AN OPTIMIZATION FRAMEWORK FOR COMBINING MULTIPLE GRAPHS

Hilmi E. Egilmez†, Antonio Ortega†, Onur G. Guleryuz, Jana Ehmann* and Sehoon Yea**

†Signal & Image Processing Institute, University of Southern California, Los Angeles, CA, USA
*LG Electronics, San Jose, CA, USA

ABSTRACT

This paper introduces a novel framework for combining multiple weighted graphs into a single optimized weighted graph. In our framework, we first develop a statistical formulation for the graph combining problem with a maximum likelihood criterion, and derive its optimality conditions. We then use these conditions to formulate the deterministic graph combining problem and propose a solution. Our experimental results show that the proposed solution provides better modeling compared to the commonly used averaging method. The introduced framework has various applications in signal processing and machine learning.

***Index Terms*—** Graph learning, graph combining, statistical learning, optimization, graphical modeling, graph signal processing.

1. INTRODUCTION

In many data-oriented applications, graphs provide a generic tool to model relations between different data sources (or data points). For such applications, a weighted graph can be employed to represent similarities between data sources where nodes of the graph denote sources, and weighted edges indicate degree of similarity between them. Multiple graphs can also be used to describe the overall relations between sources, especially when the data of interest is high-dimensional. In a social network, for example, multiple graphs can connect a number of people where each graph is constructed based on a different feature (e.g., friendship, political view and geographical location). In signal processing, multiple graphs can be used for graphical modeling of a random signal having a mixture distribution (e.g., Gaussian mixture models).

In this work, we are interested in *graph combining* which we define as designing an optimized weighted graph from multiple weighted graphs. Graph combining can be useful in various cases. For instance, given multiple graphical models for a set of signals (or data), one would like to use an optimized aggregate model that is close to the original models with respect to a metric. Especially when a signal model is uncertain (or unknown), combining multiple candidate models would allow to design a model that is robust to model uncertainties. Using a single (optimized) model would also be advantageous in terms of computational complexity. Moreover, graph combining can be used to summarize a dataset consisting of multiple graphs into a single graph, which is appealing for graph visualization in data analytics.

In this paper, we propose a novel framework for graph combining. In our framework, we first present a two-step statistical formulation for the graph combining problem with a maximum likelihood criterion. Specifically, we define (i) *common graph transform* (CGT) estimation and (ii) *common graph frequency* (CGF) estimation problems and derive their optimality conditions. Exploiting the resulting optimality conditions, we then formulate the deterministic graph combining problem and propose a two-step solution that first estimates the best CGT and then finds the optimal combined graph by estimating the best CGF. To the best of our knowledge, this is the first systematic approach to graph combining problem with a maximum likelihood criterion.

There is only limited work on graph combining in the literature. The authors in [1, 2] propose graph combining to improve spectral clustering and semi-supervised learning with multiple graphs, respectively. However, their approaches are based on weighted averaging of graphs, which have no theoretical guarantees. Additionally, our CGT estimation problem is closely related to *common principal component* (CPC) estimation originally introduced in [3] where the goal is finding an orthogonal matrix that jointly diagonalizes multiple covariance matrices in maximum likelihood sense. An iterative solution to this problem is developed in [4]. A variation of the CPC problem was later studied in blind source separation [5], and the JADE algorithm [6] was introduced for joint diagonalization with a Frobenius norm criterion. Algorithmically, our solution to the CGT estimation problem is very similar to the ones in [4, 6] which are all based on Jacobi-like iterations [7]. The main difference is the optimality conditions derived for CGT estimation, which lead to a different metric to optimize.

The rest of the paper is organized as follows. Section 2 presents the graph combining framework. In Section 3, we introduce the proposed solution. Experimental results are presented in Section 4, and Section 5 draws some conclusions.

2. GRAPH COMBINING FRAMEWORK

2.1. Notation and Basic Assumptions

In this paper, we are interested in combining undirected, weighted graphs with no self-loops. We are given k graphs $\{\mathcal{G}_i(\mathcal{V}, \mathcal{E}_i, \mathbf{W}_i)\}_{i=1}^k$ with a fixed set of nodes $\mathcal{V}$ where for graph $\mathcal{G}_i$, $\mathcal{E}_i$ is the set of edges, and $\mathbf{W}_i$ is the weighted adjacency matrix with non-negative entries. If there is an edge between nodes j and l, the element at j-th row and l-th column of $\mathbf{W}_i$ is greater than zero (i.e., $(\mathbf{W}_i)_{jl} > 0$), otherwise $(\mathbf{W}_i)_{jl} = 0$. Since graphs are undirected, $\mathbf{W}_i$ is a symmetric matrix. The graph $\mathcal{G}_i$ can also be characterized using a graph Laplacian matrix, $\mathbf{L}_i = \mathbf{D}_i - \mathbf{W}_i$ where $\mathbf{D}_i$ is the diagonal degree matrix such that the j-th diagonal element, $(\mathbf{D}_i)_{jj}$, is the sum of weights of edges incident to j-th node in $\mathcal{V}$. For an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$ with d nodes, the corresponding $(d \times d)$ graph Laplacian matrix $\mathbf{L}$ is symmetric, positive semidefinite and singular. Thus, it has a complete set of orthonormal eigenvectors $\{\mathbf{u}_j\}_{j=1}^d$, whose associated eigenvalues are $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_d$. For connected graphs, only the first eigenvalue is equal to zero (i.e., $0 = \lambda_1 < \lambda_2$) and the corresponding eigenvector is $\mathbf{u}_1 = (1/\sqrt{d})\mathbf{1}$ where $\mathbf{1}$ is the all one vector of size d. By eigen-decomposition of $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^t$, we obtain orthogonal *graph transform matrix* $\mathbf{U}$ and diagonal *graph frequency matrix* $\mathbf{\Lambda}$.

This work has been supported in part by LG Electronics.

The formulation is based on following basic assumptions:

- **Nodes**: All k graphs use the same node set $\mathcal{V}$ with d nodes.
- **Connectivity**: Each of k graphs is a connected graph.
- **Similarity**: Graph weights represent similarity between nodes.

2.2. Statistical Formulation and Optimality Conditions

In what follows, we present a two-step statistical formulation for combining k graph Laplacian matrices $\mathbf{L}_1, \mathbf{L}_2, ..., \mathbf{L}_k$ (i.e., graphs) into a single graph Laplacian $\hat{\mathbf{L}} = \hat{\mathbf{U}}\hat{\mathbf{\Lambda}}\hat{\mathbf{U}}^t$ and derive optimality conditions based on a maximum likelihood criterion. In the first step, we formulate the common graph transform (CGT) estimation problem for finding the best CGT ($\hat{\mathbf{U}}$) and associated k frequency matrices $\{\mathbf{\Lambda}_i\}_{i=1}^k$. In the second step, we use the solution from the CGT step, and formulate the common graph frequency (CGF) estimation problem to find the best frequency matrix $\hat{\mathbf{\Lambda}}$ combining $\{\mathbf{\Lambda}_i\}_{i=1}^k$. Note that, the CGT estimation is similar to the common principal component (CPC) problem which deals with optimizing a orthogonal matrix that jointly diagonalizes multiple covariance matrices [3]. Since our formulation is based on graph Laplacian matrices, the CGT problem is inherently different and leads to different optimality conditions.

Maximum Likelihood (ML) Criterion: To derive the ML criterion, we choose the following data model associated to k graph Laplacians, $\{\mathbf{L}_i\}_{i=1}^k$. Let d-variate Gaussian random vectors $\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{L}_i^+)$ for $i = 1, 2, ..., k$ be independently distributed as

$$p(\mathbf{x}_i) = \frac{1}{(2\pi)^{d/2}|\mathbf{L}_i^+|_+^{1/2}} \exp\left(-\frac{1}{2}\mathbf{x}_i^t\mathbf{L}_i\mathbf{x}_i\right), \tag{1}$$

where $|\cdot|_+$ is the pseudo-determinant operator and $\mathbf{L}_i^+$ is the pseudo-inverse of $\mathbf{L}_i$. Since the random vector $\mathbf{x}_i$ is parametrized by the graph Laplacian $\mathbf{L}_i$, it has a degenerate multivariate Gaussian distribution with the singular covariance $\mathbf{L}_i^+$. Consisting $\tilde{n}_i = n_i + 1$ independent random vectors having distribution $p(\mathbf{x}_i)$, the random matrix $\mathbf{X}_i = [\mathbf{x}_i^{(1)}\mathbf{x}_i^{(2)} \cdots \mathbf{x}_i^{(\tilde{n}_i)}]$ leads to the random covariance matrix $\mathbf{S}_i = (1/n_i)\mathbf{X}_i\mathbf{X}_i^t$. Then, the corresponding random scatter matrix $\tilde{\mathbf{S}}_i = n_i\mathbf{S}_i$ has a Wishart distribution, $\tilde{\mathbf{S}}_i \sim \mathcal{W}(\mathbf{L}_i^+, n_i)$, which is a common data model used in covariance estimation. In our statistical formulation, we also use this data model. Since $\tilde{\mathbf{S}}_i$ are independent for $i = 1, 2, ..., k$, the likelihood function of $\mathbf{L}_1, \mathbf{L}_2, ..., \mathbf{L}_k$ is

$$p(\{\tilde{\mathbf{S}}_i\}_{i=1}^k | \{\mathbf{L}_i\}_{i=1}^k) = \prod_{i=1}^k C_i |\mathbf{L}_i^+|_+^{-\frac{n_i}{2}} \exp\Big(-\frac{n_i}{2}\mathrm{Tr}\left(\mathbf{L}_i\mathbf{S}_i\right)\Big) \tag{2}$$

where C_i is a constant that does not depend on $\mathbf{L}_i$. Thus, we can write negative log-likelihood objective function as follows,

$$\mathsf{NLL}(\mathbf{L}_1, \mathbf{L}_2, ..., \mathbf{L}_k) = \sum_{i=1}^k n_i \left(\log|\mathbf{L}_i^+|_+ + \mathrm{Tr}\left(\mathbf{L}_i\mathbf{S}_i\right)\right). \tag{3}$$

Common Graph Transform (CGT) Estimation : To find the best CGT ($\hat{\mathbf{U}}$), we define the CGT hypothesis as follows,

$$\mathcal{H}_{CGT}: \ \mathbf{L}_i = \mathbf{U}\mathbf{\Lambda}_i\mathbf{U}^t \qquad \text{for } i = 1, 2, ..., k \tag{4}$$

where $\mathbf{U}$ is the orthogonal matrix and $\{\mathbf{\Lambda}_i\}_{i=1}^k$ is the set of diagonal frequency matrices that we want to estimate. This hypothesis allows different frequency matrices, so it provides a larger degree of freedom in estimating $\mathbf{U}$. Assuming that the CGT hypothesis stated in (4) holds, we can rewrite NLL objective in (3) as

$$\mathsf{NLL}(\mathbf{U}, \{\mathbf{\Lambda}_i\}_{i=1}^k) = \sum_{i=1}^k n_i \left(\log|\mathbf{\Lambda}_i^+|_+ + \mathrm{Tr}\left(\mathbf{\Lambda}_i\mathbf{U}^t\mathbf{S}_i\mathbf{U}\right)\right) \tag{5}$$

Since, $\lambda_1^{(i)} = 0$ for $i = 1, .., k$ and $\mathbf{u}_1 = (1/\sqrt{d})\mathbf{1}$ by properties of graph Laplacian matrices. We can simplify (5) as follows,

$$\mathsf{NLL}(\tilde{\mathbf{U}}, \tilde{\mathbf{\lambda}}) = \sum_{i=1}^k n_i \sum_{j=2}^d \left(\log(1/\lambda_j^{(i)}) + \lambda_j^{(i)}\mathbf{u}_j^t\mathbf{S}_i\mathbf{u}_j\right) \tag{6}$$

where $\mathbf{u}_j$ is the j-th column of $\mathbf{U}$, and $\lambda_j^{(i)}$ is the (j,j)-th entry of $\mathbf{\Lambda}_i$. $\tilde{\mathbf{U}}$ and $\tilde{\mathbf{\lambda}}$ denote variables $\{\mathbf{u}_j\}_{j=2}^d$ and $\{\lambda_2^{(i)}, ..., \lambda_d^{(i)}\}_{i=1}^k$, respectively. Thus, the maximization of likelihood function in (2) under hypothesis $\mathcal{H}_{CGT}$ is equivalent to the following problem:

$$\begin{aligned} \underset{\tilde{\mathbf{U}}, \tilde{\mathbf{\lambda}}}{\text{minimize}} \quad & \sum_{i=1}^k n_i \sum_{j=2}^d \left(-\log(\lambda_j^{(i)}) + \lambda_j^{(i)}\mathbf{u}_j^t\mathbf{S}_i\mathbf{u}_j\right) \\ \text{subject to} \quad & \mathbf{u}_l^t\mathbf{u}_j = \begin{cases} 1 & l = j \\ 0 & l \neq j \end{cases} \end{aligned} \tag{7}$$

The optimization problem in (7) is non-convex due to its orthogonality constraints. So in the following, we derive necessary conditions for local optimality using the Lagrange multiplier theorem [8]. The Lagrangian function associated to (7) is

$$\mathcal{L}(\tilde{\mathbf{U}}, \tilde{\mathbf{\lambda}}, \boldsymbol{\mu}) = \mathsf{NLL}(\tilde{\mathbf{U}}, \tilde{\mathbf{\lambda}}) + \sum_{l=2}^d \mu_l(\mathbf{u}_l^t\mathbf{u}_l - 1) + 2\sum_{2\leq l<j\leq d} \mu_{l,j}\mathbf{u}_l^t\mathbf{u}_j \tag{8}$$

where $\boldsymbol{\mu}$ denotes Lagrange multipliers $\{\mu_l\}_{l=2}^d$ and $\{\mu_{l,j}\}_{2\leq l<j\leq d}$. Note that the last summation term of (8) is simplified by exploiting the symmetry between multipliers ($\mu_{l,j} = \mu_{j,l}$). Taking partial derivatives with respect to primal variables $(\tilde{\mathbf{U}}, \tilde{\mathbf{\lambda}})$ and equating to zero, we obtain following system of equations:

$$\frac{\partial\mathcal{L}(\tilde{\mathbf{U}}, \tilde{\mathbf{\lambda}}, \boldsymbol{\mu})}{\partial\lambda_j^{(i)}} = \sum_{i=1}^k n_i \left(-\frac{1}{\lambda_j^{(i)}} + \mathbf{u}_j^t\mathbf{S}_i\mathbf{u}_j\right) = 0 \tag{9}$$

$$\frac{\partial\mathcal{L}(\tilde{\mathbf{U}}, \tilde{\mathbf{\lambda}}, \boldsymbol{\mu})}{\partial\mathbf{u}_j} = \sum_{i=1}^k 2n_i\lambda_j^{(i)}\mathbf{S}_i\mathbf{u}_j + 2\mu_j\mathbf{u}_j + 2\sum_{2\leq l<j\leq d}\mu_{l,j}\mathbf{u}_l = \mathbf{0} \tag{10}$$

Multiplying (10) with $\mathbf{u}_j^t$ from the left, we get

$$\sum_{i=1}^k 2n_i\lambda_j^{(i)}\mathbf{u}_j^t\mathbf{S}_i\mathbf{u}_j + 2\mu_j = 0 \tag{11}$$

and by reorganizing (9), we have

$$1/\lambda_j^{(i)} = \mathbf{u}_j^t\mathbf{S}_i\mathbf{u}_j \tag{12}$$

In (11), replacing $\mathbf{u}_j^t\mathbf{S}_i\mathbf{u}_j$ with $1/\lambda_j^{(i)}$ leads to

$$\mu_j = -\sum_{i=1}^k n_i \quad \text{for } j = 2, ..., d \tag{13}$$

By multiplying (10) with $(1/2)\mathbf{u}_h^t$ from the left, we get

$$\sum_{i=1}^k n_i\lambda_j^{(i)}\mathbf{u}_h^t\mathbf{S}_i\mathbf{u}_j + \mu_{h,j} = 0 \quad j = 2, .., d, \ h \neq j \tag{14}$$

Switching h and j indexes leads to

$$\sum_{i=1}^k n_i\lambda_h^{(i)}\mathbf{u}_j^t\mathbf{S}_i\mathbf{u}_h + \mu_{j,h} = 0 \quad h = 2, .., d, \ j \neq h \tag{15}$$

Subtracting (15) from (14) results in the optimality condition for common graph transform estimation,

$$\mathbf{u}_h^t\left(\sum_{i=1}^k n_i(\lambda_j^{(i)} - \lambda_h^{(i)})\mathbf{S}_i\right)\mathbf{u}_j = 0 \quad j, h = 2, .., d, \ j \neq h \tag{16}$$

where $\lambda_j^{(i)} = 1/(\mathbf{u}_j^t\mathbf{S}_i\mathbf{u}_j)$ and $\lambda_h^{(i)} = 1/(\mathbf{u}_h^t\mathbf{S}_i\mathbf{u}_h)$ by (12).

Common Graph Frequency (CGF) Estimation : To find the best CGF ($\hat{\mathbf{\Lambda}}$), we define the CGF hypothesis as follows,

$$\mathcal{H}_{CGF}: \ \mathbf{L}_i = \hat{\mathbf{U}}\mathbf{\Lambda}\hat{\mathbf{U}}^t \qquad \text{for } i = 1, 2, ..., k \tag{17}$$

where $\hat{\mathbf{U}}$ is an optimal solution to the CGT problem in (7). $\mathbf{\Lambda} =$

$\mathrm{diag}([\lambda_1, ..., \lambda_d])$ is the diagonal matrix we want to estimate. Assuming that $\mathcal{H}_{CGF}$ holds, we can rewrite $\mathsf{NLL}(\mathbf{L}_1, \mathbf{L}_2, ..., \mathbf{L}_k) = \mathsf{NLL}(\mathbf{\Lambda})$ in (3) as follows,

$$\mathsf{NLL}(\mathbf{\Lambda}) = \sum_{i=1}^{k} n_i \left(\log|\mathbf{\Lambda}^+|_+ + \mathrm{Tr}\left(\mathbf{\Lambda} \hat{\mathbf{U}}^t \mathbf{S}_i \hat{\mathbf{U}} \right) \right). \quad (18)$$

Since the first eigenvalue of a graph Laplacian matrix is zero, we simplify above equation as follows,

$$\mathsf{NLL}(\lambda_2, ..., \lambda_d) = \sum_{i=1}^{k} n_i \sum_{j=2}^{d} \left(\log(1/\lambda_j) + \lambda_j \hat{\mathbf{u}}_j^t \mathbf{S}_i \hat{\mathbf{u}}_j \right) \quad (19)$$

where $\hat{\mathbf{u}}_j$ is the j-th column of $\hat{\mathbf{U}}$, and λ_j is the (j, j)-th entry of $\mathbf{\Lambda}$. Taking the derivative of (19) with respect to λ_j and equating to zero,

$$\frac{\partial \mathsf{NLL}(\lambda_2, ..., \lambda_d)}{\partial \lambda_j} = \sum_{i=1}^{k} n_i \left(-\frac{1}{\lambda_j} + \hat{\mathbf{u}}_j^t \mathbf{S}_i \hat{\mathbf{u}}_j \right) = 0. \quad (20)$$

Since $1/\lambda_j^{(i)} = \hat{\mathbf{u}}_j^t \mathbf{S}_i \hat{\mathbf{u}}_j$ from (12), we can write the optimality condition for common graph frequency estimation as follows,

$$\frac{1}{\lambda_j} = \frac{\sum_{i=1}^{k} n_i / \lambda_j^{(i)}}{\sum_{i=1}^{k} n_i} \quad j = 2, 3, ..., d. \quad (21)$$

where $\hat{\mathbf{\Lambda}} = \mathrm{diag}([0, \lambda_2, ..., \lambda_d])$.

2.3. Deterministic Formulation of Graph Combining Problem

The above statistical formulation poses the problem of finding a combined graph ($\hat{\mathbf{L}}$) using k groups of data ($\{\mathbf{S}_i\}_{i=1}^k$), each of which is assumed to be generated based on a graph. Yet, our main goal is combining k deterministic graphs, $\mathbf{L}_1, \mathbf{L}_2, ..., \mathbf{L}_k$. In the following, we propose a model-based formulation by exploiting the optimality conditions derived in our statistical formulation. However, the optimality conditions stated in (16) and (21) are satisfied if $\hat{\mathbf{L}} = \hat{\mathbf{U}} \hat{\mathbf{\Lambda}} \hat{\mathbf{U}}^t$ maximizes (2) under the assumptions given by $\mathcal{H}_{CGT}$ and $\mathcal{H}_{CGF}$. In practice, these assumptions generally do not hold. In particular, for a given set of graph Laplacian matrices, there may be no orthogonal matrix that satisfies $\mathcal{H}_{CGT}$. So, in general, we can only approximate the best common graph transform. In order to find the best CGT for given k graph Laplacian matrices, $\{\mathbf{L}_i\}_{i=1}^k$, we replace sample covariance $\mathbf{S}_i$ with the exact covariance $\mathbf{L}_i^+$ in (16), and also remove n_i since $\mathbf{L}_i^+$ does not depend on n_i. The resulting term is used as the objective in following minimization problem,

$$\begin{aligned} \underset{\mathbf{u}_h, \mathbf{u}_j}{\text{minimize}} \quad & \mathbf{u}_h^t \left(\sum_{i=1}^{k} (\lambda_j^{(i)} - \lambda_h^{(i)}) \mathbf{L}_i^+ \right) \mathbf{u}_j \quad j, h = 2, .., d \\ \text{subject to} \quad & \mathbf{u}_h^t \mathbf{u}_j = \begin{cases} 1 & h = j \qquad \lambda_j^{(i)} = 1/(\mathbf{u}_j^t \mathbf{L}_i^+ \mathbf{u}_j) \\ 0 & h \neq j \qquad \lambda_h^{(i)} = 1/(\mathbf{u}_h^t \mathbf{L}_i^+ \mathbf{u}_h) \end{cases} \end{aligned} \quad (22)$$

which is a non-convex problem because of the orthogonality constraints. In the next section, we provide an iterative algorithm that guarantees convergence to a locally optimal solution for CGT.

To find the best CGF for given k graph Laplacian matrices, $\{\mathbf{L}_i\}_{i=1}^k$, we reformulate (20) by replacing $\mathbf{S}_i$ with $\mathbf{L}_i^+$ and by removing n_i. Thus, we get the following new optimality condition,

$$\frac{1}{\lambda_j} = \frac{1}{k} \sum_{i=1}^{k} \frac{1}{\lambda_j^{(i)}} = \frac{1}{k} \sum_{i=1}^{k} \hat{\mathbf{u}}_j^t \mathbf{L}_i^+ \hat{\mathbf{u}}_j \quad j = 2, 3, ..., d \quad (23)$$

where $\hat{\mathbf{u}}_j$ is the j-th column of the optimized CGT matrix, $\hat{\mathbf{U}}$. Basically, this condition suggests an averaging of frequency estimates obtained by solving optimization problem (22). However, the resulting frequency matrix $\hat{\mathbf{\Lambda}} = \mathrm{diag}([0, \lambda_2, ..., \lambda_d])$ may not construct a graph Laplacian matrix using $\hat{\mathbf{U}} \hat{\mathbf{\Lambda}} \hat{\mathbf{U}}^t$, because hypotheses $\mathcal{H}_{CGT}$ and $\mathcal{H}_{CGF}$ do not hold in general. Therefore, we propose to find

1: **function** GRAPH COMBINING($\{\mathbf{L}_i\}_{i=1}^k$, $\mathbf{U}_{\text{init}}$, Γ, ϵ)
2: $\quad \hat{\mathbf{U}} \leftarrow$ CGTA($\{\mathbf{L}_i\}_{i=1}^k$, $\mathbf{U}_{\text{init}}$, ϵ)
3: $\quad \hat{\mathbf{\Sigma}} \leftarrow \frac{1}{k} \sum_{i=1}^{k} \hat{\mathbf{U}}^t \mathbf{L}_i^+ \hat{\mathbf{U}}$
4: $\quad \lambda_1 \leftarrow 0 \qquad \{\lambda_j\}_{j=2}^d \leftarrow \{1/(\hat{\mathbf{\Sigma}})_{jj}\}_{j=2}^d$
5: $\quad \{(\hat{\mathbf{\Lambda}})_{jj}\}_{j=1}^d \leftarrow \{\lambda_j\}_{j=1}^d$
6: $\quad \mathbf{L}^* \leftarrow$ Solve the problem in (24) with parameters $\hat{\mathbf{U}}$, $\hat{\mathbf{\Lambda}}$ and Γ
7: $\quad$ **return** $\mathbf{L}^*$
8: **end function**

Fig. 1: Graph Combining Algorithm

1: **function** CGTA($\{\mathbf{L}_i\}_{i=1}^k$, $\mathbf{U}_{\text{init}}$, ϵ)
2: $\quad \mathbf{U} \leftarrow \mathbf{U}_{\text{init}}$
3: $\quad$ **do**
4: $\quad\quad \mathbf{U}_{\text{pre}} \leftarrow \mathbf{U}$
5: $\quad\quad$ **for** all (h, j) pairs such that $2 \leq h, j \leq d$ and $h \neq j$ **do**
6: $\quad\quad\quad \mathbf{u}_h \leftarrow \mathbf{U}(:, h) \quad \mathbf{u}_j \leftarrow \mathbf{U}(:, j)$
7: $\quad\quad\quad \{\mathbf{A}_i\}_{i=1}^k \leftarrow \left\{ \begin{bmatrix} \mathbf{u}_h^t \mathbf{L}_i^+ \mathbf{u}_h & \mathbf{u}_h^t \mathbf{L}_i^+ \mathbf{u}_j \\ \mathbf{u}_j^t \mathbf{L}_i^+ \mathbf{u}_h & \mathbf{u}_j^t \mathbf{L}_i^+ \mathbf{u}_j \end{bmatrix} \right\}_{i=1}^k$
8: $\quad\quad\quad \mathbf{R} \leftarrow$ BEST ROTATION($\{\mathbf{A}_i\}_{i=1}^k$, ϵ)
9: $\quad\quad\quad \mathbf{V} \leftarrow [\mathbf{u}_h, \mathbf{u}_j]\mathbf{R}$
10: $\quad\quad\quad \mathbf{U}(:, h) \leftarrow \mathbf{V}(:, 1) \quad \mathbf{U}(:, j) \leftarrow \mathbf{V}(:, 2)$
11: $\quad\quad$ **end for**
12: $\quad$ **while** $||\mathbf{U} - \mathbf{U}_{\text{pre}}||_F \geq \epsilon$ (stopping criterion)
13: $\quad$ **return** $\mathbf{U}$
14: **end function**

15: **function** BEST ROTATION($\{\mathbf{A}_i\}_{i=1}^k$, ϵ)
16: $\quad \mathbf{Q} \leftarrow \mathbf{I}$
17: $\quad$ **do**
18: $\quad\quad \mathbf{Q}_{\text{pre}} \leftarrow \mathbf{Q} \quad \mathbf{T} \leftarrow \mathbf{0}$
19: $\quad\quad \mathbf{q}_1 \leftarrow \mathbf{Q}(:, 1) \quad \mathbf{q}_2 \leftarrow \mathbf{Q}(:, 2)$
20: $\quad\quad$ **for** $i = 1 : k$ **do**
21: $\quad\quad\quad \delta_1^{(i)} \leftarrow 1/(\mathbf{q}_1^t \mathbf{A}_i \mathbf{q}_1) \quad \delta_2^{(i)} \leftarrow 1/(\mathbf{q}_2^t \mathbf{A}_i \mathbf{q}_2)$
22: $\quad\quad\quad \mathbf{T} \leftarrow \mathbf{T} + (\delta_1^{(i)} - \delta_2^{(i)}) \mathbf{A}_i$
23: $\quad\quad$ **end for**
24: $\quad\quad \theta \leftarrow \frac{1}{2} \arctan\left(\frac{\mathbf{T}(1,2)}{\mathbf{T}(1,1) - \mathbf{T}(2,2)} \right)$
25: $\quad\quad \mathbf{Q} \leftarrow \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$
26: $\quad$ **while** $||\mathbf{Q} - \mathbf{Q}_{\text{pre}}||_F \geq \epsilon$ (stopping criterion)
27: $\quad$ **return** $\mathbf{Q}$
28: **end function**

Fig. 2: Common Graph Transform Algorithm (CGTA)

the closest graph Laplacian matrix to $\hat{\mathbf{U}} \hat{\mathbf{\Lambda}} \hat{\mathbf{U}}^t$ in maximum likelihood sense. Thus, we formulate the following convex optimization problem,

$$\begin{aligned} \underset{\mathbf{L}(\Gamma)}{\text{maximize}} \quad & \log|\mathbf{M}|_+ - \mathrm{Tr}\left(\mathbf{M} \hat{\mathbf{\Lambda}}^+ \right) \\ \text{subject to} \quad & \mathbf{M} = \hat{\mathbf{U}}^t (\mathbf{L}(\Gamma)) \hat{\mathbf{U}} \\ & \mathbf{L}(\Gamma) := \text{graph Laplacian matrix} \end{aligned} \quad (24)$$

where $\mathbf{L}(\Gamma)$ is constrained to be a graph Laplacian matrix with edge set Γ (i.e., graph connectivity). $\hat{\mathbf{U}}$ is obtained by solving (22) and $\hat{\mathbf{\Lambda}}$ satisfies (23). Γ parameter allows us to specify graph connectivity information if it is known a priori. Otherwise, Γ can be selected as the set containing all possible edges.

3. PROPOSED SOLUTION

In order to find the best graph Laplacian matrix $\mathbf{L}^*$ that combines k graph Laplacian matrices $\{\mathbf{L}_i\}_{i=1}^k$, we propose the algorithm shown in Fig.1 where we first solve the optimization problem stated in (22) to find the best common graph transform $\hat{\mathbf{U}}$ (see line 2). We then estimate the common graph frequency (see lines 3–5), using the condition stated in (23). Finally, the best graph Laplacian is found by solving the optimization problem in (24) (see line 6). Note that, the graph combining algorithm calls CGTA to find the best CGT.

In Fig.2, we present the CGT algorithm proposed to solve the non-convex optimization problem in (22). In a nutshell, for a given

initial guess $\mathbf{U}_{\text{init}}$ and error tolerance ϵ, CGTA iteratively minimizes the objective in (22) by pairwise updating the columns of $\mathbf{U}$ (see lines 6–10). The for loop at line 5 of CGTA iterates over all (h, j) pairs for $h, j = 2, ..., d$ and $h \neq j$ until the ϵ-convergence has been reached (see line 12). At each iteration, an optimal (2×2) rotation matrix is used to update $\mathbf{u}_h$ and $\mathbf{u}_j$ vectors in $\mathbf{U}$ (see lines 8–10). To find the best rotation, another iterative procedure is used as depicted in Fig.2 between lines 15–28. This algorithm guarantees convergence to a local optimal solution, but the convergence analysis is omitted due to space limitation.

4. RESULTS

In this section, we present our experimental results demonstrating the performance of our graph combining algorithm (see Fig.1) by benchmarking against averaging method. For given k graph Laplacian matrices, $\mathbf{L}_1, ..., \mathbf{L}_k$, the averaging method combines graphs as,

$$\mathbf{L}^{\text{avg}} = \frac{1}{k} \sum_{i=1}^{k} \mathbf{L}_i. \tag{25}$$

We use two different metrics to measure the graph combining performance. First metric is called *coding gain* which is a popular metric used in information theory and compression [9]. This metric is used to measure how well a designed common graph transform $\mathbf{U}$ diagonalizes $\mathbf{L}^+$ as follows,

$$\mathsf{CG}(\mathbf{U}, \mathbf{L}) = \left(\frac{\prod_{i=1}^{d} (\mathbf{L}^+)_{ii}}{\prod_{i=2}^{d} (\mathbf{U}^t \mathbf{L}^+ \mathbf{U})_{ii}} \right)^{1/d}. \tag{26}$$

The other metric we use is called *average quadratic cost* [10] which measures average variation of p signals $(\{\mathbf{y}_s\}_{s=1}^{p})$ with respect to a graph Laplacian $\mathbf{L}$ as,

$$\mathsf{AQC}(\{\mathbf{y}_s\}_{s=1}^{p}, \mathbf{L}) = \frac{1}{p} \sum_{s=1}^{p} \mathbf{y}_s^t \mathbf{L} \mathbf{y}_s. \tag{27}$$

In our experiments, for each input graph $\mathbf{L}_1, ..., \mathbf{L}_k$ we randomly pick $p = 100$ samples from distribution in (1), and measure average quadratic cost with respect to combined graphs, $\mathbf{L}^*$ and $\mathbf{L}^{\text{avg}}$. On the other hand, the coding gain is directly calculated for each input graph $\mathbf{L}_1, ..., \mathbf{L}_k$ using graph transforms $\mathbf{U}^*$ and $\mathbf{U}^{\text{avg}}$ obtained from $\mathbf{L}^*$ and $\mathbf{L}^{\text{avg}}$, respectively. Figs. 3 and 4 illustrate input graphs with line and mesh structures and their combined graph results, $\mathbf{L}^*$ and $\mathbf{L}^{\text{avg}}$, respectively. Corresponding coding gain and quadratic cost results are presented in Tables 1 – 4. According to these results, proposed graph combining algorithm outperforms averaging method by providing larger coding gain and lower average quadratic cost.

Table 1: Coding gain results for the line graphs in Fig.3

CG	$\mathbf{L}_1$	$\mathbf{L}_2$	$\mathbf{L}_3$	$\mathbf{L}_4$	$\mathbf{L}_5$	**Average**
$\mathbf{U}^*$	0.8298	0.8586	0.9066	0.8319	0.8812	0.8616
$\mathbf{U}^{\text{avg}}$	0.8216	0.8102	0.8160	0.8125	0.8061	0.8133

Table 2: Average quadratic cost results for the line graphs in Fig.3

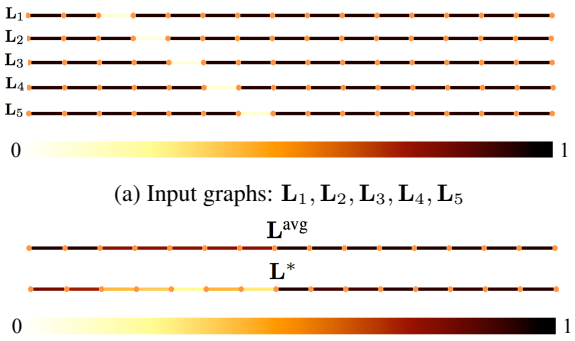
AQC	$\mathbf{L}_1$	$\mathbf{L}_2$	$\mathbf{L}_3$	$\mathbf{L}_4$	$\mathbf{L}_5$	**Average**
$\mathbf{L}^*$	14.334	14.102	13.094	15.8391	12.867	14.047
$\mathbf{L}^{\text{avg}}$	20.797	20.068	22.061	21.971	19.506	20.881

Table 3: Coding gain results for the graphs in Fig.4

CG	$\mathbf{L}_1$	$\mathbf{L}_2$	$\mathbf{L}_3$	$\mathbf{L}_4$	**Average**
$\mathbf{U}^*$	0.9802	0.9023	0.9562	0.8936	0.9331
$\mathbf{U}^{\text{avg}}$	0.9050	0.8549	0.9521	0.8801	0.8981

Table 4: Average quadratic cost results for the graphs in Fig.4

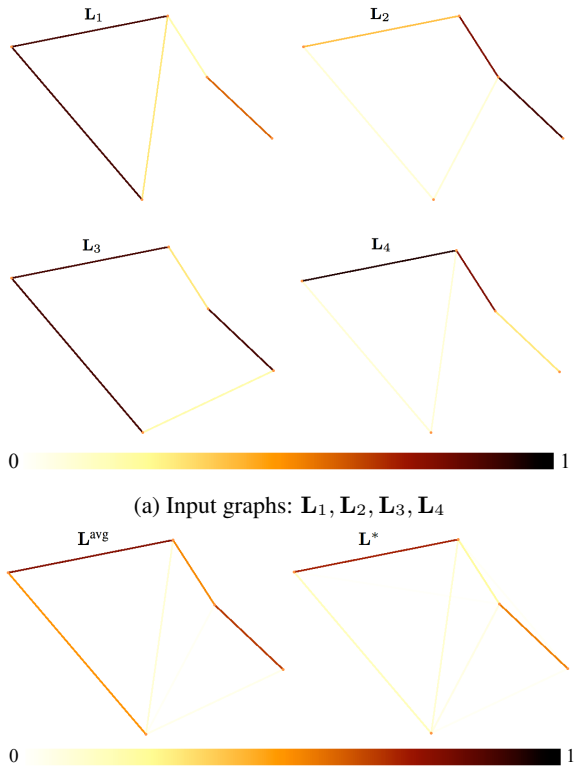
AQC	$\mathbf{L}_1$	$\mathbf{L}_2$	$\mathbf{L}_3$	$\mathbf{L}_4$	**Average**
$\mathbf{L}^*$	3.9576	4.6745	2.9700	5.1360	4.1845
$\mathbf{L}^{\text{avg}}$	5.8170	7.0549	4.2231	7.6249	6.1800



(a) Input graphs: $\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_3, \mathbf{L}_4, \mathbf{L}_5$

(b) Combined graphs: $\mathbf{L}^{\text{avg}}$ and $\mathbf{L}^*$

Fig. 3: Combining $k = 5$ line graphs with $d = 16$ nodes. Edge weights are color coded between 0 and 1. Each input graph has only one weak edge weight equal to 0.1 while all other edges are weighted as 0.95.



(a) Input graphs: $\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_3, \mathbf{L}_4$

(b) Combined graphs: $\mathbf{L}^{\text{avg}}$ and $\mathbf{L}^*$

Fig. 4: Combining $k = 4$ mesh-like graphs with $d = 5$ nodes. Edge weights are color coded between 0 and 1.

5. CONCLUSIONS

In this paper, we have introduced a novel framework for graph combining by (i) introducing a new problem formulation with a maximum likelihood criterion and by (ii) proposing a solution involving common graph transform estimation and common graph frequency estimation. The experimental results have showed that the proposed graph combining method leads to a better model compared to the average graph model in terms of two well known metrics, coding gain and average quadratic cost.

6. REFERENCES

[1] W. Tang, Z. Lu, and I. S. Dhillon, "Clustering with multiple graphs," in *Proc. IEEE International Conference on Data Mining (ICDM)*, 2009, pp. 1016–1021.

[2] A. Argyriou, M. Herbster, and M. Pontil, "Combining graph laplacians for semi-supervised learning," in *Neural Information Processing Systems, (NIPS)*, 2005, pp. 67–74.

[3] B. N. Flury, "Common principal components in k groups," *Journal of the American Statistical Association*, vol. 79, no. 388, pp. 892–898, 1984.

[4] B. N. Flury and W. Gautschi, "An algorithm for simultaneous orthogonal transformation of several positive definite symmetric matrices to nearly diagonal form," *SIAM J. Sci. Stat. Comput.*, vol. 7, no. 1, pp. 169–184, Jan. 1986.

[5] A. Belouchrani, K. Abed-Meraim, J.-F. Cardoso, and E. Moulines, "A blind source separation technique using second-order statistics," *IEEE Transactions on Signal Processing*, vol. 45, no. 2, pp. 434–444, Feb 1997.

[6] J.-F. Cardoso and A. Souloumiac, "Jacobi angles for simultaneous diagonalization," *SIAM J. Mat. Anal. Appl.*, vol. 17, no. 1, pp. 161–164, Jan. 1996.

[7] G. H. Golub and H. A. van der Vorst, "Eigenvalue computation in the 20th century," *Journal of Computational and Applied Mathematics*, vol. 123, no. 12, pp. 35 – 65, 2000.

[8] D. P. Bertsekas, *Nonlinear Programming*. Belmont, MA: Athena Scientific, 1999.

[9] K. R. Rao and P. Yip, *Discrete Cosine Transform Algorithms, Advantages, Applications*. San Diego, CA, USA: Academic Press Professional, Inc., 1990.

[10] 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, May 2013.