Fast Linear Iterations for Distributed Averaging [1]

Lin Xiao Stephen Boyd
Information Systems Laboratory, Stanford University
Stanford, CA 94305-9510
lxiao@stanford.edu, boyd@stanford.edu

Abstract

We consider the problem of finding a linear iteration that yields distributed averaging consensus over a network, *i.e.*, that asymptotically computes the average of some initial values given at the nodes. When the iteration is assumed symmetric, the problem of finding the fastest converging linear iteration can be cast as a semidefinite program, and therefore efficiently and globally solved. These optimal linear iterations are often substantially faster than several simple heuristics that are based on the Laplacian matrix of the associated graph.

Keywords: distributed consensus, linear system, spectral radius, semidefinite program.

1 Introduction

We consider a network (a connected graph) $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ consisting of a set of nodes $\mathcal{N} = \{1, \ldots, n\}$ and a set of edges $\mathcal{E}$, where each edge $\{i, j\} \in \mathcal{E}$ is an unordered pair of distinct nodes. The set of neighbors of node i is denoted $\mathcal{N}_i = \{j \mid \{i, j\} \in \mathcal{E}\}$. Each node i holds an initial scalar value $x_i(0) \in \mathbf{R}$, and $x(0) = (x_1(0), \ldots, x_n(0))$ denotes the vector of the initial values on the network. We are interested in computing the average $(1/n)\sum_{i=1}^n x_i(0)$, via a distributed algorithm, in which the nodes only communicate with their neighbors.

Distributed averaging can be done in many ways. One straightforward method is flooding. Each node maintains a table of the initial node values of all the nodes, initialized with its own node value only. At each step, the nodes exchange information from their own tables and the tables of their neighbors. After a number of steps equal to the diameter of the network, every node knows all the initial values of all the nodes, so the average (or any other function of the initial node values) can be computed.

In this paper, we only consider *distributed linear iterations*, which have the form

$$x_i(t+1) = W_{ii}x_i(t) + \sum_{j \in \mathcal{N}_i} W_{ij}x_j(t), \quad i = 1, \ldots, n,$$

where $t = 0, 1, 2, \ldots$ and W_{ij} is the *weight* on x_j at node i. Setting $W_{ij} = 0$ for $j \notin \mathcal{N}_i$, this iteration can be written in vector form as

$$x(t+1) = Wx(t). \tag{1}$$

The constraint on the sparsity pattern of the matrix W can be expressed as $W \in \mathcal{S}$, where

$$\mathcal{S} = \{W \in \mathbf{R}^{n \times n} \mid W_{ij} = 0 \text{ if } \{i, j\} \notin \mathcal{E} \text{ and } i \neq j\}.$$

Equation (1) implies that $x(t) = W^t x(0)$ for all t. We want to choose the weight matrix W so that for any initial value $x(0)$, $x(t)$ converges to the average vector $\bar{x} = (\mathbf{1}^T x(0)/n)\mathbf{1} = ((1/n)\mathbf{1}\mathbf{1}^T)x(0)$, *i.e.*,

$$\lim_{t \to \infty} x(t) = \lim_{t \to \infty} W^t x(0) = \left((1/n)\mathbf{1}\mathbf{1}^T\right) x(0).$$

(Here $\mathbf{1}$ denotes the vector with all coefficients one.) This is equivalent to the matrix equation

$$\lim_{t \to \infty} W^t = (1/n)\mathbf{1}\mathbf{1}^T. \tag{2}$$

Assuming this holds, we define the *asymptotic convergence factor* as

$$r_{\text{asym}}(W) = \sup_{x(0) \neq \bar{x}} \lim_{t \to \infty} \left(\frac{\|x(t) - \bar{x}\|_2}{\|x(0) - \bar{x}\|_2} \right)^{1/t},$$

and the associated *convergence time*

$$\tau_{\text{asym}} = \frac{1}{\log(1/r_{\text{asym}})}, \tag{3}$$

which gives the (asymptotic) number of steps for the error to decrease by the factor $1/e$.

[1] This research was supported in part by DARPA grant F33615-99-C-3014 and AFOSR award F49620-01-1-0365.

Another measure of the speed of convergence is the *per-step convergence factor* which is defined as

$$r_{\rm step}(W) = \sup_{x(t)\neq\bar{x}} \frac{\|x(t+1)-\bar{x}\|_2}{\|x(t)-\bar{x}\|_2}.$$

The convergence time $\tau_{\rm step}$ is defined similar to (3).

In this paper we consider the following problem: find the weight matrix $W \in \mathcal{S}$, consistent with the given network, that makes the convergence as fast as possible. In terms of the asymptotic convergence factor, it can be posed as the optimization problem:

$$\begin{array}{ll} \text{minimize} & r_{\rm asym}(W) \\ \text{subject to} & W \in \mathcal{S}, \quad \lim_{t\to\infty} W^t = (1/n)\mathbf{1}\mathbf{1}^T. \end{array} \quad (4)$$

Here W is the optimization variable, and the network is the problem data. We call this problem the *fastest distributed linear averaging* (FDLA) problem. A similar problem can be formulated to minimize the per-step convergence factor $r_{\rm step}(W)$.

The distributed averaging problem arises in the context of coordination of networks of autonomous agents, in particular, the *consensus* or *agreement* problem among the agents (see, *e.g.*, [1] for a treatment in the computer science literature). Recently it has found a wide range of applications, in areas such as formation flight of unmanned air vehicles and clustered satellites, and coordination of mobile robots. The recent paper [2] studies linear and nonlinear consensus protocols in these applications with fixed network topology. Related coordination problems with time-varying topologies have been studied in [3] using a switched linear system model. In these previous works, the edge weights used in the linear iteration are either constant or only dependent on the degrees of their incident nodes.

The rest of this paper is organized as follows. In §2, we give necessary and sufficient conditions on the weight matrix for the distributed linear iteration to converge, and characterize the asymptotic and per-step convergence factors. In §3, we formulate the FDLA problem as spectral radius/norm minimization problems, and show how some variations can be formulated as semidefinite programs (SDP). In §4, we describe some simple heuristics for choosing the weight matrix, based on the Laplacian matrix of the graph, which at least guarantee convergence. In §5, we give some numerical examples, and show that the optimal weights often result in substantially faster convergence than those obtained from the simple heuristics. In §6, we extend the FDLA problem to a sparse graph design problem.

2 Convergence conditions

As we have seen, the linear iteration (1) converges to the average for any initial vector $x(0)\in\mathbf{R}^n$ if and only if (2) holds. We have the following theorem:

Theorem 1 *The equation (2) holds if and only if*

$$\begin{aligned} \mathbf{1}^T W &= \mathbf{1}^T, & (5)\\ W\mathbf{1} &= \mathbf{1}, & (6)\\ \rho\left(W-(1/n)\mathbf{1}\mathbf{1}^T\right) &< 1, & (7) \end{aligned}$$

where $\rho(\cdot)$ denotes the spectral radius of a matrix. Moreover,

$$\begin{aligned} r_{\rm asym}(W) &= \rho\left(W-(1/n)\mathbf{1}\mathbf{1}^T\right), & (8)\\ r_{\rm step}(W) &= \left\|W-(1/n)\mathbf{1}\mathbf{1}^T\right\|_2. & (9) \end{aligned}$$

(Here $\|\cdot\|_2$ denotes the spectral norm.)

Before proving the theorem, we first give some interpretations of the above conditions.

- Equation (5) states that $\mathbf{1}$ is a left eigenvector of W associated with the eigenvalue one. This implies that $\mathbf{1}^T x(t+1) = \mathbf{1}^T x(t)$ for all t, *i.e.*, the sum (therefore the average) of the vector of node values is preserved at each step.
- Equation (6) states that $\mathbf{1}$ is also a right eigenvector of W associated with the eigenvalue one. This means that $\mathbf{1}$ (or any multiple of it) is a fixed point of the iteration (1).
- Together with the first two conditions, equation (7) means that one is a simple eigenvalue of W, and that all other eigenvalues are strictly less than one in magnitude.

Proof: First we prove sufficiency. If W satisfies conditions (5) and (6), then

$$\begin{aligned} W^t-(1/n)\mathbf{1}\mathbf{1}^T &= W^t\left(I-(1/n)\mathbf{1}\mathbf{1}^T\right)\\ &= W^t\left(I-(1/n)\mathbf{1}\mathbf{1}^T\right)^t\\ &= \left(W\left(I-(1/n)\mathbf{1}\mathbf{1}^T\right)\right)^t\\ &= \left(W-(1/n)\mathbf{1}\mathbf{1}^T\right)^t, \end{aligned}$$

where in the second equality, we use the fact that $I-(1/n)\mathbf{1}\mathbf{1}^T$ is a projection matrix. Now applying condition (7) leads to the desired convergence (2).

To prove necessity, we use the fact that $\lim_{t\to\infty} W^t$ exists (*i.e.*, W is *semi-convergent*) if and only if there is a nonsingular matrix T such that

$$W = T\begin{bmatrix} I_\kappa & 0\\ 0 & Z\end{bmatrix}T^{-1},$$

where I_κ is the κ-dimensional identity matrix ($0 \leq \kappa \leq n$) and Z is a convergent matrix, *i.e.*, $\rho(Z) < 1$ (see, *e.g.*, [4]). Let $u_1, \ldots, u_n$ be the columns of T and $v_1^T, \ldots, v_n^T$ be the rows of T^{-1}. Then we have

$$\begin{aligned}\lim_{t\to\infty} W^t &= \lim_{t\to\infty} T \begin{bmatrix} I_\kappa & 0 \\ 0 & Z^t \end{bmatrix} T^{-1} \\ &= T \begin{bmatrix} I_\kappa & 0 \\ 0 & 0 \end{bmatrix} T^{-1} = \sum_{i=1}^{\kappa} u_i v_i^T. \quad (10)\end{aligned}$$

Since each $u_i v_i^T$ is a rank-one matrix and their sum $\sum_{i=1}^n u_i v_i^T = TT^{-1} = I$ has rank n, the matrix $\sum_{i=1}^{\kappa} u_i v_i^T$ must have rank κ. Comparing (2) and (10) gives $\kappa = 1$ and $u_1 v_1^T = (1/n)\mathbf{1}\mathbf{1}^T$, which implies that both u_1 and v_1 are multiples of $\mathbf{1}$. In other words, one is a simple eigenvalue of W and $\mathbf{1}$ is its associated left and right eigenvectors. Hence equations (5) and (6) hold. Moreover,

$$\rho\left(W - \frac{\mathbf{1}\mathbf{1}^T}{n}\right) = \rho\left(T \begin{bmatrix} 0 & 0 \\ 0 & Z \end{bmatrix} T^{-1}\right) = \rho(Z) < 1,$$

which is precisely condition (7).

Finally equations (8) and (9) follow directly from the error dynamics

$$\begin{aligned} x(t+1) - \bar{x} &= \left(W - (1/n)\mathbf{1}\mathbf{1}^T\right) x(t) \\ &= \left(W - (1/n)\mathbf{1}\mathbf{1}^T\right) (x(t) - \bar{x}). \end{aligned}$$

In other words, r_{asym} is the spectral radius of $W - (1/n)\mathbf{1}\mathbf{1}^T$, and r_{step} is its spectral norm. ∎

3 Fastest distributed linear averaging problems

Using theorem 1, the FDLA problem (4) can be formulated a spectral radius minimization problem:

$$\begin{array}{ll} \text{minimize} & \rho\left(W - (1/n)\mathbf{1}\mathbf{1}^T\right) \\ \text{subject to} & W \in \mathcal{S}, \quad \mathbf{1}^T W = \mathbf{1}^T, \quad W\mathbf{1} = \mathbf{1} \end{array} \quad (11)$$

with the optimization variable W. Even though the constraints in problem (11) are linear equalities, the problem in general is very hard. The reason is that the objective function, *i.e.*, the spectral radius of a matrix, is in general not a convex function; indeed it is not even Lipschitz continuous (see, *e.g.*, [5]). Some related spectral radius minimization problems are NP-hard [6, 7].

We can also formulate the FDLA problem, with per-step convergence factor, as the following spectral norm minimization problem:

$$\begin{array}{ll} \text{minimize} & \|W - (1/n)\mathbf{1}\mathbf{1}^T\|_2 \\ \text{subject to} & W \in \mathcal{S}, \quad \mathbf{1}^T W = \mathbf{1}^T, \quad W\mathbf{1} = \mathbf{1} \end{array} \quad (12)$$

In contrast to (11), this problem is convex, and can be solved globally and efficiently.

Now suppose we add the additional constraint that weights are symmetric, *i.e.*, $W_{ij} = W_{ji}$ for all $\{i, j\} \in \mathcal{E}$. In this case the spectral radius minimization problem (11) and the spectral norm minimization problem (12) coincide (since the spectral norm of a symmetric matrix is also its spectral radius). In this case, both problems can be cast as

$$\begin{array}{ll} \text{minimize} & \rho(W - (1/n)\mathbf{1}\mathbf{1}^T) \\ \text{subject to} & W \in \mathcal{S}, \quad W = W^T, \quad W\mathbf{1} = \mathbf{1} \end{array} \quad (13)$$

which is a convex problem. We refer to this problem as the *symmetric FDLA* problem.

The spectral norm minimization problem (12) can be expressed as a semidefinite program, by introducing a scalar variable s to bound the spectral norm $\|W - \mathbf{1}\mathbf{1}^T/n\|_2$, and expressing the norm bound constraint as a linear matrix inequality:

$$\begin{array}{ll} \text{minimize} & s \\ \text{subject to} & \begin{bmatrix} sI & W - (1/n)\mathbf{1}\mathbf{1}^T \\ W^T - (1/n)\mathbf{1}\mathbf{1}^T & sI \end{bmatrix} \succeq 0 \\ & W \in \mathcal{S}, \quad \mathbf{1}^T W = \mathbf{1}^T, \quad W\mathbf{1} = \mathbf{1}. \end{array}$$

Here the symbol $\succeq$ denotes matrix inequality, *i.e.*, $X \succeq Y$ means that $X - Y$ is positive semidefinite.

Similarly, the symmetric FDLA problem (13) can be expressed as the SDP

$$\begin{array}{ll} \text{minimize} & s \\ \text{subject to} & -sI \preceq W - (1/n)\mathbf{1}\mathbf{1}^T \preceq sI \\ & W \in \mathcal{S}, \quad W = W^T, \quad W\mathbf{1} = \mathbf{1}. \end{array} \quad (14)$$

This problem is closely related to the problem of finding the fastest mixing Markov chain on a graph [8]; the only difference is that in the FDLA problem, the weights can be (and the optimal ones often are) negative.

General background on SDP, eigenvalue optimization and associated interior-point methods can be found in, *e.g.*, [9, 10, 11] and references therein. In [12], we show how problem structure can be exploited to speed up interior-point methods for solving the FDLA problem, for networks with up to a thousand or so edges; we also describe a simple subgradient method that handles far larger problems, with up to one hundred thousand edges.

4 Heuristics based on the Laplacian

In this section we give some simple heuristics for choosing W that guarantee convergence of the distributed linear averaging iteration, and sometimes give reasonably fast convergence. These heuristics are based on the Laplacian matrix of the associated graph, and assign symmetric edge weights.

The *Laplacian matrix* L of the graph is defined as

$$L_{ij} = \begin{cases} -1 & \{i,j\} \in \mathcal{E} \\ d_i & i = j \\ 0 & \text{otherwise} \end{cases}$$

where d_i is the degree of node i (*i.e.*, the number of neighbors of the node i). The Laplacian matrix is a useful tool in algebraic graph theory, and its eigenstructure reveals many important properties of the graph (*e.g.*, [13, 14]). It appears to be very useful in the convergence analysis of consensus protocols [2], and has also been used in control of distributed dynamic systems (*e.g.*, [15, 16]). We note for future use that L is positive semidefinite, and since our graph is assumed connected, L has a simple eigenvalue zero, with corresponding eigenvector $\mathbf{1}$.

4.1 Constant edge weights

The simplest approach is to set all the edge weights (for neighboring nodes) equal to a constant α; the self-weights on the nodes are then chosen to satisfy the condition $W\mathbf{1} = \mathbf{1}$. This choice of weights can be expressed in terms of the Laplacian matrix

$$W = I - \alpha L. \tag{15}$$

In this case, equation (1) becomes the following commonly used iteration (see, *e.g.*, [2]):

$$x_i(t+1) = x_i(t) + \alpha \sum_{j \in \mathcal{N}_i} (x_j(t) - x_i(t)), \quad i = 1, \ldots, n.$$

Since L is positive semidefinite, we must have $\alpha > 0$ for the convergence condition $\rho\left(W - (1/n)\mathbf{1}\mathbf{1}\right) < 1$ to hold. From equation (15) we can express the eigenvalues of W in terms of those of L:

$$\lambda_i(W) = 1 - \alpha \lambda_{n-i+1}(L), \quad i = 1, \ldots, n,$$

where $\lambda_i(\cdot)$ denotes the ith largest eigenvalue of a symmetric matrix. In particular, the eigenvalue zero of L corresponds to the eigenvalue one of W (*i.e.*, $\lambda_n(L) = 0$, $\lambda_1(W) = 1$). The spectral radius of $W - (1/n)\mathbf{1}\mathbf{1}^T$ can then be expressed as

$$\begin{aligned} \rho\big(W - (1/n)\mathbf{1}\mathbf{1}^T\big) &= \max\{\lambda_2(W), \ -\lambda_n(W)\} \\ &= \max\{1 - \alpha\lambda_{n-1}(L), \ \alpha\lambda_1(L) - 1\}. \end{aligned}$$

From this we conclude that $\rho\left(W - (1/n)\mathbf{1}\mathbf{1}^T\right) < 1$ if and only if $0 < \alpha < 2/\lambda_1(L)$. The choice of α that minimizes $\rho\left(W - (1/n)\mathbf{1}\mathbf{1}^T\right)$ is

$$\alpha^\star = \frac{2}{\lambda_1(L) + \lambda_{n-1}(L)}. \tag{16}$$

This gives the best possible constant edge weight.

There are some simple bounds that give choices for α that do not require exact knowledge of the Laplacian spectrum. For example, we have $\lambda_1(L) \leq 2d_{\max}$, where $d_{\max} = \max_{i \in \mathcal{N}} d_i$ (see, *e.g.*, [13, 14]). So one easily determined convergence range for α is $0 < \alpha < 1/d_{\max}$. In fact, we can always use the *maximum-degree weight*

$$\alpha^{\text{md}} = \frac{1}{d_{\max}} \tag{17}$$

provided the graph is not bipartite.

4.2 Local-degree weights

Another method is to assign the weight on an edge based on the larger degree of its two incident nodes:

$$W_{ij} = \frac{1}{\max\{d_i, d_j\}}, \quad \{i,j\} \in \mathcal{E},$$

and then determine W_{ii} using $W\mathbf{1} = \mathbf{1}$. We call these the *local-degree weights*, since they depend only on the degrees of the two incident nodes. Similar to the maximum-degree weight, the local-degree weights guarantee convergence provided the graph is not bipartite. This method is particularly suitable for distributed implementation.

5 Examples

We first consider the small graph shown in figure 1. For this graph, the maximum-degree weight given by (17) is $\alpha^{\text{md}} = 1/6$, and the best constant edge weight found from (16) is $\alpha^\star = 0.227$. By solving the SDP (14), we found the optimal symmetric

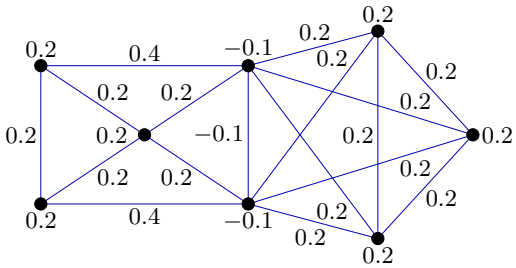

Figure 1: A small network with 8 nodes and 17 edges.

weights, which are labeled in figure 1. Note that the optimal weights for the two middle nodes, and the edge connecting them, are negative. To say the least, this is not an obvious choice of weights.

The asymptotic convergence factors and convergence times of the four choices of weights are

	$\rho(W-(1/n)\mathbf{1}\mathbf{1}^T)$	$\tau = 1/\log(1/\rho)$
max.-degree	0.746	3.413
local-degree	0.743	3.366
best constant	0.655	2.363
optimal symm.	0.600	1.958

For this example the maximum-degree and local-degree weights result in the slowest convergence, with the optimal symmetric weights substantially faster than the best constant weight.

5.1 A larger network

Next we consider the graph shown in figure 2, which has 50 nodes and 200 edges. This graph was randomly generated as follows. First we randomly generate 50 nodes, uniformly distributed on the unit square. Two nodes are connected by an edge if their distance is less than a specified threshold. Then we increase the threshold until the total number of edges is 200. (The resulting graph is connected).

The asymptotic convergence factors and convergence times, for the four different sets of weights, are summarized below

	$\rho(W-(1/n)\mathbf{1}\mathbf{1}^T)$	$\tau = 1/\log(1/\rho)$
max.-degree	0.971	33.980
local-degree	0.949	19.104
best constant	0.947	18.363
optimal symm.	0.902	9.696

It can be seen that the convergence with the optimal symmetric weights is roughly twice as fast as

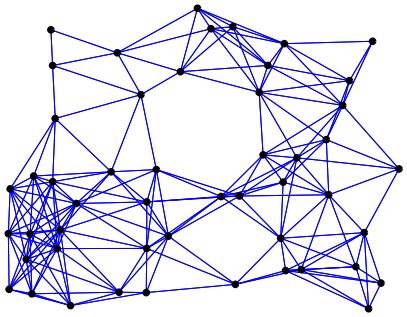

Figure 2: A randomly generated network.

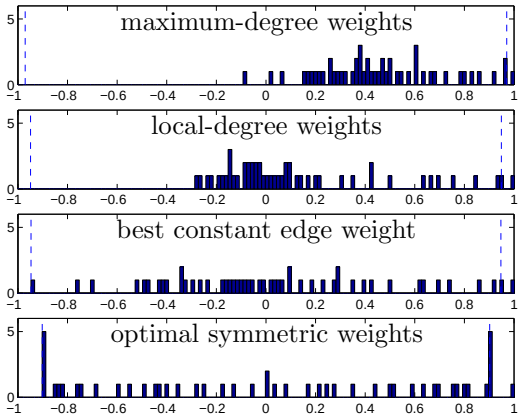

Figure 3: Distribution of the eigenvalues of W. The dashed lines indicate $\pm\rho\left(W-(1/n)\mathbf{1}\mathbf{1}^T\right)$.

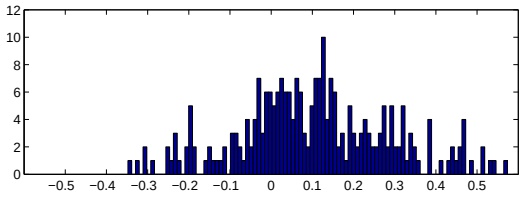

Figure 4: Distribution of optimal symmetric weights.

with the best constant edge weight and the local-degree weights, and is more than three times faster than the maximum-degree weights.

Figure 3 shows the eigenvalue distribution for the four weight matrices. Each of the distributions has a single eigenvalue at one. For the optimal symmetric weights, the eigenvalues (other than 1) have an approximately symmetric distribution, with many at or near the two critical values $\pm\rho(W-(1/n)\mathbf{1}\mathbf{1}^T)$. Figure 4 shows the distribution of the optimal symmetric edge and node weights. It shows that many of the weights are negative.

6 Extension: sparse graph design

An interesting variation on the FDLA problem is to find a sparse subgraph of the given graph, while guaranteeing a certain convergence factor. In other words, we seek a weight matrix with as many zero entries as possible, subject to a prescribed maximum for the convergence factor. This is a difficult combinatorial problem, but one very effective heuristic for this problem is to minimize the ℓ_1 norm of the vector of edge weights (*e.g.*, [17, §6]).

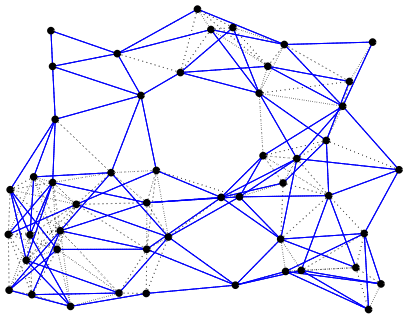

Figure 5: Sparse network design. The dotted lines show edges that are assigned zero weight.

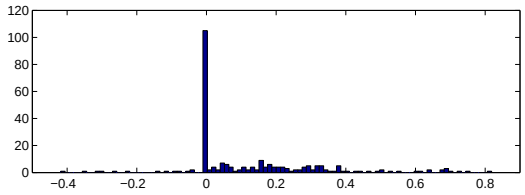

Figure 6: Weight distribution of sparse graph design.

For example, given the maximum allowed asymptotic convergence factor $r^{\max}$, the ℓ_1 heuristic for the sparse graph design problem (with symmetric weights) can be posed as the convex problem

$$\begin{array}{ll} \text{minimize} & \sum_{\{i,j\}\in\mathcal{E}} |W_{ij}| \\ \text{subject to} & -r^{\max} I \preceq W - (1/n)\mathbf{1}\mathbf{1}^T \preceq r^{\max} I \\ & W \in \mathcal{S}, \quad W = W^T, \quad W\mathbf{1} = \mathbf{1}. \end{array}$$

More sophisticated heuristics for sparse design and minimum rank problems can be found in, *e.g.*, [18].

To demonstrate this idea, we applied the ℓ_1 heuristic to the example described in §5.1. We set the guaranteed convergence factor $r^{\max} = 0.910$, which is only slightly larger than the minimum factor 0.902. The resulting edge weight vector is relatively sparse; the number of edges with non-zero weights is reduced from 200 to 96. This is illustrated in figure 5. Figure 6 shows the distribution of the weights for the sparse network, and should be compared to the distribution shown in figure 4.

There are many other interesting extensions of the basic FDLA problem. For example, one can consider criteria other than the asymptotic convergence factor or per-step factor in the selection of the weight matrix, and the distributed linear iteration needs not to converge to the average vector. We discuss many such extensions in [12].

References

[1] N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1996.

[2] R. Olfati-Saber and R. M. Murray. Consensus protocols for networks of dynamic agents. In *Proc. of American Control Conference*, Denver, CO, June 2003.

[3] A. Jadbabaie, J. Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. To appear, *IEEE Transactions Automatic Control*, 2003.

[4] C. D. Meyer and R. J. Plemmons. Convergence powers of a matrix with applications to iterative methods for singular linear systems. *SIAM Journal on Numerical Analysis*, 14(4):699–705, 1977.

[5] M. L. Overton and R. S. Womersley. On minimizing the spectral radius of a nonsymmetric matrix function — optimality conditions and duality theory. *SIAM J. Matrix Analysis & Applic.*, 9:473–498, 1988.

[6] V. Blondel and J. N. Tsitsiklis. NP-hardness of some linear control design problems. *SIAM Journal on Control and Optimization*, 35:2118–2127, 1997.

[7] A. Nemirovskii. Several NP-hard problems arising in robust stability analysis. *Mathematics of Control, Signals and Systems*, 6:99–105, 1993.

[8] S. Boyd, P. Diaconis, and L. Xiao. Fastest mixing Markov chain on a graph. Submitted to *SIAM Review*, problems and techniques section, February 2003. Available at `http://www.stanford.edu/~boyd/fmmc.html`.

[9] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

[10] A. S. Lewis and M. L. Overton. Eigenvalue optimization. *Acta Numerica*, 5:149–190, 1996.

[11] H. Wolkowicz, R. Saigal, and L. Vandengerghe, editors. *Handbook of Semidefinite Programming, Theory, Algorithms, and Applications*. Kluwer, 2000.

[12] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. Submitted to *Systems and Control Letters*, March 2003. Available on the Internet at `http://www.stanford.edu/~boyd/fastavg.html`.

[13] R. Merris. Laplacian matrices of graphs: a survey. *Linear Algebra and Its Applic.*, 197:143–176, 1994.

[14] C. Godsil and G. Royle. *Algebraic Graph Theory*. Springer, 2001.

[15] A. Fax and R. M. Murray. Graph Laplacians and vehicle formation stabilization. In *Proc. 15th IFAC World Congr. Automat. Control*, Barcelona, July 2002.

[16] M. Mesbahi. On a dynamic extension of the theory of graphs. In *Proceedings of the American Control Conference*, Anchorange, AL, May 2002.

[17] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2003. Available at `http://www.stanford.edu/~boyd/cvxbook.html`.

[18] M. Fazel, H. Hindi, and S. Boyd. A rank minimization heuristic with application to minimum order system approximation. In *Proc. American Control Conf.*, pages 4734–4739, Arlington, VA, June 2001.