Graph Weight Allocation to Meet Laplacian Spectral Constraints

S. Yusef Shafi, Murat Arcak, and Laurent El Ghaoui

***Abstract*—We adjust the node and edge weightings of graphs using convex optimization to impose bounds on their Laplacian spectra. First, we derive necessary and sufficient conditions that characterize the feasibility of spectral bounds given positive node and edge weightings. Synthesizing these conditions leads naturally to algorithms that exploit convexity to achieve several eigenvalue bounds simultaneously. The algorithms we propose apply to many graph design problems as well as multi-agent systems control. Finally, we suggest efficient ways to accommodate larger graphs, and show that dual formulations lead to substantial improvement in the size of graphs that can be addressed.**

***Index Terms*—Convex optimization, graph Laplacians, linear matrix inequalities, multiagent systems.**

I. Introduction

A well-studied tool for characterizing the interconnection topology of a network of distributed agents is the graph Laplacian matrix [1]. In particular, the spectrum of the Laplacian contains useful information about the dynamics of the network. For example, the smallest positive eigenvalue of a Laplacian, known as the algebraic connectivity, or Fiedler eigenvalue [2], is a common measure of how well connected a network is [3]–[6]. On the other hand, the largest eigenvalue must be sufficiently small for stability of discrete-time consensus algorithms [4], [7], and for continuous-time formation control algorithms when agent dynamics can be destabilized by high gain feedback [8].

We present a scheme to enforce constraints on the Laplacian spectrum by treating both node and edge weights as decision variables. Let λ_i be the ith-smallest eigenvalue of the Laplacian, whose eigenvalues are ordered from least to greatest. Given $m \in \{2, \ldots, n\}$ and $\underline{\lambda}_m > 0$, the lower eigenvalue bound assignment problem is to guarantee $\lambda_m \geq \underline{\lambda}_m$. Likewise, given $p \in \{2, \ldots, n\}$ and $\overline{\lambda}_p > 0$, the upper eigenvalue bound assignment problem is to guarantee $\lambda_p \leq \overline{\lambda}_p$. Our goal is to achieve individual upper and lower bounds for several Laplacian eigenvalues simultaneously. We show how these bounds can be recast as linear matrix inequality constraints [9] that can be applied using semidefinite programming.

Convex optimization solutions to several graph problems are well-documented in the literature, including fastest distributed linear averaging (FDLA) [10], minimization of total effective resistance on a graph [11], fastest mixing Markov chains [12] and processes [13], and Fiedler eigenvalue maximization through vertex positioning [14]. In FDLA [10], a particular interconnection structure for a discrete system with symmetric interconnections is specified. The number of iterations required for linear averaging is minimized by finding a particular weight distribution that assigns iterative update laws for each node's state. The goal in many resistor network problems [11] is to minimize the total effective resistance on a graph by assigning different weights representing resistances to the links connecting the nodes of an electrical network. The aim for fastest mixing Markov chains [12] and processes [13] is to find the optimal transition probabilities between states to reach a stationary distribution as quickly as possible. Finally, vertex positioning [14] aims to find the optimal locations of vertices, corresponding to edge weights, in order to maximize the Fiedler eigenvalue.

Our approach is unique when compared to previous literature on optimization of the Laplacian spectrum because it is applicable to any selection of eigenvalues, and assigns weights independently to both nodes and edges. As demonstrated in the paper, joint tuning of node and edge weights is an especially powerful tool that goes beyond the commonly-used edge weighting strategies for achieving spectral constraints.

The remainder of the paper is organized as follows. Section II introduces preliminary results in linear algebra and spectral graph theory that are necessary for our analysis. Section III outlines a general optimization framework that enables upper and lower bounds on several Laplacian eigenvalues simultaneously based on node and edge weighting. Section IV presents sample problems that can be formulated and solved using the methods of Section III. Section V explores applications to multi-agent systems. Larger graphs are addressed in Section VI with dual formulations of the optimization framework developed in Sections III and IV.

II. Preliminaries

We review the following results from linear algebra, which we will use in Section III. The first result concerns the eigenvalues of a product of two matrices ([15, Theor. 1.3.20]):

Lemma 2.1: Let $A \in \mathbb{R}^{r \times q}$, $B \in \mathbb{R}^{q \times r}$, and $r \geq q$. Then AB and BA have m identical eigenvalues with AB having $r - q$ additional eigenvalues at zero.

The next lemma follows from the Courant-Fischer theorem, which characterizes the eigenvalues of a symmetric matrix ([15, Coroll. 4.3.23]):

Lemma 2.2: If $A \in \mathbb{R}^{n \times n}$ is symmetric and if $x^T A x \geq 0$ for all vectors $x \in \mathbb{R}^n$ in a k-dimensional subspace, then A has at least k nonnegative eigenvalues.

Definition 2.3: The square matrices A and B are **congruent** if $B = SAS^T$ for some square, nonsingular S.

The following lemma is known as Sylvester's Law of Inertia ([15, Theor. 4.5.8]):

Lemma 2.4: Let A, $B \in \mathbb{R}^{n \times n}$ be symmetric matrices. A and B are congruent if and only if A and B have the same *inertia*, i.e., the same number of positive, negative, and zero eigenvalues.

An inequality due to Sylvester characterizes the relationship between the eigenvalues of two matrices and their products ([16, Section 3.5]):

Lemma 2.5: Given two matrices $A \in \mathbb{R}^{r \times n}$ and $B \in \mathbb{R}^{n \times q}$, the following inequality holds:

$$\operatorname{rank}(A) + \operatorname{rank}(B) - n \leq \operatorname{rank}(AB) \leq \min\{\operatorname{rank}(A), \operatorname{rank}(B)\}.$$

We next review notions from spectral graph theory that are essential to this paper. A *graph* $G = G(V, E)$ is a collection of nodes V and a corresponding set of edges E. In this paper, we consider *undirected* graphs, where two nodes are *connected* when there exists an edge incident to both. A graph itself is connected if there exists a sequence of edges connecting any pair of nodes in the graph. Given an undirected graph $G(V, E)$ with n nodes and h edges, an *incidence*

Manuscript received March 25, 2011; revised March 28, 2011 and March 28, 2011; accepted October 14, 2011. Date of publication December 26, 2011; date of current version June 22, 2012. This work was supported in part by the National Science Foundation under Grants ECCS-0852750 and ECCS-1101876 and the Air Force Office of Scientific Research (AFOSR) under Grant FA9550-11-1-0244. Recommended by Associate Editor Y. Hong.

The authors are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley CA 94720 USA (e-mail: yusef@eecs.berkeley.edu; arcak@eecs.berkeley.edu; elghaoui@eecs.berkeley.edu).

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

Digital Object Identifier 10.1109/TAC.2011.2181795

matrix $A \in \mathbb{R}^{n\times h}$ is an $n \times h$ matrix, each of whose columns, indexed by $k = 1,\ldots,h$, represents an edge in E linking nodes v_i and v_j in V, with $[A]_{ik} = 1$, $[A]_{jk} = -1$, and $[A]_{lk} = 0$ for all $l \neq i, j$. We note that the incidence matrix is not unique for an undirected graph, and choice of orientation does not change our results. We denote by $L = AA^T$ the $n \times n$ *nominal Laplacian*, and by $L_e = AKA^T$ the *edge-weighted Laplacian*, where $K \succ 0$ is the diagonal *edge weighting matrix*. We denote by $L_g = M^{-1}AKA^T$ the *node- and edge- weighted graph Laplacian* (henceforth *weighted Laplacian*), where $M \succ 0$ is the diagonal *node weighting matrix*.

We first recall key facts about Laplacian matrices [1]. The matrices L and L_e are symmetric positive semidefinite, with at least one eigenvalue at zero corresponding to an eigenvector $\mathbf{1}_n = (1/\sqrt{n})[1\ldots1]^T$. If the graph represented by L or L_e is connected, then L or L_e, respectively, has exactly one eigenvalue at zero. Although L_g is not symmetric in general, its eigenvalues possess properties similar to those of L and L_e:

Lemma 2.6: Every eigenvalue of $L_g = M^{-1}AKA^T$ is real and nonnegative. If L_g represents a connected graph, then all eigenvalues of L_g, excepting one at zero, are positive.

Proof: A similarity transformation brings L_g to the symmetric form $M^{-1/2}L_eM^{-1/2}$, and so all eigenvalues of L_g are real. Furthermore, the symmetric matrices $M^{-1/2}L_eM^{-1/2}$ and L_e are congruent, and hence Lemma 2.4 guarantees that all eigenvalues of L_g are nonnegative. When L_e represents a connected graph, and, thus, has only one eigenvalue at zero, Lemma 2.4 implies that all eigenvalues of L_g, excepting one at zero, are positive. ∎

III. Convex Characterizations of Upper and Lower Eigenvalue Constraints

Our goal is to find node and edge weighting matrices M and K, respectively, to assign individual lower and upper bounds for several eigenvalues of L_g simultaneously. Let $\lambda_k(L_g)$ denote the k-th smallest eigenvalue of L_g. Given $a, b \leq n$, define the sets of indices $\{m_i\}_{i=1}^a$ and $\{p_i\}_{i=1}^b$ with each $2 \leq m_i \leq n$ and $2 \leq p_j \leq n$ an integer contained in $\{2, n\}$. Define the sets of positive scalars $\{\underline{\lambda}_{m_i}\}_{i=1}^a$ and $\{\overline{\lambda}_{p_j}\}_{j=1}^b$. We wish to see if there exist M and K that satisfy the constraints in the following problem:

$$\begin{aligned} \text{Find} \quad & M,\ K \\ \text{subject to} \quad & \lambda_{m_i}(L_g) \geq \underline{\lambda}_{m_i},\ i = 1,\ldots,a \\ & \lambda_{p_j}(L_g) \leq \overline{\lambda}_{p_j},\ j = 1,\ldots,b. \end{aligned} \tag{1}$$

A. Bounding Eigenvalues From Below

Given $m < n$ and $\underline{\lambda}_m > 0$, we wish to design node and edge weights M and K, respectively, such that $\lambda_m(L_g) \geq \underline{\lambda}_m$. We note that by itself, the lower eigenvalue bound $\lambda_m(L_g) \geq \underline{\lambda}_m$ can be enforced by scaling M by $\lambda_m(L)/\underline{\lambda}_m$ or K by $\underline{\lambda}_m/\lambda_m(L)$. However, when the graph optimization problem imposes upper eigenvalue constraints as in (1) or objective functions, this approach would likely be infeasible. In contrast, our results make it possible to apply several upper and lower eigenvalue bounds at once. To begin, we construct a linear matrix inequality enforcing the eigenvalue constraint, making use of the following lemma:

Lemma 3.1: Suppose that $m < n$, $Q_m \in \mathbb{R}^{n\times(n-m+1)}$ is a full column rank matrix whose columns are orthogonal, and S is a symmetric matrix. If $Q_m^TSQ_m \succeq 0$, then $\lambda_m(S) \geq 0$.

Proof: The result follows immediately from Lemma 2.2: the subspace spanned by the columns of Q_m is $n - m + 1$ dimensional, so $\lambda_m(S) \geq 0$. ∎

The next theorem provides a sufficient condition in the form of a linear matrix inequality constraint to enforce lower eigenvalue bounds:

Theorem 3.2: Let Q_m be as in Lemma 3.1. The constraint

$$Q_m^T(L_e - \underline{\lambda}_m M)Q_m \succeq 0 \tag{2}$$

implies that $\lambda_m(L_g) \geq \underline{\lambda}_m$.

Proof: First, we note by Lemma 3.1 that if (2) holds, then the matrix $L_e - \underline{\lambda}_m M$ has at most $m-1$ negative eigenvalues. By congruence, $M^{-1/2}L_eM^{-1/2} - \underline{\lambda}_m I$ has at most $m-1$ negative eigenvalues, which means that the symmetric positive semidefinite matrix $L_s \triangleq M^{-1/2}L_eM^{-1/2}$ has at most $m-1$ eigenvalues less than $\underline{\lambda}_m$. Similarity of L_g to L_s implies that L_g has at most $m-1$ eigenvalues less than $\underline{\lambda}_m$, implying that $\lambda_m(L_g) \geq \underline{\lambda}_m$. ∎

We now present a convex feasibility program that enforces the lower eigenvalue bound sufficient linear matrix inequality condition of Theorem 3.2, as follows:

$$\begin{aligned} \text{Find} \quad & M,\ K \\ \text{subject to} \quad & Q_m^T(AKA^T - \underline{\lambda}_m M)Q_m \succeq 0 \\ & M \succ 0,\ K \succeq 0;\ M,\ K \text{ diagonal}. \end{aligned} \tag{3}$$

Theorem 3.2 provides only a sufficient condition to imply $\lambda_m(L_g) \geq \underline{\lambda}_m$, because the choice of Q_m is arbitrary. We now present a necessary and sufficient condition enabled by a specific choice of Q_m:

Theorem 3.3: The inequality $\lambda_m(L_g) \geq \underline{\lambda}_m$ holds if and only if $\mathcal{Q}_m^T(L_e - \underline{\lambda}_m M)\mathcal{Q}_m \succeq 0$, where $\mathcal{Q}_m \in \mathbb{R}^{n\times(n-m+1)}$ is the matrix whose columns are the eigenvectors corresponding to the $n - m + 1$ largest eigenvalues of $L_e - \underline{\lambda}_m M$.

Proof: Necessity follows from Theorem 3.2. To prove sufficiency, suppose that $\lambda_m(L_g) \geq \underline{\lambda}_m$. By similarity, $L_s = M^{-1/2}L_eM^{-1/2}$ has the same spectrum as L_g. Then $L_s - \underline{\lambda}_m I$ has at most $m-1$ negative eigenvalues. By congruence, so does $L_e - \underline{\lambda}_m M$. Considering the projection matrix $\mathcal{Q}_m\mathcal{Q}_m^T$, it follows that $(L_e - \underline{\lambda}_m M)\mathcal{Q}_m\mathcal{Q}_m^T$ must have exclusively nonnegative eigenvalues. Lemma 2.1 then implies that $\mathcal{Q}_m^T(L_e - \underline{\lambda}_m M)\mathcal{Q}_m \succeq 0$. ∎

Theorem 3.3 is the basis for an iterative procedure presented in Section IV-A that allows for improved performance when the constraints of (3) are paired with an objective.

B. Bounding Eigenvalues From Above

Given $p \leq n$ and $\overline{\lambda}_p \geq 0$, we wish to design node and edge weights M and K, respectively, such that $\lambda_p(L_g) \leq \overline{\lambda}_p$. We construct a linear matrix inequality enforcing this eigenvalue constraint. The analysis is similar to that of the previous section, and so the proofs are omitted.

Theorem 3.4: Let $U_p \in \mathbb{R}^{n\times p}$ be a full column rank matrix whose columns are orthogonal. The constraint

$$U_p^T(\overline{\lambda}_p M - L_e)U_p \succeq 0 \tag{4}$$

implies that $\lambda_p(L_G) \leq \overline{\lambda}_p$.

We now present a convex feasibility program that enforces the upper eigenvalue bound sufficient linear matrix inequality condition of Theorem 3.4, as follows:

$$\begin{aligned} \text{Find} \quad & M,\ K \\ \text{subject to} \quad & U_p^T(\overline{\lambda}_p M - AKA^T)U_p \succeq 0 \\ & M \succ 0,\ K \succeq 0;\ M,\ K \text{ diagonal}. \end{aligned} \tag{5}$$

As in the case of bounding eigenvalues from below, Theorem 3.4 provides only a sufficient condition to imply $\lambda_p(L_g) \leq \overline{\lambda}_p$. The following theorem gives a necessary and sufficient condition enabled by a specific choice of U_p:

Theorem 3.5: The inequality $\lambda_p(L_g) \leq \overline{\lambda}_p$ holds if and only if $\mathcal{U}_p^T(\overline{\lambda}_p M - L_e)\mathcal{U}_p \succeq 0$, where $\mathcal{U}_p \in \mathbb{R}^{n\times p}$ is the matrix whose

columns are the eigenvectors corresponding to the p smallest eigenvalues of $\overline{\lambda}_p M - L_e$.

An iterative procedure presented in Section IV-B employs Theorem 3.5 and allows for improved performance when the constraints of (3) and (5) are paired with an objective. A special case of Theorem 3.5 is when $p = n$. In this case, the $\mathcal{U}_p$ that satisfies Theorem 3.5 is a square, orthogonal matrix, and thus the eigenvalues of $\mathcal{U}_p^T(\overline{\lambda}_p M - AKA^T)\mathcal{U}_p$ and $\overline{\lambda}_p M - AKA^T$ are equal by Lemma 2.1. Theorem 3.5 therefore simplifies to the following corollary:

Corollary 3.6: The inequality $\lambda_n(L_g) \leq \overline{\lambda}_n$ holds if and only if $\overline{\lambda}_n M - L_e \succeq 0$.

IV. Examples of Graph Design Problems

We provide two sample problems that can be addressed by combining (3) and (5), demonstrating the flexibility of our formulation to impose individual constraints on several eigenvalues simultaneously. In our numerical examples, we require that all node and edge weights be contained in $[\epsilon, \epsilon^{-1}]$, where $\epsilon < 1$ is a small positive parameter that guarantees that the largest and smallest weights do not have too great a relative difference. Smaller values of ϵ increase the number of feasible M and K matrices, making it more likely that any given constraint set is feasible, but run the risk of numerical loss of precision with very large differences in individual node and edge weightings. We perform our numerical examples using CVX, a package for *disciplined convex programming* [17], [18], and the SDPT3 interior point solver [19].

A. Minimizing the Largest Eigenvalue Given a Minimum Connectivity Constraint

In formation control problems (see, e.g., Section V), it is desirable to have a lower bound on λ_2 to ensure adequate convergence time while at the same time imposing an upper bound on λ_n for stability. We present the problem of minimizing the largest eigenvalue $\lambda_n(L_g)$ of a graph given the requirement $\lambda_2(L_g) \geq \underline{\lambda}_2$, making use of (3) and (5) as well as including upper and lower bounds on the entries of M and K, as follows:

$$\begin{aligned} \underset{\kappa, M, K}{\text{minimize}} \quad & \kappa \\ \text{subject to} \quad & \kappa\underline{\lambda}_2 M - AKA^T \succeq 0 \\ & Q_2^T(AKA^T - \underline{\lambda}_2 M)Q_2 \succeq 0 \\ & \epsilon^{-1} I \succeq M \succeq \epsilon I, \epsilon^{-1} I \succeq K \succeq \epsilon I;\ M,\ K \text{ diagonal.} \end{aligned} \tag{6}$$

The problem is quasiconvex for any $Q_2 \in \mathbb{R}^{n\times(n-1)}$. To find the optimal $\kappa = \lambda_n(L_g)/\underline{\lambda}_2$ for the problem, we perform a bisection on the interval $[\underline{\lambda}_2, \lambda_n(L)]$, where in each iteration, a convex feasibility problem is solved for the value of κ given by the bisection. As discussed in Section III-A, an arbitrary choice of Q_2 may lead to conservatism in the optimal κ achieved. To improve the value of κ, we propose Algorithm 1, which makes use of Theorem 3.3 and updates Q_2.

Algorithm 1 Iterative Updates for Q_2

1: $M \Leftarrow I$, $K \Leftarrow I$, $\mu > 0$.

2: **repeat**

3: Set Q_2 to be the matrix whose columns are the eigenvectors corresponding to the $n-1$ largest eigenvalues of $AKA^T - \underline{\lambda}_2 M$.

4: Solve **(6)** and update M, K.

5: **until** $|\kappa_i - \kappa_{i-1}| \leq \mu$ OR $(\max(M) = \epsilon^{-1}$ AND $\min(M) = \epsilon)$ OR $(\max(K) = \epsilon^{-1}$ AND $\min(K) = \epsilon)$.

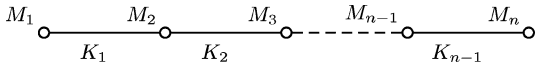

Fig. 1. Chain graph with n nodes.

We note that when M and K are identity and Q_2 is initialized as in Algorithm 1, the columns of Q_2 are orthogonal both to each other and to $\mathbf{1}_n$. In our implementation, we choose μ to be very small, and the effective stopping criteria are the conditions on M and K involving ϵ. The parameter ϵ can be tuned by being made smaller or larger to achieve improved or worsened values of κ with the resulting difference $|\kappa_i - \kappa_{i-1}|$ relatively smaller or larger, respectively, when the algorithm terminates.

1) Numerical Example: For an unweighted chain graph with twenty nodes obeying the structure of Fig. 1, we have $\lambda_2(L) = 0.0246$ and $\kappa = \lambda_{20}(L)/\lambda_2(L) = 161.6016$. We set $\epsilon = 10^{-2}$, and apply our method to reduce κ. For the first three experiments, (6) was solved with Q_2 set to be a matrix whose $n-1$ columns are orthogonal to $\mathbf{1}_n$. The lower eigenvalue bound was set to be $\underline{\lambda}_2 = \lambda_2(L)$. Solving for edges only, with nodes weighted to identity, produced no re-weighting of edges, and so κ was unchanged. In contrast, solving for nodes only, with edges weighted to identity, resulted in $\kappa = 123.5286$. Simultaneous optimization with both the nodes and edges as decision variables produced a marked improvement to $\kappa = 52.8616$. Allowing Q_2 to vary in accordance with Algorithm 1 described above resulted in $\kappa = 13.0499$. By setting $\epsilon = 10^{-3}$, we achieved $\kappa = 6.3021$.

B. Minimizing the Gap Between λ_p and λ_{p+1}

We consider graphs with *clusters*, that is, groupings of densely connected nodes with sparse external links. The Laplacian of a graph with p clusters exhibits, in addition to the first eigenvalue at zero, $p-1$ additional eigenvalues close to zero. Thus, in such graphs, there is a gap between the first p eigenvalues and the rest. Examples of systems obeying the clustered structure have been studied in building sensor networks [20] and power systems [21], where distributed estimation algorithms are increasingly prevalent. The gap in the eigenvalues may be undesirable because it leads to a two-time-scale behavior in the convergence of these algorithms [22].

To obtain uniform convergence rates for nodes in different clusters, we maximize λ_2 while requiring $\lambda_{p+1} \leq \overline{\lambda}_{p+1}$, and in so doing, minimize the gap between $\lambda_p(L_g)$ and $\lambda_{p+1}(L_g)$. Additionally, we fix $\lambda_n(L_g) \leq \overline{\lambda}_n$, so that the rest of the spectrum of the weighted Laplacian does not deviate far from its original location. The problem is solved with a bisection to maximize κ on the interval $[\lambda_2(L), \overline{\lambda}_{p+1}]$. We impose upper and lower bounds on the entries of M and K, and introduce $Q_2 \in \mathbb{R}^{n\times(n-1)}$ and $U_{p+1} \in \mathbb{R}^{n\times(p+1)}$ defined according to Theorems 3.2 and 3.4, respectively. We now write the quasiconvex problem, with $\kappa = \overline{\lambda}_{p+1}/\lambda_2(L_g)$, as follows:

$$\begin{aligned} \underset{\kappa, M, K}{\text{maximize}} \quad & \kappa \\ \text{subject to} \quad & \overline{\lambda}_n M - AKA^T \succeq 0 \\ & Q_2^T(AKA^T - \kappa M)Q_2 \succeq 0 \\ & U_{p+1}^T(\overline{\lambda}_{p+1} M - AKA^T)U_{p+1} \succeq 0 \\ & \epsilon^{-1} I \succeq M \succeq \epsilon I,\ \epsilon^{-1} I \succeq K \succeq \epsilon I,\ M,\ K \text{ diagonal.} \end{aligned} \tag{7}$$

We can realize significant improvements in reducing the gap between $\lambda_p(L_g)$ and $\lambda_{p+1}(L_g)$ by employing Algorithm 2, an iterative procedure similar to Algorithm 1 of Section IV-A.

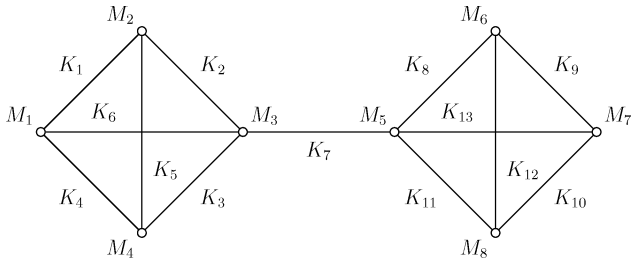

Fig. 2. Eight-node graph with two clusters.

Algorithm 2 Iterative Updates for Q_2, U_{p+1}

1: $M \Leftarrow I$, $K \Leftarrow I$, $\mu > 0$.

2: **repeat**

3: Set Q_2 to be the matrix whose columns are the eigenvectors corresponding to the $n-1$ largest eigenvalues of $AKA^T - \underline{\lambda}_2 M$.

4: Set U_{p+1} to be the matrix whose columns are the eigenvectors corresponding to the $p+1$ smallest eigenvalues of $\overline{\lambda}_{p+1}M - AKA^T$.

5: Solve **(7)** and update M, K.

6: **until** $|\kappa_i - \kappa_{i-1}| \leq \mu$ OR $(\max(M) = \epsilon^{-1}$ AND $\min(M) = \epsilon)$ OR $(\max(K) = \epsilon^{-1}$ AND $\min(K) = \epsilon)$.

1) Numerical Example: Consider the eight node graph with two clusters in Fig. 2. Such a graph, with identical weights, exhibits a significant gap between λ_2 and λ_3. We have the relation $\lambda_3(L)/\lambda_2(L) = 11.2931$, with the eigenvalues of the unweighted graph at $\{0.0000,\ 0.3542,\ 4.0000,\ 4.0000,\ \ldots,\ 4.0000,\ 5.6458\}$.

Our goal is to reduce the gap $\lambda_3(L)/\lambda_2(L)$ by increasing the second eigenvalue while bounding the third and eighth eigenvalues from above. To do so, we employ Algorithm 2, iteratively updating both Q_2 and U_3 while requiring $\lambda_3(L_g) \leq 4.0000$ and $\lambda_n(L_g) \leq 5.6458$ and setting $\epsilon = 10^{-2}$. We find the optimal value $\lambda_2(L_g) = 4.0000$, with $\lambda_3(L_g)/\lambda_2(L_g) = 1$.

V. Application to Multi-Agent Systems

We now apply the results of Section IV-A to multi-agent systems whose feedback structure is described by a graph Laplacian.

Each of the n subsystems possesses identical dynamics, as follows:

$$P_i : \begin{cases} \dot{x}_i &= Fx_i + Gu_i \\ y_i &= Hx_i. \end{cases} \tag{8}$$

and is controlled according to the feedback law

$$u_i = -M_i^{-1} \sum_{j \in \mathcal{N}_i} K_{\{i,j\}}(y_i - y_j) \tag{9}$$

where $F \in \mathbb{R}^{n_s \times n_s}$, $G \in \mathbb{R}^{n_s \times n_o}$, and $H \in \mathbb{R}^{n_o \times n_s}$, with n_s and n_o the dimension of the state space and input and output, respectively. $\mathcal{N}_i$ denotes the neighbors of agent i, that is, the other agents whom agent i senses. We assume that the individual plants are stable or can be stabilized by local state feedback (see the numerical example below). Therefore, we assume that F is Hurwitz. M_i denotes entry i of the diagonal node weighting matrix M, while $K_{\{i,j\}}$ denotes the entry of the edge weighting matrix K that corresponds to the edge linking nodes v_i and v_j. The block diagram of the system is shown in Fig. 3, with each

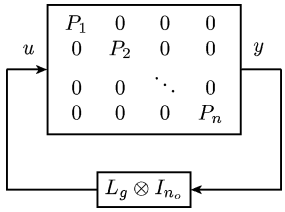

Fig. 3. Block diagram of the multi-agent system (8)–(9).

subsystem P_i having input given by (9). We let $x = [x_1^T, \ldots, x_n^T]^T$, and rewrite (8) and (9) as

$$\dot{x} = [I_n \otimes F - L_g \otimes (GH)]x. \tag{10}$$

As a consequence of the identical dynamics of each subsystem, the system can be decoupled into n identical subsystems by a change of coordinates using the basis of eigenvectors of L_g [3]. Let U be a change-of-coordinates matrix that diagonalizes L_g and let D be the diagonal matrix of eigenvalues of L_g. Then $D = U^{-1}L_gU$. Now let $V = U \otimes I$, and let $\tilde{x} = V^{-1}x$. In the new coordinates, the dynamics are given by

$$\dot{\tilde{x}} = [I_n \otimes F - D \otimes (GH)]\tilde{x}, \tag{11}$$

and, thus, the eigenvalues are determined from the characteristic polynomials of $F - \lambda_i(L_g)GH$, $i = 1, \ldots, n$. This means that the multi-agent system can be analyzed as n decoupled feedback systems with constant gain $\lambda_i(L_g)$, $i = 1, \ldots, n$. In particular, larger Laplacian eigenvalues imply higher gains for these decoupled systems, which is often undesirable. For example, if the transfer function $H(sI - F)^{-1}G$ has non-minimum phase zeros or relative degree higher than two, high gain will result in right half plane poles, rendering the multi-agent system unstable. The largest eigenvalue minimization method of Section IV-A can mitigate this instability by finding a node and edge weighting such that the spectrum of L_g spectrum falls within a range specified by design requirements.

1) Numerical Example: We consider formation control for four planar vertical takeoff and landing, or PVTOL, aircraft, as described in [23]. We model the state of the aircraft by its lateral position, x, vertical position y, and its roll, θ. The equations of motion, in input-output linearized form, are given by the following:

$$\begin{aligned} \ddot{x} &= u_1 \\ \ddot{y} &= u_2 \\ \ddot{\theta} &= \epsilon^{-1}(\sin\theta + \cos\theta u_1 + \sin\theta u_2). \end{aligned} \tag{12}$$

The zero dynamics of the system are unstable and the system is non-minimum phase

$$\ddot{\theta} = \epsilon^{-1} \sin\theta. \tag{13}$$

We assume that the aircraft are in hover operation and are stabilized vertically, so we discard y and v_2, the vertical thrust input. We set $\epsilon = 0.1$ and see that the linearized dynamics around $x = 0, \theta = 0$ are:

$$\tilde{F} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 10 & 0 \end{bmatrix} \quad G = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 10 \end{bmatrix} \quad H = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 0 \end{bmatrix}^T. \tag{14}$$

The input to each aircraft is dictated by the input term of (8), with the graph structure of a four node chain. We choose a state feedback $J = [\,0 \quad -90.6157 \quad 42.1472 \quad 13.2155\,]$, which renders $F = \tilde{F} - GJ$ stable. To achieve a reasonable response time and to maintain stability,

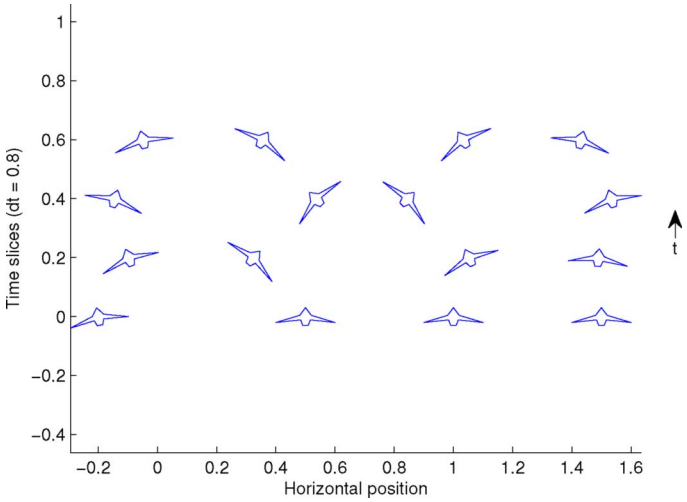

Fig. 4. PVTOL formation of four aircraft with unweighted, scaled graph. Each row represents a snapshot in time in ascending order. Each aircraft's maximum roll angle and amplitude of deviation from the desired relative position increases in time, indicating instability.

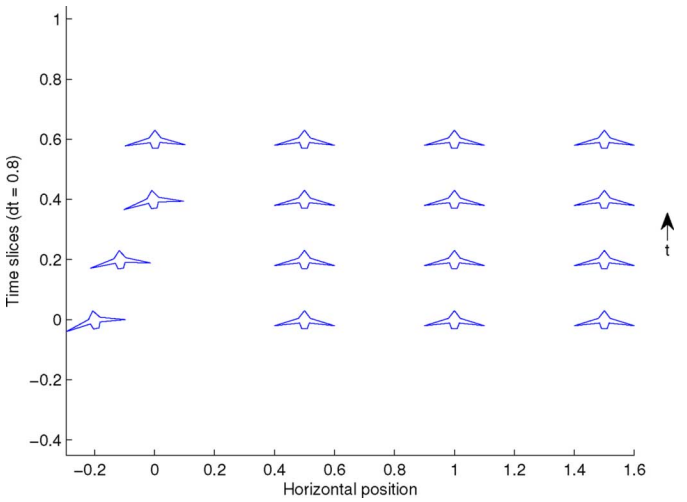

Fig. 5. PVTOL formation of four aircraft with weighted graph. Each row represents a snapshot in time in ascending order. Each aircraft's maximum roll angle and amplitude of deviation from the desired relative position of the aircraft decrease in time as it converges to formation.

we wish to contain the eigenvalues of the weighted Laplacian in the interval $[50, 125]$. In particular, the upper bound of this interval guarantees a damping ratio greater than 0.6. For the unweighted Laplacian, we have $\kappa = \lambda_4(L)/\lambda_2(L) = 5.8283$, which means that scaling the Laplacian by a constant $\alpha = 50/\lambda_2(L)$ to meet the lower eigenvalue constraint $\lambda_2(\alpha L) \geq 50$ will violate the upper eigenvalue constraint $\lambda_4(\alpha L) \leq 125$ and lead to instability as illustrated in Fig. 4. In contrast, applying the node and edge weights found by applying Algorithm 1 results in an improvement to $\kappa = 1.0202$. We show simulation results with the new weights in Fig. 5.

VI. Dual Formulation of the Quasiconvex Largest Eigenvalue Minimization Problem

Popular interior point methods, such as SDPT3 [19], when applied to the convex problems derived Section IV, are limited as to the size of graph they can handle, breaking down for many graphs with more than several tens of nodes. We show how using Lagrangian dual formulations enables our eigenvalue optimization framework to accommodate graphs with several hundred nodes and edges. In the following exposition, we derive the dual of the largest eigenvalue minimization problem of Section IV-A.

We begin with the convex problem solved as part of the solution to the largest eigenvalue minimization problem. We denote by $\langle \cdot, \cdot \rangle$ the trace inner product of two matrices of appropriate dimension. We set $Q \in \mathbb{R}^{n \times (n-1)}$ to be a matrix with columns orthogonal to each other and to $\mathbf{1}_n$. The convex problem we dualize is

$$\begin{aligned} \text{Find} \quad & K, M \\ \text{subject to} \quad & \kappa M - AKA^T \succeq 0 \\ & Q^T(AKA^T - M)Q \succeq 0 \\ & M \succeq \epsilon I,\ K \succeq 0; M,\ K \text{ diagonal}. \end{aligned} \tag{15}$$

To derive the dual problem, we note that the Lagrangian function is

$$\begin{aligned} L(K, M, Z, R, T, P) = & -\langle Z, \kappa M - AKA^T \rangle \\ & - \langle R, Q^T(AKA^T - M)Q \rangle \\ & - v^T(\text{diag}(M) - \epsilon \mathbf{1_n}) - w^T \text{diag}(K). \end{aligned} \tag{16}$$

We seek to obtain a finite minimization of the Lagrangian function with respect to M and K. Thus, minimizing (16) with respect to M yields the constraint:

$$\text{diag}(\tilde{R} - \kappa Z) = v \tag{17}$$

where we have defined a new variable $\tilde{R} = QRQ^T$, from which we observe that $\tilde{R}\mathbf{1}_n = 0$. Likewise, minimizing the Lagrangian with respect to K yields the constraint:

$$\text{diag}(A^T(Z - \tilde{R})A) = w. \tag{18}$$

To express the nullspace constraint on $\tilde{R}$ as a single equality constraint, we begin by defining the matrix $E \in \mathbb{R}^{n \times n}$ to be $E = \mathbf{1}_n \mathbf{1}_n^T$, and state and prove the following lemma:

Lemma 6.1: Let G be a positive semidefinite matrix. Then $G\mathbf{1}_n = 0$ if and only if $\langle G, E \rangle = 0$.

Proof:

$$\begin{aligned} G\mathbf{1}_n = 0 \iff & \mathbf{1}_n^T G \mathbf{1}_n = 0 \\ \iff & \text{tr}(\mathbf{1}_n^T G \mathbf{1}_n) = 0 \iff \langle G, E \rangle = 0. \end{aligned}$$

■

Combining the constraints found by minimizing (16) with respect to M and K, we now write the resulting dual problem, where we have eliminated the slack variable w:

$$\begin{aligned} \underset{Z, \tilde{R}, v}{\text{maximize}} \quad & \mathbf{1}_n^T v \\ \text{subject to} \quad & \text{diag}(A^T(Z - \tilde{R})A) \geq 0 \\ & \text{diag}(\tilde{R} - \kappa Z) = v \\ & \langle \tilde{R}, E \rangle = 0 \\ & Z, \tilde{R} \succeq 0,\ v \geq 0. \end{aligned} \tag{19}$$

TABLE I
RANDOM GRAPHS WITH v NODES AND e EDGES

n, m, κ	Primal Runtime (s)	Dual Runtime (s)
50, 100, 3	16.0664	4.5942
50, 500, 1.5	48.6661	7.9857
100, 200, 3	141.2667	11.7235
100, 1000, 1.5	N/A	27.1863
200, 400, 3	N/A	32.9802

TABLE II
CHAIN GRAPHS WITH n NODES

n	Primal Runtime (s)	Dual Runtime (s)
50	21.0757	7.2228
100	N/A	18.8605
200	N/A	44.9283
400	N/A	207.3877

The dual problem we have derived explicitly separates semidefinite matrix variables and linear variables as well as limits the growth of the number of equality constraints to scale linearly with the number of nodes and edges in the graph. Solving it using interior point methods is reasonably fast for graphs with up to 1000 edges, meaning that in addition to being substantially faster than the primal formulation, the dual formulation can accommodate significantly larger graphs.

We compare the performance of the SDPT3 algorithm on the primal and dual problems. The goal is to find feasible M and K matrices for (15) and (19) given κ. We characterize the performance in terms of CPU runtime in seconds on an Intel Quad Core 2 Duo 2.2 GHz system with 8 GB of RAM. In Table I, we consider random graphs with n nodes, m edges, and parameter κ, while in Table II, we consider chain graphs with n nodes with $\kappa = 3$. While the primal formulation cannot accommodate chain graphs with more than 65 nodes, the dual formulation can accommodate graphs with more than 650 nodes, an order of magnitude improvement.

VII. CONCLUSION

The graph Laplacian is an indispensable tool for assessing the dynamics of a multi-agent system. In this paper, we have presented a novel approach to impose bounds on the Laplacian spectrum. We have shown how node and edge weights can be adjusted using convex optimization to impose individual constraints on several eigenvalues simultaneously. In future work, we will quantitatively characterize optimality gaps and convergence properties for Algorithms 1 and 2. We plan to explore related problems such as synthesis of relative sensing networks [24]. We are also examining first order methods [25], [26] that will allow us to accommodate systems with thousands of agents.

REFERENCES

[1] F. Chung, Spectral Graph Theory American Mathematical Society, 1997.

[2] M. Fiedler, "Algebraic connectivity of graphs," *Czech. Mathemat. J.*, vol. 23, no. 2, pp. 298–305, 1973.

[3] J. Fax and R. Murray, "Information flow and cooperative control of vehicle formations," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1465–1476, Sep. 2004.

[4] A. Jadbabaie, J. Lin, and A. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 988–1001, Jun. 2003.

[5] R. Olfati-Saber and R. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1520–1533, Sep. 2004.

[6] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[7] M. Arcak, "Passivity as a design tool for group coordination," *IEEE Trans. Autom. Control*, vol. 52, no. 8, pp. 1380–1390, Aug. 2007.

[8] H. Bai and M. Arcak, "Instability mechanisms in cooperative control," *IEEE Trans. Autom. Control*, vol. 55, no. 1, pp. 258–263, Jan. 2010.

[9] S. Boyd, L. E. Ghaoui, E. Feron, and V. Balakrishnan, Linear Matrix Inequalities in System and Control Theory Society for Industrial Mathematics, Philadelphia, PA, 1994.

[10] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Syst. & Control Lett.*, vol. 53, no. 1, pp. 65–78, 2004.

[11] A. Ghosh, S. Boyd, and A. Saberi, "Minimizing effective resistance of a graph," *SIAM Rev.*, vol. 50, no. 1, pp. 37–66, 2008.

[12] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing Markov chain on a graph," *SIAM Rev.*, vol. 46, no. 4, pp. 667–689, 2004.

[13] J. Sun, S. Boyd, L. Xiao, and P. Diaconis, "The fastest mixing Markov process on a graph and a connection to a maximum variance unfolding problem," *SIAM Rev.*, vol. 48, no. 4, pp. 681–699, 2006.

[14] Y. Kim and M. Mesbahi, "On maximizing the second smallest eigenvalue of a state-dependent graph Laplacian," *IEEE Trans. Autom. Control*, vol. 51, no. 1, pp. 116–120, Jan. 2006.

[15] R. Horn and C. Johnson, *Matrix Analysis*. New York: Cambridge Univ. Press, 1990.

[16] F. Gantmacher, *The Theory of Matrices*. New York: Chelsea Pub. Co., 2000.

[17] M. Grant and S. Boyd, CVX: Matlab Software for Disciplined Convex Programming 2010 [Online]. Available: http://cvxr.com/cvx, 1.21

[18] M. Grant and S. Boyd, "Graph implementations for nonsmooth convex programs," *Recent Adv. Learn. and Control*, pp. 95–110, 2008.

[19] K. Toh, M. Todd, and R. Tutuncu, "SDPT3–A Matlab software package for semidefinite programming," *Optimiz. Meth. and Softw.*, vol. 11, no. 12, pp. 545–581, 1999.

[20] J. Kim, M. West, E. Scholte, and S. Narayanan, "Multiscale consensus for decentralized estimation and its application to building systems," in *Proc. American Control Conf.*, 2008, pp. 888–893.

[21] J. Chow, Ed., Time-Scale Modeling of Dynamic Networks With Applications to Power Systems Berlin/Heidelberg, Germany, Springer-Verlag, 1982.

[22] E. Biyik and M. Arcak, "Area aggregation and time-scale modeling for sparse nonlinear networks," *Syst. & Control Lett.*, vol. 57, no. 2, pp. 142–149, 2008.

[23] S. Sastry, *Nonlinear Systems: Analysis, Stability, and Control*. Berlin/Heidelberg, Germany: Springer–Verlag, 1999.

[24] D. Zelazo and M. Mesbahi, "Graph-theoretic analysis and synthesis of relative sensing networks," *IEEE Trans. Autom. Control*, vol. 56, no. 5, pp. 971–982, May 2011.

[25] Z. Wen, D. Goldfarb, and W. Yin, "Alternating direction augmented Lagrangian methods for semidefinite programming," *Math. Programm. Comput.*, vol. 3, no. 2, pp. 203–230, 2010.

[26] Z. Wen, D. Goldfarb, S. Ma, and K. Scheinberg, "Row by Row Methods for Semidefinite Programming," Tech. Rep. Dept. IEOR, Columbia Univ., New York, 2009.