Exploiting Memory in Dynamic Average Consensus

Bryan Van Scoy, Randy A. Freeman, and Kevin M. Lynch

***Abstract*— In the discrete-time average consensus problem, each agent in a network has a local input and communicates with neighboring agents to calculate the global average of all agent inputs. We analyze diffusion-like algorithms where each agent maintains an internal state which it updates at each time step using its local input together with information it receives from neighboring agents. The agent's estimate of the global average input is then a local function of its internal state.**

Local memory on each agent can be used to enhance the performance of average consensus estimators in several ways. Agents can use memory to store both internal state variables as well as intermediate diffusion calculations within each time step. We exploit memory to design two types of estimators. First, we design feedback estimators which track constant input signals with zero steady-state error. Such estimators produce estimates that converge exponentially to the global average, and we consider the cost of an estimator to be the largest time constant of the exponential decay of its estimation errors. However, we measure time using normalized units of communicated real variables per agent, so that estimators requiring more communication per time step are potentially costlier even if they converge in fewer time steps. We then show that a certain estimator having two internal state variables and one diffusion calculation per time step achieves the minimal cost over all graphs and all estimators with one or two states no matter how many intermediate diffusion calculations are stored. Second, we design a feedforward estimator which tracks time-varying signals whose frequencies lie below some cut-off frequency. The steady-state error is finite, but can be made arbitrarily small using enough diffusion calculations per time step.

I. INTRODUCTION

Given a group of agents, the average consensus problem is for each agent to calculate the average of agent inputs using only information obtained from neighboring agents. This is a fundamental problem in distributed control which has numerous applications such as formation control [1], [2], distributed Kriged Kalman filtering [3], and distributed merging of feature-based maps [4]. We study diffusive algorithms which are scalable, distributed, independent of graph structure, and use small amounts of memory, communication, and computation on each agent.

With diffusive algorithms, agents can average internal variables with their local neighbors as well as perform any necessary internal calculations. The amount of internal variables and calculations depends on the capabilities of each agent. If extra internal memory is available, several diffusion calculations can be performed and the results stored before updating the internal state variables on each agent, or more internal state variables could be used. Both uses of memory allow for estimators which have potentially better performance. We investigate this trade-off between using more state variables and using more intermediate diffusion calculations to determine which use of memory yields estimators with better performance.

We design two types of average consensus estimators. The first estimator uses feedback to track constant input signals with zero steady-state error where local memory is used to increase the convergence rate. The estimators are designed to achieve exact consensus for constant inputs, to be internally stable, and to have the steady-state value not depend on the initial conditions. The second estimator is a feedforward estimator which is designed to track time-varying signals whose frequencies are upper bounded by some known cut-off frequency. Although nonzero, the steady-state error can be made arbitrarily small using enough internal memory on each agent. The estimator achieves arbitrarily small steady-state error, estimates the average at the current iteration (as opposed to the previous iteration), and can transmit all variables in a single broadcast packet to neighboring agents at each iteration.

Diffusive average consensus estimators produce estimates which converge exponentially to the global average by performing weighted averages of variables with neighboring agents. To design estimators which converge quickly, we take the cost to be the largest time constant of the exponential decay of the estimation errors. Within each iteration, multiple diffusion calculations can take place which require time and communication. Multiple diffusion calculations can occur either simulatenously where all information is sent in a single packet, or sequentially where the result of one diffusion calculation is needed for the second, and so on. If many sequential diffusion calculations are performed at each iteration, then the communication cost is high so the estimator may converge slowly even if it takes few iterations to converge. Therefore, we normalize the cost by the number of sequential diffusion calculations per iteration. All estimators designed in this paper must perform diffusion calculations sequentially.

Much work has been done on increasing the convergence rate of average consensus estimators. For known graphs, Xiao and Boyd [5] showed that the optimal estimator with one state and one diffusion calculation per iteration can be found using semidefinite programming. Faster consensus can be achieved using a two-state estimator with one diffusion calculation at each iteration [6], [7], [8]. Other estimators have been designed which use two diffusion calculations

This work is supported in part by the Office of Naval Research. The authors are with the Department of Electrical Engineering and Computer Science (Van Scoy and Freeman) and the Department of Mechanical Engineering (Lynch), Northwestern University, Evanston, IL 60208, USA. Emails: bryanvanscoy2012@u.northwestern.edu, freeman@eecs.northwestern.edu, kmlynch@northwestern.edu.

at each iteration (although both diffusion calculations can be performed simultaneously) [9], [10]. Estimators with a single internal state and an arbitrary number of diffusion calculations at each iteration were first designed in [11] with closed-form expressions in terms of Chebyshev polynomials given in [12]. We expand these works by giving closed-form expressions for the cost of estimators which use an arbitrary number of diffusion calculations at each iteration in terms of the graph structure. We consider both one- and two-state estimators, give explicit expressions for the cost, and calculate the limiting cost as the number of diffusion calculations per iteration approaches infinity.

For the feedback design which is not robust to the initial conditions, it is shown that the cost of the one-state estimator decreases as the number of diffusion calculations increases while the cost of the two-state estimator is independent of the number of diffusion calculations. Furthermore, the limiting cost for the one-state estimator as the number of diffusion calculations approaches infinity is identical to that of the two-state estimator. Designing the estimators to be robust to initial conditions decreases the performance for any finite number of intermediate diffusion calculations, but the performance is the same as the equivalent non-robust estimator in the limit as the number of diffusion calculations approaches infinity.

The rest of the paper is organized as follows. Section II sets up the average consensus problem. Section III designs the optimal one-state and two-state feedback estimators using an arbitrary number of intermediate diffusion calculations and gives the associated cost of each estimator. The feed-forard estimators are designed in Section IV, and conclusions are given in Section V.

II. AVERAGE CONSENSUS

To setup the average consensus problem, consider a group of N agents each having a local scalar input signal $u_i(k)$ which may or may not change at each iteration k. Each agent runs an estimator which takes in its local input along with information from its neighbors to produce a local scalar output signal $y_i(k)$. The goal of each estimator is to have its output signal $y_i(k)$ track the global average of all the local inputs, $1/N \sum_{i=1}^{N} u_i(k)$. We model the communication topology as a weighted undirected graph G. Define the adjacency matrix of G to be $A = [a_{ij}] \in \mathbb{R}^{N\times N}$ where $a_{ij} = a_{ji} > 0$ if agents i and j can communicate and zero otherwise (with $a_{ii} = 0$). The neighbors of agent i, denoted $\mathcal{N}_i$, is the set of agents with which agent i can communicate. The degree of agent i, denoted $\deg(i)$, is the number of agents in $\mathcal{N}_i$. Define the $N \times 1$ vectors 1_N and 0_N of all ones and zeros, respectively. Then the Laplacian matrix is $L = \operatorname{diag}(A1_N) - A$ which is positive semidefinite and satisfies $L1_N = 0_N$. The algebraic connectivity of the graph is the second smallest eigenvalue of L, denoted λ_{min}. The graph is connected if and only if $\lambda_{\text{min}} > 0$.

If the graph topology is known, then the weights a_{ij} can be chosen to optimize system performance [13]. When the graph is unknown, however, it is often useful to choose a weighting scheme which bounds the eigenvalues of the Laplacian. For example, the decentralized weighting scheme $a_{ij} = 1/[\deg(i) + \deg(j)]$ restricts the eigenvalues of L to the interval $[0, 1]$ which allows us to use $\lambda_{\text{max}} = 1$ when the graph is unknown [14].

We now define several properties of average consensus estimators. For simplicity, we assume we have a single estimator design and that each agent runs a local copy of this estimator. We stack the local scalar inputs u_i and outputs y_i into vectors u and y, each of which can depend on the discrete time variable k.

Definition 1 (Exact): An estimator is said to achieve *exact* average consensus when for any constant input u, the output $y(k)$ converges to $\frac{1}{N} 1_N \sum_{i=1}^{N} u_i$ as $k \to \infty$.

Definition 2 (Internally stable): An estimator is said to be *internally stable* when for any initial internal states and any bounded inputs $u(\cdot)$, all internal states remain bounded in forward time.

Definition 3 (Robust to initial conditions): An estimator is said to be *robust to initial conditions* when the limit of $y(k)$ as $k \to \infty$ does not depend on the initial internal states.

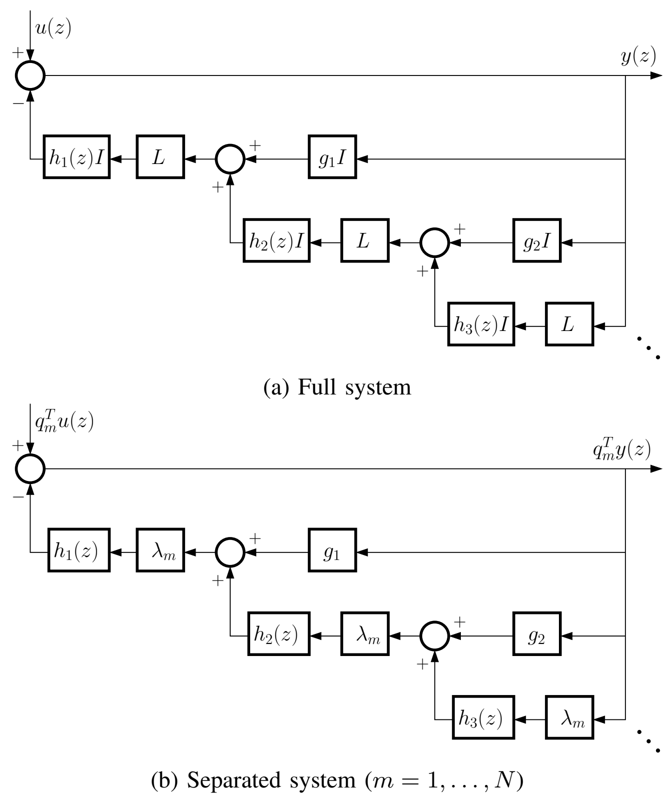


Fig. 1: Block diagram of a general feedback average consensus estimator. The diagram is shown with $r = 3$ variables communicated per iteration, but the ellipses indicate how to generalize the diagram for general r.

A useful way to characterize average consensus estimators is using their block diagram. Properties of the estimator can be easily identified based on the structure of the block diagram (see [10] for details). Specifically, for the estimator in Figure 1 we have the following properties as shown in

Figure 2:

- To be exact, the estimator must contain an integrator.
- To be internally stable, the output must pass through the Laplacian before reaching any integrator.
- To be robust to initial conditions, any integrator states must pass through the Laplacian before reaching the output.

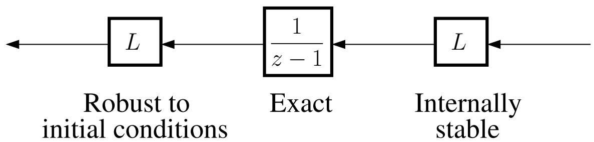


Fig. 2: Properties of a feedback estimator based on the structure of the block diagram.

In the block diagram, multiplying a signal x by L is implemented on agent i by taking a weighted average of the difference between neighbors as follows,

$$(Lx)_i = \sum_{j\in\mathcal{N}_i} a_{ij}(x_i - x_j).$$

To simplify the design, we separate the system using the eigenvalues of the Laplacian matrix. For undirected graphs, the Laplacian is symmetric and can be diagonalized as $D = Q^T L Q$ where $D = \text{diag}(\lambda_1, \dots, \lambda_N)$, $Q = \begin{bmatrix} q_1 & \dots & q_N \end{bmatrix} \in \mathbb{R}^{N\times N}$ is orthogonal, and λ_m are the eigenvalues of L. Without loss of generality, we can assume $\lambda_1 = 0$ and $q_1 = 1_N/\sqrt{N}$. To get the separated system, we multiply the input $u(z)$ by q_m^T so that the output is $q_m^T y(z)$. The output of the full system can then be recovered using

$$y(z) = Q^T Q y(z) = q_1[q_1^T y(z)] + \sum_{m=2}^{N} q_m[q_m^T y(z)] \tag{1}$$

which is the sum of the output of the separated system in the consensus direction ($m = 1$) and the disagreement directions ($m = 2, \dots, N$). This transformation diagonalizes the Laplacian blocks so that the system can be analyzed for each eigenvalue of the Laplacian separately, and then the results combined using equation (1). The resulting block diagram is shown in Figure 1b.

Note that the Laplacian can be scaled by any positive constant without loss of generality since this can be accounted for by scaling the gains of $h_i(z)$ and g_i in the block diagram in Figure 1. Therefore, we scale the Laplacian by λ_{max} so that the scaled Laplacian has maximum eigenvalue one and minimum eigenvalue $\lambda_r := \lambda_{\text{min}}/\lambda_{\text{max}}$. This is useful in presenting the results so that the estimator designs depend only on the single parameter λ_r.

III. FEEDBACK ESTIMATOR DESIGN

We want to design estimators with the fastest convergence rate normalized by the amount of communicated variables per iteration. Both the exponential convergence rate and the number of communicated variables per iteration depend only on the characteristic polynomial of the estimator. Therefore we can design the characteristic polynomial and then use the block diagram to obtain an estimator with the desired characteristic polynomial.

Let $F(z, \lambda)$ be the characteristic polynomial of the average consensus estimator using the separated system where λ is an eigenvalue of L. Let n be the order of $F(z, \lambda)$ which is the number of internal state variables on each agent. Then we want to design $F(z, \lambda)$ to minimize α such that all roots of $F(z, \lambda)$ are inside the circle of radius α centered at the origin of the complex plane for all $\lambda \in \{0\} \cup [\lambda_r, 1]$. For the estimator to be exact for constant inputs, we need the estimator to contain a model of the input in the consensus direction, meaning that $F(z, \lambda)$ has a root at $z = 1$ when $\lambda = 0$. Therefore we need $F(1, 0) = 0$.

Define α to be the worst-case asymptotic convergence factor of the estimator for all connected undirected weighted graphs whose nonzero Laplacian eigenvalues lie in the interval $[\lambda_r, 1]$. The error then decreases as α^k. If we allow for an arbitrary amount of communication during each iteration, then α can be made arbitrarily small, but the estimator may still take a long time to converge because of the large communication cost of each iteration. To take this into account, we define the normalized worst-case asymptotic convergence factor to be $\sqrt[d]{\alpha}$ where d is the number of sequential diffusion calculations per iteration. We can now state the problem to be solved.

Problem 1: Given $\lambda_r \in (0, 1]$, design an estimator which is exact, internally stable, and robust to initial conditions for all connected undirected weighted graphs whose nonzero Laplacian eigenvalues lie in the interval $[\lambda_r, 1]$. Furthermore, the estimator should have the minimum normalized worst-case asymptotic convergence factor over all such graphs.

We first consider estimators which are not robust to initial conditions, and then show how to modify the design to achieve robustness to the initial conditions.

A. Non-robust to initial conditions

1) $n = 1$: For an estimator with one state, the characteristic polynomial is

$$F(z, \lambda) = z - p(\lambda) \tag{2}$$

where $p(\lambda)$ is a polynomial in λ and the number of sequential diffusion calculations per iteration is given by the degree of $p(\lambda)$. In this case $F(z, \lambda)$ has a single root at $p(\lambda)$, so we want to choose $p(\lambda)$ to have minimum absolute value on the interval $[\lambda_r, 1]$. Chebyshev polynomials of the first-kind are known to have the minimax property meaning that they have the smallest maximum absolute value on the interval $[-1, 1]$. As shown in [12], the solution is simply a shifted and scaled Chebyshev polynomial of the first-kind. Therefore, we have

$$p(\lambda) = (-1)^d \alpha T_d \left(\frac{2}{1-\lambda_r}\lambda - \frac{1+\lambda_r}{1-\lambda_r} \right)$$

where $T_d(\cdot)$ is the d^{th} Chebyshev polynomial of the first-kind. Choosing α such that $F(1,0)=0$ gives

$$\alpha = \frac{(-1)^d}{T_d\left(-\frac{1+\lambda_r}{1-\lambda_r}\right)}.$$

The convergence factor is given by α, so the estimation error decreases as α^k. To normalize by the number of sequential diffusion calculations per iteration we take the d^{th} root of α where d is the degree of $p(\lambda)$. In the limit as $d \to \infty$, the cost of the estimator is given by the following theorem which is proved in the appendix.

Theorem 1: For the non-robust one-state estimator, the limit of the normalized worst-case convergence factor as the number of sequential diffusion calculations per iteration approaches infinity is

$$\lim_{d\to\infty} \sqrt[d]{\alpha} = \frac{1-\sqrt{\lambda_r}}{1+\sqrt{\lambda_r}}.$$

Note that for $d=1$, we have

$$\sqrt[d]{\alpha} = \frac{1-\lambda_r}{1+\lambda_r}$$

which is the same as that of the optimal estimator found in [5] and whose block diagram is shown in Figure 3a. By increasing the number of intermediate diffusion calculations stored, not only does the convergence factor α decrease, but it also decreases when normalized for the extra communication. Therefore, using more intermediate diffusion calculations increases the convergence rate for the one-state estimator.

2) $n=2$: The two-state estimator has the characteristic polynomial

$$F(z,\lambda) = z^2 + p_1(\lambda)z + p_0(\lambda). \tag{3}$$

The roots of $F(z,\lambda)$ are inside a circle of radius α if and only if

$$0 < H = \frac{\alpha^2 - p_0(\lambda)}{\alpha^2}\begin{bmatrix} \alpha^2 + p_0(\lambda) & p_1(\lambda) \\ p_1(\lambda) & \frac{\alpha^2+p_0(\lambda)}{\alpha^2}\end{bmatrix}$$

(see Lemma 1 in [9]). This condition is equivalent to the two conditions

$$|p_0(\lambda)| \le \alpha^2$$
$$|p_1(\lambda)| \le \frac{\alpha^2 + p_0(\lambda)}{\alpha}.$$

Setting $p_0(\lambda)=\alpha^2$, we need $|p_1(\lambda)| \le 2\alpha$. Once again, the optimal solution is given by a shifted and scaled Chebyshev polynomial,

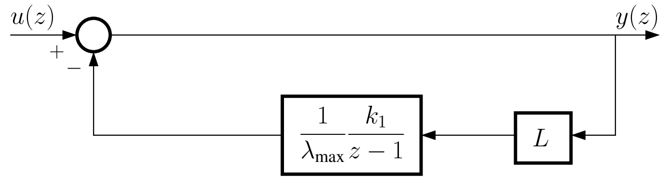
$$p_1(\lambda) = (-1)^{d+1} 2\alpha T_d\left(\frac{2}{1-\lambda_r}\lambda - \frac{1+\lambda_r}{1-\lambda_r}\right).$$

Choosing α such that $F(1,0)=0$ gives

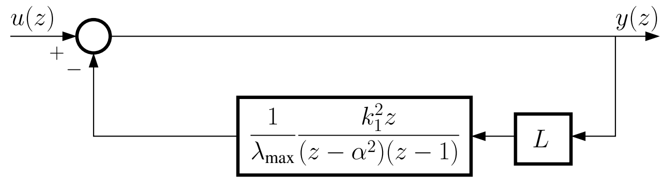
$$0 = F(1,0) = 1 + (-1)^{d+1} 2\alpha T_d(x_r) + \alpha^2$$

where $x_r = -(1+\lambda_r)/(1-\lambda_r)$. Solving for α gives

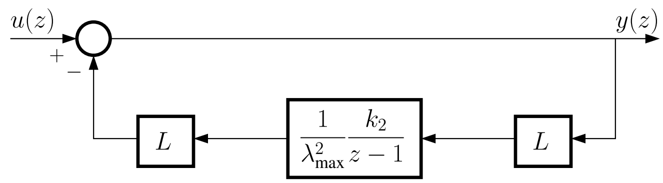
$$\alpha = (-1)^d T_d(x_r) - \sqrt{T_d^2(x_r) - 1}$$



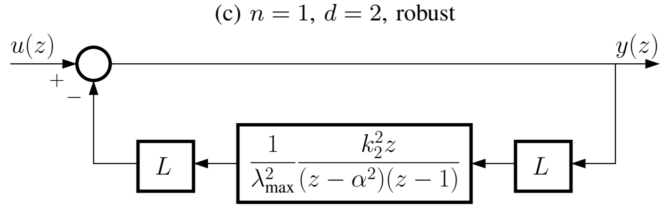
(a) $n=1$, $d=1$ non-robust



(b) $n=2$, $d=1$, non-robust



(c) $n=1$, $d=2$, robust



(d) $n=2$, $d=2$, robust

Fig. 3: Block diagrams of feedback estimators. The parameters are $k_1 = 2/(1+\lambda_r)$ and $k_2 = 2/(1+\lambda_r^2)$. Each estimator has n state variables and performs d sequential diffusion calculations per iteration. The normalized asymptotic convergence factor is given in Table I.

which can be simplified using the following theorem whose proof is given in the appendix.

Theorem 2: For the two-state estimator, the normalized worst-case convergence factor is

$$\sqrt[d]{\alpha} = \frac{1-\sqrt{\lambda_r}}{1+\sqrt{\lambda_r}}$$

for all $d \ge 1$.

Note that the performance of the two-state estimator is always better than that of the one-state estimator, and the performance of the one-state estimator approaches that of the two-state estimator as the number of sequential diffusion calculations per iteration approaches infinity for the one-state estimator.

Since it does not benefit the two-state estimator to use any additional diffusion calculations, the optimal use of memory is to use the two-state estimator with only one diffusion calculation per iteration. The block diagram for this estimator is shown in Figure 3b.

B. Robust to initial conditions

The estimators designed in the previous subsection are not robust to initial conditions. To see this, note that the integrator state in Figures 3a and 3b is observable from the output without going through the Laplacian, so the initial conditions affect the steady-state output (see [10] for details). We now show how to modify the non-robust designs to achieve robustness. The resulting estimators are sub-optimal, but their performance approaches that of the non-robust estimators in the limit as the number of sequential diffusion calculations per iteration approaches infinity. Since the performance of the robust estimators cannot be better than that of the equivalent non-robust estimators, this shows that the proposed robust estimators are optimal in the limit.

In the characteristic polynomial, we need $z-1$ to factor out of the zero-order λ term for the estimator to be exact. To be both internally stable and robust to initial conditions, $z-1$ must also factor out of the first-order λ term. This allows for a Laplacian block before and after the integrator in the block diagram as described in Section II. Therefore, we need $\frac{\partial F}{\partial \lambda}(1,0)=0$ for the estimator to be robust to initial conditions.

1) $n=1$: For a robust estimator with one state, the characteristic polynomial is still given by equation (2), except that we now have the extra robustness condition $p'(0)=0$. For low degree polynomials, we can solve for the optimal solution. For example, when $p(\lambda)$ is degree two (the lowest possible degree for robustness), we have $p(\lambda)=1+p_2\lambda^2$ with $p(\lambda_r)=\alpha$ and $p(1)=-\alpha$. Solving these equations gives

$$p_2=-\frac{2}{1+\lambda_r^2}, \qquad \alpha=\frac{1-\lambda_r^2}{1+\lambda_r^2}$$

which gives the estimator shown in Figure 3c. This design method is infeasible for high-order polynomials. The optimal solution is no longer given by Chebyshev polynomials since they do not satisfy the robustness condition. To handle the extra condition, we multiply a Chebyshev polynomial by a linear term and use the extra degrees of freedom from the linear term to satisfy the robustness condition. This allows us to analyze the performance of higher-order estimators. Assume that $p(\lambda)$ is of the form

$$p(\lambda)=a(\lambda-b)T_{d-1}\left(\frac{2}{1-\lambda_r}\lambda-\frac{1+\lambda_r}{1-\lambda_r}\right).$$

For robustness, we need $p'(0)=0$ which implies that $b=T_{d-1}(x_r)/T'_{d-1}(x_r)<0$ where $x_r=-(1+\lambda_r)/(1-\lambda_r)$. To be exact, we need $p(0)=1$ which gives $a=-T'_{d-1}(x_r)/T^2_{d-1}(x_r)$. The worst-case asymptotic convergence factor is then

$$\begin{aligned}\alpha &= |p(1)|=(-1)^{d-1}p(1)=(-1)^{d-1}a(1-b)\\ &= \frac{(-1)^{d-1}}{T_{d-1}(x_r)}\left[1-\frac{T'_{d-1}(x_r)}{T_{d-1}(x_r)}\right].\end{aligned}$$

This expression is simplified in the limit in the following theorem whose proof is provided in the appendix.

Theorem 3: For the robust one-state estimator, the limit of the normalized worst-case convergence factor as the number of sequential diffusion calculations per iteration approaches infinity is

$$\lim_{d\to\infty}\sqrt[d]{\alpha}=\frac{1-\sqrt{\lambda_r}}{1+\sqrt{\lambda_r}}.$$

2) $n=2$: The design of the two-state robust estimator is similar to that of the two-state non-robust estimator except for the additional robustness condition, $p'_1(0)+p'_0(0)=0$. Once again we choose $p_0(\lambda)=\alpha^2$ and

$$p_1(\lambda)=a(\lambda-b)T_{d-1}\left(\frac{2}{1-\lambda_r}\lambda-\frac{1+\lambda_r}{1-\lambda_r}\right).$$

Enforcing the conditions for the estimator to be exact and robust gives

$$a=(1+\alpha^2)\frac{T'_{d-1}(x_r)}{T^2_{d-1}(x_r)}, \qquad b=\frac{T_{d-1}(x_r)}{T'_{d-1}(x_r)}.$$

The asymptotic convergence factor α is then chosen so that $|p_1(1)|=2\alpha$.

Theorem 4: For the robust two-state estimator, the limit of the normalized worst-case convergence factor as the number of sequential diffusion calculations per iteration approaches infinity is

$$\lim_{d\to\infty}\sqrt[d]{\alpha}=\frac{1-\sqrt{\lambda_r}}{1+\sqrt{\lambda_r}}.$$

The results for both non-robust and robust estimators with one and two states are summarized in Table I with the normalized convergence factors plotted in Figure 4 and

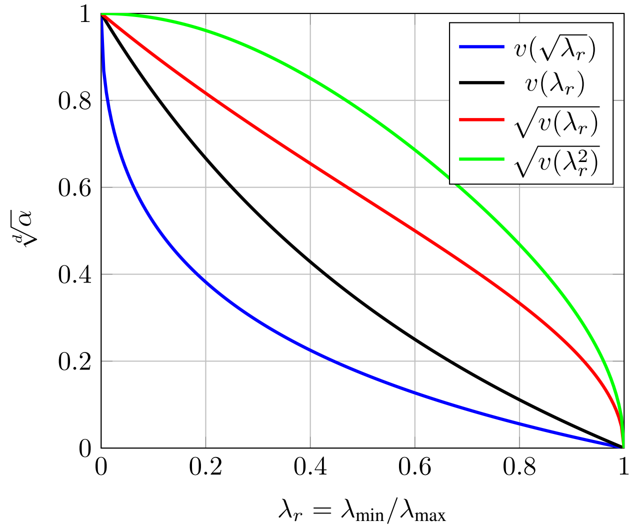


Fig. 4: Normalized worst-case asymptotic convergence factors of the estimators in Table I where $v(x)=(1-x)/(1+x)$.

TABLE I: Normalized worst-case asymptotic convergence factor $\sqrt[d]{\alpha}$ where $v(x)=(1-x)/(1+x)$.

	Non-robust $d=1$	Non-robust $d\to\infty$	Robust $d=2$	Robust $d\to\infty$
$n=1$	$v(\lambda_r)$	$v(\sqrt{\lambda_r})$	$\sqrt{v(\lambda_r^2)}$	$v(\sqrt{\lambda_r})$
$n=2$	$v(\sqrt{\lambda_r})$	$v(\sqrt{\lambda_r})$	$\sqrt{v(\lambda_r)}$	$v(\sqrt{\lambda_r})$

block diagrams shown in Figure 3. Note that the non-robust estimators achieve better performance for small values of d, but their performance is the same in the limit as $d \to \infty$.

IV. FEEDFORWARD ESTIMATOR DESIGN

The second estimator design uses memory to track time-varying signals whose frequencies lie below some cut-off frequency with small steady-state error. The model of the input signal is unknown in this case, only an upper-bound on the frequency is required. Instead of the steady-state error being zero, the error is now finite although it can be made arbitrarily small by increasing the amount of memory required to implement the estimator. We solve the following problem.

Problem 2: Given $\lambda_r \in (0,1]$ and $\theta_c \in [0,\pi)$, design an estimator which has minimum steady-state error for all connected undirected weighted graphs whose nonzero Laplacian eigenvalues lie in the interval $[\lambda_r, 1]$ and all inputs whose frequencies are upper-bounded by θ_c.

Consider the transfer function of an estimator $F(z,\lambda)$ using the separated system. We want the estimator to pass through all signals in the consensus direction ($\lambda = 0$) unchanged while attenuating all signals in the disagreement directions ($\lambda \in [\lambda_r, 1]$) for any signal with frequencies below the cut-off frequency θ_c. Therefore, we want to choose $F(z,\lambda)$ to solve

$$\min_{F} \max_{\substack{\lambda\in[\lambda_r,1]\\ \theta\in[-\theta_c,\theta_c]}} \left|F\left(e^{j\theta},\lambda\right)\right| \quad \text{s.t.} \quad F(z,0)=1. \tag{4}$$

A simple design is to choose $F(z,\lambda) = p(\lambda)$ where $p(\lambda)$ is a polynomial in λ, that is, the estimator does not have any internal states. Problem (4) is then the same polynomial minimax problem as in the case of the one-state feedback estimator design, and the solution is given by the Chebyshev polynomial

$$F(\lambda) = (-1)^d \beta T_d\left(\frac{2}{1-\lambda_r}\lambda - \frac{1+\lambda_r}{1-\lambda_r}\right)$$

where

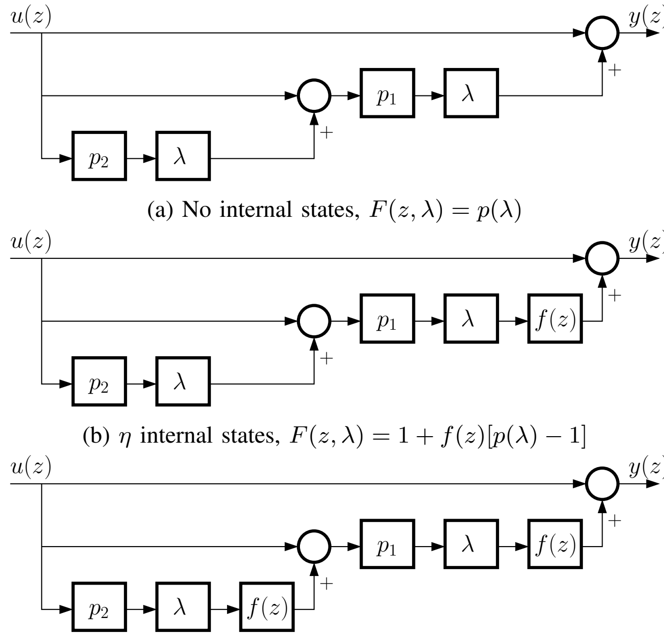
$$\beta = \max_{\substack{\lambda\in[\lambda_r,1]\\ \theta\in[-\theta_c,\theta_c]}} |F(\lambda)| = \frac{(-1)^d}{T_d\left(-\frac{1+\lambda_r}{1-\lambda_r}\right)} \tag{5}$$

is the level of attenuation and d is the degree of the estimator. The block diagram for such an estimator is shown in Figure 5a where

$$p(\lambda) = 1 + p_1\lambda\Big(1 + p_2\lambda\big[1 + p_3\lambda(\ldots)\big]\Big).$$

A recursive version of this estimator is given in [12].

This simple design has a drawback, however. The output is a d^{th} degree polynomial in λ, so all d diffusion calculations must be performed sequentially before the output can be calculated. Since the output is not calculated until the iteration is complete, the estimate is for the *previous* iteration. While this may be allowable in certain applications, we would like the estimate to be a function of only the current internal state variables (and not the Laplacian) so that the estimate



(a) No internal states, $F(z,\lambda) = p(\lambda)$

(b) η internal states, $F(z,\lambda) = 1 + f(z)[p(\lambda) - 1]$

(c) $d\eta$ internal states, $F(z,\lambda) = p(\lambda f(z))$

Fig. 5: Block diagrams of feedforward estimators.

is for the *current* iteration. To separate the output from the Laplacian, we introduce a strictly proper transfer function between the Laplacian blocks and the output in the block diagram as shown in Figure 5b. The output is then only a function of the internal states on each agent and the local input. Ideally, we would like $f(e^{j\theta}) = 1$ for all $\theta \in [-\theta_c, \theta_c]$ so that we recover the performance of the stateless estimator. This cannot be achieved exactly since $f(z)$ must be strictly proper, so we design $1 - f(z)$ to be a highpass filter with cut-off frequency θ_c. Then $f(e^{j\theta}) \approx 1$ for all $\theta \in [-\theta_c, \theta_c]$. The attenuation of the estimator can then be increased both by using a higher-degree polynomial $p(\lambda)$ and by using a higher-order filter $f(z)$.

The estimators in Figures 5a and 5b both require d diffusion calculations per iteration which must be done sequentially (the result of the first calculation is needed for the second, and so on). Instead, we would like to perform all d diffusion calculations simultaneously so that each agent can send all of the required information in a single broadcast packet. To do this, we need a strictly proper transfer function between each Laplacian block in Figure 5b. Inserting the same filter $f(z)$ after *each* Laplacian block (not just the last one) produces the estimator shown in Figure 5c. To implement the estimator, each agent now has $d\eta$ internal state variables where η is the order of $f(z)$. The transfer function of the estimator is given by $p(\lambda f(z))$. As the order of the filter increases, $f(e^{j\theta})$ approaches unity for $\theta \in [-\theta_c, \theta_c]$ so that

$$\lim_{\eta\to\infty} \max_{\substack{\lambda\in[\lambda_r,1]\\ \theta\in[-\theta_c,\theta_c]}} \left|F\left(\lambda f(e^{j\theta})\right)\right| = \beta$$

where β is given by equation (5).

Therefore, the estimator in Figure 5c

- achieves the same performance as the stateless estimator in Figure 5a in the limit as the order of $f(z)$ approaches infinity,
- estimates the average of the current iteration, and
- can transmit all transmission variables in a single packet at each iteration.

V. CONCLUSIONS

We have designed two types of average consensus estimators. The first design tracks constant input signals with zero steady-state error. It is shown that using two internal state variables and one diffusion calculation per iteration achieves the optimal asymptotic convergence rate when normalized for the number of sequential diffusion calculations per iteration. The second design tracks frequency-bounded input signals with small steady-state error. The steady-state error can be made arbitrarily small by adding more internal state variables on each agent, the estimate is for the current average of the inputs, and all transmission variables can be sent in a single packet to neighboring agents at each iteration. Furthermore, the two estimator designs have the ability to be combined in series to obtain the benefits of both estimators.

APPENDIX

We now provide proofs to the theorems involving the normalized convergence factors. To prove Theorem 1, we need the following lemma.

Lemma 1: For $x \geq 0$, $\lim_{d\to\infty} \sqrt[d]{\cosh(dx)} = e^x$.

Proof: Since $x \geq 0$, we can upper bound $\cosh(dx)$ by

$$\cosh(dx) = \frac{e^{dx} + e^{-dx}}{2} \leq \frac{e^{dx} + e^{dx}}{2} = e^{dx}.$$

Since $e^{-dx} > 0$, a lower bound is

$$\frac{1}{2} e^{dx} \leq \cosh(dx).$$

Using the squeeze theorem on the inequalities

$$\frac{1}{\sqrt[d]{2}} e^x \leq \sqrt[d]{\cosh(dx)} \leq e^x$$

gives that $\lim_{d\to\infty} \sqrt[d]{\cosh(dx)} = e^x$. ■

We now give the proof for Theorem 1.

Proof: [Theorem 1] Define $x_r = -(1+\lambda_r)/(1-\lambda_r)$. Using the fact that $T_d(x) = (-1)^d \cosh(d\cosh^{-1}(x))$ for $x \leq -1$,

$$\lim_{d\to\infty} \sqrt[d]{\alpha} = \frac{1}{\lim_{d\to\infty} \sqrt[d]{(-1)^d T_d(x_r)}} = \frac{1}{\lim_{d\to\infty} \sqrt[d]{\cosh\left(d\cosh^{-1}(-x_r)\right)}}.$$

Note that $x_r \leq -1$ since $0 < \lambda_r < 1$, so $\cosh^{-1}(-x_r) \geq 0$. Then we can apply Lemma 1 to obtain

$$\lim_{d\to\infty} \sqrt[d]{\alpha} = \frac{1}{e^{\cosh^{-1}(-x_r)}}.$$

Using $\cosh^{-1}(x) = \ln[x + \sqrt{x^2-1}]$ and simplifying gives

$$\lim_{d\to\infty} \sqrt[d]{\alpha} = \frac{1}{e^{\ln[-x_r + \sqrt{x_r^2 - 1}]}} = \frac{1}{-x_r + \sqrt{x_r^2-1}} = \frac{1-\sqrt{\lambda_r}}{1+\sqrt{\lambda_r}}$$

which completes the proof. ■

The proof of Theorem 2 is given below.

Proof: [Theorem 2] Similar to the proof of Theorem 1, define $x_r = -(1+\lambda_r)/(1-\lambda_r)$ and use the fact that $T_d(x) = (-1)^d \cosh(d\cosh^{-1}(x))$ for $x \leq -1$ to write

$$\alpha = (-1)^d T_d(x_r) - \sqrt{T_d^2(x_r) - 1} = \cosh(d\cosh^{-1}(-x_r)) - \sqrt{\cosh^2(d\cosh^{-1}(-x_r)) - 1}.$$

Using the common identities $\cosh^2(x) - 1 = \sinh^2(x)$ and $\cosh(x) - \sinh(x) = e^{-x}$ gives

$$\alpha = \cosh(d\cosh^{-1}(-x_r)) - \sinh(d\cosh^{-1}(-x_r)) = e^{-d\cosh^{-1}(-x_r)}.$$

Once again using $\cosh^{-1}(x) = \ln[x + \sqrt{x^2-1}]$ and simplifying gives

$$\alpha = e^{-d\ln[-x_r + \sqrt{x_r^2-1}]} = e^{\ln[(-x_r+\sqrt{x_r^2-1}]^{-d}} = \frac{1}{(-x_r + \sqrt{x_r^2-1})^d} = \left(\frac{1-\sqrt{\lambda_r}}{1+\sqrt{\lambda_r}}\right)^d$$

which completes the proof. ■

The following Lemma is needed for the proof of Theorem 3.

Lemma 2: For $|x| > 1$,

$$\lim_{d\to\infty} \sqrt[d]{1 - \frac{T_d'(x)}{T_d(x)}} = 1. \tag{6}$$

Proof: Let $c = \sqrt{x^2-1}$ and $a = x + c$. Then $1/a = x - c$ and $|a| < 1$. Then we can write $T_d(x)$ and $T_d'(x)$ as

$$T_d(x) = \frac{a^d + a^{-d}}{2}, \qquad T_d'(x) = d\frac{a^d - a^{-d}}{2c}.$$

Taking the logarithm of the left-hand side of (6), we have

$$\lim_{d\to\infty} \log \sqrt[d]{1 - \frac{T_d'(x)}{T_d(x)}} = \lim_{d\to\infty} \frac{\log\left[1 - \frac{T_d'(x)}{T_d(x)}\right]}{d} = \lim_{d\to\infty} \frac{\log\left[1 - \frac{d}{c}\frac{a^d - a^{-d}}{a^d + a^{-d}}\right]}{d}.$$

Using L'Hôpital's rule, this gives

$$\lim_{d\to\infty} \frac{\frac{1 - a^{4d} - 4da^{2d}\log(a)}{c(1+a^{2d})^2}}{1 - \frac{d}{c}\frac{a^d - a^{-d}}{a^d + a^{-d}}} = \lim_{d\to\infty} \frac{1/c}{-d/c} = 0.$$

Since $\log(0) = 1$ and the $\log(\cdot)$ function is continuous, this gives the result. ■

Proof: [Theorem 3] Splitting the limit to be evaluated, we have

$$\lim_{d\to\infty} \sqrt[d]{\alpha} = \lim_{d\to\infty} \sqrt[d]{\frac{(-1)^{d-1}}{T_{d-1}(x_r)}\left[1-\frac{T'_{d-1}(x_r)}{T_{d-1}(x_r)}\right]}$$

$$= \lim_{d\to\infty} \left(\sqrt[d-1]{\frac{(-1)^{d-1}}{T_{d-1}(x_r)}}\right)^{\frac{d-1}{d}} \left(\sqrt[d-1]{\left[1-\frac{T'_{d-1}(x_r)}{T_{d-1}(x_r)}\right]}\right)^{\frac{d-1}{d}}$$

where Theorem 1 provides the solution to the limit of the first expression while Lemma 2 gives that the limit of the second expression is equal to one, so

$$\lim_{d\to\infty} \sqrt[d]{\alpha} = \frac{1-\sqrt{\lambda_r}}{1+\sqrt{\lambda_r}}.$$

■

Proof: [Theorem 4] To calculate the asymptotic convergence of the robust two-state estimator, we compare the performance to that of the robust one-state estimator. Let α_1 and α_2 be the asymptotic convergence factor of the robust one-state and two-state estimators, respectively. The one-state estimator requires a polynomial which satisfies $p(0) = 1$ and $p(\lambda) \le \alpha$ for all $\lambda \in [\lambda_r, 1]$. For the two-state estimator, we need a polynomial which satisfies $p(0) = 1 + \alpha^2$ and $p(\lambda) \le 2\alpha$ for all $\lambda \in [\lambda_r, 1]$. Therefore, we have $\alpha_2 = 2\alpha_1/(1+\alpha_1^2)$. In the limit, this gives

$$\lim_{d\to\infty} \sqrt[d]{\alpha_2} = \lim_{d\to\infty} \sqrt[d]{\frac{2}{1+\alpha_1^2}} \lim_{d\to\infty} \sqrt[d]{\alpha_1} = \lim_{d\to\infty} \sqrt[d]{\alpha_1}.$$

Therefore, the normalized asymptotic convergence factor of the robust two-state estimator is the same as that of the robust one-state estimator as the number of diffusion calculations per iteration approaches infinity. ■

References

[1] P. Yang, R. Freeman, and K. Lynch, "Multi-agent coordination by decentralized estimation and control," *IEEE Trans. Autom. Control*, vol. 53, no. 11, pp. 2480–2496, 2008.

[2] C. K. Peterson and D. A. Paley, "Distributed estimation for motion coordination in an unknown spatially varying flowfield," *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 3, pp. 894–898, 2013.

[3] J. Cortes, "Distributed kriged kalman filter for spatial estimation," *IEEE Trans. Autom. Control*, vol. 54, no. 12, pp. 2816–2827, 2009.

[4] R. Aragues, J. Cortes, and C. Sagues, "Distributed consensus on robot networks for dynamically merging feature-based maps," *IEEE Trans. Robot.*, vol. 28, no. 4, pp. 840–854, 2012.

[5] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," in *Proc. of the 42nd IEEE Conf. on Decision and Control*, vol. 5, Dec 2003, pp. 4997–5002.

[6] T. Erseghe, D. Zennaro, E. Dall'Anese, and L. Vangelista, "Fast consensus by the alternating direction multipliers method," *IEEE Trans. Signal Process.*, vol. 59, no. 11, pp. 5523–5537, Nov. 2011.

[7] B. N. Oreshkin, M. J. Coates, and M. G. Rabbat, "Optimization and analysis of distributed averaging with short node memory," *IEEE Trans. Signal Process.*, vol. 58, no. 5, pp. 2850–2865, May 2010.

[8] J. Liu and A. S. Morse, "Accelerated linear iterations for distributed averaging," *Annual Reviews in Control*, vol. 35, no. 2, pp. 160 – 165, 2011.

[9] B. Van Scoy, R. A. Freeman, and K. M. Lynch, "Optimal worst-case dynamic average consensus," in *Proc. of the 2015 Amer. Control Conf.*, July 2015, pp. 5324–5329.

[10] ——, "Design of robust dynamic average consensus estimators," in *Proc. of the 54th IEEE Conf. on Decision and Control*, Dec 2015.

[11] E. Kokiopoulou and P. Frossard, "Polynomial filtering for fast convergence in distributed consensus," *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 342–354, Jan 2009.

[12] E. Montijano, J. Montijano, and C. Sagues, "Chebyshev polynomials in distributed consensus applications," *IEEE Trans. Signal Process.*, vol. 61, no. 3, pp. 693–706, Feb 2013.

[13] P. Yang, R. Freeman, and K. Lynch, "Optimal information propagation in sensor networks," in *Proc. of the 2006 IEEE Int. Conf. on Robotics and Automation*, 2006, pp. 3122–3127.

[14] R. Freeman, T. Nelson, and K. Lynch, "A complete characterization of a class of robust linear average consensus protocols," in *Proc. of the 2010 Amer. Control Conf.*, 2010, pp. 3198–3203.