag{13}$$

which provides a means to obtain the coefficients $\{c_l\}_{l=0}^L$ when (13) admits a solution.

To understand when the latter is the case, assume w.l.o.g. that $L = N-1$ since the existence of a solution to (12) for some L implies its existence for $L = N-1$. Since $\boldsymbol{\Psi}$ is a square Vandermonde matrix, any two rows corresponding to distinct eigenvalues are linearly independent. Looking at the left-hand side of (12), it is easy to see that the system (12) admits a solution iff $\boldsymbol{\Lambda}_\parallel$ and $\boldsymbol{\Lambda}_\perp$ do not share eigenvalues.

This conclusion can be combined with Lemma 1 as follows:

Theorem 1: Let $\boldsymbol{S} \in \mathbb{R}^{N\times N}$ be symmetric. Then, $\boldsymbol{S} \in \mathcal{F}_\boldsymbol{P}$ iff both the following two conditions hold:

(C1) $\boldsymbol{S} \in \tilde{\mathcal{F}}_\boldsymbol{P}$, i.e., it satisfies (8) for some symmetric $\boldsymbol{F}_\perp$ and $\boldsymbol{F}_\parallel$,

(C2) $\text{evals}(\boldsymbol{F}_\parallel) \cap \text{evals}(\boldsymbol{F}_\perp) = \varnothing$.

C. Filter Order Minimization

Theorem 1 implies that (P1) can be reformulated as the minimization of $\mathcal{O}_\boldsymbol{P}(\boldsymbol{S})$ subject to $\boldsymbol{S} \in \mathcal{S}_\mathcal{G}$, (C1), and (C2). This section and the next develop a reformulation more amenable to application of a numerical solver.

The first step is to express the objective function $\mathcal{O}_\boldsymbol{P}(\boldsymbol{S})$ more explicitly. To that end, consider the following result:

Lemma 2: If $\boldsymbol{S} = \boldsymbol{U}_\parallel \boldsymbol{F}_\parallel \boldsymbol{U}_\parallel^\top + \boldsymbol{U}_\perp \boldsymbol{F}_\perp \boldsymbol{U}_\perp^\top \in \mathcal{F}_\boldsymbol{P}$, then

$$\mathcal{O}_\boldsymbol{P}(\boldsymbol{S}) \leq \mathcal{L}(\boldsymbol{F}_\parallel) + \mathcal{L}(\boldsymbol{F}_\perp) - 1, \tag{14}$$

where $\mathcal{L}(\cdot)$ is the number of distinct eigenvalues of its argument.

Proof: Computing $\mathcal{O}_\boldsymbol{P}(\boldsymbol{S})$ for a given $\boldsymbol{S}$ amounts to determining the minimum L for which (12), or equivalently its compact version (13), admits a solution. Since $\boldsymbol{S} \in \mathcal{F}_\boldsymbol{P}$ by hypothesis, one has that (13) is satisfied for at least one value of L. Let L_0 denote the smallest value for which (13) holds. Since $\boldsymbol{\Psi}$ is Vandermonde, it has at most $\mathcal{L}(\boldsymbol{S})$ linearly independent rows, which implies that $L_0 + 1 \leq \mathcal{L}(\boldsymbol{S})$. The proof is completed by noting from Theorem 1 that $\mathcal{L}(\boldsymbol{S}) = \mathcal{L}(\boldsymbol{F}_\parallel) + \mathcal{L}(\boldsymbol{F}_\perp)$. ■

In practice, the bound in (14) will hold with equality unless in degenerate cases. For example, when $N = 4$, $r = 2$, $\lambda_1 = -\lambda_2$ and $\lambda_3 = -\lambda_4 \neq |\lambda_1|$, it can be seen that $\mathcal{O}_{\boldsymbol{P}}(\boldsymbol{S}) = 2$ whereas $\mathcal{L}(\boldsymbol{F}_{\parallel}) + \mathcal{L}(\boldsymbol{F}_{\perp}) - 1 = 3$. However, in general, such an $\boldsymbol{S}$ will only be in $\mathcal{S}_{\mathcal{G}} \cap \mathcal{F}_{\boldsymbol{P}}$ if $\boldsymbol{P}$ and $\mathcal{G}$ are jointly selected to achieve this end, which will not occur in a practical application. In words, Lemma 2 implies that one may seek the shift matrix of an approximately minimal-order projection filter as the matrix with the smallest number of distinct eigenvalues among all matrices in the feasible set of (P1).

Next, this feasible set is rewritten more explicitly. To this end, split the constraint $\boldsymbol{S} \in \mathcal{S}_{\mathcal{G}} \cap \mathcal{F}_{\boldsymbol{P}}$ into the two constraints $\boldsymbol{S} \in \mathcal{S}_{\mathcal{G}}$ and $\boldsymbol{S} \in \mathcal{F}_{\boldsymbol{P}}$. Regarding $\boldsymbol{S} \in \mathcal{S}_{\mathcal{G}}$, recall from the definition of $\mathcal{S}_{\mathcal{G}}$ in Sec. II-C that $\boldsymbol{S} \in \mathcal{S}_{\mathcal{G}}$ iff $(\boldsymbol{S})_{n,n'} = 0$ for all (n, n') such that $(n, n') \notin \mathcal{E}$. Because any $\boldsymbol{S}$ that is feasible for (P1) has to be in $\mathcal{F}_{\boldsymbol{P}}$ and all the matrices in this set are symmetric, any feasible $\boldsymbol{S}$ is necessarily symmetric. Thus, one can just require that $(\boldsymbol{S})_{n,n'} = 0$ only for those (n, n') such that $(n, n') \notin \mathcal{E}$ and $n < n'$. With $\boldsymbol{e}_n$ the n-th column of the identity matrix $\boldsymbol{I}_N$, it follows that $(\boldsymbol{S})_{n,n'} = \boldsymbol{e}_n^{\top} \boldsymbol{S} \boldsymbol{e}_{n'} = (\boldsymbol{e}_{n'} \otimes \boldsymbol{e}_n)^{\top} \mathrm{vec}(\boldsymbol{S})$. Thus, the constraint $\boldsymbol{S} \in \mathcal{S}_{\mathcal{G}}$ can be expressed as $\boldsymbol{W}\mathrm{vec}(\boldsymbol{S}) = \boldsymbol{0}$, where $\boldsymbol{W}$ is a matrix whose rows are given by the vectors $\{(\boldsymbol{e}_{n'} \otimes \boldsymbol{e}_n)^{\top}, \forall (n, n')$ such that $(n, n') \notin \mathcal{E}$ and $n < n'\}$. As expected, the fewer edges in the graph, the more rows $\boldsymbol{W}$ has and, consequently, the smaller the feasible set. In the extreme case of a fully disconnected graph, only the diagonal matrices satisfy $\boldsymbol{S} \in \mathcal{S}_{\mathcal{G}}$ (recall that $\mathcal{E}$ contains all self-loops; cf. Sec. II-A).

On the other hand, the constraint $\boldsymbol{S} \in \mathcal{F}_{\boldsymbol{P}}$ can be easily expressed invoking Theorem 1 and introducing two auxiliary optimization variables $\boldsymbol{F}_{\parallel}$ and $\boldsymbol{F}_{\perp}$.

In view of these observations and Lemma 2, problem (P1) becomes

$$\text{(P1')} \quad \begin{aligned} \underset{\boldsymbol{S}, \boldsymbol{F}_{\parallel}, \boldsymbol{F}_{\perp}}{\text{minimize}} \quad & \mathcal{L}(\boldsymbol{F}_{\parallel}) + \mathcal{L}(\boldsymbol{F}_{\perp}) \\ \text{s.t.} \quad & \boldsymbol{W}\mathrm{vec}(\boldsymbol{S}) = \boldsymbol{0} \\ & \boldsymbol{S} = \boldsymbol{U}_{\parallel} \boldsymbol{F}_{\parallel} \boldsymbol{U}_{\parallel}^{\top} + \boldsymbol{U}_{\perp} \boldsymbol{F}_{\perp} \boldsymbol{U}_{\perp}^{\top} \\ & \boldsymbol{F}_{\parallel} = \boldsymbol{F}_{\parallel}^{\top}, \ \boldsymbol{F}_{\perp} = \boldsymbol{F}_{\perp}^{\top} \\ & \lambda_n(\boldsymbol{F}_{\parallel}) \neq \lambda_{n'}(\boldsymbol{F}_{\perp}) \ \forall n, n' \end{aligned}$$

for an arbitrary choice of $\boldsymbol{U}_{\perp} \in \mathbb{R}^{N \times N-r}$ with orthonormal columns spanning $\mathcal{R}^{\perp}(\boldsymbol{U}_{\parallel})$.

Two further modifications are in order. First, note that (P1') is invariant to scalings in the sense that if $(\boldsymbol{S}, \boldsymbol{F}_{\parallel}, \boldsymbol{F}_{\perp})$ is feasible, then $(\kappa \boldsymbol{S}, \kappa \boldsymbol{F}_{\parallel}, \kappa \boldsymbol{F}_{\perp})$ is also feasible and attains the same objective value $\forall \kappa \neq 0$. For this reason, the constraint $\mathrm{Tr}(\boldsymbol{F}_{\parallel}) = r$ will be introduced w.l.o.g. to eliminate this ambiguity.

Second, the feasible set of (P1') is not a closed set due to the constraint $\lambda_n(\boldsymbol{F}_{\parallel}) \neq \lambda_{n'}(\boldsymbol{F}_{\perp}) \ \forall n, n'$, implying that the optimum may not be attained through an iterative algorithm. In practice, this constraint holds so long as the eigenvalues of $\boldsymbol{F}_{\parallel}$ differ from those of $\boldsymbol{F}_{\perp}$, even if there is an eigenvalue of $\boldsymbol{F}_{\parallel}$ arbitrarily close to an eigenvalue of $\boldsymbol{F}_{\perp}$. But in the latter case, the numerical conditioning of (12) would be poor, implying that the target projection cannot be implemented as a graph filter using finite-precision arithmetic. Thus, the aforementioned constraint should be replaced with another one that ensures that (i) the feasible set is closed, and (ii) the eigenvalues of $\boldsymbol{F}_{\parallel}$ are sufficiently different from those of $\boldsymbol{F}_{\perp}$. One natural possibility is $|\lambda_n(\boldsymbol{F}_{\parallel}) - \lambda_{n'}(\boldsymbol{F}_{\perp})| \geq \epsilon \ \forall n, n'$, where $\epsilon > 0$ is a user-selected parameter. Unfortunately, this constraint is not convex, but an effective relaxation will be presented in the next section.

To sum up, the optimization problem to be solved is:

$$\text{(P1")} \quad \begin{aligned} \underset{\boldsymbol{S}, \boldsymbol{F}_{\parallel}, \boldsymbol{F}_{\perp}}{\text{minimize}} \quad & \mathcal{L}(\boldsymbol{F}_{\parallel}) + \mathcal{L}(\boldsymbol{F}_{\perp}) \\ \text{s.t.} \quad & \boldsymbol{W}\mathrm{vec}(\boldsymbol{S}) = \boldsymbol{0}, \ \mathrm{Tr}(\boldsymbol{F}_{\parallel}) = r \\ & \boldsymbol{S} = \boldsymbol{U}_{\parallel} \boldsymbol{F}_{\parallel} \boldsymbol{U}_{\parallel}^{\top} + \boldsymbol{U}_{\perp} \boldsymbol{F}_{\perp} \boldsymbol{U}_{\perp}^{\top} \\ & \boldsymbol{F}_{\parallel} = \boldsymbol{F}_{\parallel}^{\top}, \ \boldsymbol{F}_{\perp} = \boldsymbol{F}_{\perp}^{\top} \\ & |\lambda_n(\boldsymbol{F}_{\parallel}) - \lambda_{n'}(\boldsymbol{F}_{\perp})| \geq \epsilon \ \forall n, n'. \end{aligned}$$

Remarkably, the set of topologies for which (P1") is feasible is strictly larger than the set of topologies for which the method proposed in [24] (and reviewed in Sec. II-D) can be applied. It can be easily seen that any feasible matrix for the problem in [24] must be in $\mathcal{S}_{\mathcal{G}} \cap \mathcal{F}_{\boldsymbol{P}}$ and, additionally, $N - r$ of its eigenvalues must be equal to 1 whereas the rest must be less than 1. It follows that the feasible set therein is strictly contained in the feasible set of (P1').

D. Exact Projection Filters via Convex Relaxation

Problem (P1") is non-convex because (i) the objective function is not convex and (ii) the last constraint, which does not define a convex set. This section proposes a convex problem to approximate the solution to (P1").

To address (i), recall that $\mathcal{L}(\boldsymbol{F}_{\parallel})$ equals the number of distinct eigenvalues of $\boldsymbol{F}_{\parallel}$ and note that the larger $\mathcal{L}(\boldsymbol{F}_{\parallel})$, the larger the number of non-zero elements of the vector $[\lambda_1 - \lambda_2, \lambda_1 - \lambda_3, \ldots, \lambda_1 - \lambda_r, \lambda_2 - \lambda_3, \ldots, \lambda_{r-1} - \lambda_r]^{\top}$. A similar observation applies to $\mathcal{L}(\boldsymbol{F}_{\perp})$. This suggests replacing the objective in (P1") with

$$\sum_{n=1}^{r} \sum_{n'=1}^{r} ||\lambda_n(\boldsymbol{F}_{\parallel}) - \lambda_{n'}(\boldsymbol{F}_{\parallel})||_0 + \sum_{n=1}^{N-r} \sum_{n'=1}^{N-r} ||\lambda_n(\boldsymbol{F}_{\perp}) - \lambda_{n'}(\boldsymbol{F}_{\perp})||_0, \tag{15}$$

where $\|\boldsymbol{x}\|_0$ is the so-called *zero norm* or number of non-zero elements of vector $\boldsymbol{x}$. A typical convex surrogate of the zero norm is the l_1-norm [41]. However, just replacing $||\cdot||_0$ in (15) with an l_1-norm would still give rise to a non-convex objective since it involves the functions $\lambda_n(\cdot)$. One of the key ideas in this paper is to apply the following result:

Lemma 3: Let $\boldsymbol{A}$ be an $N \times N$ matrix with eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_N$. Then,

$$\|\boldsymbol{A} \otimes \boldsymbol{I}_N - \boldsymbol{I}_N \otimes \boldsymbol{A}\|_* = \sum_{n=1}^{N} \sum_{n'=1}^{N} |\lambda_n - \lambda_{n'}|. \tag{16}$$

Proof: See Appendix B. ∎

Applying Lemma 3 to (15) suggests replacing the objective in (P1") with $||\boldsymbol{F}_{\parallel} \otimes \boldsymbol{I}_r - \boldsymbol{I}_r \otimes \boldsymbol{F}_{\parallel}||_{\star} + ||\boldsymbol{F}_{\perp} \otimes \boldsymbol{I}_{N-r} - \boldsymbol{I}_{N-r} \otimes$

$\boldsymbol{F}_\perp||_\star$. The implications of this relaxation are further discussed in Appendix D.

Regarding (ii), note that constraint $|\lambda_n(\boldsymbol{F}_\parallel) - \lambda_{n'}(\boldsymbol{F}_\perp)| \geq \epsilon \ \forall n, n'$ renders the feasible set non-convex due to (ii-1) the functions $\lambda_n(\cdot)$ and (ii-2) the absolute value. To deal with (ii-1), a sensible approach is to relax the constraints $|\lambda_n(\boldsymbol{F}_\parallel) - \lambda_{n'}(\boldsymbol{F}_\perp)| \geq \epsilon \ \forall n, n'$ into a single constraint which requires only that the average of the eigenvalues of $\boldsymbol{F}_\parallel$ must differ sufficiently from the average of the eigenvalues of $\boldsymbol{F}_\perp$, that is

$$\left| \frac{1}{r} \sum_{n=1}^{r} \lambda_n(\boldsymbol{F}_\parallel) - \frac{1}{N-r} \sum_{n=1}^{N-r} \lambda_n(\boldsymbol{F}_\perp) \right| \geq \epsilon$$

for some user-selected $\epsilon > 0$. Clearly, this constraint is equivalent to $\left|\mathrm{Tr}(\boldsymbol{F}_\parallel)/r - \mathrm{Tr}(\boldsymbol{F}_\perp)/(N-r)\right| \geq \epsilon$ and, using the constraint $\mathrm{Tr}(\boldsymbol{F}_\perp) = r$ introduced in (P1"), becomes equivalent to $|1 - \mathrm{Tr}(\boldsymbol{F}_\perp)/(N-r)| \geq \epsilon$. To deal with (ii-2), one can now use the definition of absolute value to conclude that this constraint is satisfied if either $\mathrm{Tr}(\boldsymbol{F}_\perp) \leq (1-\epsilon)(N-r)$ or $\mathrm{Tr}(\boldsymbol{F}_\perp) \geq (1+\epsilon)(N-r)$. Both of these inequalities are affine and therefore convex, however the OR condition renders the feasible set non-convex. However, a solution can be readily found by first solving the problem with the constraint $\mathrm{Tr}(\boldsymbol{F}_\perp) \leq (1-\epsilon)(N-r)$, then solving it with the constraint $\mathrm{Tr}(\boldsymbol{F}_\perp) \geq (1+\epsilon)(N-r)$ instead, and finally comparing the objective values achieved at the optimum of both these sub-problems. To simplify the task of solving the resulting problem, note that the solution in both subproblems will satisfy the corresponding constraint with equality. To see this, note that if the inequality constraint is removed, the optimum of the resulting convex problem becomes $(\boldsymbol{S}, \boldsymbol{F}_\parallel, \boldsymbol{F}_\perp) = (\boldsymbol{I}_N, \boldsymbol{I}_r, \boldsymbol{I}_{N-r})$. Due to convexity and since this optimum does not satisfy the removed constraint, such a constraint will become necessarily active when introduced.

In view of these observations, the last constraint in (P1") will be replaced with $\mathrm{Tr}(\boldsymbol{F}_\perp) = (1 \pm \epsilon)(N-r)$, where the $\pm$ sign indicates that both subproblems must be solved separately. Except for degenerate cases, one expects that these relaxed constraints will suffice to ensure that $\boldsymbol{F}_\perp$ and $\boldsymbol{F}_\parallel$ do not share eigenvalues. However, although this was indeed observed in our experiments, sometimes an eigenvalue of $\boldsymbol{F}_\perp$ may become sufficiently close to an eigenvalue of $\boldsymbol{F}_\parallel$ in such a way that the Vandermonde system (12) becomes poorly conditioned. To alleviate this situation, a penalty proportional to $||\boldsymbol{F}_\perp||_{\mathrm{F}}^2$ will be added to the objective to "push" the eigenvalues of $\boldsymbol{F}_\perp$ towards zero.

With all these considerations, the resulting relaxed problem becomes:

$$\begin{aligned}
\underset{\boldsymbol{S}, \boldsymbol{F}_\parallel, \boldsymbol{F}_\perp}{\text{minimize}} \quad & \eta_\parallel ||\boldsymbol{F}_\parallel \otimes \boldsymbol{I}_r - \boldsymbol{I}_r \otimes \boldsymbol{F}_\parallel||_\star \\
& + \eta_\perp ||\boldsymbol{F}_\perp \otimes \boldsymbol{I}_{N-r} - \boldsymbol{I}_{N-r} \otimes \boldsymbol{F}_\perp||_\star + ||\boldsymbol{F}_\perp||_{\mathrm{F}}^2 \\
\text{s.t.} \quad & \boldsymbol{W}\mathrm{vec}(\boldsymbol{S}) = \boldsymbol{0}, \\
\text{(P1-R)} \quad & \boldsymbol{S} = \boldsymbol{U}_\parallel \boldsymbol{F}_\parallel \boldsymbol{U}_\parallel^\top + \boldsymbol{U}_\perp \boldsymbol{F}_\perp \boldsymbol{U}_\perp^\top, \\
& \boldsymbol{F}_\parallel = \boldsymbol{F}_\parallel^\top, \ \boldsymbol{F}_\perp = \boldsymbol{F}_\perp^\top, \\
& \mathrm{Tr}(\boldsymbol{F}_\perp) = (1 \pm \epsilon)(N-r), \mathrm{Tr}(\boldsymbol{F}_\parallel) = r,
\end{aligned}$$

where $\eta_\parallel > 0$ and $\eta_\perp > 0$ are user-specified parameters to control the relative weight of each term in the objective. Having two parameters rather than just one that scales the term $||\boldsymbol{F}_\perp||_{\mathrm{F}}^2$ empowers the user with more flexibility to alleviate possible numerical issues, as described earlier. A study of the ability of (P1-R) to approximate the solution of (P1) is presented in Appendix D.

An iterative solver for (P1-R) is proposed in Appendix E in the supplementary material based on ADMM, therefore inheriting its solid convergence guarantees.

IV. Approximate Projection Filters

When the graph is too sparse for (P1) to admit a solution, one may instead seek a low-order graph filter that approximates $\boldsymbol{P}$ reasonably well. Before presenting an optimization criterion to obtain the corresponding shift matrix, the next section analyzes in which situations there exists an exact projection filter.

A. Feasibility of Exact Projection Filters

The following result provides a necessary condition for the existence of an exact projection shift matrix. Let $\bar{\mathcal{E}} := \{(n,n') \in \mathcal{E} : \ n \leq n'\} = \{(n_1, n_1'), \ldots, (n_{\bar{E}}, n_{\bar{E}}')\}$ denote the reduced edge set, where each of the $\bar{E}$ undirected edges shows up only once. Consider also the symmetric shift matrices $\boldsymbol{\Phi}_i := \boldsymbol{e}_{n_i}\boldsymbol{e}_{n_i'}^\top + \boldsymbol{e}_{n_i'}\boldsymbol{e}_{n_i}^\top, \ i = 1, \ldots, \bar{E}$, where only a single edge is used. Clearly, all feasible shift matrices are linear combinations of these matrices.

Theorem 2: Let $\boldsymbol{\Phi} := [\mathrm{vec}(\boldsymbol{\Phi}_1), \ldots, \mathrm{vec}(\boldsymbol{\Phi}_{\bar{E}})]$. It holds that

$$\dim(\mathcal{S}_\mathcal{G} \cap \tilde{\mathcal{F}}_{\boldsymbol{P}}) = \bar{E} - \mathrm{rank}((\boldsymbol{U}_\parallel^\top \otimes \boldsymbol{U}_\perp^\top)\boldsymbol{\Phi}). \tag{17}$$

Proof: See Appendix C. ∎

Due to the presence of the scaled identity matrices in $\mathcal{S}_\mathcal{G} \cap \tilde{\mathcal{F}}_{\boldsymbol{P}}$ (see Sec. III-B), a necessary condition for a feasible projection filter to exist, i.e. $\mathcal{S}_\mathcal{G} \cap \mathcal{F}_{\boldsymbol{P}} \neq \varnothing$, is that $\dim(\mathcal{S}_\mathcal{G} \cap \tilde{\mathcal{F}}_{\boldsymbol{P}}) > 1$. From (17), this condition becomes $\mathrm{rank}((\boldsymbol{U}_\parallel^\top \otimes \boldsymbol{U}_\perp^\top)\boldsymbol{\Phi}) \leq \bar{E} - 2$. For future reference, this is summarized as follows:

Corollary 1: If there exists an exact projection filter, i.e. $\mathcal{S}_\mathcal{G} \cap \mathcal{F}_{\boldsymbol{P}} \neq \varnothing$, then

$$\mathrm{rank}((\boldsymbol{U}_\parallel^\top \otimes \boldsymbol{U}_\perp^\top)\boldsymbol{\Phi}) \leq \bar{E} - 2. \tag{18}$$

Note that this provides a condition that can be easily checked before attempting to solve (P1-R). It also provides a guideline on the minimum number of edges required to exactly implement a projection with a given r. Since $(\boldsymbol{U}_\parallel^\top \otimes \boldsymbol{U}_\perp^\top)\boldsymbol{\Phi} \in \mathbb{R}^{r(N-r)\times\bar{E}}$, it follows that $\mathrm{rank}((\boldsymbol{U}_\parallel^\top \otimes \boldsymbol{U}_\perp^\top)\boldsymbol{\Phi}) \leq \min(r(N-r), \bar{E})$. To obtain a reference for the number of edges required for the existence of exact projection filters, suppose that $(\boldsymbol{U}_\parallel^\top \otimes \boldsymbol{U}_\perp^\top)\boldsymbol{\Phi}$ is full rank, as often is the case. Then, (18) becomes $\min(r(N-r), \bar{E}) \leq \bar{E} - 2$, which is equivalent to $r(N-r) \leq \bar{E} - 2$. Thus, the number of edges required for an exact projection filter to exist is in the order of $r(N-r)$. This agrees with intuition because the difficulty to implement

a projection must depend equally on r and $N-r$. This follows by noting that (i) due to the term $c_0\boldsymbol{I}_N$ in (7), implementing $\boldsymbol{P}$ with a graph filter is equally difficult as implementing $\boldsymbol{I}_N-\boldsymbol{P}$; and (ii) $\boldsymbol{P}$ has rank r whereas $\boldsymbol{I}_N-\boldsymbol{P}$ has rank $N-r$. In any case, note that this is just a guideline on the number of edges. The existence of an exact projection filter will not only depend on the number of edges but also on their locations.

Note that Corollary 1 provides a necessary condition for the existence of an exact projection filter. Obtaining a sufficient condition, in turn, is much more challenging. However, one expects that $\mathcal{S}_\mathcal{G}\cap\mathcal{F}_{\boldsymbol{P}}\neq\varnothing$ whenever $\dim(\mathcal{S}_\mathcal{G}\cap\tilde{\mathcal{F}}_{\boldsymbol{P}})>1$. In fact we conjecture that, given a topology and generating $\boldsymbol{U}_\parallel$ by orthonormalizing r vectors in $\mathbb{R}^N$ drawn i.i.d. from a continuous distribution, the event where $\dim(\mathcal{S}_\mathcal{G}\cap\tilde{\mathcal{F}}_{\boldsymbol{P}})>1$ and $\mathcal{S}_\mathcal{G}\cap\mathcal{F}_{\boldsymbol{P}}=\varnothing$ has zero probability. In practice, however, whether $\mathcal{S}_\mathcal{G}\cap\mathcal{F}_{\boldsymbol{P}}$ is empty or not is not as relevant as it may seem. To see this, remember from Theorem 1 that $\mathcal{S}_\mathcal{G}\cap\mathcal{F}_{\boldsymbol{P}}\neq\varnothing$ when there exists a pre-feasible shift matrix for which the eigenvalues of $\boldsymbol{F}_\parallel$ differ from those of $\boldsymbol{F}_\perp$. However, even when this is the case, if one of the eigenvalues of $\boldsymbol{F}_\parallel$ lies too close to an eigenvalue of $\boldsymbol{F}_\perp$, the corresponding filter cannot be implemented due to poor conditioning of (12). In short, even when an exact projection filter may exist from a theoretical perspective, such a filter may not be implementable in practice. Instead, having a sufficiently large $\dim(\mathcal{S}_\mathcal{G}\cap\mathcal{F}_{\boldsymbol{P}})$ seems more important since that would allow the user to choose a shift that leads to a good conditioning of (12). In this sense, Theorem 2 suggests that the *margin* $\bar{E}-\operatorname{rank}((\boldsymbol{U}_\parallel^\top\otimes\boldsymbol{U}_\perp^\top)\boldsymbol{\Phi})$ would be a reasonable indicator of how easy it is to obtain an exact projection filter.

B. Approximate Projection Criterion

The method proposed in Sec. III-D to obtain an exact projection filter relies on solving (P1-R). Using this problem as a starting point, the present section develops an optimization criterion that yields low-order graph filters that approximate a given projection operator.

To this end, note from the second constraint in (P1-R) and the orthogonality of $\boldsymbol{U}_\parallel$ and $\boldsymbol{U}_\perp$ that $\boldsymbol{F}_\parallel=\boldsymbol{U}_\parallel^\top\boldsymbol{S}\boldsymbol{U}_\parallel$ and $\boldsymbol{F}_\perp=\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\perp$. It is also easy to see that $\boldsymbol{S}$ is symmetric iff $\boldsymbol{F}_\parallel$ and $\boldsymbol{F}_\perp$ are symmetric; cf. Appendix A. Additionally, one can also easily prove that any $\boldsymbol{S}$ can be expressed as $\boldsymbol{S}=\boldsymbol{U}_\parallel\boldsymbol{F}_\parallel\boldsymbol{U}_\parallel^\top+\boldsymbol{U}_\perp\boldsymbol{F}_\perp\boldsymbol{U}_\perp^\top$ for some $\boldsymbol{F}_\parallel$ and $\boldsymbol{F}_\perp$ iff $\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\parallel=\boldsymbol{0}$. In words, the latter condition states that each eigenvector of $\boldsymbol{S}$ must be either in the signal subspace or in its orthogonal complement.

Then, problem (P1-R) can be equivalently expressed in terms of $\boldsymbol{S}$ as

$$\begin{aligned}\underset{\boldsymbol{S}}{\text{minimize}}\quad & \eta_\parallel||\boldsymbol{U}_\parallel^\top\boldsymbol{S}\boldsymbol{U}_\parallel\otimes\boldsymbol{I}_r-\boldsymbol{I}_r\otimes\boldsymbol{U}_\parallel^\top\boldsymbol{S}\boldsymbol{U}_\parallel||_\star+\\ &\eta_\perp||\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\perp\otimes\boldsymbol{I}_{N-r}-\boldsymbol{I}_{N-r}\otimes\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\perp||_\star+\\ \text{(P1-R')}\quad & ||\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\perp||_{\mathrm{F}}^2\\ \text{s.t.}\quad & \boldsymbol{W}\operatorname{vec}(\boldsymbol{S})=\boldsymbol{0},\ \boldsymbol{S}=\boldsymbol{S}^\top,\ \boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\parallel=\boldsymbol{0},\\ &\operatorname{Tr}(\boldsymbol{U}_\parallel^\top\boldsymbol{S}\boldsymbol{U}_\parallel)=r,\\ &\operatorname{Tr}(\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\perp)=(1\pm\epsilon)(N-r).\end{aligned}$$

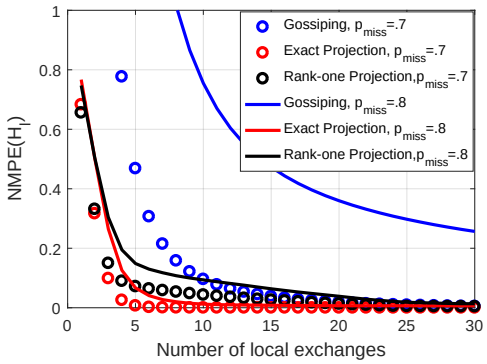

Fig. 1: NMPE as a function of the number of communications performed per node for the Erdős–Rényi networks ($N=30, r=1$, $\rho=0.1$, $I_{\max}=1000$, $\eta_\perp=0.9, \eta_\parallel=0.1, \epsilon=0.1$).

The constraint that renders (P1-R') infeasible when an exact projection filter does not exist is precisely $\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\parallel=\boldsymbol{0}$. This suggests finding a shift for an approximate projection filter by solving

$$\begin{aligned}\underset{\boldsymbol{S}}{\text{minimize}}\quad & \eta_\parallel||\boldsymbol{U}_\parallel^\top\boldsymbol{S}\boldsymbol{U}_\parallel\otimes\boldsymbol{I}_r-\boldsymbol{I}_r\otimes\boldsymbol{U}_\parallel^\top\boldsymbol{S}\boldsymbol{U}_\parallel||_\star+\\ &\eta_\perp||\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\perp\otimes\boldsymbol{I}_{N-r}-\boldsymbol{I}_{N-r}\otimes\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\perp||_\star+\\ \text{(P2-R)}\quad & ||\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\perp||_{\mathrm{F}}^2+\lambda||\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\parallel||_{\mathrm{F}}^2\\ \text{s.t.}\quad & \boldsymbol{W}\operatorname{vec}(\boldsymbol{S})=\boldsymbol{0},\ \boldsymbol{S}=\boldsymbol{S}^\top,\\ &\operatorname{Tr}(\boldsymbol{U}_\parallel^\top\boldsymbol{S}\boldsymbol{U}_\parallel)=r,\\ &\operatorname{Tr}(\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\perp)=(1\pm\epsilon)(N-r),\end{aligned}$$

where the separation of eigenspaces *enforced* by the constraint $\boldsymbol{U}_\perp^\top\boldsymbol{S}\boldsymbol{U}_\parallel=\boldsymbol{0}$ in (P1-R') is now simply *promoted* through the last term in the objective. The parameter $\lambda>0$ is selected by the user to balance the trade off between approximation error and filter order. An iterative solver for (P2-R) is proposed in Appendix F of the supplementary material based on ADMM, therefore inheriting its solid convergence guarantees.

V. Numerical Experiments

This section validates the performance of the proposed algorithms by means of numerical experiments.[5]

The data generation process is as follows. Matrix $\boldsymbol{U}_\parallel$ is obtained by orthonormalizing an $N\times r$ matrix with i.i.d. standard Gaussian entries. This basis is assumed exactly known, meaning that the error due to the selection of the basis is disregarded; see Sec. II-B. To generate the observations $\boldsymbol{z}=\boldsymbol{\xi}+\boldsymbol{v}$, the noise is drawn as $\boldsymbol{v}\sim\mathcal{N}(\boldsymbol{0}_N,\boldsymbol{I}_N)$ and the signal as $\boldsymbol{\xi}=\boldsymbol{U}_\parallel\boldsymbol{\alpha}$, where $\boldsymbol{\alpha}$ is obtained as $\boldsymbol{\alpha}=\sqrt{\beta N/r}\,\boldsymbol{\alpha}_0$ with $\boldsymbol{\alpha}_0\sim\mathcal{N}(\boldsymbol{0}_r,\boldsymbol{I}_r)$ and with $\beta:=\mathbb{E}\|\boldsymbol{\xi}\|_2^2/\mathbb{E}\|\boldsymbol{v}\|_2^2$ the *signal-to-noise ratio* (SNR).

[5]The code necessary to reproduce the experiments is available at `github.com/uiano/fast_decentralized_projections`.

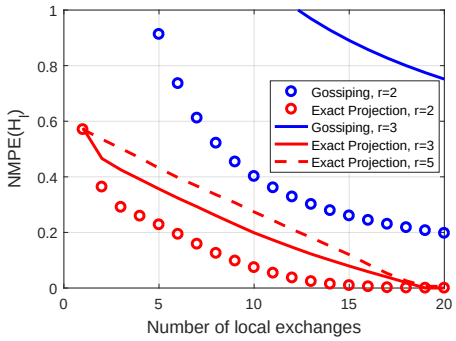

Fig. 2: NMPE as a function of the number of communications performed per node for the Erdős–Rényi networks ($N = 20$, $p_{\text{miss}} = 0.6$, $\rho = 0.1$, $I_{\max} = 1000$, $\eta_\perp = 0.9, \eta_\| = 0.1, \epsilon = 0.1$).

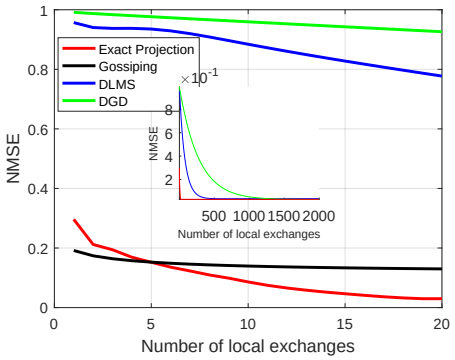

Fig. 4: NMSE as a function of the number of communications performed per node ($N = 20$, $r = 3$, $\beta = 5$, Erdős–Rényi graph with $p_{\text{miss}} = 0.6$, $\rho = 0.1$, $I_{\max} = 1000$, $\eta_\perp = 0.9, \eta_\| = 0.1, \epsilon = 0.1$).

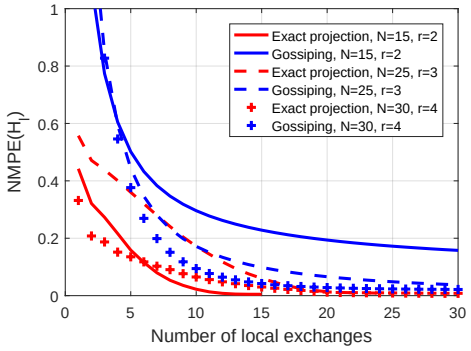

Fig. 3: NMPE as a function of the number of communications performed per node for WSN ($d_{max} = .6, \rho = 0.1$, $I_{\max} = 1000$, $\eta_\perp = 0.9, \eta_\| = 0.1, \epsilon = 0.1$).

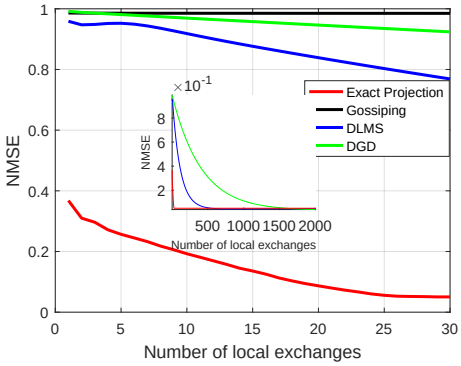

Fig. 5: NMSE as a function of the number of communications performed per node ($N = 30, r = 5, \beta = 5$, Erdős–Rényi graph with $p_{\text{miss}} = 0.7$, $\rho = 0.1$, $I_{\max} = 1000$, $\eta_\perp = 0.9, \eta_\| = 0.1, \epsilon = 0.1$).

The following kinds of networks are generated: (i) Erdős–Rényi graphs, where the presence of each undirected edge is an i.i.d. Bernoulli$(1 - p_{\text{miss}})$ random variable with p_{miss} the missing edge probability. (ii) wireless sensor network (WSN) graphs, generated by deploying the nodes uniformly at random over a square area of unit side length and connecting them with an edge if the internode distance is smaller than $d_{\max}$. If the generated graph in (i) and (ii) is not connected, then additional edges are introduced to ensure connectivity. More specifically, if the graph has c components, $c-1$ eges are added between nodes chosen at random from each component.

Among the compared methods, those for decentralized optimization obtain the projection by solving the least squares problem $\arg\min_{\boldsymbol{\alpha}} \|\boldsymbol{z} - \boldsymbol{U}_\| \boldsymbol{\alpha}\|^2$. This includes (ii) the *distributed least mean squares* (DLMS) method in [20] with augmented Lagrangian parameter ρ_{DLMS} and step size μ_{DLMS}, which builds upon ADMM, and (ii) the *decentralized gradient descent* (DGD) method in [23] with step size μ_{DGD}, which is based on gradient descent. Other methods iteratively apply a shift matrix: (iii) The *gossiping* scheme in [24] obtains the shift matrix $\boldsymbol{S}$ that provides fastest convergence of $\boldsymbol{S}^l$ to $\boldsymbol{P}$ as $l \to \infty$ according to a certain criterion. Then, the nodes collaboratively obtain $\boldsymbol{z}^{(l)} = \boldsymbol{H}_l \boldsymbol{z}$ with $\boldsymbol{H}_l = \boldsymbol{S}^l$ at the l-th exchange round; see also Sec. II-C. Signal $\boldsymbol{z}^{(l)}$ asymptotically converges to the desired projection. (iv) The *rank-1* method in [31] obtains a shift matrix $\boldsymbol{S}$ when $\boldsymbol{P}$ is of rank 1 and then applies a graph filter. The resulting graph filter is generally of maximum order $N - 1$.

Some implementation details follow. To alleviate numerical issues, the method in [31] and the proposed algorithms use node-dependent coefficients [31, Sec. II-B]. For comparison purposes, an order l filter $\boldsymbol{H}_l$ is obtained for each number l of local exchanges by fitting the node-dependent coefficients to minimize $\|\boldsymbol{P} - \boldsymbol{H}_l\|_{\text{F}}^2$. Regarding feasibility (see Sec. IV), the solver for (P1-R) in Appendix E declares the problem as infeasible if the convergence criterion is not met after $I_{\max}$ updates. Both proposed methods use the same parameters in most experiments. This illustrates that a single set of

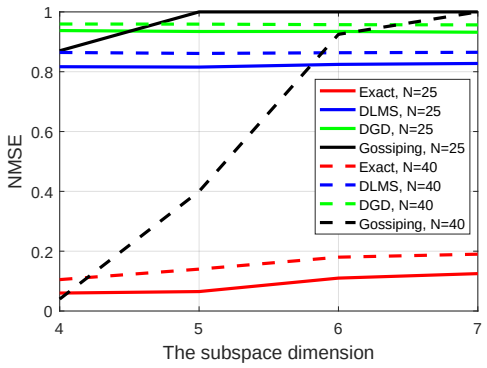

Fig. 6: NMSE as a function of the subspace dimension (Erdős–Rényi network with $p_{\text{miss}} = .7$, $\beta = 5, L = 20$, $\rho = 0.1$, $I_{\max} = 1000$, $\eta_{\perp} = 0.9, \eta_{\|} = 0.1, \epsilon = 0.1$).

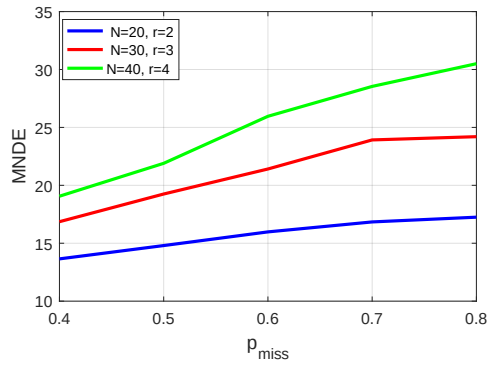

Fig. 7: MNDE for the exact projection method as a function of p_{miss} ($\rho = 0.1$, $I_{\max} = 1000$, $\eta_{\perp} = 0.9, \eta_{\|} = 0.1, \epsilon = 0.1$, $\tau = 0.005$, Erdős–Rényi network).

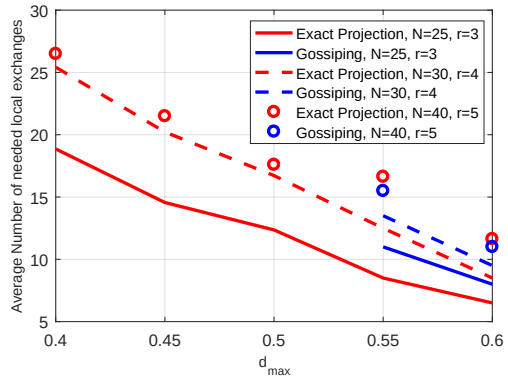

Fig. 8: Average number of local exchanges required to satisfy NMPE $< \gamma_{\text{target}} = .1$ vs. the WSN graph parameter $d_{\max}$ ($\rho = 0.1$, $I_{\max} = 1000$, $\eta_{\perp} = 0.9, \eta_{\|} = 0.1, \epsilon = 0.1$.)

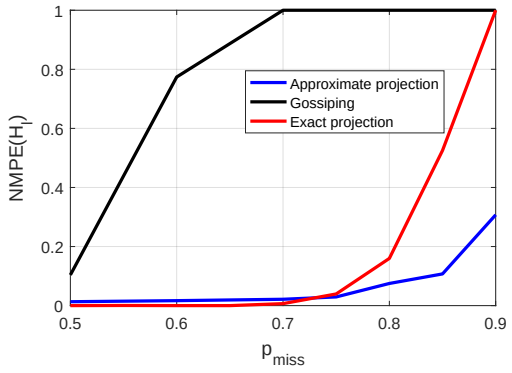

Fig. 9: NMPE as a function of p_{miss} for the Erdős–Rényi networks ($L = N - 1$, $N = 20, r = 3$, $\rho = 0.1$, $I_{\max} = 1000$, $\eta_{\perp} = 0.9, \eta_{\|} = 0.1, \epsilon = 0.2, \lambda = 10$).

parameters works reasonably well in a wide range of scenarios, which facilitates parameter tuning. For fairness, the competing methods use the same parameters in all experiments: $\rho_{\text{DLMS}} = 0.001, \mu_{\text{DLMS}} = 1$, $\mu_{\text{DGD}} = 0.1$. These values were adjusted to approximately yield the best performance in these scenarios.

Two main performance metrics will be adopted. To quantify estimation error, some experiments obtain the *normalized mean square error* (NMSE), defined as $\text{NMSE}(\boldsymbol{H}_l) := \mathbb{E}\left[||\boldsymbol{\xi} - \boldsymbol{H}_l \boldsymbol{z}||_2^2\right] / \mathbb{E}\left[||\boldsymbol{\xi}||_2^2\right]$, where the expectation is taken over $\mathcal{G}$, $\boldsymbol{U}_{\|}, \boldsymbol{\alpha}$, and $\boldsymbol{v}$. For the methods based on graph shift operators, the error between the obtained $\boldsymbol{H}_l$ and the target $\boldsymbol{P}$ is measured through the *normalized mean projection error* $\text{NMPE}(\boldsymbol{H}_l) \triangleq \mathbb{E}\|\boldsymbol{P} - \boldsymbol{H}_l\|_{\text{F}}^2 / r$, where $\mathbb{E}$ averages over $\mathcal{G}$, and $\boldsymbol{U}_{\|}$. The normalization factor r was chosen so that $\text{NMPE}(\boldsymbol{H}_l)$ equals the NMSE when $\boldsymbol{\xi} = \boldsymbol{P}\boldsymbol{z}'$ with $\boldsymbol{z}' \sim \mathcal{N}(\boldsymbol{0}_N, \boldsymbol{I}_N)$. Both the NMSE and NMPE will be estimated by averaging across 500 Monte Carlo realizations. The optimization problems solved by some of these methods, namely the exact projection and the gossiping method, may be infeasible for certain realizations of $\mathcal{G}$ and $\boldsymbol{U}_{\|}$; see Sec. IV. In that case, $\boldsymbol{H}_l$ is set to $\boldsymbol{0}$, which would penalize the tested algorithm by setting its NMSE or NMPE closer to 1.

A. Exact Projection Filters

Figs. 1-3 depict the NMPE for those algorithms that rely on shift matrices. Whereas Figs. 1-2 adopt an Erdős–Rényi random graph, a WSN is used in Fig. 3. The method in [31] is not displayed in Figs. 2-3 because it cannot be applied when $r > 1$. Relative to the competing alternatives, the proposed exact projection method is seen to generally require a significantly smaller number of local exchange rounds to obtain an NMPE close to 0. The aforementioned figures also reveal that the proposed method provides an exact projection in a finite number of iterations. As predicted by the Cayley-Hamilton Theorem, the order is never greater than $N - 1$.

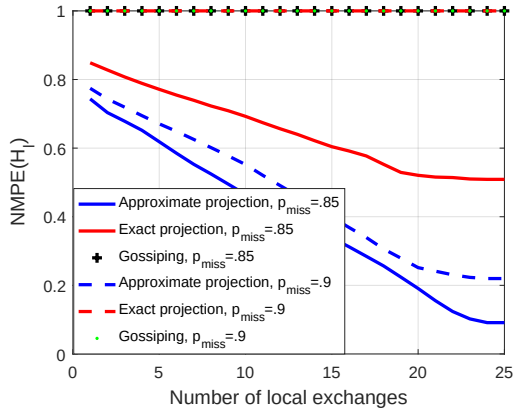

Fig. 10: NMPE as a function of the number of communications performed per node for the Erdős–Rényi networks ($N = 25, r = 3$, $\rho = 0.1$, $I_{\max} = 1000$, $\eta_\perp = 0.9, \eta_\| = 0.1, \epsilon = 0.2, \lambda = 10$).

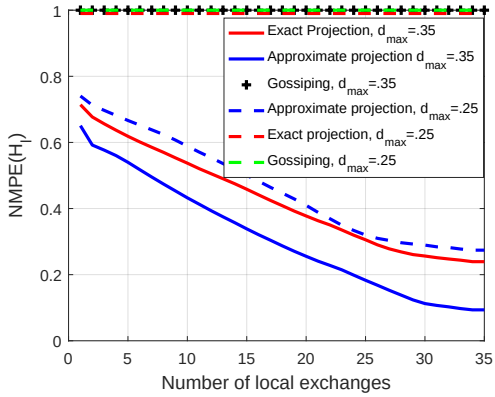

Fig. 11: NMPE as a function of the number of communications performed per node for the WSN networks ($N = 35, r = 5$, $\rho = 0.1$, $I_{\max} = 1000$, $\eta_\perp = 0.9, \eta_\| = 0.1, \epsilon = 0.2, \lambda = 10$).

However, most of the times, the actual order is much smaller than this upper bound. This phenomenon may not be clear at first glance from Figs. 1-3 because they reflect *average* behavior across a large number of Monte Carlo iterations. For this reason, NMPE($\boldsymbol{H}_l$) is generally positive for all $l < N-1$ when one or more realizations yield a filter with maximum order $l = N-1$. On the other hand, the rank-1 method generally yields exact projection filters with order $N-1$ since it is not designed to minimize the order. The gossiping method, however, does not necessarily produce an exact projection in a finite number of iterations. In exchange its implementation is simpler; cf. Sec. II-D.

To compare with the competing algorithms that are not based on graph filtering, Figs. 4-5 depict NMSE($\boldsymbol{H}_l$) vs. the number l of local exchanges for different scenarios. Observe

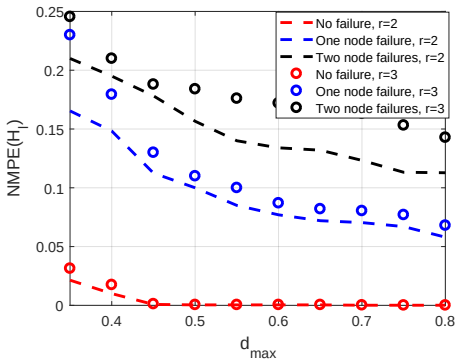

Fig. 12: NMPE as a function of $d_{\max}$ for the WSN networks ($N = 20$, $\rho = 0.1$, $I_{\max} = 1000$, $\eta_\perp = 0.9, \eta_\| = 0.1, \epsilon = 0.1, \lambda = 10$).

that the NMSE of the exact projection method does not converge to 0 due to the observation noise $\boldsymbol{v}$. The miniatures demonstrate the different convergence time scales of DGD, DLMS, and the exact projection method. The cause for the slower convergence of DGD and DLMS is twofold: first, DGD and DLMS are general-purpose methods whereas the proposed method is tailored to the subspace projection problem. Second, DGD and DLMS require the selection of step size parameters, which in the simulations here were set to ensure convergence in virtually all Monte Carlo realizations. For certain specific realizations, though, one could find step sizes that yield a faster convergence. An alternative perspective to the comparison in Figs. 4-5 is offered by Fig. 6, which depicts the NMSE vs. the subspace dimension r when the number of local exchange rounds is fixed to L. This analysis would be necessary e.g. in real-time applications. It is interesting to observe that the sensitivity to r is higher for the gossiping algorithm as compared to the other methods.

To analyze the impact of the graph sparsity in the required number of local exchanges to implement a projection exactly with the proposed method, Fig. 7 shows the *mean number of distinct eigenvalues* (MNDE) yielded by the exact projection method vs. p_{miss}. The NMDE is defined as the mean of the number of distinct eigenvalues of $\boldsymbol{F}_\|$ plus the number of distinct eigenvalues of $\boldsymbol{F}_\perp$, which equals the order of the filter; cf. Sec. III-C. A threshold τ is used to determine whether two eigenvalues are different. As expected, the filter order increases as the graph becomes sparser; see Sec. IV-A. Remarkably, the increase is more pronounced for larger networks and higher r.

In the case of WSN graphs, the impact of sparsity is studied in Fig. 8, which displays the number of local exchange rounds required to reach an NMPE below a target value γ_{target}. The gossiping method is also shown for comparison purposes, yet it is not capable of attaining this target for certain values of $d_{\max}$ due to the infeasibility of the optimization problem that it solves in a significant fraction of the Monte Carlo realizations; see explanation after the definition of NMSE earlier in Sec. V. As $d_{\max}$ increases, the network becomes more densely connected and therefore an exact projection can be

obtained in a smaller number of local exchanges. This agrees with intuition and with Fig. 7.

B. Approximate Projection Filters

As explained in Sec. IV, a graph filter capable of exactly implementing a projection may not exist when the graph is highly sparsely connected. The approximate projection method in that section provides a graph filter capable of approximating such a projection. To illustrate how this algorithm complements the exact projection method from Sec. III, Fig. 9 shows NMPE after $L = N - 1$ local exchanges for both methods along with the gossiping algorithm. The exact projection method is seen to provide a filter that exactly implements the target projection when p_{miss} is below approximately 0.7. For larger p_{miss}, the NMPE becomes strictly positive. It is important to emphasize that the reason is *not* that this method provides filters that do not exactly implement the target projection. When the problem is feasible, the filters yield an exact projection. However, as p_{miss} increases, a smaller fraction of Monte Carlo realizations give rise to a feasible problem and this is penalized in the NMSE computation; see explanation after the definition of NMSE earlier in Sec. V. As expected, the exact projection method performs better than its approximate counterpart when exact projection filters exist. However, for sufficiently sparse graphs, the NMPE of the exact projection method explodes, whereas the approximate projection method remains low.

Figs. 10 and 11 present the evolution of the NMPE vs. the number of local exchange rounds for Erdős–Rényi and WSN graphs respectively with several degrees of sparsity. These figures showcase that the proposed approximate projection method can reasonably approximate a projection with a small number of local exchanges even if the graph is highly sparse. Note that for some of the sparsity levels used in these figures, the problems solved by the gossiping algorithm and the exact projection method become infeasible.

To analyze the robustness of the approximate projection method, Fig. 12 depicts its NMPE for different numbers of node failures. Note that the exact projection method cannot be applied since the removal of one or more sensors renders the exact projection problem infeasible except for trivial cases where their measurements bear no information about the estimated signal. Whenever the network detects that a node has failed, the graph filter is recomputed with the updated topology. As expected, the degradation is more pronounced when the network is sparser (smaller $d_{\max}$). Due to the distribution used to generate $\boldsymbol{U}_{\parallel}$, each measurement contains, on average, a fraction $1/N$ of the signal energy. This informally indicates that n/N lower bounds the error attainable in the presence of n node failures. Observe that the NMPE is indeed close to such fundamental limits, thereby establishing that the approximate projection method is reasonably robust to node failures.

VI. Conclusions and Discussion

This paper develops methods to obtain graph filters that can be used to compute subspace projections in a decentralized fashion. The approach relies on transforming the filter design task into a shift-matrix design problem. The first method addresses the latter by exploiting the eigenstructure of feasible shift matrices. The second method builds upon the first to approximate projections when no feasible projection filter exists. An exhaustive simulation analysis demonstrates the ability of the designed filters to effectively produce subspace projections in a small number of iterations. This contrasts with existing methods, whose convergence is generally asymptotic.

The main strength of graph filters lies in their simplicity. As described before, the inference operations considered in this paper can be implemented with LN transmissions. Lower and higher values can be attained by alternative communication strategies, including centralized processing. The approach adopted in practice will depend on factors such as energy and hardware constraints.

The filters obtained through the proposed methods also inherit the general numerical limitations of graph filters. In turn, these limitations stem from the well-known conditioning issues of Vandermonde systems, e.g. (12). This limits the application of graph filters to networks whose number of nodes is comparable to the ones in Sec. V. Further work is required by the graph signal processing community to enable the implementation of graph filters in significantly larger networks.

Another direction along which the proposed schemes can be extended is by lifting the symmetry constraint on $\boldsymbol{S}$. Since the target matrix $\boldsymbol{P}$ is symmetric, the gain may not be significant. Future work will investigate this possibility.

References

[1] T. Weerasinghe, D. Romero, C. Asensio-Marco, and B. Beferull-Lozano, "Fast distributed subspace projection via graph filters," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Calgary, Canada, Apr. 2018, pp. 4639–4643.

[2] C. S. Raghavendra, K. M. Sivalingam, and T. Znati, *Wireless Sensor Networks*, Springer, 2006.

[3] C. Chen, J. Yan, N. Lu, Y. Wang, X. Yang, and X. Guan, "Ubiquitous monitoring for industrial cyber-physical systems over relay-assisted wireless sensor networks," *IEEE Trans. Emerging Topics in Computing.*, vol. 3, no. 3, pp. 352–362, Sep. 2015.

[4] I. Nevat, G. W. Peters, F. Septier, and T. Matsui, "Estimation of spatially correlated random fields in heterogeneous wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 63, no. 10, pp. 2597–2609, May. 2015.

[5] C. Asensio-Marco and B. Beferull-Lozano, "Energy efficient consensus over complex networks," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 2, pp. 292–303, March. 2015.

[6] R. Nowak, U. Mitra, and R. Willett, "Estimating inhomogeneous fields using wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 6, pp. 999–1006, Aug. 2004.

[7] P. A. Forero, K. Rajawat, and G. B. Giannakis, "Prediction of partially observed dynamical processes over networks via dictionary learning," *IEEE Trans. Signal Process.*, vol. 62, no. 13, pp. 3305–3320, Jul. 2014.

[8] S. M. Kay, *Fundamentals of Statistical Signal Processing, Vol. I: Estimation Theory*, Prentice-Hall, 1993.

[9] J. C. Harsanyi and C-I Chang, "Hyperspectral image classification and dimensionality reduction: An orthogonal subspace projection approach," *IEEE Trans. Geoscience Remote Sensing*, vol. 32, no. 4, pp. 779–785, Jan. 1994.

[10] R. T. Behrens and L. L. Scharf, "Signal processing applications of oblique projection operators," *IEEE Trans. Signal Process.*, vol. 42, no. 6, pp. 1413–1424, jun. 1994.

[11] A. Nedić, A. Olshevsky, and M.G. Rabbat, "Network topology and communication-computation tradeoffs in decentralized optimization," *Proc. IEEE*, vol. 106, no. 5, pp. 953–976, May. 2018.

[12] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, May. 2013.

[13] A. Sandryhaila and J. M. F. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 80–90, Sep. 2014.

[14] M. Belkin, I. Matveeva, and P. Niyogi, "Regularization and semi-supervised learning on large graphs," in *Proc. Annual Conf. Learning Theory*, Banff, Canada, Jul. 2004, Springer, vol. 3120, pp. 624–638.

[15] D. Romero, V. N. Ioannidis, and G. B. Giannakis, "Kernel-based reconstruction of space-time functions on dynamic graphs," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 1–14, Sep. 2017.

[16] D. Romero, M. Ma, and G. B. Giannakis, "Kernel-based reconstruction of graph signals," *IEEE Trans. Signal Process.*, vol. 65, no. 3, pp. 764–778, Feb. 2017.

[17] S. L. Lauritzen, *Graphical Models*, vol. 17, Clarendon Press, 1996.

[18] E. Isufi, A. Loukas, N. Perraudin, and G. Leus, "Forecasting time series with VARMA recursions on graphs," *IEEE Trans. Signal Process.*, vol. 67, no. 18, pp. 4870–4885, Jul. 2019.

[19] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Networks.*, vol. 20, no. 1, pp. 61–80, Jan. 2008.

[20] G. Mateos, I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Performance analysis of the consensus-based distributed LMS algorithm," *EURASIP J. Advances Signal Process.*, pp. 1–19, Nov. 2009.

[21] G. B. Giannakis, Q. Ling, G. Mateos, I. D. Schizas, and H. Zhu, "Decentralized learning for wireless communications and networking," in *Splitting Methods in Communication, Imaging, Science, and Engineering*, pp. 461–497. Springer, 2016.

[22] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[23] L. Shi, L. Zhao, W. Song, G. Kamath, Y. Wu, and X. Liu, "Distributed least-squares iterative methods in large-scale networks: A survey," *ZTE Commun.*, vol. 3, pp. 37–45, Aug. 2017.

[24] S. Barbarossa, G. Scutari, and T. Battisti, "Distributed signal subspace projection algorithms with maximum convergence rate for sensor networks with topological constraints," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Taipei,Taiwan, Apr. 2009, pp. 2893–2896.

[25] X. Insausti, P. Crespo, and B. Beferull-Lozano, "In-network computation of the transition matrix for distributed subspace projection," in *Int. Conf. on Dist. Comp. in Sensor Systems.*, Hangzhou, China, May. 2012, pp. 124–131.

[26] P. Di Lorenzo, S. Barbarossa, and S. Sardellitti, "Distributed signal processing and optimization based on in-network subspace projections," *IEEE Trans. Signal Process.*, vol. 68, pp. 2061–2076, May. 2020.

[27] F. Camaro-Nogues, D. Alonso-Roman, C. Asensio-Marco, and B. Beferull-Lozano, "Reducing the observation error in a wsn through a consensus-based subspace projection," in *Proc. IEEE Int. Wireless Commun. and Networking Conf.*, Shanghai,China, Apr. 2013, pp. 3643–3648.

[28] Y. A. Kibangou, "Graph laplacian based matrix design for finite-time distributed average consensus," in *American Control Conference*, Montreal,Canada, Jun. 2012, pp. 1901–1906.

[29] S. Safavi and U. A. Khan, "Revisiting finite-time distributed algorithms via successive nulling of eigenvalues," *IEEE Signal Process. Lett.*, vol. 22, no. 1, pp. 54–57, Jan. 2015.

[30] A. Sandryhaila, S. Kar, and J. M. F. Moura, "Finite-time distributed consensus through graph filters," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Florence, Italy, May. 2014, pp. 1080–1084.

[31] S. Segarra, A. G. Marques, and A. Ribeiro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Trans. Signal Process.*, vol. 65, no. 15, pp. 4117–4131, Aug. 2017.

[32] M. Coutino, E. Isufi, and G. Leus, "Advances in distributed graph filtering," *IEEE Trans. Signal Process.*, vol. 67, no. 9, pp. 2320–2333, May. 2019.

[33] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, Cambridge, UK, 2004.

[34] E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Autoregressive moving average graph filtering," *IEEE Trans. Signal Process.*, vol. 65, no. 2, pp. 274–288, Jan. 2017.

[35] D. B. Tay and Z. Lin, "Design of near orthogonal graph filter banks," *IEEE Signal Process. Lett.*, vol. 22, no. 6, pp. 701–704, Nov. 2014.

[36] S. K. Narang and A. Ortega, "Perfect reconstruction two-channel wavelet filter banks for graph structured data," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 2786–2799, Feb. 2012.

[37] S. Mollaebrahim, C. Asensio-Marco, D. Romero, and B. Beferull-Lozano, "Decentralized subspace projection in large networks," in *Proc. IEEE Global Conf. Signal Inf. Process.*, Anaheim, CA, Nov. 2018, pp. 788–792.

[38] D. Romero, S-J. Kim, G. B. Giannakis, and R. López-Valcarce, "Learning power spectrum maps from quantized power measurements," *IEEE Trans. Signal Process.*, vol. 65, no. 10, pp. 2547–2560, May. 2017.

[39] V. Cherkassky and F. M. Mulier, *Learning from Data: Concepts, Theory, and Methods*, John Wiley & Sons, 2007.

[40] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, 1990.

[41] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2008.

[42] P. Billingsley, "Probability and measure," *Jown Wiley & Sons*, 1995.

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

APPENDIX A
PROOF OF LEMMA 1

Let $\boldsymbol{H}(\boldsymbol{c},\boldsymbol{S}) := \sum_{l=0}^{L} c_l \boldsymbol{S}^l$ and note from the definition of $\mathcal{F}_{\boldsymbol{P}}$ that $\boldsymbol{S} \in \mathcal{F}_{\boldsymbol{P}}$ iff there exists $\boldsymbol{c}$ such that

$$\boldsymbol{H}(\boldsymbol{c},\boldsymbol{S}) = \boldsymbol{U}_{\parallel}\boldsymbol{U}_{\parallel}^{\top} = \begin{bmatrix}\boldsymbol{U}_{\parallel} & \boldsymbol{U}_{\perp}\end{bmatrix} \begin{bmatrix}\boldsymbol{I}_r & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix} \begin{bmatrix}\boldsymbol{U}_{\parallel}^{\top} \\ \boldsymbol{U}_{\perp}^{\top}\end{bmatrix}. \quad (19)$$

Since $\boldsymbol{S}$ is symmetric, it admits an eigenvalue decomposition $\boldsymbol{S} = \boldsymbol{W}\boldsymbol{\Lambda}\boldsymbol{W}^{\top}$. Thus,

$$\boldsymbol{H}(\boldsymbol{c},\boldsymbol{S}) = \boldsymbol{W}\left(\sum_{l=0}^{L} c_l \boldsymbol{\Lambda}^l\right)\boldsymbol{W}^{\top} \quad (20a)$$

$$= \boldsymbol{W}\begin{bmatrix}\boldsymbol{I}_r & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\boldsymbol{W}^{\top} \quad (20b)$$

$$= \boldsymbol{W}_{\parallel}\boldsymbol{W}_{\parallel}^{\top}, \quad (20c)$$

where $\boldsymbol{W}_{\parallel}$ comprises the first r columns of $\boldsymbol{W} := [\boldsymbol{W}_{\parallel}, \boldsymbol{W}_{\perp}]$. The second and third equalities in (20) follow from the fact that (19) implies that $\boldsymbol{H}(\boldsymbol{c},\boldsymbol{S})$ has an eigenvalue 1 with multiplicity r and an eigenvalue 0 with multiplicity $N-r$ and, therefore, (20b) and (20c) must hold for some ordering of the eigenvectors in the eigendecomposition $\boldsymbol{S} = \boldsymbol{W}\boldsymbol{\Lambda}\boldsymbol{W}^{\top}$.

From (19) and (20c), it follows that $\boldsymbol{W}_{\|}\boldsymbol{W}_{\|}^{\top} = \boldsymbol{U}_{\|}\boldsymbol{U}_{\|}^{\top}$ and, therefore, $\mathcal{R}\{\boldsymbol{W}_{\|}\boldsymbol{W}_{\|}^{\top}\} = \mathcal{R}\{\boldsymbol{U}_{\|}\boldsymbol{U}_{\|}^{\top}\}$, which in turn is equivalent to $\mathcal{R}\{\boldsymbol{W}_{\|}\} = \mathcal{R}\{\boldsymbol{U}_{\|}\}$. Consequently, $\boldsymbol{W}_{\|} = \boldsymbol{U}_{\|}\boldsymbol{Q}_{\|}$ for some $\boldsymbol{Q}_{\|}$. Moreover, since $\boldsymbol{I}_r = \boldsymbol{W}_{\|}^{\top}\boldsymbol{W}_{\|} = \boldsymbol{Q}_{\|}^{\top}\boldsymbol{U}_{\|}^{\top}\boldsymbol{U}_{\|}\boldsymbol{Q}_{\|} = \boldsymbol{Q}_{\|}^{\top}\boldsymbol{Q}_{\|}$, one can conclude that $\boldsymbol{Q}_{\|}$ is orthonormal. Similarly, $\mathcal{R}\{\boldsymbol{W}_{\perp}\} = \mathcal{R}^{\perp}\{\boldsymbol{W}_{\|}\} = \mathcal{R}^{\perp}\{\boldsymbol{U}_{\|}\} = \mathcal{R}\{\boldsymbol{U}_{\perp}\}$, which implies that $\boldsymbol{W}_{\perp} = \boldsymbol{U}_{\perp}\boldsymbol{Q}_{\perp}$ for some orthogonal $\boldsymbol{Q}_{\perp}$.

Upon letting $\boldsymbol{\Lambda}_{\|}$ and $\boldsymbol{\Lambda}_{\perp}$ be such that

$$\boldsymbol{S} = \boldsymbol{W}\boldsymbol{\Lambda}\boldsymbol{W}^{\top} := \begin{bmatrix}\boldsymbol{W}_{\|} & \boldsymbol{W}_{\perp}\end{bmatrix}\begin{bmatrix}\boldsymbol{\Lambda}_{\|} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{\Lambda}_{\perp}\end{bmatrix}\begin{bmatrix}\boldsymbol{W}_{\|}^{\top}\\ \boldsymbol{W}_{\perp}^{\top}\end{bmatrix} \tag{21}$$

and applying the above relations to (21), it follows that

$$\boldsymbol{S} = \begin{bmatrix}\boldsymbol{U}_{\|}\boldsymbol{Q}_{\|} & \boldsymbol{U}_{\perp}\boldsymbol{Q}_{\perp}\end{bmatrix}\begin{bmatrix}\boldsymbol{\Lambda}_{\|} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{\Lambda}_{\perp}\end{bmatrix}\begin{bmatrix}\boldsymbol{Q}_{\|}^{\top}\boldsymbol{U}_{\|}^{\top}\\ \boldsymbol{Q}_{\perp}^{\top}\boldsymbol{U}_{\perp}^{\top}\end{bmatrix} = \begin{bmatrix}\boldsymbol{U}_{\|} & \boldsymbol{U}_{\perp}\end{bmatrix}\begin{bmatrix}\boldsymbol{F}_{\|} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{F}_{\perp}\end{bmatrix}\begin{bmatrix}\boldsymbol{U}_{\|}^{\top}\\ \boldsymbol{U}_{\perp}^{T}\end{bmatrix}, \tag{22}$$

where $\boldsymbol{F}_{\|} := \boldsymbol{Q}_{\|}\boldsymbol{\Lambda}_{\|}\boldsymbol{Q}_{\|}^{\top}$ and $\boldsymbol{F}_{\perp} := \boldsymbol{Q}_{\perp}\boldsymbol{\Lambda}_{\perp}\boldsymbol{Q}_{\perp}^{\top}$.

Appendix B
Proof of Lemma 3

Let $\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{\top}$ be an eigenvalue decomposition. Then,

$$\|\boldsymbol{A}\otimes\boldsymbol{I}_N - \boldsymbol{I}_N\otimes\boldsymbol{A}\|_* = \|\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{\top}\otimes\boldsymbol{I}_N - \boldsymbol{I}_N\otimes\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{\top}\|_*,$$

Applying the properties of the Kronecker product and the invariance of the nuclear norm to orthogonal transformations,

$$\begin{aligned}\|\boldsymbol{A}\otimes\boldsymbol{I}_N - \boldsymbol{I}_N\otimes\boldsymbol{A}\|_* =& \|(\boldsymbol{V}\otimes\boldsymbol{I}_N)(\boldsymbol{\Lambda}\otimes\boldsymbol{I}_N)(\boldsymbol{V}\otimes\boldsymbol{I}_N)^{\top}\\ &-(\boldsymbol{I}_N\otimes\boldsymbol{V})(\boldsymbol{I}_N\otimes\boldsymbol{\Lambda})(\boldsymbol{I}_N\otimes\boldsymbol{V})^{\top}\|_*\\ =& \|(\boldsymbol{\Lambda}\otimes\boldsymbol{I}_N) - (\boldsymbol{I}_N\otimes\boldsymbol{\Lambda})\|_*.\end{aligned}$$

From the definition of the nuclear and ℓ_1 norms,

$$\begin{aligned}\|(\boldsymbol{\Lambda}\otimes\boldsymbol{I}_N)-(\boldsymbol{I}_N\otimes\boldsymbol{\Lambda})\|_* &= \|\mathrm{diag}(\boldsymbol{\Lambda}\otimes\boldsymbol{I}_N - \boldsymbol{I}_N\otimes\boldsymbol{\Lambda})\|_1\\ &= \|\boldsymbol{\lambda}\otimes\boldsymbol{1}_N - \boldsymbol{1}_N\otimes\boldsymbol{\lambda}\|_1 &\text{(23a)}\\ &= \sum_{i=1}^{N}|\,(\boldsymbol{\lambda}\otimes\boldsymbol{1}_N - \boldsymbol{1}_N\otimes\boldsymbol{\lambda})_i\,| &\text{(23b)}\\ &= \sum_{i=1}^{N}|\,(\boldsymbol{\lambda}\otimes\boldsymbol{1}_N - \boldsymbol{1}_N\otimes\boldsymbol{\lambda})^{\top}\boldsymbol{e}_{N^2,i}\,|, &\text{(23c)}\end{aligned}$$

where $\boldsymbol{\lambda} := \mathrm{diag}(\boldsymbol{\Lambda})$ and $\boldsymbol{e}_{M,i}$ is the i-th column of $\boldsymbol{I}_M$. Splitting the summation in (23c) and applying the fact that $\boldsymbol{e}_{N^2,N(j-1)+k} = \boldsymbol{e}_{N,j}\otimes\boldsymbol{e}_{N,k}\ \forall j,k$, it follows that

$$\begin{aligned}\|\boldsymbol{A}\otimes\boldsymbol{I}_N-\boldsymbol{I}_N\otimes\boldsymbol{A}\|_* &&\text{(24)}\\ =\sum_{j=1}^{N}\sum_{k=1}^{N}&|\,(\boldsymbol{\lambda}\otimes\boldsymbol{1}_N - \boldsymbol{1}_N\otimes\boldsymbol{\lambda})^{\top}\boldsymbol{e}_{N^2,N(j-1)+k}\,|\\ =\sum_{j=1}^{N}\sum_{k=1}^{N}&|\,(\boldsymbol{\lambda}\otimes\boldsymbol{1}_N - \boldsymbol{1}_N\otimes\boldsymbol{\lambda})^{\top}(\boldsymbol{e}_{N,j}\otimes\boldsymbol{e}_{N,k})\,|\end{aligned}$$

Finally, from the properties of the Kronecker product,

$$\begin{aligned}\|\boldsymbol{A}\otimes\boldsymbol{I}_N-\boldsymbol{I}_N\otimes\boldsymbol{A}\|_* &&\text{(25a)}\\ =&\sum_{j=1}^{N}\sum_{k=1}^{N}\left|\boldsymbol{\lambda}^{\top}\boldsymbol{e}_{N,j}\otimes\boldsymbol{1}_N^{\top}\boldsymbol{e}_{N,k} - \boldsymbol{1}_N^{\top}\boldsymbol{e}_{N,j}\otimes\boldsymbol{\lambda}^{\top}\boldsymbol{e}_{N,k}\right|\\ =&\sum_{j=1}^{N}\sum_{k=1}^{N}|\lambda_j\otimes 1 - 1\otimes\lambda_k| = \sum_{j=1}^{N}\sum_{k=1}^{N}|\lambda_j-\lambda_k|.\end{aligned}$$

Appendix C
Proof of Theorem 2

Consider first the following auxiliary result:

Lemma 4: $\boldsymbol{S}\in\mathcal{S}_{\mathcal{G}}\cap\tilde{\mathcal{F}}_{\boldsymbol{P}}$ iff the following three conditions simultaneously hold

$$\boldsymbol{W}\mathrm{vec}(\boldsymbol{S}) = \boldsymbol{0}, \tag{26a}$$

$$\boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|} = \boldsymbol{0}, \tag{26b}$$

$$\boldsymbol{S} = \boldsymbol{S}^{\top}. \tag{26c}$$

Proof: The first step is proving that $\boldsymbol{S}\in\mathcal{S}_{\mathcal{G}}\cap\tilde{\mathcal{F}}_{\boldsymbol{P}}$ implies (26). Conditions (26a) and (26c) follow from the definition of $\mathcal{S}_{\mathcal{G}}$ and Definition 2. To verify (26b), note from Definition 2 that $\boldsymbol{S}\in\tilde{\mathcal{F}}_{\boldsymbol{P}}$ iff $\boldsymbol{S}$ satisfies (8) for some symmetric $\boldsymbol{F}_{\|}$ and $\boldsymbol{F}_{\perp}$. Multiplying (8) on the right by $\boldsymbol{U} := [\boldsymbol{U}_{\|},\boldsymbol{U}_{\perp}]$ and on the left by $\boldsymbol{U}^{\top}$ yields

$$\begin{bmatrix}\boldsymbol{U}_{\|}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|} & \boldsymbol{U}_{\|}^{\top}\boldsymbol{S}\boldsymbol{U}_{\perp}\\ \boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|} & \boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\perp}\end{bmatrix} = \begin{bmatrix}\boldsymbol{F}_{\|} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{F}_{\perp}\end{bmatrix} \tag{27}$$

Condition (26b) corresponds to the block $(2,1)$ in this equality.

Conversely, if $\boldsymbol{S}$ satisfies (26), then it follows from (26a) and[6] (26c) that $\boldsymbol{S}\in\mathcal{S}_{\mathcal{G}}$. To show that $\mathcal{S}_{\mathcal{G}}\in\tilde{\mathcal{F}}_{\boldsymbol{P}}$, one needs to find symmetric $\boldsymbol{F}_{\|}$ and $\boldsymbol{F}_{\perp}$ such that (8) or, equivalently, (27) holds. This can be easily accomplished just by setting $\boldsymbol{F}_{\|} = \boldsymbol{U}_{\|}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|}$ and $\boldsymbol{F}_{\perp} = \boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\perp}$ since the (1,2) and (2,1) blocks of equality (27) will automatically hold due to (26b) and (26c). ■

Let $\mathbb{S} := \{\boldsymbol{S}\in\mathbb{R}^{N\times N} : \boldsymbol{S} = \boldsymbol{S}^{\top}\}$ and note from Definition 2 that $\tilde{\mathcal{F}}_{\boldsymbol{P}}\subset\mathbb{S}$. Then, $\mathcal{S}_{\mathcal{G}}\cap\tilde{\mathcal{F}}_{\boldsymbol{P}} = \mathcal{S}_{\mathcal{G}}\cap(\mathbb{S}\cap\tilde{\mathcal{F}}_{\boldsymbol{P}}) = (\mathcal{S}_{\mathcal{G}}\cap\mathbb{S})\cap\tilde{\mathcal{F}}_{\boldsymbol{P}}$.

Now, consider the following parameterization of $\mathcal{S}_{\mathcal{G}}\cap\mathbb{S}$:

$$\mathcal{S}_{\mathcal{G}}\cap\mathbb{S} = \left\{\boldsymbol{S} = \sum_{i=1}^{\bar{E}}\kappa_i\overbrace{(\boldsymbol{e}_{n_i}\boldsymbol{e}_{n_i'}^{\top} + \boldsymbol{e}_{n_i'}\boldsymbol{e}_{n_i}^{\top})}^{\boldsymbol{\Phi}_i}, \kappa_i\in\mathbb{R}\right\}, \tag{28}$$

which can be expressed in vector form as

$$\begin{aligned}\mathrm{vec}(\mathcal{S}_{\mathcal{G}}\cap\mathbb{S}) &= \left\{\mathrm{vec}(\boldsymbol{S}) = \sum_{i=1}^{\bar{E}}\kappa_i\mathrm{vec}(\boldsymbol{\Phi}_i)\right\} &\text{(29)}\\ &= \left\{[\mathrm{vec}(\boldsymbol{\Phi}_1),\cdots,\mathrm{vec}(\boldsymbol{\Phi}_{\bar{E}})]\boldsymbol{\kappa}, \boldsymbol{\kappa}\in\mathbb{R}^{\bar{E}}\right\}.\end{aligned}$$

[6]Please keep in mind that $\boldsymbol{S}\in\mathcal{S}_{\mathcal{G}}$ does not imply that $\boldsymbol{S}$ is symmetric and (26a) alone only imposes support constraints on the upper-triangular entries of $\boldsymbol{S}$; cf. Sec. III-C.

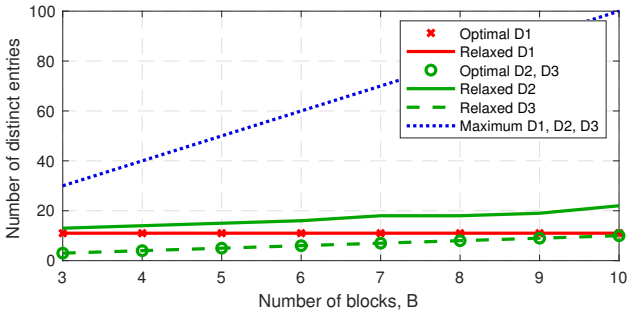

Fig. 13: Comparison between the solution of (34) and that of $\inf\{\mathcal{L}(\boldsymbol{x}) : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\}$ for $M = N = 10$.

Combining (26) and (29) yields

$$\mathcal{S}_{\mathcal{G}} \cap \tilde{\mathcal{F}}_{\boldsymbol{P}} = \left\{ \text{vec}^{-1}(\boldsymbol{\Phi}\boldsymbol{\kappa}) \; \forall \boldsymbol{\kappa} : \boldsymbol{U}_{\perp}^{\top} \text{vec}^{-1}(\boldsymbol{\Phi}\boldsymbol{\kappa}) \boldsymbol{U}_{\parallel} = \boldsymbol{0} \right\}. \tag{30}$$

Since the columns of $\boldsymbol{\Phi}$ are linearly independent,

$$\begin{aligned} \dim(\mathcal{S}_{\mathcal{G}} \cap \tilde{\mathcal{F}}_{\boldsymbol{P}}) =& \dim\{\boldsymbol{\kappa} : \boldsymbol{U}_{\perp}^{\top} \text{vec}^{-1}(\boldsymbol{\Phi}\boldsymbol{\kappa}) \boldsymbol{U}_{\parallel} = \boldsymbol{0}\} \\ =& \dim\{\boldsymbol{\kappa} : (\boldsymbol{U}_{\parallel}^{\top} \otimes \boldsymbol{U}_{\perp}^{\top}) \boldsymbol{\Phi}\boldsymbol{\kappa} = \boldsymbol{0}\}. \end{aligned}$$

Since $(\boldsymbol{U}_{\parallel}^{\top} \otimes \boldsymbol{U}_{\perp}^{\top})\boldsymbol{\Phi} \in \mathbb{R}^{r(N-r)\times \bar{E}}$, it follows that $\dim(\mathcal{S}_{\mathcal{G}} \cap \tilde{\mathcal{F}}_{\boldsymbol{P}}) = \bar{E} - \text{rank}((\boldsymbol{U}_{\parallel}^{\top} \otimes \boldsymbol{U}_{\perp}^{\top})\boldsymbol{\Phi})$.

Appendix D
The Tightness of the Relaxed Solution

This appendix further justifies why the objective of the relaxed problem (P1-R) is a reasonable surrogate for the objective of the original problem (P1). To simplify notation, some of the symbols used earlier will be reused here.

To focus on the fundamental aspects, consider the optimization in (P1-R) for a fixed $\boldsymbol{F}_{\perp}$. The resulting problem is a special case of

$$\underset{\boldsymbol{F}_{\parallel}}{\text{minimize}} \quad ||\boldsymbol{F}_{\parallel} \otimes \boldsymbol{I}_r - \boldsymbol{I}_r \otimes \boldsymbol{F}_{\parallel}||_{\star} \tag{31a}$$

$$\text{s.t.} \quad \boldsymbol{A}_{\parallel} \text{vec}(\boldsymbol{F}_{\parallel}) = \boldsymbol{b}_{\parallel} \tag{31b}$$

for some $\boldsymbol{A}_{\parallel}$ and $\boldsymbol{b}_{\parallel}$. Ideally, one would like to see how well the solution to (31) approximates the solution to

$$\underset{\boldsymbol{F}_{\parallel}}{\text{minimize}} \quad \mathcal{L}(\boldsymbol{F}_{\parallel}) \tag{32a}$$

$$\text{s.t.} \quad \boldsymbol{A}_{\parallel} \text{vec}(\boldsymbol{F}_{\parallel}) = \boldsymbol{b}_{\parallel}. \tag{32b}$$

To this end, apply the eigendecomposition $\boldsymbol{F}_{\parallel} = \boldsymbol{Q}_{\parallel} \boldsymbol{\Lambda}_{\parallel} \boldsymbol{Q}_{\parallel}^{\top}$ to rewrite the solution of (31) as

$$\inf_{\text{orthogonal } \boldsymbol{Q}_{\parallel}} \left[\begin{array}{l} \inf\limits_{\text{diagonal } \boldsymbol{\Lambda}_{\parallel}} ||\boldsymbol{\Lambda}_{\parallel} \otimes \boldsymbol{I}_r - \boldsymbol{I}_r \otimes \boldsymbol{\Lambda}_{\parallel}||_{\star} \\ \text{s.t.} \quad \boldsymbol{A}_{\parallel} \text{vec}(\boldsymbol{Q}_{\parallel} \boldsymbol{\Lambda}_{\parallel} \boldsymbol{Q}_{\parallel}^{\top}) = \boldsymbol{b}_{\parallel} \end{array} \right]. \tag{33a}$$

The inner problem, which captures the essence of the question to be addressed here, is a special case of

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad \|\boldsymbol{x} \otimes \boldsymbol{1} - \boldsymbol{1} \otimes \boldsymbol{x}\|_1 \tag{34a}$$

$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{34b}$$

for some $\boldsymbol{A}$ and $\boldsymbol{b}$. Thus, the ability of (31) to promote solutions with a reduced number of distinct eigenvalues can be understood by analyzing how well (34) promotes solutions with a reduced number of distinct entries. With $\boldsymbol{x}^*$ denoting a minimizer of (34) and with $\mathcal{L}(\boldsymbol{x})$ denoting the number of distinct entries of $\boldsymbol{x}$, one is therefore interested in analyzing how far $\mathcal{L}(\boldsymbol{x}^*)$ is from $\mathcal{L}^* := \inf\{\mathcal{L}(\boldsymbol{x}) : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\}$ for given $\boldsymbol{A}$ and $\boldsymbol{b}$. Unfortunately, this is a challenging comparison since obtaining $\mathcal{L}^*$ generally entails combinatorial complexity. However, $\mathcal{L}^*$ can be obtained exactly for certain families of $(\boldsymbol{A}, \boldsymbol{b})$. Specifically, the rest of this section will compare $\mathcal{L}(\boldsymbol{x}^*)$ and $\mathcal{L}^*$ when $\boldsymbol{A}$ is drawn from 3 probability distributions.

Let $\boldsymbol{A}$ comprise B block columns:

$$\boldsymbol{A} := \left[\begin{array}{ccc} \boldsymbol{A}_1 & \dots & \boldsymbol{A}_B \\ \boldsymbol{E}_1 & \dots & \boldsymbol{E}_B \\ \boldsymbol{1}^{\top} & \dots & \boldsymbol{1}^{\top} \end{array} \right]. \tag{35}$$

Let also $\boldsymbol{A}_i := [\boldsymbol{a}_{i,1}, \dots, \boldsymbol{a}_{i,N}] \in \mathbb{R}^{M \times N}$ and consider the following distributions:

- Distribution D_1: $\{\boldsymbol{E}_i\}_{i=1}^{B}$ are empty and $\{\boldsymbol{a}_{i,j}\}_{i,j}$ are drawn independently from a continuous distribution.[7]
- Distribution D_2: $\{\boldsymbol{a}_{i,j}\}_i$ are drawn independently from a continuous distribution for $j = 1, \dots, N-1$ and $\boldsymbol{a}_{i,N} := -\sum_{j=1}^{N-1} \boldsymbol{a}_{i,j}$. On the other hand, $\boldsymbol{E}_i = (-1/i)\boldsymbol{e}_i \boldsymbol{e}_1^{\top} \in \mathbb{R}^{B-1 \times N}$ for $i = 1, \dots, B-1$, where $\boldsymbol{e}_i$ is, with an abuse of notation, the i-th column of the identity matrix with appropriate dimension. For $i = B$, $\boldsymbol{E}_B = (1/i)\boldsymbol{1}\boldsymbol{e}_1^{\top}$.
- Distribution D_3: $\{\boldsymbol{a}_{i,j}\}_{i,j}$ are as in D_2. Matrix $\boldsymbol{E}_i$ is $\boldsymbol{E}_i = (-1/i)\boldsymbol{e}_i \boldsymbol{1}^{\top} \in \mathbb{R}^{B-1 \times N}$ for $i = 1, \dots, B-1$ and $\boldsymbol{E}_B = (1/i)\boldsymbol{1}\boldsymbol{1}^{\top}$ for $i = B$.

With these distributions, we have the following:

Theorem 3: Let $\mathcal{L}^* := \inf\{\mathcal{L}(\boldsymbol{x}) : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\}$ and let $\boldsymbol{b}$ be a vector of the appropriate dimensions with all zeros except for the last entry, which equals $N \sum_{i=1}^{B} i = NB(B+1)/2$.

(a) If $\boldsymbol{A} \sim D_1$ with $NB > M$, then $\mathcal{L}^* = M + 1$ with probability 1.

(b) If $\boldsymbol{A} \sim D_i$, $i \in 2, 3$, with $M \geq B$, then $\mathcal{L}^* = B$ with probability 1.

Proof: See Appendix G in the supplementary material. ■

Fig. 13 compares $\mathcal{L}(\boldsymbol{x}^*)$ and $\mathcal{L}^* := \inf\{\mathcal{L}(\boldsymbol{x}) : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\}$, where $\boldsymbol{x}^*$ is found by applying a convex solver to (34). Each point is obtained for a realization of $\boldsymbol{A}$. Although we have not proved it formally, it seems that the relaxed solution $\boldsymbol{x}^*$ coincides with the exact one with probability 1 when $\boldsymbol{A}$ is drawn from D_1 or D_3. For D_2, the number of distinct entries of $\boldsymbol{x}^*$ is higher than optimal, but in any case it is considerably lower than the number NB of entries of $\boldsymbol{x}$, shown as "Maximum".

To sum up, the solution to the relaxed problem (34) is optimal in some cases and close to the optimal in other tested cases. This supports the choice of $||\boldsymbol{F}_{\parallel} \otimes \boldsymbol{I}_r - \boldsymbol{I}_r \otimes \boldsymbol{F}_{\parallel}||_{\star}$ as a convex surrogate for the number of distinct eigenvalues of $\boldsymbol{F}_{\parallel}$.

[7] Formally, a continuous distribution in this context is a distribution that is absolutely continuous with respect to Lebesgue measure [42].

SUPPLEMENTARY MATERIAL

APPENDIX E
ITERATIVE SOLVER FOR (P1-R)

This appendix develops an iterative method to solve (P1-R) based on ADMM. To this end, the first step is to rewrite the objective and constraints in a form that is amenable to the application of this method.

To rewrite the objective, let $\boldsymbol{e}_i$ and $\boldsymbol{t}_j$ be respectively the i-th and j-th columns of $\boldsymbol{I}_r$ and $\boldsymbol{I}_{N-r}$. The first and second norms in the objective of (P1-R) can be expressed as:

$$\big\|\boldsymbol{F}_\parallel \otimes \boldsymbol{I}_r - \boldsymbol{I}_r \otimes \boldsymbol{F}_\parallel\big\|_* = \big\|\mathrm{vec}^{-1}(\boldsymbol{A}\mathrm{vec}(\boldsymbol{F}_\parallel))\big\|_*$$
$$\|\boldsymbol{F}_\perp \otimes \boldsymbol{I}_{N-r} - \boldsymbol{I}_{N-r} \otimes \boldsymbol{F}_\perp\|_* = \big\|\mathrm{vec}^{-1}(\boldsymbol{B}\mathrm{vec}(\boldsymbol{F}_\perp))\big\|_*,$$

where $\boldsymbol{A} := [\boldsymbol{a}_{11}, \boldsymbol{a}_{21}, ..., \boldsymbol{a}_{rr}]$, $\boldsymbol{B} := [\boldsymbol{b}_{11}, \boldsymbol{b}_{21}, ..., \boldsymbol{b}_{N-r,N-r}]$, $\boldsymbol{a}_{ij} := \mathrm{vec}(\boldsymbol{e}_i\boldsymbol{e}_j^\top \otimes \boldsymbol{I}_r - \boldsymbol{I}_r \otimes \boldsymbol{e}_i\boldsymbol{e}_j^\top)$, $\boldsymbol{b}_{ij} := \mathrm{vec}(\boldsymbol{t}_i\boldsymbol{t}_j^\top \otimes \boldsymbol{I}_{N-r} - \boldsymbol{I}_{N-r} \otimes \boldsymbol{t}_i\boldsymbol{t}_j^\top)$.

To rewrite the constraints of (P1-R), invoke the properties of the Kronecker product to combine the first and second constraints as $\boldsymbol{W}((\boldsymbol{U}_\parallel \otimes \boldsymbol{U}_\parallel)\mathrm{vec}(\boldsymbol{F}_\parallel) + (\boldsymbol{U}_\perp \otimes \boldsymbol{U}_\perp)\mathrm{vec}(\boldsymbol{F}_\perp)) = \boldsymbol{0}$. The third and fourth constraints can be rewritten as $\boldsymbol{G}_\parallel\mathrm{vec}(\boldsymbol{F}_\parallel) = \boldsymbol{0}$, $\boldsymbol{G}_\perp\mathrm{vec}(\boldsymbol{F}_\perp) = \boldsymbol{0}$ where the rows of $\boldsymbol{G}_\parallel \in \mathbb{R}^{(r^2-r)/2\times r^2}$ and $\boldsymbol{G}_\perp \in \mathbb{R}^{((N-r)^2-(N-r))/2\times(N-r)^2}$ are respectively given by $(\boldsymbol{e}_j^\top \otimes \boldsymbol{e}_i^\top - \boldsymbol{e}_i^\top \otimes \boldsymbol{e}_j^\top)$ and $(\boldsymbol{t}_j^\top \otimes \boldsymbol{t}_i^\top - \boldsymbol{t}_i^\top \otimes \boldsymbol{t}_j^\top)$ for $i < j$. Regarding the trace constraints, rewrite $\mathrm{tr}(\boldsymbol{F}_\parallel) = r$ as $\mathrm{tr}(\boldsymbol{I}_r\boldsymbol{F}_\parallel) = r$, which in turn can be expressed as $\mathrm{vec}^\top(\boldsymbol{I}_r)\mathrm{vec}(\boldsymbol{F}_\parallel) = r$. Similarly, one can rewrite $\mathrm{tr}(\boldsymbol{F}_\perp) = (1 \pm \epsilon)(N - r)$ as $\mathrm{vec}^\top(\boldsymbol{I}_{N-r})\mathrm{vec}(\boldsymbol{F}_\perp) = (1 \pm \epsilon)(N - r)$.

Therefore, (P1-R) can be written as:

$$\begin{aligned}\min_{\boldsymbol{Y}_\perp,\boldsymbol{Y}_\parallel,\boldsymbol{F}_\parallel,\boldsymbol{F}_\perp}\quad & \eta_\parallel\big\|\boldsymbol{Y}_\parallel\big\|_* + \eta_\perp\|\boldsymbol{Y}_\perp\|_* + \|\boldsymbol{F}_\perp\|_{\mathrm{F}}^2\\ \text{s. t.}\quad & \boldsymbol{T}_\parallel\mathrm{vec}(\boldsymbol{F}_\parallel) + \boldsymbol{T}_\perp\mathrm{vec}(\boldsymbol{F}_\perp) = \boldsymbol{b} && (36a)\\ & \mathrm{vec}(\boldsymbol{Y}_\parallel) = \boldsymbol{A}\mathrm{vec}(\boldsymbol{F}_\parallel) && (36b)\\ & \mathrm{vec}(\boldsymbol{Y}_\perp) = \boldsymbol{B}\mathrm{vec}(\boldsymbol{F}_\perp), && (36c)\end{aligned}$$

where

$$\boldsymbol{T}_\parallel \triangleq [\boldsymbol{W}(\boldsymbol{U}_\parallel \otimes \boldsymbol{U}_\parallel); \boldsymbol{G}_\parallel; \boldsymbol{0}; \mathrm{vec}^\top(\boldsymbol{I}_r); \boldsymbol{0}] \quad (37a)$$
$$\boldsymbol{T}_\perp \triangleq [\boldsymbol{W}(\boldsymbol{U}_\perp \otimes \boldsymbol{U}_\perp); \boldsymbol{0}; \boldsymbol{G}_\perp; \boldsymbol{0}; \mathrm{vec}^\top(\boldsymbol{I}_{N-r})] \quad (37b)$$
$$\boldsymbol{b} \triangleq [\boldsymbol{0}; \boldsymbol{0}; \boldsymbol{0}; r; (1 \pm \epsilon)(N - r)]. \quad (37c)$$

The ADMM method in scaled form [22, Sec. 3.1.1] applied to (36) obtains the k-th iterate as follows:

$$\begin{aligned}&(\boldsymbol{Y}_\parallel^{k+1}, \boldsymbol{Y}_\perp^{k+1}) && (38a)\\ &\quad := \operatorname*{argmin}_{\boldsymbol{Y}_\parallel,\boldsymbol{Y}_\perp}\ L_\rho(\boldsymbol{Y}_\parallel, \boldsymbol{Y}_\perp, \boldsymbol{F}_\parallel^k, \boldsymbol{F}_\perp^k, \boldsymbol{q}_1^k, \boldsymbol{q}_2^k, \boldsymbol{q}_3^k)\\ &(\boldsymbol{F}_\parallel^{k+1}, \boldsymbol{F}_\perp^{k+1}) && (38b)\\ &\quad := \operatorname*{argmin}_{\boldsymbol{F}_\parallel,\boldsymbol{F}_\perp}\ L_\rho(\boldsymbol{Y}_\parallel^{k+1}, \boldsymbol{Y}_\perp^{k+1}, \boldsymbol{F}_\parallel, \boldsymbol{F}_\perp, \boldsymbol{q}_1^k, \boldsymbol{q}_2^k, \boldsymbol{q}_3^k)\\ &\boldsymbol{q}_1^{k+1} := \boldsymbol{q}_1^k + \boldsymbol{T}_\parallel\mathrm{vec}(\boldsymbol{F}_\parallel^{k+1}) + \boldsymbol{T}_\perp\mathrm{vec}(\boldsymbol{F}_\perp^{k+1}) - \boldsymbol{b} && (38c)\\ &\boldsymbol{q}_2^{k+1} := \boldsymbol{q}_2^k + \mathrm{vec}(\boldsymbol{Y}_\parallel^{k+1}) - \boldsymbol{A}\mathrm{vec}(\boldsymbol{F}_\parallel^{k+1}) && (38d)\\ &\boldsymbol{q}_3^{k+1} := \boldsymbol{q}_3^k + \mathrm{vec}(\boldsymbol{Y}_\perp^{k+1}) - \boldsymbol{B}\mathrm{vec}(\boldsymbol{F}_\perp^{k+1}), && (38e)\end{aligned}$$

where

$$\begin{aligned}&L_\rho(\boldsymbol{Y}_\parallel, \boldsymbol{Y}_\perp, \boldsymbol{F}_\parallel, \boldsymbol{F}_\perp, \boldsymbol{q}_1, \boldsymbol{q}_2, \boldsymbol{q}_3)\\ &:= \eta_\parallel\big\|\boldsymbol{Y}_\parallel\big\|_* + \eta_\perp\|\boldsymbol{Y}_\perp\|_* + \|\boldsymbol{F}_\perp\|_{\mathrm{F}}^2\\ &\quad + (\rho/2)\big\|\boldsymbol{T}_\parallel\mathrm{vec}(\boldsymbol{F}_\parallel) + \boldsymbol{T}_\perp\mathrm{vec}(\boldsymbol{F}_\perp) - \boldsymbol{b} + \boldsymbol{q}_1\big\|_2^2\\ &\quad + (\rho/2)\big\|\mathrm{vec}(\boldsymbol{Y}_\parallel) - \boldsymbol{A}\mathrm{vec}(\boldsymbol{F}_\parallel) + \boldsymbol{q}_2\big\|_2^2\\ &\quad + (\rho/2)\|\mathrm{vec}(\boldsymbol{Y}_\perp) - \boldsymbol{B}\mathrm{vec}(\boldsymbol{F}_\perp) + \boldsymbol{q}_3\|_2^2\end{aligned}$$

is the so-called augmented Lagrangian with user-defined *penalty parameter* $\rho > 0$. Variables $\boldsymbol{q}_1$, $\boldsymbol{q}_2$ and $\boldsymbol{q}_3$ correspond to the Lagrange multipliers of (36).

To evaluate (38a), one can leverage the proximal operator of the nuclear norm [43, Th. 2.1]

$$\mathrm{prox}_\tau(\boldsymbol{Z}) := \operatorname*{arg\,min}_{\boldsymbol{Y}}\ ||\boldsymbol{Y}||_* + \frac{1}{2\tau}||\boldsymbol{Y} - \boldsymbol{Z}||_{\mathrm{F}}^2 = D_\tau(\boldsymbol{Z}),$$

where the *singular value shrinkage* operator D_τ is defined for $\boldsymbol{Z}$ with SVD $\boldsymbol{Z} = \boldsymbol{V}_{\boldsymbol{Z}}\boldsymbol{\Sigma}_{\boldsymbol{Z}}\boldsymbol{V}_{\boldsymbol{Z}}^\top$ as $D_\tau(\boldsymbol{Z}) := \boldsymbol{V}_{\boldsymbol{Z}}D_\tau(\boldsymbol{\Sigma}_{\boldsymbol{Z}})\boldsymbol{V}_{\boldsymbol{Z}}^\top$ with $D_\tau(\boldsymbol{\Sigma}_{\boldsymbol{Z}})$ a diagonal matrix whose (i,i)-th entry equals $\max(0, (\boldsymbol{\Sigma}_{\boldsymbol{Z}})_{i,i} - \tau)$. With this operator, (38a) becomes

$$\boldsymbol{Y}_\parallel^{k+1} = \mathrm{prox}_{\eta_\parallel/\rho}(\mathrm{vec}^{-1}(\boldsymbol{A}\mathrm{vec}(\boldsymbol{F}_\parallel^k) - \boldsymbol{q}_2^k)) \quad (39a)$$
$$\boldsymbol{Y}_\perp^{k+1} = \mathrm{prox}_{\eta_\perp/\rho}(\mathrm{vec}^{-1}(\boldsymbol{B}\mathrm{vec}(\boldsymbol{F}_\perp^k) - \boldsymbol{q}_3^k)). \quad (39b)$$

On the other hand, (38b) can be obtained in closed-form as

$$\begin{aligned}&\mathrm{vec}(\boldsymbol{F}_\parallel^{k+1}) = [\boldsymbol{T}_\parallel^\top\boldsymbol{T}_\parallel + \boldsymbol{A}^\top\boldsymbol{A}]^{-1} && (40a)\\ &[\boldsymbol{T}_\parallel^\top(\boldsymbol{b} - \boldsymbol{q}_1^k - \boldsymbol{T}_\perp\mathrm{vec}(\boldsymbol{F}_\perp^k)) + \boldsymbol{A}^\top(\boldsymbol{q}_2^k + \mathrm{vec}(\boldsymbol{Y}_\parallel^{k+1}))],\\ &\mathrm{vec}(\boldsymbol{F}_\perp^{k+1}) = [2\boldsymbol{I} + \rho\boldsymbol{T}_\perp^\top\boldsymbol{T}_\perp + \rho\boldsymbol{B}^\top\boldsymbol{B}]^{-1} && (40b)\\ &[\rho\boldsymbol{T}_\perp^\top(\boldsymbol{b} - \boldsymbol{q}_1^k - \boldsymbol{T}_\parallel\mathrm{vec}(\boldsymbol{F}_\parallel^k)) + \rho\boldsymbol{B}^\top(\boldsymbol{q}_3^k + \mathrm{vec}(\boldsymbol{Y}_\perp^{k+1}))].\end{aligned}$$

The overall ADMM algorithm is summarized as Algorithm 1 and drastically improves the converegence rate of our previous algorithm, reported in [37], since the latter is based on subgradient descent.

The computational complexity is dominated either by (39a) or by (39b), whose SVDs respectively require $O(r^6)$ and $O((N-r)^6)$ arithmetic operations, depending on which quantity is larger. This complexity is much lower than the one of general-purpose convex solvers and does not limit the values of N and r to be used in practice given the intrinsic limitations of graph filters; see Sec. VI.

APPENDIX F
ITERATIVE SOLVER FOR (P2-R)

This section presents a method to solve P2-R via ADMM. To this end, the first step is to express the objective function in a suitable form. With $\boldsymbol{A}$ and $\boldsymbol{B}$ defined in Appendix E, the first and second terms of the objective of (P2-R) are respectively proportional to

$$\begin{aligned}&\Big\|\mathrm{vec}^{-1}(\boldsymbol{A}\mathrm{vec}(\boldsymbol{U}_\parallel^\top\boldsymbol{S}\boldsymbol{U}_\parallel))\Big\|_* =\\ &\Big\|\mathrm{vec}^{-1}(\boldsymbol{A}(\boldsymbol{U}_\parallel^\top \otimes \boldsymbol{U}_\parallel^\top)\mathrm{vec}(\boldsymbol{S}))\Big\|_* = \big\|\mathrm{vec}^{-1}(\boldsymbol{A}_\parallel\mathrm{vec}(\boldsymbol{S}))\big\|_*\end{aligned}$$

Algorithm 1 Iterative solver for (P1-R).

Require: $\mathcal{E}$, $\boldsymbol{U}_{\|}$, ρ, $\eta_{\|}$, $\eta_{\perp}$, ϵ.
1: Obtain $\boldsymbol{U}_{\perp}$ s.t. $\boldsymbol{U}_{\perp}^{\top}\boldsymbol{U}_{\perp} = \boldsymbol{I}_{N-r}$ and $\boldsymbol{U}_{\|}^{\top}\boldsymbol{U}_{\perp} = \boldsymbol{0}_{r\times N}$.
2: Initialize $\boldsymbol{F}_{\|}^{0}$, $\boldsymbol{F}_{\perp}^{0}$, $\boldsymbol{q}_1^0$, $\boldsymbol{q}_2^0$, $\boldsymbol{q}_3^0, k = 0$.
3: **while** stopping_criterion_not_met() **do**
4: obtain $\boldsymbol{Y}_{\|}^{k+1}$ via (39a).
5: obtain $\boldsymbol{Y}_{\perp}^{k+1}$ via (39b).
6: obtain $\boldsymbol{F}_{\|}^{k+1}$ via (40a).
7: obtain $\boldsymbol{F}_{\perp}^{k+1}$ via (40b).
8: obtain $\boldsymbol{q}_1^{k+1}, \boldsymbol{q}_2^{k+1}, \boldsymbol{q}_3^{k+1}$ via (38).
9: $k \leftarrow k+1$.
10: **end while**
11: **return** $\boldsymbol{S} = \boldsymbol{U}_{\|}\boldsymbol{F}_{\|}^{k}\boldsymbol{U}_{\|}^{\top} + \boldsymbol{U}_{\perp}\boldsymbol{F}_{\perp}^{k}\boldsymbol{U}_{\perp}^{\top}$.

and

$$\Big\|\mathrm{vec}^{-1}(\boldsymbol{B}\mathrm{vec}(\boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\perp}))\Big\|_{*} = \Big\|\mathrm{vec}^{-1}(\boldsymbol{B}(\boldsymbol{U}_{\perp}^{\top}\otimes\boldsymbol{U}_{\perp}^{\top})\mathrm{vec}(\boldsymbol{S}))\Big\|_{*} = \big\|\mathrm{vec}^{-1}(\boldsymbol{B}_{\perp}\mathrm{vec}(\boldsymbol{S}))\big\|_{*},$$

where $\boldsymbol{A}_{\|} := \boldsymbol{A}(\boldsymbol{U}_{\|}^{\top}\otimes\boldsymbol{U}_{\|}^{\top})$ and $\boldsymbol{B}_{\perp} := \boldsymbol{B}(\boldsymbol{U}_{\perp}^{\top}\otimes\boldsymbol{U}_{\perp}^{\top})$. Noting that the nuclear norm of a block diagonal matrix equals the sum of the nuclear norms of each block enables one to compactly express the first two terms in (P2-R) as

$$\eta_{\|}\big\|\mathrm{vec}^{-1}(\boldsymbol{A}_{\|}\mathrm{vec}(\boldsymbol{S}))\big\|_{*} + \eta_{\perp}\big\|\mathrm{vec}^{-1}(\boldsymbol{B}_{\perp}\mathrm{vec}(\boldsymbol{S}))\big\|_{*} = \|\boldsymbol{Y}\|_{*}$$

where

$$\boldsymbol{Y} := \begin{bmatrix}\eta_{\|}\mathrm{vec}^{-1}(\boldsymbol{A}_{\|}\mathrm{vec}(\boldsymbol{S})) & \boldsymbol{0}_{r^2\times(N-r)^2}\\ \boldsymbol{0}_{(N-r)^2\times r^2} & \eta_{\perp}\mathrm{vec}^{-1}(\boldsymbol{B}_{\perp}\mathrm{vec}(\boldsymbol{S}))\end{bmatrix}. \quad (41)$$

On the other hand, due to the properties of the Kronecker product, the third and fourth terms in the objective of (P2-R) can be written as $\|(\boldsymbol{U}_{\perp}^{\top}\otimes\boldsymbol{U}_{\perp}^{\top})\mathrm{vec}(\boldsymbol{S})\|_2^2 + \lambda\|(\boldsymbol{U}_{\|}^{\top}\otimes\boldsymbol{U}_{\perp}^{\top})\mathrm{vec}(\boldsymbol{S})\|_2^2$.

Regarding the constraints, note that $\boldsymbol{S} = \boldsymbol{S}^{\top}$ can be expressed as $\boldsymbol{G}\mathrm{vec}(\boldsymbol{S}) = \boldsymbol{0}$, where the rows of $\boldsymbol{G} \in \mathbb{R}^{(N^2-N)/2\times N^2}$ are given by $(\boldsymbol{e}_j^{\top}\otimes\boldsymbol{e}_i^{\top} - \boldsymbol{e}_i^{\top}\otimes\boldsymbol{e}_j^{\top})$ for all $i,j = 1,\ldots,N$ such that $i<j$. Here, $\boldsymbol{e}_i$ denotes the i-th column of $\boldsymbol{I}_N$. From these considerations and using the definition of $\boldsymbol{W}$ in Sec. III-C, problem (P2-R) can be written as:

$$\begin{aligned}\underset{\boldsymbol{S},\boldsymbol{Y}}{\text{minimize}}\quad & \|\boldsymbol{Y}\|_{*} + \Big\|(\boldsymbol{U}_{\perp}^{\top}\otimes\boldsymbol{U}_{\perp}^{\top})\mathrm{vec}(\boldsymbol{S})\Big\|_2^2\\ \text{(P2-R')}\qquad & + \lambda\Big\|(\boldsymbol{U}_{\|}^{\top}\otimes\boldsymbol{U}_{\perp}^{\top})\mathrm{vec}(\boldsymbol{S})\Big\|_2^2\\ \text{s.t.}\quad & (41),\\ & \boldsymbol{T}\mathrm{vec}(\boldsymbol{S}) = \boldsymbol{b}',\end{aligned}$$

where $\boldsymbol{T} \triangleq [\boldsymbol{W}; \boldsymbol{G}; \mathrm{vec}^{\top}(\boldsymbol{U}_{\|}\boldsymbol{U}_{\|}^{\top}); \mathrm{vec}^{\top}(\boldsymbol{U}_{\perp}\boldsymbol{U}_{\perp}^{\top})]$ and $\boldsymbol{b}' \triangleq [\boldsymbol{0}; \boldsymbol{0}; r; (1\pm\epsilon)(N-r)]$. By considering each block separately, (41) holds iff

$$\begin{aligned}&\eta_{\|}\boldsymbol{A}_{\|}\mathrm{vec}(\boldsymbol{S}) = \mathrm{vec}(\boldsymbol{T}_1\boldsymbol{Y}\boldsymbol{T}_1^{\top}) = (\boldsymbol{T}_1\otimes\boldsymbol{T}_1)\mathrm{vec}(\boldsymbol{Y}),\\ &\boldsymbol{0} = \mathrm{vec}(\boldsymbol{T}_2\boldsymbol{Y}\boldsymbol{T}_1^{\top}) = (\boldsymbol{T}_1\otimes\boldsymbol{T}_2)\mathrm{vec}(\boldsymbol{Y}),\\ &\boldsymbol{0} = \mathrm{vec}(\boldsymbol{T}_1\boldsymbol{Y}\boldsymbol{T}_2^{\top}) = (\boldsymbol{T}_2\otimes\boldsymbol{T}_1)\mathrm{vec}(\boldsymbol{Y}),\\ &\eta_{\perp}\boldsymbol{B}_{\perp}\mathrm{vec}(\boldsymbol{S}) = \mathrm{vec}(\boldsymbol{T}_2\boldsymbol{Y}\boldsymbol{T}_2^{\top}) = (\boldsymbol{T}_2\otimes\boldsymbol{T}_2)\mathrm{vec}(\boldsymbol{Y}),\end{aligned}$$

where $\boldsymbol{T}_1 := [\boldsymbol{I}_{r^2\times r^2}, \boldsymbol{0}_{r^2\times(N-r)^2}]$ and $\boldsymbol{T}_2 := [\boldsymbol{0}_{(N-r)^2\times r^2}, \boldsymbol{I}_{(N-r)^2\times(N-r)^2}]$. More compactly, (41) holds iff

$$\boldsymbol{C}\mathrm{vec}(\boldsymbol{Y}) = \boldsymbol{D}\mathrm{vec}(\boldsymbol{S}) \quad (42)$$

where $\boldsymbol{C} = [\boldsymbol{T}_1\otimes\boldsymbol{T}_1; \boldsymbol{T}_1\otimes\boldsymbol{T}_2; \boldsymbol{T}_2\otimes\boldsymbol{T}_1; \boldsymbol{T}_2\otimes\boldsymbol{T}_2]$ and $\boldsymbol{D} = [\eta_{\|}\boldsymbol{A}_{\|}; \boldsymbol{0}; \boldsymbol{0}; \eta_{\perp}\boldsymbol{B}_{\perp}]$. Noting that $\boldsymbol{C}$ is orthogonal enables one to rewrite (P2-R') as

$$\begin{aligned}\underset{\boldsymbol{S},\boldsymbol{Y}}{\text{minimize}}\quad & \|\boldsymbol{Y}\|_{*} + \Big\|(\boldsymbol{U}_{\perp}^{\top}\otimes\boldsymbol{U}_{\perp}^{\top})\mathrm{vec}(\boldsymbol{S})\Big\|_2^2\\ \text{(P2-R")}\qquad & + \lambda\Big\|(\boldsymbol{U}_{\|}^{\top}\otimes\boldsymbol{U}_{\perp}^{\top})\mathrm{vec}(\boldsymbol{S})\Big\|_2^2\\ \text{s.t.}\quad & \mathrm{vec}(\boldsymbol{Y}) = \boldsymbol{C}^{\top}\boldsymbol{D}\ \mathrm{vec}(\boldsymbol{S}),\\ & \boldsymbol{T}\mathrm{vec}(\boldsymbol{S}) = \boldsymbol{b}'.\end{aligned}$$

The ADMM method in scaled form [22, Sec. 3.1.1] applied to (P2-R") reads as:

$$\boldsymbol{S}^{k+1} := \underset{\boldsymbol{S}}{\mathrm{argmin}}\ \bar{L}_{\rho}(\boldsymbol{S}, \boldsymbol{Y}^{k}, \boldsymbol{Q}_1^{k}, \boldsymbol{Q}_2^{k}) \quad (43a)$$

$$\boldsymbol{Y}^{k+1} := \underset{\boldsymbol{Y}}{\mathrm{argmin}}\ \bar{L}_{\rho}(\boldsymbol{S}^{k+1}, \boldsymbol{Y}, \boldsymbol{Q}_1^{k}, \boldsymbol{Q}_2^{k}) \quad (43b)$$

$$\boldsymbol{q}_1^{k+1} := \boldsymbol{q}_1^{k} + \mathrm{vec}(\boldsymbol{Y}^{k+1}) - \boldsymbol{C}^{\top}\boldsymbol{D}\mathrm{vec}(\boldsymbol{S}^{k+1}) \quad (43c)$$

$$\boldsymbol{q}_2^{k+1} := \boldsymbol{q}_2^{k} + \boldsymbol{T}\mathrm{vec}(\boldsymbol{S}^{k+1}) - \boldsymbol{b}', \quad (43d)$$

where

$$\begin{aligned}\bar{L}_{\rho}(\boldsymbol{S},\boldsymbol{Y},\boldsymbol{q}_1,\boldsymbol{q}_2) := {}& \|\boldsymbol{Y}\|_{*} + \Big\|(\boldsymbol{U}_{\perp}^{\top}\otimes\boldsymbol{U}_{\perp}^{\top})\mathrm{vec}(\boldsymbol{S})\Big\|_2^2\\ & + \lambda\Big\|(\boldsymbol{U}_{\|}^{\top}\otimes\boldsymbol{U}_{\perp}^{\top})\mathrm{vec}(\boldsymbol{S})\Big\|_2^2\\ & + (\rho/2)\Big\|\mathrm{vec}(\boldsymbol{Y}) - \boldsymbol{C}^{\top}\boldsymbol{D}\mathrm{vec}(\boldsymbol{S}) + \boldsymbol{q}_1\Big\|_2^2\\ & + (\rho/2)\|\boldsymbol{T}\mathrm{vec}(\boldsymbol{S}) - \boldsymbol{b}' + \boldsymbol{q}_2\|_2^2,\end{aligned}$$

where is the augmented Lagrangian with Lagrange multipliers $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$.

To express (43c) in a more convenient form, observe first from the definitions of $\boldsymbol{C}$ and $\boldsymbol{D}$ that $\boldsymbol{C}^{\top}\boldsymbol{D} = \eta_{\|}(\boldsymbol{T}_1^{\top}\otimes\boldsymbol{T}_1^{\top})\boldsymbol{A}_{\|} + \eta_{\perp}(\boldsymbol{T}_2^{\top}\otimes\boldsymbol{T}_2^{\top})\boldsymbol{B}_{\perp}$. Second, note from the properties of the Kronecker product that $\boldsymbol{A}_{\|}\mathrm{vec}(\boldsymbol{S}) = \boldsymbol{A}(\boldsymbol{U}_{\|}^{\top}\otimes\boldsymbol{U}_{\|}^{\top})\mathrm{vec}(\boldsymbol{S}) = \boldsymbol{A}\mathrm{vec}(\boldsymbol{U}_{\|}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|}) = \mathrm{vec}(\boldsymbol{U}_{\|}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|}\otimes\boldsymbol{I}_r - \boldsymbol{I}_r\otimes\boldsymbol{U}_{\|}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|})$. Similarly, $\boldsymbol{B}_{\perp}\mathrm{vec}(\boldsymbol{S}) = \mathrm{vec}(\boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\perp}\otimes\boldsymbol{I}_{N-r} - \boldsymbol{I}_{N-r}\otimes\boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\perp})$. Consequently,

$$\begin{aligned}&\boldsymbol{C}^{\top}\boldsymbol{D}\mathrm{vec}(\boldsymbol{S})\\ =&\eta_{\|}(\boldsymbol{T}_1^{\top}\otimes\boldsymbol{T}_1^{\top})\mathrm{vec}(\boldsymbol{U}_{\|}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|}\otimes\boldsymbol{I} - \boldsymbol{I}\otimes\boldsymbol{U}_{\|}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|})\\ &+\eta_{\perp}(\boldsymbol{T}_2^{\top}\otimes\boldsymbol{T}_2^{\top})\mathrm{vec}(\boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\perp}\otimes\boldsymbol{I} - \boldsymbol{I}\otimes\boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\perp})\\ =&\eta_{\|}\mathrm{vec}(\boldsymbol{T}_1^{\top}(\boldsymbol{U}_{\|}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|}\otimes\boldsymbol{I} - \boldsymbol{I}\otimes\boldsymbol{U}_{\|}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|})\boldsymbol{T}_1) +\\ &\eta_{\perp}\mathrm{vec}(\boldsymbol{T}_2^{\top}(\boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\perp}\otimes\boldsymbol{I} - \boldsymbol{I}\otimes\boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\perp})\boldsymbol{T}_2)\\ =&\mathrm{vec}[\eta_{\|}(\boldsymbol{U}_{\|}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|}\otimes\boldsymbol{I} - \boldsymbol{I}\otimes\boldsymbol{U}_{\|}^{\top}\boldsymbol{S}\boldsymbol{U}_{\|}), \boldsymbol{0};\\ &\boldsymbol{0}, \eta_{\perp}(\boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\perp}\otimes\boldsymbol{I} - \boldsymbol{I}\otimes\boldsymbol{U}_{\perp}^{\top}\boldsymbol{S}\boldsymbol{U}_{\perp})].\end{aligned} \quad (44)$$

Algorithm 2 Iterative solver for (P2-R).

Require: $\mathcal{E}$, $\boldsymbol{U}_\parallel$, ρ, λ, $\eta_\parallel$, $\eta_\perp$, ϵ.
1: Obtain $\boldsymbol{U}_\perp$ s.t. $\boldsymbol{U}_\perp^\top \boldsymbol{U}_\perp = \boldsymbol{I}_{N-r}$ and $\boldsymbol{U}_\parallel^\top \boldsymbol{U}_\perp = \boldsymbol{0}_{r\times N}$.
2: Initialize $\boldsymbol{Y}_1^0$, $\boldsymbol{Y}_2^0$, $\boldsymbol{Q}_{1,1}^0$, $\boldsymbol{Q}_{1,2}^0$, $\boldsymbol{q}_2^0$, $k=0$
3: **while** stopping_criterion_not_met() **do**
4: obtain $\boldsymbol{S}^{k+1}$ via (49).
5: obtain $\boldsymbol{Y}_1^{k+1}$, $\boldsymbol{Y}_2^{k+1}$, $\boldsymbol{Q}_{1,1}^{k+1}$, $\boldsymbol{Q}_{1,2}^{k+1}$ via (47).
6: obtain $\boldsymbol{q}_2^{k+1}$ via (43d).
7: $k \leftarrow k+1$.
8: **end while**
9: **return** $\boldsymbol{S}^k$

The update (43c) can therefore be expressed upon defining $\boldsymbol{Q}_1^k := \mathrm{vec}^{-1}(\boldsymbol{q}_1^k)$ as

$$\begin{aligned}\boldsymbol{Q}_1^{k+1} &:= \boldsymbol{Q}_1^k + \boldsymbol{Y}^{k+1} \\ &- [\eta_\parallel(\boldsymbol{U}_\parallel^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\parallel \otimes \boldsymbol{I}_r - \boldsymbol{I}_r \otimes \boldsymbol{U}_\parallel^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\parallel), \boldsymbol{0}; \\ &\quad \boldsymbol{0}, \eta_\perp(\boldsymbol{U}_\perp^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\perp \otimes \boldsymbol{I}_{N-r} - \boldsymbol{I}_{N-r} \otimes \boldsymbol{U}_\perp^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\perp)].\end{aligned} \tag{45}$$

Furthermore, by using the proximal operator of the nuclear norm (see Appendix E), the update in (43b) becomes

$$\begin{aligned}\boldsymbol{Y}^{k+1} &= \mathrm{prox}_{1/\rho}(\mathrm{vec}^{-1}(\boldsymbol{C}^\top \boldsymbol{D}\mathrm{vec}(\boldsymbol{S}^k) - \boldsymbol{q}_1^k)) \\ &= \mathrm{prox}_{1/\rho}([\eta_\parallel(\boldsymbol{U}_\parallel^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\parallel \otimes \boldsymbol{I}_r - \boldsymbol{I}_r \otimes \boldsymbol{U}_\parallel^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\parallel), \boldsymbol{0}; \\ &\boldsymbol{0}, \eta_\perp(\boldsymbol{U}_\perp^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\perp \otimes \boldsymbol{I}_{N-r} - \boldsymbol{I}_{N-r} \otimes \boldsymbol{U}_\perp^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\perp)] - \boldsymbol{Q}_1^k),\end{aligned} \tag{46}$$

where the second equality follows from (44).

If $\boldsymbol{Q}_1^0$ is initialized as a block diagonal matrix with diagonal blocks of size $r\times r$ and $(N-r)\times(N-r)$, then it follows from (45) and (46) that $\boldsymbol{Y}^k$ and $\boldsymbol{Q}_1^k$ will remain block diagonal with blocks of the same size. These blocks can be updated as

$$\begin{aligned}\boldsymbol{Y}_1^{k+1} =& \mathrm{prox}_{1/\rho}\Big(\eta_\parallel(\boldsymbol{U}_\parallel^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\parallel \otimes \boldsymbol{I}_r \\ &- \boldsymbol{I}_r \otimes \boldsymbol{U}_\parallel^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\parallel) - \boldsymbol{Q}_{1,1}^k\Big)\end{aligned} \tag{47a}$$

$$\begin{aligned}\boldsymbol{Y}_2^{k+1} =& \mathrm{prox}_{1/\rho}\Big(\eta_\perp(\boldsymbol{U}_\perp^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\perp \otimes \boldsymbol{I}_{N-r} \\ &- \boldsymbol{I}_{N-r} \otimes \boldsymbol{U}_\perp^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\perp) - \boldsymbol{Q}_{1,2}^k\Big)\end{aligned} \tag{47b}$$

$$\begin{aligned}\boldsymbol{Q}_{1,1}^{k+1} =& \boldsymbol{Q}_{1,1}^k + \boldsymbol{Y}_1^{k+1} \\ &- \eta_\parallel(\boldsymbol{U}_\parallel^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\parallel \otimes \boldsymbol{I} - \boldsymbol{I} \otimes \boldsymbol{U}_\parallel^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\parallel)\end{aligned} \tag{47c}$$

$$\begin{aligned}\boldsymbol{Q}_{1,2}^{k+1} =& \boldsymbol{Q}_{1,2}^k + \boldsymbol{Y}_2^{k+1} \\ &- \eta_\perp(\boldsymbol{U}_\perp^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\perp \otimes \boldsymbol{I} - \boldsymbol{I} \otimes \boldsymbol{U}_\perp^\top \boldsymbol{S}^{k+1}\boldsymbol{U}_\perp).\end{aligned} \tag{47d}$$

The $\boldsymbol{S}$-update (43a) can be obtained in closed form as

$$\begin{aligned}\mathrm{vec}(\boldsymbol{S}^{k+1}) =& \Big[\boldsymbol{D}^\top \boldsymbol{D} + \boldsymbol{T}^\top \boldsymbol{T} + (2\lambda/\rho)(\boldsymbol{U}_\parallel \boldsymbol{U}_\parallel^\top \otimes \boldsymbol{U}_\perp \boldsymbol{U}_\perp^\top) \\ &+ (2/\rho)(\boldsymbol{U}_\perp \boldsymbol{U}_\perp^\top \otimes \boldsymbol{U}_\perp \boldsymbol{U}_\perp^\top)\Big]^{-1} \\ &\Big[\boldsymbol{D}^\top \boldsymbol{C}(\mathrm{vec}(\boldsymbol{Y}^k) + \boldsymbol{q}_1^k) + \boldsymbol{T}^\top(\boldsymbol{b}' - \boldsymbol{q}_2^k)\Big]\end{aligned} \tag{48}$$

by using the orthogonality of $\boldsymbol{C}$. To reduce the computational complexity of (48), note that

$$\begin{aligned}&\boldsymbol{D}^\top \boldsymbol{C}\mathrm{vec}(\boldsymbol{Y}^k) = \eta_\parallel \boldsymbol{A}_\parallel^\top(\boldsymbol{T}_1 \otimes \boldsymbol{T}_1)\mathrm{vec}(\boldsymbol{Y}^k) + \\ &\eta_\perp \boldsymbol{B}_\perp^\top(\boldsymbol{T}_2 \otimes \boldsymbol{T}_2)\mathrm{vec}(\boldsymbol{Y}^k) = \eta_\parallel \boldsymbol{A}_\parallel^\top \mathrm{vec}(\boldsymbol{Y}_1^k) + \\ &\eta_\perp \boldsymbol{B}_\perp^\top \mathrm{vec}(\boldsymbol{Y}_2^k)\end{aligned}$$

and

$$\boldsymbol{D}^\top \boldsymbol{C}\boldsymbol{q}_1^k = \eta_\parallel \boldsymbol{A}_\parallel^\top \mathrm{vec}(\boldsymbol{Q}_{1,1}^k) + \eta_\perp \boldsymbol{B}_\perp^\top \mathrm{vec}(\boldsymbol{Q}_{1,2}^k)$$

to obtain

$$\begin{aligned}\mathrm{vec}(\boldsymbol{S}^{k+1}) =& \Big[\boldsymbol{D}^\top \boldsymbol{D} + \boldsymbol{T}^\top \boldsymbol{T} + (2\lambda/\rho)(\boldsymbol{U}_\parallel \boldsymbol{U}_\parallel^\top \otimes \boldsymbol{U}_\perp \boldsymbol{U}_\perp^\top) \\ &+ (2/\rho)(\boldsymbol{U}_\perp \boldsymbol{U}_\perp^\top \otimes \boldsymbol{U}_\perp \boldsymbol{U}_\perp^\top)\Big]^{-1} \Big[\eta_\parallel \boldsymbol{A}_\parallel^\top \mathrm{vec}(\boldsymbol{Y}_1^k + \boldsymbol{Q}_{1,1}^k) \\ &+ \eta_\perp \boldsymbol{B}_\perp^\top \mathrm{vec}(\boldsymbol{Y}_2^k + \boldsymbol{Q}_{1,2}^k) + \boldsymbol{T}^\top(\boldsymbol{b}' - \boldsymbol{q}_2^k)\Big].\end{aligned} \tag{49}$$

The overall procedure is summarized as Algorithm 2. Its complexity is dominated by the inversion in (49), which involves $O(N^6)$ arithmetic operations. This complexity is much lower than the one of general-purpose convex solvers and does not limit the values of N to be used in practice given the intrinsic limitations of graph filters; see Sec. VI.

Appendix G
Proof of Theorem 3

Given any vector $\boldsymbol{x} := [x_1, \ldots, x_{NB}]^\top$, one can collect its $L := \mathcal{L}(\boldsymbol{x})$ distinct entries into the vector $\bar{\boldsymbol{z}} := [\bar{z}_1, \ldots, \bar{z}_L]^\top$ and construct a partition map $\ell : \{1, \ldots, NB\} \to \{1, \ldots, L\}$ such that $x_i = \bar{z}_{\ell(i)}$. With this notation and letting $\boldsymbol{a}_i$ denote the i-th column of $[\boldsymbol{A}_1, \ldots, \boldsymbol{A}_B]$, one can express the first block row of $\boldsymbol{A}\boldsymbol{x}$ as $\sum_{l=1}^L \bar{\boldsymbol{a}}_l \bar{z}_l$, where $\bar{\boldsymbol{a}}_l := \sum_{i:\ell(i)=l} \boldsymbol{a}_i$, $l = 1, \ldots, L$. Thus, the system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ admits a solution iff there exist $\boldsymbol{x}$, $\bar{\boldsymbol{z}}$, and ℓ satisfying the aforementioned relations such that $\bar{\boldsymbol{A}}\bar{\boldsymbol{z}} = \boldsymbol{0}$, $\sum_i \boldsymbol{E}_i \boldsymbol{x}_i = \boldsymbol{0}$ and $\boldsymbol{1}^\top \boldsymbol{x} = NB(B+1)/2$, where $\bar{\boldsymbol{A}} := [\bar{\boldsymbol{a}}_1, \ldots, \bar{\boldsymbol{a}}_L]$ and $\boldsymbol{x}_i \in \mathbb{R}^N$, $i = 1, \ldots, B$, is such that $\boldsymbol{x} = [\boldsymbol{x}_1^\top, \ldots, \boldsymbol{x}_B^\top]^\top$.

To prove part (a), observe that, regardless of the partition ℓ, the fact that the columns of $\boldsymbol{A}_i$ are independently drawn from a continuous distribution implies that $\{\bar{\boldsymbol{a}}_l\}_l$ also adhere to a continuous distribution and are independent of each other. Thus $\mathrm{rank}(\bar{\boldsymbol{A}}) = \min(M, L)$ with probability 1. A vector $\boldsymbol{x}$ satisfying $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ exists iff the homogeneous system $\bar{\boldsymbol{A}}\bar{\boldsymbol{z}} = \boldsymbol{0}$ admits at least a non-zero solution. This is because any non-zero solution $\bar{\boldsymbol{z}}$ can be normalized to satisfy $\boldsymbol{1}^\top \boldsymbol{x} = NB(B+1)/2$ and thus $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$. Given that $\mathrm{rank}(\bar{\boldsymbol{A}}) = \min(M, L)$ with probability 1, the system $\bar{\boldsymbol{A}}\bar{\boldsymbol{z}} = \boldsymbol{0}$ admits a non-zero solution iff $L > M$. The proof is concluded by noting that the minimum L satisfying this condition is $M+1$. The hypothesis $NB > M$ is necessary because $L \leq NB$.

To prove part (b) let $\check{\boldsymbol{A}} := [\boldsymbol{a}_{1,1}, \ldots, \boldsymbol{a}_{1,N-1}, \boldsymbol{a}_{2,1}, \ldots, \boldsymbol{a}_{B,N-1}]$ collect the independent vectors in $\{\boldsymbol{a}_{i,j}\}_{i,j}$ and note that $[\boldsymbol{A}_1, \ldots, \boldsymbol{A}_B] = \check{\boldsymbol{A}}\boldsymbol{T}$, where

$$\boldsymbol{T} := \begin{bmatrix} \boldsymbol{I}_{N-1} & -\boldsymbol{1} & \boldsymbol{0} & \boldsymbol{0} & \ldots & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I}_{N-1} & -\boldsymbol{1} & \ldots & \boldsymbol{0} & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \ldots & \boldsymbol{I}_{N-1} & -\boldsymbol{1} \end{bmatrix}.$$

Thus, the system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ admits a solution iff there exists $\boldsymbol{x}$ such that $\check{\boldsymbol{A}}\boldsymbol{T}\boldsymbol{x} = \boldsymbol{0}$, $\sum_i \boldsymbol{E}_i\boldsymbol{x}_i = \boldsymbol{0}$ and $\boldsymbol{1}^\top\boldsymbol{x} = NB(B+1)/2$. Observe that, by setting $\boldsymbol{x}_i = i\boldsymbol{1}\ \forall i$, the resulting $\boldsymbol{x}$ satisfies these three conditions, which establishes that $\mathcal{L}^* \leq \mathcal{L}(\boldsymbol{x}) = B$. Besides, it can be easily seen that this solution is the only that satisfies $\boldsymbol{T}\boldsymbol{x} = \boldsymbol{0}$, $\sum_i \boldsymbol{E}_i\boldsymbol{x}_i = \boldsymbol{0}$, and $\boldsymbol{1}^\top\boldsymbol{x} = NB(B+1)/2$. In particular, this implies that $\check{\boldsymbol{A}}\boldsymbol{T}\boldsymbol{x} = \boldsymbol{0}$ holds regardless of the value of $\check{\boldsymbol{A}}$. It remains only to show that, with probability 1, there exists no solution $\boldsymbol{x}'$ with $\mathcal{L}(\boldsymbol{x}') < B$.

To this end, note that the set of realizations of $\check{\boldsymbol{A}}$ for which such an $\boldsymbol{x}'$ exists is the union over partitions ℓ of the sets of realizations of $\check{\boldsymbol{A}}$ for which a solution $\boldsymbol{x}'$ with partition ℓ exists. Since there are only finitely many partitions, it suffices to show that the probability of finding such an $\boldsymbol{x}'$ is 0 for a single generic partition. Let ℓ denote such a partition and observe that $\boldsymbol{x}$ can be expressed as $\boldsymbol{x} = \boldsymbol{S}_\ell\bar{\boldsymbol{z}}$ for some $NB \times L$ matrix $\boldsymbol{S}_\ell$ with ones and zeros. The probability that there exists a solution is $\mathbb{P}[\exists\bar{\boldsymbol{z}} \in \mathcal{Z}_\ell : \check{\boldsymbol{A}}\boldsymbol{T}\boldsymbol{S}_\ell\bar{\boldsymbol{z}} = \boldsymbol{0}]$ where $\mathcal{Z}_\ell := \{\bar{\boldsymbol{z}} :\ \boldsymbol{x} = \boldsymbol{S}_\ell\bar{\boldsymbol{z}}$ satisfies $\sum_i \boldsymbol{E}_i\boldsymbol{x}_i = \boldsymbol{0}$ and $\boldsymbol{1}^\top\boldsymbol{x} = NB(B+1)/2\}$. If $L < B$, the product $\boldsymbol{T}\boldsymbol{S}_\ell\bar{\boldsymbol{z}}$ is necessarily non-zero for $\bar{\boldsymbol{z}} \in \mathcal{Z}_\ell$, as described earlier. Thus, the aforementioned probability equals

$$\mathbb{P}[\exists\bar{\boldsymbol{z}} \in \mathcal{Z}_\ell : \check{\boldsymbol{A}}\boldsymbol{T}\boldsymbol{S}_\ell\bar{\boldsymbol{z}} = \boldsymbol{0}, \boldsymbol{T}\boldsymbol{S}_\ell\bar{\boldsymbol{z}} \neq \boldsymbol{0}] \tag{50a}$$

$$\leq \mathbb{P}[\exists\bar{\boldsymbol{t}} \in \mathcal{R}\{\boldsymbol{T}\boldsymbol{S}_\ell\} : \check{\boldsymbol{A}}\bar{\boldsymbol{t}} = \boldsymbol{0}, \bar{\boldsymbol{t}} \neq \boldsymbol{0}] \tag{50b}$$

$$= \mathbb{P}[\exists\boldsymbol{v} \in \mathbb{R}^r : \check{\boldsymbol{A}}\boldsymbol{Q}_\ell\boldsymbol{v} = \boldsymbol{0}, \boldsymbol{v} \neq \boldsymbol{0}] \tag{50c}$$

$$= \mathbb{P}[\mathrm{rank}[\check{\boldsymbol{A}}\boldsymbol{Q}_\ell] < r], \tag{50d}$$

where $\boldsymbol{Q}_\ell$ is a matrix whose r linearly independent columns constitute a basis for $\mathcal{R}\{\boldsymbol{T}\boldsymbol{S}_\ell\}$. Noting that $r \leq L < B$ and, by hypothesis, $M \geq B$, it follows that $\mathrm{rank}[\check{\boldsymbol{A}}\boldsymbol{Q}_\ell] < r$ is only possible if the M rows of $\check{\boldsymbol{A}}$ lie in a proper subspace of $\mathcal{R}\{\boldsymbol{T}\boldsymbol{S}_\ell\}$. Since proper subspaces have zero Lebesgue measure and $\check{\boldsymbol{A}}$ obeys a continuous probability distribution, it follows that $\mathbb{P}[\mathrm{rank}[\check{\boldsymbol{A}}\boldsymbol{Q}_\ell] < r] = 0$ and, consequently, $\mathbb{P}[\exists\bar{\boldsymbol{z}} \in \mathcal{Z}_\ell : \check{\boldsymbol{A}}\boldsymbol{T}\boldsymbol{S}_\ell\bar{\boldsymbol{z}} = \boldsymbol{0}] = 0$.