Topics on the average consensus problems

The research field of networked systems has burgeoned during the last years attracting the attention of outstanding mathematicians, computer scientists and electrical engineers. This is due in no small part to the recent technological advances in communication and computation and to the miniaturization of electronic components which have made large groups of embedded systems, such as sensors and robotic networks, a reality. Such systems are expected to pervade our technological life during the next years performing several complex tasks, such as search and rescue, health case, environmental monitoring and surveillance. The potential advantages of employing networks of agents are numerous. For instance, certain tasks are difficult, if not impossible, when performed by a single agent. Further, a group of agents inherently provides robustness to failures of single agents or communication links.

The existence of such networks raise compelling questions: Can large numbers of such autonomous agents be successfully deployed as a team to cooperatively carry out a prescribed task and to respond as a group to high-level management commands? Can such a group reliably function robustly and adaptively, without a centralized controller, with limited communications between its members? These scenarios motivate the study of algorithms for the autonomous adaptation and coordination of large-scale network systems. So is born the emerging discipline of multi-agent systems.

This thesis is related to this context. Its focus is on the *consensus problem*, which in few words, can be described as the problem of making the individuals agents of a network to reach an agreement on key pieces of information or on a common decision (represented by scalar or vector values) that enable them to cooperate in a coordinate fashion.

This problem has been massively analyzed in the last years. Indeed, it can be viewed as an intermediate but fundamental step toward a comprehensive understanding of how to assemble the members of a large-scale system into a coherent whole. Moreover consensus algorithms are starting to be applied in a wide-range of applications. In **Multi-Vehicle cooperative control**, large numbers of autonomous vehicles (air, ground and water) are envisioned to have a significant impact in numerous civilian and homeland security applications (e.g., monitoring forest fires, oil fields, pipelines, tracking wildlife, patrolling borders, monitoring the perimeter of nuclear power plants, surveillance, reconnaissance). To enable these applications various cooperative control capabilities need to be developed, including formation control, rendezvous, attitude alignment, flocking and cooperative search. Lately, consensus algorithms have shown their effectiveness in reaching these capabilities. In **load balancing** consensus algorithms are used to balance as much as possible the tasks among the various processors, computers in order to speed up the whole computation. Then, for several applications of a wireless sensor networks, such as mobile target tracking, event detection, efficient TDMA scheduling, and sleep scheduling with very low duty cycle, it is essential that the nodes act in a coordinated and synchronized fashion. All these applications required global **clock synchronization**, that is, all the nodes of the network need to refer to a common notion of time. Recently novel consensus-based protocol, for synchronizing a wireless has been elaborated. During the last years **distributed estimation** problems are attracting a large interest. It is possible to see that the solution of some linear inference problems, as for instance the computation of the best linear unbiased estimator, can be reduced to the applications of two or more consensus algorithms. Consensus algorithms have shown their effectiveness also in an other important distributed inference application: tracking of noisy signals. Moreover interesting consensus ideas have been applied, recently, also in more general **distributed computation** (in particular distributed optimization) in **computer vision** and in modeling information and decisions dynamics in **social networks** (learning, opinion dynamics, rules of thumb). The rest of the present document is organized in the following way. First we briefly review the standard consensus algorithm we consider; then we summarize the main results obtained in each chapter.

A standard average consensus algorithm

A natural and widely studied consensus algorithm involves, at each time step, every system taking a weighted average of its own value with values received from some of the other systems. To be more precise, we consider a set V of N nodes (or agents), numbered 1 through N. Each node i starts with a scalar value $x_i(0)$. At each nonnegative integer time t, node i receives from some of the other nodes j a message with the value of $x_j(t)$, and updates its value according to:

$$x_i(t+1) = \sum_{j=1}^{N} P_{ij}(t) x_j(t) \tag{1}$$

where the $P_{ij}(t)$ are weights with the property that $P_{ij}(t) \neq 0$ only if node i receives information from node j at time t. We use the notation $P(t)$ to denote the *weighted matrix* $\{P_{ij}(t)\}_{i,j=1,\dots,N}$. Given a matrix P, let $\mathcal{E}(P)$ denote the set of directed edges (j,i), including self-edges (i,i), such that $P_{ij} \neq 0$. At each time t, the nodes' connectivity can be represented by the directed graph $\mathcal{G}(t) = (V, \mathcal{E}(P(t)))$. The goal is to study the convergence of the iterates $x_i(t)$ to some

common value $\bar{x}$, as t approaches infinity. In the case of the *average consensus problem* this value $\bar{x}$ is the average of the initial values, $\bar{x} = x_{ave} := \frac{1}{N}\sum_{i=1}^{N} x_i(0)$. In this thesis we focus particularly on the *average consensus* algorithm.

The main results obtained in the thesis are illustrated in Chapter 3, 4, 5, 6, 7 and now we summarize them.

Chapter 3: *The symmetries in the consensus problem*

This chapter aims to provide an answer to the following natural question:" How much does the amount of information exchanged by the systems in the consensus algorithm influence the rate of convergence towards the asymptotic agreement?" Intuitively one should expect that the larger is the communication effort between the systems, the better are the performance achievable by the consensus algorithms. The main result of the chapter is a mathematical characterization of this fact for a particular class of graphs and matrices exhibiting symmetries: the Cayley graphs. It is worth noting that the circulant graphs and all d-dimensional grids on toruses are Cayley graphs. In this chapter, by modelling the communication networks as Cayley graphs defined on Abelian groups we were able to extend the available bounds on the mixing time of Markov chains defined on such groups.

In more formal terms, we have analyzed the time-invariant case of Equation (1), i.e.,

$$x(t+1) = Px(t), \tag{2}$$

where P is assumed to be a Cayley matrix. The Cayley matrices can be described as follows. Let G be any finite Abelian group of order $|G| = N$. A matrix $P \in \mathbb{R}^{G\times G}$ is said to be a Cayley matrix over the group G if

$$P_{i,j} = P_{i+h,j+h} \qquad \forall\ i,j,h \in G\,.^{1}$$

Moreover we assume that the matrix P is nonnegative. It turns out from the considerations made in Chapter 2 of the thesis, that P is a doubly stochastic matrix. In Chapter 2 it is shown that, in the context of the average consensus algorithms, a meaningful way to quantify the speed of convergence toward the consensus is represented by the *essential spectral radius*, denoted by $\rho_{ess}(P)$, which represents the second largest eigenvalue in absolute value of the matrix P. Let now $\mathcal{G}_P$ be the communication graph associated to the matrix P as described in the previous section; $\mathcal{G}_P$ is a Cayley graph and it is easy to see that, in this case, all the nodes have the same number ν of arcs incoming. In this context ν can be viewed as a proper measure of the connectivity of the Cayley graph $\mathcal{G}_P$. The main result of Chapter 3 establishes a lower bound on $\rho_{ess}(P)$ as a function of ν; precisely, if P is any N-dimensional Cayley matrix with ν denoting the number of incoming arcs in each node of $\mathcal{G}_P$, then

$$\rho_{ess}(P) \geq 1 - CN^{-2/\nu}$$

where C is a constant independent from P. The interpretation of the previous result is the following. Consider a family of Cayley matrices $\{P_N\}$ of increasing size, whose associated communication graphs $\mathcal{G}_{P_N}$ have a bounded in-degree; then

$$\lim_{N\to\infty} \rho_{ess}(P_N) = 1.$$

Hence if we impose symmetries in the communication network and we keep the number of incoming arcs in each vertex bounded, then the convergence rate degrades as the number of agents increases; in other words the speed of convergence in such graphs could be very slow in the case of large N.

It is worth saying that the interest for graphs with symmetries is motivated by two facts. First the use in applications with large number of agents (particularly in computer science), where imposing symmetries in the communication graph allows a much more compact representation. Second the conjectured similarity of the spectral properties of Cayley matrices with the spectral properties of the diffusion matrices built over an another important family of graphs: the random geometric graphs. The random geometric graphs have recently shown their effectiveness in modeling wireless communication in sensors network applications but a precise analysis of their spectral properties is very challenging and still a open problem.

Chapter 4: *Randomized consensus algorithms*

Roughly speaking a randomized algorithm works as follows. Assume we have a family $\mathcal{P}$ of stochastic matrices. Then, at each iteration of the consensus algorithm, the consensus matrix $P(t)$ is chosen randomly inside this set of matrices, according to a certain distribution probability. For a randomized algorithm, convergence is considered in a probabilistic

[1] Analogous definition of Cayley symmetries can be provided also for graphs.

sense and performance are studied in mean square sense. The use of the *random algorithms* is quite appealing, since they take into consideration some fundamental limitations arising in the realistic implementation of the consensus algorithms. Indeed, it should be pointed out that, in many practical applications, a node can not simultaneously receive data from two different neighbor nodes (for instance collision can delete messages in wireless environment) and in some applications it cannot simultaneously transmit to more than a node (this happens for instance for processors nets).

We have considered three interesting examples of randomized consensus algorithms. By means of the first two, the *time-varying strategy with bounded in-degree* and the *time-varying Cayley graphs strategy*, we have emphasized a remarkable feature of the randomized consensus algorithms, i.e., that they allow to achieve better performance than the deterministic ones with comparable complexity. This comparison has been done by considering the consensus algorithms having as communication graphs the Abelian Cayley graphs: in the randomized case the speed of convergence is independent from the size of the network!

The last example has reviewed well-studied average consensus algorithms: the symmetric gossip algorithms. In these algorithms, at each step, a node communicates with a randomly chosen neighbor. To be more precise assume we are given an undirected graph $\mathcal{G} = (V, \mathcal{E})$, $\mathcal{E} \subset \{(i,j) : i,j \in V\}$. At each time step, one edge (i,j) is randomly selected in $\mathcal{E}$ with a certain probability; the two agents connected by that edge average their states according to

$$x_i(t+1) = x_j(t+1) = \frac{1}{2}x_i(t) + \frac{1}{2}x_j(t)$$

while $x_h(t+1) = x_h(t)$, if $h \neq i, j$. The gossip algorithms are mainly motivated by applications to sensors, peer-to-peer and *ad hoc* networks. While the convergence properties of this kind of algorithms have been recently analyzed in a comprehensive way, the evaluation of the performance is still an open problem. In our research we have provided interesting results in this direction for the symmetric gossip algorithms working over Cayley graphs.

Quantized Consensus: Chapter 5, 6, 7

In the application of distributed algorithms over wireless sensor networks, power and bandwith limitations naturally lead to consider finite capacity digital transmission channels among the various sensors. As a consequence, this forces quantization at the level of the transmitted values. In this part of our research we have considered the more realistic and practical situation in which the communication network between the systems is constituted of only rate-constrained digital links. This, in general, prevents the nodes from having a precise knowledge about the state of the other nodes. Indeed, through digital channels the nodes can exchange only symbolic data in a finite alphabet and using this information they can build at most an estimate of their neighbors. We have assumed that the nodes quantize their information before transmitting it. In particular we have introduced two quantizers, well-studied in the literature: the *deterministic* uniform quantizer and the *probabilistic* uniform quantizer. The *deterministic quantizer* is defined as follows. Let $q_d \colon \mathbb{R} \to \mathbb{Z}$ be the mapping sending $z \in \mathbb{R}$ into its nearest integer, namely,

$$q_d(z) = n \in \mathbb{Z} \quad \Leftrightarrow \quad \begin{array}{ll} z & \in [n-1/2, n+1/2[, \text{ if } z \geq 0 \\ z & \in]n-1/2, n+1/2], \text{ if } z < 0. \end{array} \tag{3}$$

The *probabilistic quantizer* $q_p \colon \mathbb{R} \to \mathbb{Z}$ is defined as follows. For any $x \in \mathbb{R}$, $q_p(x)$ is a random variable on $\mathbb{Z}$ defined by

$$q_p(x) = \begin{cases} \lfloor x \rfloor & \text{with probability} \quad \lceil x \rceil - x \\ \lceil x \rceil & \text{with probability} \quad x - \lfloor x \rfloor. \end{cases} \tag{4}$$

Through both analytical results and simulations, we investigate two questions: (i) Is it still possible to reach the average consensus? and (ii) If not, how far is it the asymptotic behavior of the states of the systems from the average consensus?

Chapter 5: *Quantized Consensus: Time-Invariant case*

Here we have focused on the time-invariant case. It is worth saying that this chapter has been motivated by a previous work whose authors, in order to overcome the effects due to the quantization, analyzed the following strategy (we refer to it as *globally quantized* strategy)

$$x(t+1) = Pq_p(x(t)) \ ^2$$

[2]With a slight abuse of notation $q_p(x(t)) = [q_p(x_1(t)), \ldots, q_p(x_N(t))]^T$

It is proved that this strategy converges to a consensus, which, in general, differs from the average of the initial conditions. However the authors provided bounds quantifying this displacement, but these bounds suffer from a big limitation: they depend strictly on the matrix P and in many cases they tend to infinity as the size of the network increases [3].

The main contribution of this chapter is the introduction of a simple and effective adaptation of the standard average consensus algorithm which does not converge to an asymptotic agreement, but is able to preserve the average of states and to drive the systems reasonably near to the consensus. The crucial assumption is that the each agent has access to exact information regarding its own state. Inspired by the observation that the law $x(t+1) = Px(t)$ can be rewritten as $x(t+1) = x(t) + (P-I)x(t)$ we have proposed the following law $x(t+1) = x(t) + (P-I)q_d(x(t))$; observe that each system i in computing $x_i(t+1)$ uses both quantized and exact information regarding its own state at time t. In the thesis we refer to this strategy as *partially quantized* strategy . The main feature of this strategy is that it preserves the average of the initial conditions at each iteration of the algorithm. But, is it converging to the average consensus? If not, how big is it the disagreement? By simulations it is possible to see that in general the *partially quantized strategy* does not converge. To evaluate the asymptotic disagreement we have introduced the quantity

$$d_\infty(P, x(0)) = \limsup_{t\to\infty} \frac{1}{\sqrt{N}} ||x(t) - x_{ave}\mathbf{1}|| \text{ [4]}. \tag{5}$$

We have run extensive simulations for many different family of graphs extrapolating the following numerical evidence: $d_\infty(P, x(0))$ is typically smaller than 1 and, remarkably, it is independent from the size of the matrix P. In other words the states are driven very close to the average consensus. We have undertaken two analysis: a worst-case analysis and a probabilistic analysis. The worst-case analysis produces bounds which are very conservative and that depend on the matrix P. In particular they grow logarithmically and, in some cases, polynomially on the size on the network, behavior which is in disagreement with the experimental evidences. The fact that $d_\infty(P, x(0))$ is, in general, smaller than 1 is, instead, captured by the probabilistic analysis that consists in modeling the quantization error as additive random noise. A classical mean squared analysis can be carried on in this case. The drawback is that the probabilistic analysis, in principle, does not offer any rigorous bound. Indeed there is no theoretical reason explaining why a white noise should be a suitable model for analyzing the deterministic quantization. However, we have provided an interesting result: showing that the probabilistic model, due to the structure of the probabilistic quantizer, represents the correct way to analyze $x(t+1) = x(t) + (P-I)q_p(x(t))$, i.e., the case in which the information transmitted is quantized through probabilistic quantizers. In this setup the probabilistic bounds provided analyzing the deterministic quantization as a white noise and describing closely the experimental evidences, are also theoretical sound.

Chapter 6: *Quantized Consensus: Gossip Algorithms*

In this context we have introduced two particular strategies, the *partially quantized* strategy and the *globally quantized* strategy, depending, as in the previous chapter, whether the systems use exact information regarding their own state or not to update their states. We have analyzed these strategies both via the deterministic quantizer and via the probabilistic quantizer. To describe them, let us assume that (i, j) be the edge selected at the t-th iteration. In the *globally quantized* strategy, i and j, in order to update their states, use only the estimates of their states, as follows,

$$x_i(t+1) = \frac{1}{2}\hat{x}_i(t) + \frac{1}{2}\hat{x}_j(t) \qquad x_j(t+1) = \frac{1}{2}\hat{x}_j(t) + \frac{1}{2}\hat{x}_i(t).$$

The above update rules does not preserve the average of states, which can be a significant drawback in some applications. To cope with this problem, we propose the *partially quantized* strategy, in which agents use both their real-valued state and their quantized values

$$x_i(t+1) = x_i(t) - \frac{1}{2}\hat{x}_i(t) + \frac{1}{2}\hat{x}_j(t) \qquad x_j(t+1) = x_j(t) - \frac{1}{2}\hat{x}_j(t) + \frac{1}{2}\hat{x}_i(t),$$

The *globally quantized* strategy has the positive feature that drives the systems to a consensus, but does not maintain the average of the initial conditions. On the other hand, the *partially quantized* strategy maintains the average of the initial conditions, but not converge to the consensus, even though the agents are driven closed to each other (we quantify how much). The results obtain are summarized in the following table.

The convergence results are all proved theoretically. The evaluation, in the *globally quantized* strategy, of the disagreement between the consensus point and the average of the initial conditions is performed numerically. In relation to this point, it is interesting to notice how the randomness introduced by the probabilistic quantizers allows to reach much better performance than the use of deterministic quantizers.

[3] Similar considerations hold also for the strategy $x(t+1) = Pq_d(x(t))$, that, however does not lead the agents to a consensus.

[4] $\mathbf{1} = [1, \ldots, 1]^T$

	Globally Quant.	Partially Quant.
q_d	(i) *Finite time convergence* to consensus (ii) *Large displacement* between the consensus point and the average of the initial conditions (increasing with the size of the network)	(i) *Finite time convergence* to $\|x - x_{\text{ave}}\mathbf{1}\|_\infty \leq 1$ (ii) Average preserved
q_p	(i) *Finite time convergence* to consensus (ii) *Small displacement* between the consensus point and the average of the initial conditions (bounded with the size of the network)	(i) *Asymptotic convergence* to $\frac{1}{N}\sqrt{\mathbb{E}[\|x - x_{\text{ave}}\mathbf{1}\|_2^2]} \leq \frac{1}{2}$ (ii) Average preserved

Chapter 7: *Quantized average consensus via dynamic coding/decoding schemes*

In the previous two chapters, in order to face with the effects due to the forced quantization, we have elaborated strategies which either preserves the average of the state but do not converge to a consensus or do not preserve the average but converge to a consensus which, in general, does not coincide with the initial average. In this context we have introduced a novel quantized strategy that permits both to maintain the initial average and to reach it asymptotically. The main idea is to adapt coding/decoding strategies, that were proposed for centralized control and communication problems, to the distributed consensus problem. In this chapter, differently form the previous two, we assume that the coder/ decoder schemes are endowed with memory and hence that, in some sense, they have their own internal dynamics. For the sake of clarity, we now briefly describe the coder/decoder scheme adopted.

Suppose a source wants to communicate to a receiver some time-varying data $x : \mathbb{N} \to \mathbb{R}$ via repeated transmissions at time instants in $\mathbb{N}$. Each transmission takes place through a digital channel, i.e., messages can only be symbols in a finite or countable set (to be designed). The channel is assumed to be reliable, that is, each transmitted symbol is received without error. A coder/decoder pair for a digital channel is defined by the sets:

(i) a set Ξ, serving as *state space* for the coder/decoder; a fixed $\xi_0 \in \Xi$ is the *initial coder/decoder state*;

(ii) a finite or countable set $\mathcal{A}$, serving as *transmission alphabet*; elements $\alpha \in \mathcal{A}$ are called message;

and by the maps

(i) a map $F : \Xi \times \mathcal{A} \to \Xi$, called the *coder/decoder dynamics*;

(ii) a map $Q : \Xi \times \mathbb{R} \to \mathcal{A}$, being the *quantizer function*;

(iii) a map $H : \Xi \times \mathcal{A} \to \mathbb{R}$, called the *decoder function*.

The coder computes the symbols to be transmitted according to, for $t \in \mathbb{N}$,

$$\xi(t+1) = F(\xi(t), \alpha(t)), \quad \alpha(t) = Q(\xi(t), x(t)).$$

Correspondingly, the decoder implements, for $t \in \mathbb{N}$,

$$\xi(t+1) = F(\xi(t), \alpha(t)), \quad \hat{x}(t) = H(\xi(t), \alpha(t)).$$

Coder and decoder are jointly initialized at $\xi(0) = \xi_0$.

We have designed two interesting coder/decoder pairs: the logarithmic quantizer strategy and the zoom in - zoom out uniform quantizer strategy. We have analyzed theoretically the former and via numerical simulations the latter. The main result is that both the strategies converge asymptotically to the average of the initial conditions. Moreover we have shown that the convergence factors depend smoothly on the accuracy parameter of the quantizers used and that, remarkably, the critical quantizer accuracy sufficient to guarantee convergence is independent from the network dimension.

Chapter 8: *Distributed Kalman filtering*

Here we deal with a possible application of the consensus ideas to the wide field of the distributed estimation. This research field has witnessed a large interest during the last years. This is mainly due to the recent technological advances in wireless communication that are promoting the appearance of large inexpensive interconnected systems, each with computational and sensing capabilities. However collecting measurements from distributed wireless sensors nodes at a single location for on-line data processing may not be feasible due to several reasons among which long packet delay (e.g. due to multi-hop transmission) and/or limited bandwidth of the wireless network, due e.g. to energy consumption

requirements. This problem is particularly relevant in wireless ad-hoc sensor networks where information needs to be multi-hopped from one node to another using closer neighbors. Therefore there is a growing need for in-network data processing tools and algorithms that provide high performance in terms of on-line estimation while (i) reducing the communication load among all sensor nodes, (ii) being very robust to sensor node failures or replacements and packet losses, and (iii) being suitable for distributed control applications. The literature addressing several aspects of distributed estimation is very rich. We can classify the various contributions into few large classes: *static vs dynamic* estimation, *distributed vs hierarchical* estimation, and *all-to-all vs multi-hop* communication networks. The distinction between static and dynamic estimation depends on whether the quantities to be estimated are constant or time-varying. The second distinction between hierarchical and distributed depends on whether the global estimate is required to be computed at a specific location on the network or at many sensing locations. The last distinction between all-to-all vs multi-hop depends on whether one node can directly send a message to any other node, or it has to route it through intermediate nodes. Our work belongs to the class of dynamic distributed estimation under multi-hop communication, while most of the works in the literature focus on different combinations of the three classes mentioned above.

Precisely the approach that we have followed focus on distributed estimation of dynamical systems for which sensor nodes are not physically co-located and exchange information only with their neighbors. For example, suppose that we want to track a dynamic quantity that changes according to a random walk, i.e $x(t+1) = x(t)+w(t)$, where $w(t)$ is a zero mean white noise process with covariance q, and we have N sensors that can measure this quantity corrupted by some noise, i.e. $y_i(t) = x(t)+n_i(t)$, where $n_i(t)$ are uncorrelated zero mean white noise processes with same covariance r. If all measurements were instantaneously available to a single location, it is well known from the centralized Kalman filter that the sufficient statistics necessary to reconstruct the optimal estimate would be given by the mean of all measurements, i.e. $\text{mean}(y(t)) := \frac{1}{N}\sum_{i=1}^{N} y_i(t)$. In a multi-hop setting, it is not possible to assume that all measurements are instantaneously available at a specific location, since data from distant nodes, at best, arrives with some delay which depends on the specific network topology. Moreover, in a distributed setting where each sensor node is required to compute a global state estimate, the number of messages from every node to every other node can congest the network.
However, if it was possible to provide an algorithm that computes the mean of a set of numbers only through local communication, then the optimal estimate could be computed at each sensor node as follows:

$$\begin{aligned}\hat{x}_i(t+1) &= (1-\ell_0)\text{mean}(\hat{x}(t)) + \ell_0 \text{mean}(y(t)) \\ &= \text{mean}\big((1-\ell_0)\hat{x}(t) + \ell_0 y(t)\big)\end{aligned}$$

where $\hat{x}_i$ is the local estimate of $i-$th sensor. An efficient way to compute the average of a set of numbers in a distributed way is represented by a linear *average consensus algorithm*. Indeed, if we consider the linear iterations $z^+ = Pz$, where z is the vector whose entries are the quantities to be averaged and P is the weighting matrix, this algorithm guarantees, under weak connectivity properties, that $\lim_{m\to\infty}[P^m z]_i = \text{mean}(z)$, i.e. all elements of vector $P^m z$ converge to their initial mean $\text{mean}(z)$. Therefore, provided that it is possible to communicate sufficiently fast within two subsequent sensor measurements, i.e. $m \gg 1$, then intuitively we can assume that the following distributed estimation strategy yields the optimal global state estimate:

$$\begin{aligned} z &= (1-\ell_0)\hat{x}_i(t) + \ell_0 y_i(t) && \textbf{measur. \& predict.} \\ \hat{x}_i(t+1) &= [P^m z]_i && \textbf{consensus}\end{aligned}$$

In this context of fast communication, i.e. $m \gg 1$, it is natural to optimize P for fastest convergence rate of P^m. The assumption $m \gg 1$ is reasonable in applications for which communication is inexpensive as compared to sensing. This is the case, for example, in rendezvous control or coordination of mobile sensors where moving and sensing is energetically more expensive than communicating. However, there are many other important applications in which the number m of messages exchanged per sampling time per node needs to be small, as required in battery-powered wireless sensor networks. Therefore the assumption that $[P^m z]_i \approx \text{mean}(z)$ is not valid. In this context, for example, it is not clear whether maximizing the rate of convergence of P is the best strategy. Moreover, also the optimal gain ℓ becomes a function of the matrix P and the number of exchanged messages m; this will unlikely coincide with the optimal centralized Kalman gain.
In our work, we have analyzed the interaction between the consensus matrix P, the number of messages per sampling time m, and the gain ℓ. In general the joint optimization of P and ℓ, in order to minimize the asymptotic variance of the estimate-error, is a hard problem to solve; in particular the problem is not convex. Nevertheless we were able to provide interesting results by exploring some important regimes, namely fast communication $m \to \infty$, "small" measurement noise ($r/q \to 0$) and "small" process noise ($q/r \to 0$).
As a side result of our analysis, we have also seen that the standard recipe of choosing the consensus matrix P which yields the fastest convergence is not necessarily the best thing to do; similarly choosing the centralized optimal gain ℓ_c is not necessarily the optimal strategy.