A Regularization Framework for Learning over Multitask Graphs

Roula Nassif, *Member, IEEE*, Stefan Vlaski, *Student Member, IEEE*,
Cédric Richard, *Senior Member, IEEE*, Ali H. Sayed, *Fellow Member, IEEE*

***Abstract*—This letter proposes a general regularization framework for inference over multitask networks. The optimization approach relies on minimizing a global cost consisting of the aggregate sum of individual costs regularized by a term that allows to incorporate global information about the graph structure and the individual parameter vectors into the solution of the inference problem. An adaptive strategy, which responds to streaming data and employs stochastic approximations in place of actual gradient vectors, is devised and studied. Methods allowing the distributed implementation of the regularization step are also discussed. The work shows how to blend real-time adaptation with graph filtering and a generalized regularization framework to result in a graph diffusion strategy for distributed learning over multitask networks.**

***Index Terms*—Multitask graphs, spectral based regularization, gradient noise, distributed implementation.**

I. Introduction

Learning over networks allows a collection of interconnected agents to perform parameter estimation tasks from streaming data by relying on local computations and communications with immediate neighbors [1]–[8]. In recent years, there has also been interest in learning algorithms that operate over multitask networks, where agents need to estimate and track multiple objectives simultaneously [9]–[18]. Although agents may generally have distinct though related tasks to perform, they may still be able to capitalize on inductive transfer between them to improve their estimation accuracy. Regularization is one of the most fundamental techniques that allows to incorporate prior information about how tasks are related to each other into the formulation and solution of the inference problem [15]–[20].

This work introduces a family of regularization operators for multitask learning over networks. We consider multitask estimation problems where each agent in the network seeks to minimize an individual cost expressed as the expectation of some loss function while enforcing graph constraints, which may include consensus [1], [2] and smoothness [16], [18] as special cases. We formulate the problem as the minimization of the aggregate sum of individual costs regularized by a term that allows to incorporate information about the structure of the tasks in the graph spectral domain into the solution of the inference problem. An adaptive strategy is devised that responds to streaming data and employs stochastic approximations in place of actual gradient vectors, which are generally unavailable. We establish, under conditions on the step-size learning parameter μ, that the strategy converges in the mean-square-error sense within $O(\mu)$ from the solution of the regularized problem. While most existing multitask strategies assume network proximity constraints and formulate convex optimization problems with appropriate co-regularizers between neighboring agents [11]–[18], the current regularization framework is concerned with the spectral properties of the graph signal to be estimated. This distinctive feature favors solutions that cannot be directly implemented in a distributed manner. Based on the concept of graph filters [21]–[25], methods allowing the distributed implementation of the regularization step are also provided. In this way, the main novelty in this work is to show how to blend three concepts: real-time adaptation, graph filtering, and generalized regularization, in order to obtain an effective graph diffusion strategy for distributed learning over multitask networks.

II. Problem formulation

Consider a connected network of N nodes. Let $w_k \in \mathbb{R}^M$ denote some parameter vector at node k and let $\mathcal{W} = \text{col}\{w_1, \ldots, w_N\}$ denote the collection of parameter vectors from across the network. We associate with each agent k a risk function $J_k(w_k) : \mathbb{R}^M \rightarrow \mathbb{R}$ assumed to be strongly convex. In most learning and adaptation problems, the risk function is expressed as the expectation of a loss function $Q_k(\cdot)$ and is written as $J_k(w_k) = \mathbb{E}Q_k(w_k; \boldsymbol{x}_k)$, where $\boldsymbol{x}_k$ denotes the random data. The expectation is computed over the distribution of the data (note that, in our notation, we use boldface letters for random quantities and normal letters for deterministic quantities). We denote the unique minimizer of $J_k(w_k)$ by w_k^o. In many situations, there is available some information about $\mathcal{W}^o = \text{col}\{w_1^o, \ldots, w_N^o\}$, such as knowing that $\mathcal{W}^o$ is smooth with respect to the underlying graph [18]. One way to exploit this information is to employ regularization to favor solutions with the desired properties. We consider in this work multitask learning problems of the form:

$$\mathcal{W}_\eta^o = \arg\min_{\mathcal{W}} J^{\text{glob}}(\mathcal{W}) = \sum_{k=1}^{N} J_k(w_k) + \frac{\eta}{2}\mathcal{W}^\top \mathcal{R}\mathcal{W}, \qquad (1)$$

where $\mathcal{R} \in \mathbb{R}^{MN\times MN}$ is a positive semi-definite regularization matrix. The tuning parameter $\eta \geq 0$ controls the tradeoff between the two components of the objective function. In practice, and as we shall see in the sequel, the selection of the regularizer $\mathcal{R}$ must account for prior information on the structure of $\mathcal{W}^o$ in the graph spectral domain [26]–[30].

A. Theoretical motivation for the optimization framework

For motivational purposes, we provide in the following a probabilistic interpretation for problem (1). Let us consider for example MSE networks [1] where each agent k, at every instant i, has access to a measurement $\boldsymbol{d}_k(i)$ and a regression vector $\boldsymbol{u}_{k,i}$, assumed to be related via the linear model:

The work of A. H. Sayed was supported in part by NSF grant CCF-1524250. R. Nassif, S. Vlaski, and A. H. Sayed are with Institute of Electrical Engineering, EPFL, Switzerland (e-mail: {roula.nassif, stefan.vlaski, ali.sayed}@epfl.ch). C. Richard is with Université de Nice Sophia-Antipolis, France (e-mail: cedric.richard@unice.fr).

$$\boldsymbol{d}_k(i) = \boldsymbol{u}_{k,i}^\top w_k^o + \boldsymbol{v}_k(i), \quad k = 1, \ldots, N, \tag{2}$$

for some unknown $M \times 1$ vector w_k^o with $\boldsymbol{v}_k(i)$ denoting a measurement noise. For these networks, the risk functions take the form of mean-square-errors (MSE):

$$J_k(w_k) = \frac{1}{2}\mathbb{E}|\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i}^\top w_k|^2, \quad k = 1, \ldots, N. \tag{3}$$

The processes $\{\boldsymbol{d}_k(i), \boldsymbol{u}_{k,i}, \boldsymbol{v}_k(i)\}$ are assumed to represent zero-mean jointly wide-sense stationary random processes satisfying: i) $\mathbb{E}\boldsymbol{u}_{k,i}\boldsymbol{u}_{\ell,j}^\top = R_{u,k}\delta_{k,\ell}\delta_{i,j}$ where $R_{u,k} > 0$ and the Kronecker delta $\delta_{m,n} = 1$ if $m = n$ and zero otherwise; ii) $\mathbb{E}\boldsymbol{v}_k(i)\boldsymbol{v}_\ell(j) = \sigma_{v,k}^2\delta_{k,\ell}\delta_{i,j}$; iii) the regression and noise processes $\{\boldsymbol{u}_{\ell,j}, \boldsymbol{v}_k(i)\}$ are independent of each other.

Lemma 1. *If the network vector is degenerate Gaussian multivariate distributed $\boldsymbol{\mathcal{W}} \sim \mathcal{N}(0, \mathcal{R}^\dagger)$ and if the noise process is Gaussian $\boldsymbol{v}_k(i) \sim \mathcal{N}(0, \sigma_{v,k}^2)$ independent over space and time and identically distributed, then problem* (1) *is a MAP estimator for $\boldsymbol{\mathcal{W}}$ conditioned on $\{\boldsymbol{d}_k(i), \boldsymbol{u}_{k,i}\}$.*

Proof. The proof is similar to Lemma 1 in [18] with the matrix $\mathcal{L}$ in [18] replaced by the matrix $\mathcal{R}$. When $\sigma_{v,k}^2 = \sigma_v^2$ $\forall k$, the optimal choice of η is σ_v^2. □

B. Regularization via the Graph Laplacian

Let us assume that the graph is endowed with a symmetric weighted adjacency $N \times N$ *block* matrix $\mathcal{A}$. If there is an edge connecting nodes k and ℓ, then the (k, ℓ)-th $M \times M$ positive semi-definite block $[\mathcal{A}]_{k\ell} = A_{k\ell} = A_{\ell k}$ reflects the relation between k and ℓ; otherwise, $A_{k\ell} = 0$[1]. By analogy to the scalar setting [18], we introduce the graph Laplacian, which is a differential operator defined as $\mathcal{L} = \mathcal{D} - \mathcal{A}$, where the degree matrix $\mathcal{D}$ is an $N \times N$ block diagonal matrix with k-th block entry $D_{kk} = \sum_{\ell=1}^N A_{k\ell}$. Let $\mathcal{N}_k$ denote the set of neighbors of k, i.e., the set of nodes connected to agent k by an edge. Since $\mathcal{W}^\top\mathcal{L}\mathcal{W} = \frac{1}{2}\sum_{k=1}^N\sum_{\ell\in\mathcal{N}_k}\|w_k - w_\ell\|_{A_{k\ell}}^2 \geq 0$ $\forall \mathcal{W}$, the matrix $\mathcal{L}$ is symmetric positive semi-definite and possesses a complete set of orthonormal eigenvectors. We denote them by $\{v_1, \ldots, v_{MN}\}$. For convenience, we order the set of real, non-negative eigenvalues of $\mathcal{L}$ as $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_{MN} = \lambda_{\max}$. Thus, the Laplacian can be decomposed as $\mathcal{L} = \mathcal{V}\Lambda\mathcal{V}^\top$ with $\Lambda = \text{diag}\{\lambda_1, \ldots, \lambda_{MN}\}$ and $\mathcal{V} = [v_1, \ldots, v_{MN}]$.

A class of regularization functionals on graphs, which is built upon the notion of graph smoothness, can be defined as [27]:

$$S(\mathcal{W}) = \langle \mathcal{W}, r(\mathcal{L})\mathcal{W}\rangle = \mathcal{W}^\top r(\mathcal{L})\mathcal{W}, \tag{4}$$

where $r(\cdot)$ is some well-defined non-negative function on the spectrum $\sigma(\mathcal{L}) = \{\lambda_1, \ldots, \lambda_{MN}\}$ and $r(\mathcal{L})$ is the corresponding *matrix function* defined as [31]:

$$r(\mathcal{L}) = \mathcal{V}r(\Lambda)\mathcal{V}^\top = \sum_{m=1}^{MN} r(\lambda_m)v_m v_m^\top. \tag{5}$$

TABLE I
Examples of spectral graph functions $r(\lambda)$

Kernel name	Spectral function $r(\lambda)$
Laplacian with $p \geq 1$ [22], [27]	λ^p
Diffusion process [26], [27]	$e^{\sigma^2\lambda/2}$
p-step random walk ($p \geq 1$, $a > \lambda_{\max}$) [27]	$1/(a - \lambda)^p$
$\lvert\mathcal{B}\rvert$-bandlimited [29], [32]	$r(\lambda_m) = \begin{cases} 0, & \text{if } \lambda_m \in \mathcal{B} \\ \beta, & \text{otherwise} \end{cases}$

Construction (4) uses the Laplacian as a means to design regularization operators. Requiring positive semi-definite $\mathcal{R}$ in (1) imposes the constraint $r(\lambda) \geq 0$ for all $\lambda \in \sigma(\mathcal{L})$. Replacing (5) into (4), we obtain:

$$S(\mathcal{W}) = \overline{\mathcal{W}}^\top r(\Lambda)\overline{\mathcal{W}} = \sum_{m=1}^{MN} r(\lambda_m)|\overline{\mathcal{W}}_m|^2, \tag{6}$$

where $\overline{\mathcal{W}} = \mathcal{V}^\top\mathcal{W} = \text{col}\{\overline{\mathcal{W}}_m\}_{m=1}^{MN}$, and $\overline{\mathcal{W}}_m = v_m^\top\mathcal{W}$. The regularization $S(\mathcal{W})$ in (6) promotes a particular structure in the graph spectral domain. It strongly penalizes $|\overline{\mathcal{W}}_m|^2$ for which the corresponding $r(\lambda_m)$ is large. Thus, one prefers $r(\lambda_m)$ to be large for those $|\overline{\mathcal{W}}_m|^2$ that are small and vice versa. The function $r(\lambda)$ is commonly chosen to be monotonically increasing in λ [27]. Table I lists some examples of typical choices for $r(\lambda)$ [27].

III. Adaptive solution

A. Adaptive solution

Our objective is to devise and study a strategy that solves problem (1) where each agent k is interested in estimating the k-th subvector $w_{k,\eta}^o$ of $\mathcal{W}_\eta^o = \text{col}\{w_{1,\eta}^o, \ldots, w_{N,\eta}^o\}$. We are particularly interested in solving the problem in the stochastic setting when the distribution of the data $\boldsymbol{x}_k$ in $J_k(w_k) = \mathbb{E}Q_k(w_k; \boldsymbol{x}_k)$ is generally unknown. As such, approximate gradient vectors need to be employed. A common construction in stochastic approximation theory is to employ the following approximation at iteration i [1]:

$$\widehat{\nabla_{w_k} J}_k(w_k) = \nabla_{w_k} Q_k(w_k; \boldsymbol{x}_{k,i}), \tag{7}$$

where $\boldsymbol{x}_{k,i}$ represents the data observed at iteration i. The difference between the true gradient and its approximation is called the gradient noise denoted by:

$$\boldsymbol{s}_{k,i}(w) = \nabla_{w_k} J_k(w) - \widehat{\nabla_{w_k} J}_k(w). \tag{8}$$

In order to estimate $\mathcal{W}_\eta^o$, we may start by employing a stochastic gradient descent update of the form:

$$\boldsymbol{\mathcal{W}}_i = \boldsymbol{\mathcal{W}}_{i-1} - \mu\,\text{col}\left\{\widehat{\nabla_{w_k} J}_k(\boldsymbol{w}_{k,i-1})\right\}_{k=1}^N - \mu\eta\mathcal{R}\boldsymbol{\mathcal{W}}_{i-1}, \tag{9}$$

where $\mu > 0$ is a small step-size parameter and $\boldsymbol{\mathcal{W}}_i = \text{col}\{\boldsymbol{w}_{1,i}, \ldots, \boldsymbol{w}_{N,i}\}$ is the estimate of $\mathcal{W}_\eta^o$ at time instant i. By introducing an auxiliary variables $\boldsymbol{\psi}_{k,i}$ at each agent k, strategy (9) can be implemented in an incremental manner:

$$\begin{cases} \boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu\widehat{\nabla_{w_k} J}_k(\boldsymbol{w}_{k,i-1}), \quad k = 1, \ldots, N, \\ \boldsymbol{\mathcal{W}}_i = \boldsymbol{\psi}_i - \mu\eta\mathcal{R}\boldsymbol{\psi}_i, \end{cases} \tag{10}$$

where $\boldsymbol{\psi}_i = \text{col}\{\boldsymbol{\psi}_{1,i}, \ldots, \boldsymbol{\psi}_{N,i}\}$ and where we replaced $\mathcal{R}\boldsymbol{\mathcal{W}}_{i-1}$ by $\mathcal{R}\boldsymbol{\psi}_i$ since we expect $\boldsymbol{\psi}_i$ to be an improved estimate compared to $\boldsymbol{\mathcal{W}}_{i-1}$. Let $R_{k\ell}$ denote the (k, ℓ)-th block of $\mathcal{R}$. In order to compute $\boldsymbol{w}_{k,i}$, agent k needs to evaluate from the second step in (10) the following expression:

$$\boldsymbol{w}_{k,i} = \boldsymbol{\psi}_{k,i} - \mu\eta\sum_{\ell=1}^N R_{k\ell}\boldsymbol{\psi}_{\ell,i}. \tag{11}$$

[1] Note that, it is common in the literature to associate non-negative scalars $a_{k\ell}$ with links [15]–[18]. In this work, we propose to associate non-negative *block* matrices instead, denoted by $A_{k\ell}$, since matrices are able to capture more thoroughly relationships between the components of the tasks (or vectors) at the agents. The scalar case can be recovered from the current framework by replacing $A_{k\ell}$ by $a_{k\ell}I_M$.

This calculation requires exchange of information between agent k and every agent ℓ (for which $R_{k\ell} \neq 0$), and some of these agents may not be in the direct neighborhood of k. Thus, although the first step in (10) can be performed locally at agent k, the second step may require non-local communications and is still non-distributed. In Section IV, we shall explain how strategy (10) can be implemented in a distributed manner.

B. Performance analysis

Since the iterates $\boldsymbol{w}_{k,i}$ generated by (10) are random, we shall measure performance by examining the average squared distance between $\boldsymbol{w}_{k,i}$ and $w^o_{k,\eta}$, $\lim_{i\to\infty}\mathbb{E}\|w^o_{k,\eta}-\boldsymbol{w}_{k,i}\|^2$. We analyze (10) under the following assumptions on the risks $\{J_k(\cdot)\}$ and on the gradient noise processes $\{\boldsymbol{s}_{k,i}(\cdot)\}$ defined in (8). As explained in [1], these conditions are satisfied by many objective functions of interest in learning and adaptation such as quadratic and logistic risks. Besides, regularization is a common technique to ensure strong convexity.

Assumption 1. *The individual costs $J_k(w_k)$ are assumed to be twice differentiable and strongly convex such that:*

$$0 < \lambda_{k,\min} I_M \leq \nabla^2_{w_k} J_k(w_k) \leq \lambda_{k,\max} I_M, \qquad (12)$$

where $\lambda_{k,\min} > 0$ for $k = 1, \ldots, N$.

Assumption 2. *The gradient noise process defined in* (8) *satisfies for any $\boldsymbol{w} \in \boldsymbol{\mathcal{F}}_{i-1}$ and for all $k, \ell = 1, \ldots, N$:*

$$\mathbb{E}[\boldsymbol{s}_{k,i}(\boldsymbol{w})|\boldsymbol{\mathcal{F}}_{i-1}] = 0, \qquad (13)$$

$$\mathbb{E}[\|\boldsymbol{s}_{k,i}(\boldsymbol{w})\|^2|\boldsymbol{\mathcal{F}}_{i-1}] \leq \beta_k^2\|\boldsymbol{w}\|^2 + \sigma^2_{s,k}, \qquad (14)$$

for some $\beta_k^2 \geq 0$, $\sigma^2_{s,k} \geq 0$, and where $\boldsymbol{\mathcal{F}}_{i-1}$ denotes the filtration generated by the random processes $\{\boldsymbol{w}_{\ell,j}\}$ for all $\ell = 1, \ldots, N$ and $j \leq i-1$.

Theorem 1. *Under Assumptions 1 and 2, strategy* (10) *converges for sufficiently small step-sizes satisfying:*

$$0 < \mu < \min\left\{\frac{2}{\eta\lambda_{\max}(\mathcal{R})}, \min_{1\leq k\leq N}\overline{\mu}_k\right\}, \qquad (15)$$

where

$$\overline{\mu}_k \triangleq \min\left\{\frac{2\lambda_{k,\min}}{\lambda^2_{k,\min}+3\beta_k^2}, \frac{2\lambda_{k,\max}}{\lambda^2_{k,\max}+3\beta_k^2}\right\}. \qquad (16)$$

Specifically, it holds that for small μ

$$\limsup_{i\to\infty}\mathbb{E}\|\mathcal{W}^o_\eta - \boldsymbol{\mathcal{W}}_i\|^2 = O(\mu). \qquad (17)$$

Proof. The argument is a simplification of the proofs presented in [33]. □

The first bound in (15) ensures stability of $I_{MN} - \mu\eta\mathcal{R}$ and the second bound ensures mean-square-error stability of each agent. Theorem 1 states that the expected squared distance between $\boldsymbol{w}_{k,i}$ and $w^o_{k,\eta}$ is on the order of μ at steady-state. This implies that when μ is chosen to be sufficiently small, the expected error can be made arbitrarily small.

C. Illustrative example

To illustrate the benefit of our multitask learning framework, we consider an MSE network of $N = 50$ nodes and $M = 5$, generated randomly with the link matrix shown in Fig. 1 (left). We set $A_{k\ell} = a_{k\ell} I_M$ with $a_{k\ell} = \frac{1}{\max\{|\mathcal{N}_k|,|\mathcal{N}_\ell|\}}$ if $\ell \in \mathcal{N}_k$ and 0 otherwise. We generate $\mathcal{W}^o$ according to $\mathcal{W}^o = \mathcal{V}e^{-\tau\Lambda}\mathcal{V}^\top\mathcal{W}_o$ with $\tau = 6$ and $\mathcal{W}_o$ a randomly generated vector from the Gaussian distribution $\mathcal{N}(0.1 \times \mathbb{1}_{MN}, I_{MN})$. Figure 1 (right) illustrates the spectral content of $\mathcal{W}^o$ given by $\mathcal{V}^\top\mathcal{W}^o$. It can be observed that the signal is mainly localized in $[0, 0.8]$. We assume that $R_{u,k} = \sigma^2_{u,k} I_M$. The variances $\sigma^2_{u,k}$ and $\sigma^2_{v,k}$ are generated from the uniform distributions $\mathcal{U}(1, 1.5)$ and $\mathcal{U}(0.15, 0.25)$, respectively. In Fig. 2 (left), we characterize the influence of the regularization on the performance of algorithm (10) relative to $\mathcal{W}^o$. We set $\mu = 5 \times 10^{-3}$. We run (10) for 5 different choices of regularizer $\mathcal{R}$: Laplacian ($p = 1$, $\mathcal{R} = \mathcal{L}$) [18], Laplacian ($p = 4$, $\mathcal{R} = \mathcal{L}^4$), diffusion process ($\sigma^2 = 12$, $\mathcal{R} = e^{6\mathcal{L}}$), p-step random walk ($p = 7$, $a = 1.8$, $\mathcal{R} = (aI_{MN} - \mathcal{L})^{-p}$), and bandlimited ($\mathcal{B} = [0, \lambda_c]$, $\lambda_c = 0.8$, $\beta = 1.3$). In each case, we report the steady-state $\text{MSD}^o = \lim_{i\to\infty}\frac{1}{N}\mathbb{E}\|\mathcal{W}^o - \boldsymbol{\mathcal{W}}_i\|^2$ for $\eta = \gamma\eta_o$ with $\eta_o \in [0, 100]$ and γ given in Fig. 2 (left). For each η, the results are averaged over 20 Monte-Carlo runs and over 500 samples after convergence. The results show that although the signal $\mathcal{W}^o$ is generated by smoothing a signal $\mathcal{W}_o$ by a diffusion kernel, setting $\mathcal{R} = \mathcal{L}$, which is a common choice for promoting smoothness [18], is not optimal in our setting and considering higher powers of Laplacian (such as $\mathcal{L}^4$) allows us to obtain better performance compared to the non-cooperative setting ($\eta = 0$). This is due to the fact that, by increasing p in $r(\lambda) = \lambda^p$, we penalize less $\overline{w}_m$ for which $\lambda_m < 1$ and we penalize more those for which $\lambda_m > 1$. The bandlimited regularizer provides the best performance. As we shall see in Section IV, due to the discontinuity at λ_c, this improvement may not be observed in a distributed implementation.

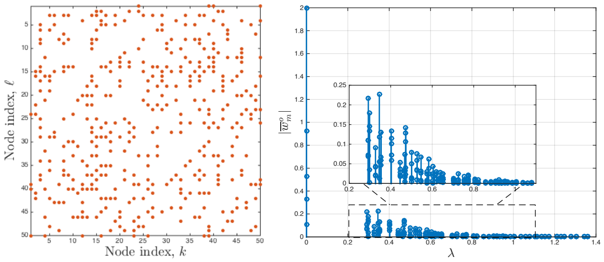

Fig. 1. Illustrative example. *(Left)* Link matrix. *(Right)* Graph spectral content of $\mathcal{W}^o$ with $\overline{w}^o_m = v_m^\top\mathcal{W}^o$.

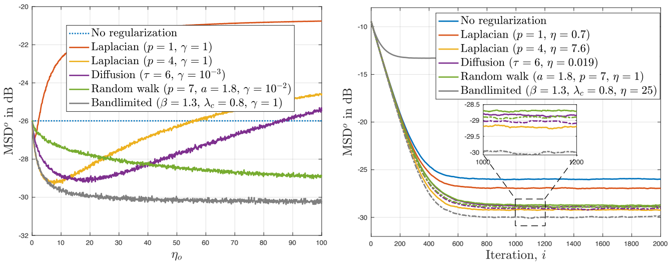

Fig. 2. Network performance relative to $\mathcal{W}^o$. *(Left)* Performance of algorithm (10) as a function of the regularization strength $\eta = \gamma\eta_o$ with $\eta_o \in [0, 100]$ for five different choices of regularizer $\mathcal{R}$. *(Right)* Performance of Algorithms 1 and 2 (solid curves). Dashed curves correspond to the centralized implementation (10).

IV. Distributed implementation

A. Regularization via Graph Laplacian polynomials

When the regularizer $\mathcal{R}$ can be written as a P-th degree polynomial of the Laplacian $\mathcal{L}$, i.e., $\mathcal{R} = \sum_{p=0}^{P}\beta_p\mathcal{L}^p$, for some scalar constants $\{\beta_p\}$, or equivalently, when $r(\lambda)$ in (5) can be written as $r(\lambda) = \sum_{p=0}^{P} \beta_p \lambda^p$, algorithm (10) can be implemented in a decentralized fashion since the second step in (10) can be implemented in P communication steps according to:

$$\left\{ \begin{array}{ll} \boldsymbol{\psi}_i^p = \beta_{P-p}\boldsymbol{\psi}_i + \mathcal{L}\boldsymbol{\psi}_i^{p-1}, & p = 1, \ldots, P \\ \boldsymbol{\mathcal{W}}_i = \boldsymbol{\psi}_i - \mu\eta\boldsymbol{\psi}_i^P & \end{array} \right. \tag{18}$$

with $\boldsymbol{\psi}_i^0 = \beta_P \boldsymbol{\psi}_i$. The P steps above involve product of $\boldsymbol{\psi}_i^{p-1}$ by $\mathcal{L}$ and this product can be computed at each node by just exchanging information with neighbors. Particularly, the k-th subvector of $\mathcal{L}\boldsymbol{\psi}_i^{p-1}$ can be computed locally at agent k according to $[\mathcal{L}\boldsymbol{\psi}_i^{p-1}]_k = \sum_{\ell \in \mathcal{N}_k} L_{k\ell}\boldsymbol{\psi}_{\ell,i}^{p-1}$ where $L_{k\ell}$ is the (k, ℓ)-th block of $\mathcal{L}$. Thus, replacing (18) into (10), we arrive at the *graph diffusion* strategy I.

Algorithm 1: Graph diffusion strategy I

When $r(\lambda) = \sum_{p=0}^{P} \beta_p \lambda^p$, run at each agent k:

$$\left\{ \begin{array}{ll} \boldsymbol{\psi}_{k,i} & = \boldsymbol{w}_{k,i-1} - \mu \widehat{\nabla_{w_k} J}_k(\boldsymbol{w}_{k,i-1}), \\ \boldsymbol{\psi}_{k,i}^0 & = \beta_P \boldsymbol{\psi}_{k,i}, \\ \boldsymbol{\psi}_{k,i}^p & = \beta_{P-p}\boldsymbol{\psi}_{k,i} + \sum\limits_{\ell \in \mathcal{N}_k} L_{k\ell}\boldsymbol{\psi}_{\ell,i}^{p-1}, \quad p = 1, \ldots, P, \\ \boldsymbol{w}_{k,i} & = \boldsymbol{\psi}_{k,i} - \mu\eta\, \boldsymbol{\psi}_{k,i}^P. \end{array} \right.$$

B. More general regularization form

For more general regularization forms, one would like to benefit from the sparsity of $\mathcal{L}$ (i.e., the graph). As long as we can approximate $\mathcal{R}$ by some low order polynomial in $\mathcal{L}$, say $\mathcal{R} \approx \sum_{p=0}^{P} \beta_p \mathcal{L}^p$, significant communication savings can be made and distributed implementations are possible. Problems of this type have already been considered in graph filters design [21]–[23]. A graph filter is an operator that acts upon a graph signal $\mathcal{W}$ by amplifying or attenuating its graph spectral content $\mathcal{V}^\top \mathcal{W} = \text{col}\{\overline{w}_m\}_{m=1}^{MN}$ as: $\Phi\mathcal{W} = \mathcal{V}\Phi(\Lambda)\mathcal{V}^\top\mathcal{W} = \sum_{m=1}^{MN} \Phi(\lambda_m)\overline{w}_m v_m$. The spectral function $\Phi(\lambda)$ controls how much Φ amplifies the spectrum. When $\mathcal{R} = r(\mathcal{L})$ in (5), $\mathcal{R}\boldsymbol{\psi}_i$ in (10) reduces to:

$$\mathcal{R}\boldsymbol{\psi}_i = r(\mathcal{L})\boldsymbol{\psi}_i = \sum_{m=1}^{MN} r(\lambda_m)\overline{\boldsymbol{\psi}}_{m,i} v_m, \tag{19}$$

where $\overline{\boldsymbol{\psi}}_{m,i} = v_m^\top \boldsymbol{\psi}_i$. By identification, we observe that $r(\mathcal{L})$ is a graph filter that acts upon $\boldsymbol{\psi}_i$.

Different methods for computing $r(\mathcal{L})\boldsymbol{\psi}_i$ in a distributed setting exist in the literature. In the following, we shall briefly describe the Chebyshev polynomial approximation method [22] which allows to approximate $r(\mathcal{L})\boldsymbol{\psi}_i$ by $\widetilde{r}(\mathcal{L})\boldsymbol{\psi}_i$, where $\widetilde{r}(\cdot)$ is a polynomial approximation of $r(\cdot)$ computed by truncating a shifted Chebyshev series expansion of $r(\cdot)$ on $[0, \lambda_{\max}]$. Doing so circumvents the need to compute the full set of eigenvalues and eigenvectors of $\mathcal{L}$. Accessible overview of other existing methods can be found in [22, Section V].

We approximate $r(\cdot)$ by the first $P + 1$ terms of its Chebyshev polynomial expansion according to [22], [34]:

$$r(\lambda) \approx \widetilde{r}(\lambda) = \frac{1}{2}\theta_0 + \sum_{p=1}^{P} \theta_p T_p\left(\frac{\lambda - \alpha}{\alpha}\right), \tag{20}$$

for $\lambda \in [0, \lambda_{\max}]$, where $\alpha = \frac{\lambda_{\max}}{2}$, θ_p are the Chebyshev coefficients given by $\theta_p = \frac{2}{\pi}\int_0^\pi \cos(px) r(\alpha(\cos(x) + 1))dx$, and $\{T_p(\cdot)\}_{p=0}^{P}$ are the polynomials that can be computed recursively according to $T_p(x) = 2xT_{p-1}(x) - T_{p-2}(x)$, for $p \geq 2$, with $T_0(x) = 1$ and $T_1(x) = x$. Thus, the second step in (10) can be approximated by:

$$\boldsymbol{\mathcal{W}}_i \approx \boldsymbol{\psi}_i - \mu\eta\left(\frac{1}{2}\theta_0\boldsymbol{\psi}_i + \sum_{p=1}^{P}\theta_p\boldsymbol{\psi}_i^p\right), \tag{21}$$

where $\boldsymbol{\psi}_i^p = T_p\left(\frac{1}{\alpha}(\mathcal{L} - \alpha I_{MN})\right)\boldsymbol{\psi}_i$. The vectors $\boldsymbol{\psi}_i^p$ can be computed recursively according to:

$$\boldsymbol{\psi}_i^p = \frac{2}{\alpha}(\mathcal{L} - \alpha I_{MN})\boldsymbol{\psi}_i^{p-1} - \boldsymbol{\psi}_i^{p-2}, \quad \text{if } p \geq 2, \tag{22}$$

with $\boldsymbol{\psi}_i^0 = \boldsymbol{\psi}_i$ and $\boldsymbol{\psi}_i^1 = \frac{1}{\alpha}(\mathcal{L} - \alpha I_{MN})\boldsymbol{\psi}_i$. Replacing (21), and (22) into (10), we arrive at the *graph diffusion* strategy II.

Algorithm 2: Graph diffusion strategy II

When $r(\lambda)$ is some non-negative function, evaluate the coefficients α and θ_p, and run at each agent k:

$$\left\{ \begin{array}{l} \boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu \widehat{\nabla_{w_k} J}_k(\boldsymbol{w}_{k,i-1}), \\ \boldsymbol{\psi}_{k,i}^0 = \boldsymbol{\psi}_{k,i}, \\ \boldsymbol{\psi}_{k,i}^1 = \frac{1}{\alpha}\sum\limits_{\ell \in \mathcal{N}_k} L_{k\ell}\boldsymbol{\psi}_{\ell,i} - \boldsymbol{\psi}_{k,i}, \\ \boldsymbol{\psi}_{k,i}^p = \frac{2}{\alpha}\sum\limits_{\ell \in \mathcal{N}_k} L_{k\ell}\boldsymbol{\psi}_{\ell,i}^{p-1} - 2\boldsymbol{\psi}_{k,i}^{p-1} - \boldsymbol{\psi}_{k,i}^{p-2},\ p = 2, \ldots, P, \\ \boldsymbol{w}_{k,i} = \boldsymbol{\psi}_{k,i} - \mu\eta\left(\frac{1}{2}\theta_0\boldsymbol{\psi}_{k,i} + \sum_{p=1}^{P}\theta_p\boldsymbol{\psi}_{k,i}^p\right). \end{array} \right.$$

This method allows the nodes in the network to perform the second step in (10) locally in P communication steps. Each node requires knowledge of $\{\theta_p\}$ that may be computed locally from knowledge of $r(\cdot)$, and $\alpha = \frac{\lambda_{\max}}{2}$. Note that, instead of using the exact value of $\lambda_{\max}$, an upper bound $\overline{\lambda_{\max}}$ can be used, and in this case α is replaced by $\overline{\alpha} = \frac{\overline{\lambda_{\max}}}{2}$. When $r(\cdot)$ is continuous, the Chebyshev approximation $\widetilde{r}(\cdot)$ converges to $r(\cdot)$ rapidly as P increases [22], [34].

C. Simulation results

We consider the same setting as in Section III-C. When $\mathcal{R} = \mathcal{L}^p$, we run Algorithm 1. For the three other choices of $\mathcal{R}$, we run Algorithm 2 with $\overline{\lambda_{\max}} = 1.5$, $P = 8$ in the diffusion case, $P = 22$ in the random walk case, and $P = 30$ in the bandlimited case. Figure 2 (right) reports the network MSD learning curves $\frac{1}{N}\mathbb{E}\|\mathcal{W}^o - \boldsymbol{\mathcal{W}}_i\|^2$. In each case, the value of η that gives the lower MSD (from the left plot in Fig. 2) is used. As it can be observed, when $r(\cdot)$ is continuous, the distributed implementation performs well compared to the centralized one. For the bandlimited case where $r(\cdot)$ is discontinuous on $[0, \lambda_{\max}]$, a performance degradation compared to the centralized implementation is observed for finite P. We note that, for $\lambda < \lambda_c$, we use $r(\lambda) = 0.07$ instead of $r(\lambda) = 0$ in the distributed case in order to ensure a positive semi-definite approximation $\widetilde{r}(\mathcal{L})$, and thus reducing the effect of ripples resulting from the Chebyshev approximation.

V. Conclusion

In this work, we proposed and studied an adaptive strategy that allows a multitask network to minimize a global cost consisting of the aggregate sum of individual costs regularized by a general regularization term enforcing a specific structure in the graph spectral domain. Approximation methods allowing the distributed implementation were also provided.

References

[1] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Foundations and Trends in Machine Learning*, vol. 7, no. 4-5, pp. 311–801, 2014.

[2] A. H. Sayed, S. Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks: An examination of distributed strategies and network behavior," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 155–171, 2013.

[3] A. H. Sayed, "Adaptive networks," *Proceedings of the IEEE*, vol. 102, no. 4, pp. 460–497, Apr. 2014.

[4] P. Braca, S. Marano, and V. Matta, "Enforcing consensus while monitoring the environment in wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3375–3380, Jul. 2008.

[5] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.

[6] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1847–1864, Nov. 2010.

[7] S. Sardellitti, M. Giona, and S. Barbarossa, "Fast distributed average consensus algorithms based on advection-diffusion processes," *IEEE Transactions on Signal Processing*, vol. 58, no. 2, pp. 826–842, Feb. 2010.

[8] S. Chouvardas, K. Slavakis, and S. Theodoridis, "Adaptive robust distributed learning in diffusion sensor networks," *IEEE Transactions on Signal Processing*, vol. 59, no. 10, pp. 4692–4707, Oct. 2011.

[9] J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Püschel, "Distributed optimization with local domains: Applications in MPC and network flows," *IEEE Transactions on Automatic Control*, vol. 60, no. 7, pp. 2004–2009, Jul. 2015.

[10] A. Bertrand and M. Moonen, "Distributed adaptive node-specific signal estimation in fully connected sensor networks – Part I: Sequential node updating," *IEEE Transactions on Signal Processing*, vol. 58, no. 10, pp. 5277–5291, Oct. 2010.

[11] A. Koppel, B. M. Sadler, and A. Ribeiro, "Proximity without consensus in online multi-agent optimization," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Shanghai, China, May 2016, pp. 3726–3730.

[12] J. Plata-Chaves, A. Bertrand, M. Moonen, S. Theodoridis, and A. M. Zoubir, "Heterogeneous and multitask wireless sensor networks – Algorithms, applications, and challenges," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 3, pp. 450–465, 2017.

[13] N. Bogdanovié, J. Plata-Chaves, and K. Berberidis, "Distributed diffusion-based LMS for node-specific parameter estimation over adaptive networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Florence, Italy, May 2014, pp. 7223–7227.

[14] S. Pan, J. Wu, X. Zhu, G. Long, and C. Zhang, "Task sensitive feature exploration and learning for multitask graph classification," *IEEE Transactions on Cybernetics*, vol. 47, no. 3, pp. 744–758, Mar. 2017.

[15] D. Hallac, J. Leskovec, and S. Boyd, "Network LASSO: Clustering and optimization in large graphs," in *Proc. ACM International Conference on Knowledge Discovery and Data Mining*, Sydney, Australia, Aug. 2015, pp. 387–396.

[16] J. Chen, C. Richard, and A. H. Sayed, "Multitask diffusion adaptation over networks," *IEEE Transactions on Signal Processing*, vol. 62, no. 16, pp. 4129–4144, 2014.

[17] R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed, "Proximal multitask learning over networks with sparsity-inducing coregularization," *IEEE Transactions on Signal Processing*, vol. 64, no. 23, pp. 6329–6344, Dec. 2016.

[18] R. Nassif, S. Vlaski, and A. H. Sayed, "Distributed inference over multitask graphs under smoothness," in *Proc. IEEE International Workshop on Signal Processing Advances in Wireless Communications*, Kalamata, Greece, Jun. 2018, pp. 1–5.

[19] Q. Xu and Q. Yang, "A survey of transfer and multitask learning in bioinformatics," *Journal of Computing Science and Engineering*, vol. 5, no. 3, pp. 257–268, Sep. 2011.

[20] X. Chen, S. Kim, Q. Lin, J. G. Carbonell, and E. P. Xing, "Graph-structured multi-task regression and an efficient optimization method for general fused Lasso," *Computer Research Repository (CoRR),* available as arXiv:1005.3579, 2010.

[21] 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 Processing Magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[22] D. I. Shuman, P. Vandergheynst, D. Kressner, and P. Frossard, "Distributed signal processing via Chebyshev polynomial approximation," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 4, pp. 736 – 751, Dec. 2018.

[23] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Frequency analysis.," *IEEE Transactions on Signal Processing*, vol. 62, no. 12, pp. 3042–3054, 2014.

[24] X. Shi, H. Feng, M. Zhai, T. Yang, and B. Hu, "Infinite impulse response graph filters in wireless sensor networks," *IEEE Signal Processing Letters*, vol. 22, no. 8, pp. 1113–1117, Aug. 2015.

[25] A. Loukas, A. Simonetto, and G. Leus, "Distributed autoregressive moving average graph filters," *IEEE Signal Processing Letters*, vol. 22, no. 11, pp. 1931–1935, Nov. 2015.

[26] R. I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete structures," in *Proc. 19th International Conference on Machine Learning (ICML)*, San Francisco, CA, USA, 2002, pp. 315–322.

[27] A. J. Smola and R. Kondor, "Kernels and regularization on graphs," in *Learning Theory and Kernel Machines*, B. Schölkopf and M. K. Warmuth, Eds. 2003, pp. 144–158, Springer.

[28] D. Zhou and B. Schölkopf, "A regularization framework for learning from graph data," in *ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields*, Banff, Canada, 2004, vol. 15, pp. 67–68.

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

[30] D. Zhou and B. Schölkopf, "Regularization on discrete spaces," in *Pattern Recognition*, W. G. Kropatsch, R. Sablatnig, and A. Hanbury, Eds. 2005, pp. 361–368, Springer.

[31] N. J. Higham, *Functions of Matrices: Theory and Computation*, SIAM, PA, 2008.

[32] S. K. Narang, A. Gadde, E. Sanou, and A. Ortega, "Localized iterative methods for interpolation in graph structured data," in *Proc. IEEE Global Conference on Signal and Information Processing*, Texas, USA, Dec. 2013, pp. 491–494.

[33] J. Chen and A. H. Sayed, "Distributed Pareto optimization via diffusion strategies," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 2, pp. 205–220, Apr. 2013.

[34] T. J. Rivlin, *The Chebyshev Polynomials*, Wiley, NY, 1974.