2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Improved Consensus Algorithms using Memory Effects

Gabriel Rodrigues de Campos, Alexandre Seuret

***Abstract*—This paper deals with simple integrator consensus problems. The objective is the design of an improved consensus algorithm for continuous-time multi-agent systems using memory effects. The novel algorithm proposes to sample, in an appropriate manner, part of the multi-agent systems information such that the algorithm converges, assuming that at each instant, agent's control laws will also consider the sampled past information of its neighbors. Stability conditions expressed in terms of LMI's and based on algebraic communication matrix structure are provided. The efficiency of the method is tested for different network communication schemes.**

I. INTRODUCTION

Networked control systems (NCS) are systems which are spatially distributed with a communication network used between sensors and actuators. Their primary advantages include their low cost, reduced weight and power requirements, simple installation and maintenance, and higher reliability. This means NCS's applications can be found in a large range of areas such as mobile sensor networks ([1]), remote surgery, haptic collaboration over Internet, multi-robot systems ([2]), automated highway systems, averaging in communication networks ([3]) and formation control ([4]). Several results have appeared in recent literature that consider systems with different motion models, symmetry of communication and network interactions. A recent review of the vast literature in the field can be found in [5], [6], [7] and [8].

A "consensus" algorithm represents an interaction rule that specifies the information exchange between an dynamic system, or *agent*, and all of its neighbors over the network in order to reach an agreement regarding a certain quantity of interest that depends on the state of all agents. We should also remark that using shared network introduces new challenges, such as delays over communications, packet losses or even communication blackout, witch can dramatically affect "consensus" convergence rate and cooperative control laws efficiency, that have been extensively studied in literature,as for example, in [9]. Here, we consider that agents are assumed to obey a simple integrator model. Knowing that classical consensus algorithms converge with a decay rate equal to the second smallest eigenvalue of Laplacian L, we propose to study improved behaviors for such algorithms. Accelerating the convergence of synchronous distributes averaging algorithms have been studied in literature based on two main approaches: optimizing the topology-respecting weight matrix summarizing the updates at each node ([3]) or incorporating memory into the distributed averaging algorithm. In this scope, even if for most applications, delays lead to a reduction of performances or can even lead to instability, there exists some cases where the introduction of a delay in the control loop can help to stabilize a system which would not be stable without it. This have been studied in [10] and [11]. Also, *local memory* effect's on consensus algorithms performances have also been studied in [12], considering naturally unstable systems and using the stabilizing delay concept (sampled approach) in order to achieve consensus. Theoretical guarantees for a distributed averaging algorithm with memory are also provided in [13] and [14].

In this article, in order to present better behaviors, we will provide an improved consensus algorithm with local memory based on sampling approach. We will prove, in a theoretical way, that the proposed algorithm always improves standard performances, and a method to design the algorithm parameters, including the appropriated sampling period T, on an "optimal" way is proposed based on a LMI's formulation. The communication graphs are supposed to be directed and undirected. This paper is organized as follows: Section II presents the problem treated in this article, as Section III will be dedicated to the establishment of the appropriated model. In Section IV we will motivate our work, and in Section V we stability analysis of the algorithm will be provided. Section VI includes illustrating simulation results and performance analysis, and finally, Section VII will present our conclusions and indicate possible future research efforts.

Throughout the paper, $\mathbb{R}^n$ denote the n-dimensional Euclidean space, and $\mathbb{R}^{n\times m}$ is the set of $n \times m$ real matrices. The set $\mathbb{S}^n$ stands for the set of symmetric matrices of $\mathbb{R}^{n\times n}$. The superscript 'T' stands for matrix transposition. For any matrix P in $\mathbb{S}^n$, the notation $P > 0$ means that the matrix P is positive definite. For any matrix A in $\mathbb{R}^n$, the notation $2He\{A\}$ corresponds to the following sum $A+A^T$. The matrix I represents the identity matrix. Finally, for any matrix M, the notation $(M)_i$ denotes the i^{th} line of M and $\lambda_k(M)$ represents the k^{th} eigenvalue of M. For the graph G with N vertices and edge set given by $E = \{(i, j) : j \in \mathcal{N}_i\}$ the *adjacency matrix* $A = A(G) = (a_{ij})$ is the $N \times N$ matrix given by $a_{ij} = 1$, if $(i, j) \in E$ and $a_{ij} = 0$, otherwise. The *degree* d_i of vertex i is defined as the number of its neighboring vertices, i.e. $d_i = \#j : (i, j) \in E$. Let Δ be the $N \times N$ diagonal matrix of d_i's. The *Laplacian* of G is the matrix $L = \Delta - A$. For an undirected graph the Laplacian matrix is symmetric positive semidefinite. Zero is a simple eigenvalue of L (the corresponding eigenvector is the vector of ones, $\vec{\mathbf{1}}$) if and

This work was supported by the European Project FeedNetBack: http://www.feednetback.eu/.

G. Rodrigues de Campos and A. Seuret are with NeCS Team, Automatic Control Department of GIPSA-Lab, UMR CNRS 5216, Grenoble, France E-mail: gabriel.rodrigues-de-campos, alexandre.seuret@gipsa-lab.grenoble-inp.fr

978-1-61284-799-3/11/$26.00 ©2011 IEEE

only if the associated directed graph has a directed spanning tree.

II. Problem Statement

A. Consensus Algorithm

In this paper the following problem is addressed. Consider the classical simple integrator consensus algorithm

$$\left\{ \begin{array}{l} \dot{x}_i(t) = u_i(t) \\ u_i(t) = \sum_{j \in \mathcal{N}_i} a_{ij}(x_j(t) - x_i(t)) \end{array} \right. \qquad i \in \{1, \dots, N\}, \tag{1}$$

where x_i represents variables of agent i. Introducing the vector $x(t) = [x_1(t), .., x_N(t)]^T$ containing the state of all agents, we then derive:

$$\dot{x}(t) = -Lx(t) \ , \tag{2}$$

where L is the Laplacian matrix.

This algorithm is distributed in the sense that each agent has only access to information from its neighbors. Moreover, consensus algorithms can be archived asymptotically if and only if the graph associated to the Laplacian L has a directed spanning tree (page 25 [8]).

In this paper, we will propose an improved algorithm for simple integrator agents. The goal is a performance comparaison between the proposed and the classical algorithm, where memory's effects on system's stability will be bring forward. Assuming that there exists a constant and positive scalar μ such that:

$$\sum_{j \in \mathcal{N}_i} a_{ij} = \mu, \quad i \in \{1, \dots, N\}.$$

The previous algorithm is modified into a new algorithm shown in Figure 1. To do so, we introduce a periodic sampling denoted by the sequence of instants $\{t_k\}_{k \geq 0}$ and such that $t_0 = 0$ and $T = t_{k+1} - t_k$. The improved algorithm is defined by

$$\forall t \in [t_k \ t_{k+1}[, \quad \dot{x}(t) = (-L - \delta A)x(t) + \delta A x(t_k) \tag{3}$$

where A is the adjacency matrix of the communication graph, $\delta \in \mathbb{R}$ and $T = t_{k+1} - t_k > 0$ are additional parameters of the improved algorithm. From the point of view of agent i, the state x_i is available at every time t. However, both continuous and sampled data from the neighbor agents of agent i are used.

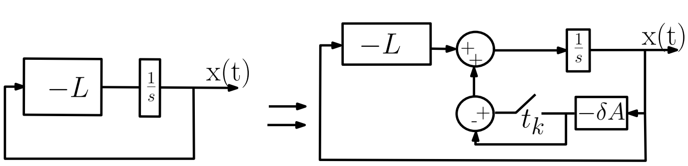


Fig. 1. Bloc diagrams of the classical and the improved algorithms.

Note that if δ and/or T are taken as zero, the classical algorithm is retrieved. In this article, we consider a sampling delay approach, using the time-varying delay defined by $\tau(t) = t - t_k$, for all $t \in [t_k, \ t_{k+1}]$ introduced in [15] and used in the context of multi-agent consensus algorithm in [12]. From computational point of view, this choice is relevant. One may have consider a constant delay τ instead of the sampling delay. However all values of x in the interval $[t - \tau, \ t]$ should be kept in memory whereas only one data is held when using the sampling approach. An inherent assumption is that all agents are synchronized and share the same clock to ensure that the agents also share the same sampling. For the sake of simplicity, we assumed that the sampling process is periodic. This makes sense in the situation of multi-agents systems. However the latter analysis could be extended to asynchronous samplings.

B. Preliminary definition

In order to clarify the presentation, a definition of exponential stability will be stated here.

Definition 1: ([16]) Let $\alpha > 0$ be some positive, constant, real number. The system is said to be exponentially stable with the decay rate α, or α-stable, if there exists a scalar $\beta \geq 1$ such that the solution $x(t; t_0, \phi)$ satisfies:

$$|x(t; t_0, \phi)| \leq \beta |\phi|_\tau e^{-\alpha(t - t_0)}. \tag{4}$$

III. Definition of an appropriate model

This section focuses on the definition of a suitable modeling of the consensus algorithm (3) to analyze its convergence. Knowing that the vector $\overrightarrow{\mathbf{1}}$ is an eigenvector of the Laplacian matrix associated to the eigenvalue 0, it is possible to find a change of coordinates $x = Wz$ such that:

$$U(-\mu I + A)W = \begin{bmatrix} B & \vec{0} \\ \vec{0}^T & 0 \end{bmatrix}, \tag{5}$$

where $U = \begin{bmatrix} U_1 \\ U_2 \end{bmatrix} = W^{-1}$ and $U_2 = (U)_N$. For graphs containing a directed spanning tree the Laplacian eigenvalues are all positive and we denote them by $0 < \lambda_2 \leq \ldots \leq \lambda_N$. Let also $B \in \mathbb{R}^{(N-1)\times(N-1)}$ be a diagonal matrix with $-\lambda_i$. The following lemma, which is taken from [17], provides an appropriate way to rewrite (3) based on the properties of the matrix L.

Lemma 1: The system (3) can be rewritten in the following way:

$$\dot{z}_1(t) = (-B + \delta(B + \mu I))z_1(t) - \delta(B + \mu I)z_1(t_k), \tag{6a}$$

$$\dot{z}_2(t) = -\mu z_2(t) + \mu z_2(t_k), \tag{6b}$$

where $z_1 \in R^{N-1}$, $z_2 \in R$ and the matrix B in given in (10).

Proof: By the Leibnitz formula, we have $x(t_k) = x(t) - \int_{t_k}^{t} \dot{x}(s)ds$, for all differentiable functions x. System (3) can be rewritten as:

$$\dot{x}(t) = -Lx(t) - \delta A \int_{t_k}^{t} \dot{x}(s)ds. \tag{7}$$

This representation is a way to understand how memory components affect the algorithm. We then rewrite (3) into two equations defined by $z_1 = U_1 x \in \mathbb{R}^{(N-1)}$ and $z_2 = U_2 x \in \mathbb{R}^N$ representing, respectively, the $N-1$ first components and the last component of z. Then (3) is rewritten as

$$\begin{bmatrix} \dot{z}_1(t) \\ \dot{z}_2(t) \end{bmatrix} = - \begin{bmatrix} B & \vec{0} \\ \vec{0}^T & 0 \end{bmatrix} \begin{bmatrix} z_1(t) \\ z_2(t) \end{bmatrix} - \begin{bmatrix} A_1' \\ A_2' \end{bmatrix} \int_{t-\tau}^{t} \dot{z}(s) ds,$$

where $\begin{bmatrix} A_1' \\ A_2' \end{bmatrix} = UAW$ and $A_2' = (UAW)_N$. From (5), simple matrix calculations lead us to

$$\begin{bmatrix} A_1' \\ A_2' \end{bmatrix} = UAW = ULW + \mu I = \begin{bmatrix} B + \mu I & \vec{0} \\ \vec{0}^T & \mu \end{bmatrix} \tag{8}$$

Using the Leibnitz formula, (3) can be rewritten as

$$\begin{aligned} \dot{z}_1(t) &= -B z_1(t) + \delta(B + \mu I) \textstyle\int_{t_k}^{t} \dot{z}_1(s) ds, \\ \dot{z}_2(t) &= -\delta\mu \textstyle\int_{t_k}^{t} \dot{z}_2(s) ds. \end{aligned} \tag{9}$$

■

The consensus problem is now expressed into an appropriate form to perform stability criteria. In the case of a symmetric network, the matrix W is an orthogonal matrix which means $U = W^T$. Then if the last column of W is $\beta \overrightarrow{\mathbf{1}}$, then $U_2 = 1/(\beta N) \overrightarrow{\mathbf{1}}$, which means that z_2 corresponds to the average of the position of all agents. This does not hold always for asymmetric communication network.

In the sequel, a stability analysis of the algorithm is proposed for any graph with a directed spanning tree, represented by the Laplacian L. Requiring a directed spanning tree is less stringent than requiring a strongly connected and balanced graph ([8]). This analysis is composed by two parts, one dealing with the stability of the algorithm and another concerning the agreement of the agents. More particularly, we will propose a method to choose appropriately the algorithm parameters δ and T for a given L, considering a performance optimisation. Next section will motivate this study.

IV. Does this algorithm always improve standard performances?

Assume for the moment that the Laplacian matrix corresponds to a symmetric graph. Let B be the diagonal matrix of the Laplacian eigenvalues defined before. We know that

$$B = \begin{bmatrix} -\lambda_2 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & -\lambda_N \end{bmatrix}. \tag{10}$$

Thus, we establish for all $i = 1, \dots, N-1$

$$\dot{z}_{1i}(t) = (-\lambda_{i+1} + b) z_{1i}(t) - b z_{1i}(t_k). \tag{11}$$

with $b = \delta(\lambda_{i+1} + \mu)$.

By integrating the previous equation, the following recurrence equation represents the discrete dynamics of the algorithm.

$$z_{1i}(t_{k+1}) = A(\lambda_{i+1}, \delta, T) z_{1i}(t_k), \tag{12}$$

with

$$A(\lambda_{i+1}, \delta, T) = \exp^{(-\lambda_{i+1}+b)T} \frac{-\lambda_{i+1}}{-\lambda_{i+1} + b} + \frac{b}{-\lambda_{i+1} + b}.$$

Note that system's (12) stability increases as $A(\lambda_{i+1}, \delta, T)$ decreases. We will prove that by varying δ and T values close to zero, we achieve a performance improvement for $\forall \lambda_{i+1}$, if

$$\frac{\partial A(\lambda_{i+1}, \delta, T)}{\partial T} \leq 0, \text{ for some } \delta \text{ values} \tag{13a}$$

$$\frac{\partial A(\lambda_{i+1}, \delta, T)}{\partial \delta} \leq 0, \text{ for some T values} \tag{13b}$$

From (12), by derivation of $A(\lambda_{i+1}, \delta, T)$, we have

$$\begin{aligned} \frac{\partial A(\lambda_{i+1}, \delta, T)}{\partial T} &= - e^{(-\lambda_{i+1}+b)T} \lambda_{i+1} \\ \frac{\partial A(\lambda_{i+1}, \delta, T)}{\partial \delta} &= \frac{-\lambda_{i+1} e^{(-\lambda_{i+1}+b)T}}{(-\lambda_{i+1} + b)} \left[T(\lambda_{i+1} + \mu) - \frac{(\lambda_{i+1} + \mu)}{-\lambda_{i+1} + b} \right] \\ &\quad + \frac{(\lambda_{i+1} + \mu)}{(-\lambda_{i+1} + b)^2} (\lambda_{i+1} + 2b) \end{aligned}$$

When we evaluate the previous equation for $T \simeq 0$ and for $\delta \simeq 0$, respectively, we have

$$\begin{aligned} \frac{\partial A(\lambda_{i+1}, \delta, T)}{\partial T} &= - \lambda_{i+1} \leq 0 \\ \frac{\partial A(\lambda_{i+1}, \delta, T)}{\partial \delta} &= e^{-\lambda_{i+1} T} (\lambda_{i+1} + \mu) \left(T + \frac{1}{\lambda_{i+1}} \right) \\ &\quad - \left(\frac{\lambda_{i+1} + \mu}{\lambda_{i+1}} \right) \leq 0 \end{aligned}$$

As $\frac{\partial A(\lambda_{i+1}, \delta, T)}{\partial T} = -\lambda_{i+1}$ is negative for all value of δ, and $\frac{\partial A(\lambda_{i+1}, \delta, T)}{\partial \delta}$ is also negative for small values of T, we can then conclude that for small values of δ and T system (12) tends to converge more rapidly when compare with the trivial algorithm. The pertinent problem of how to chose theses parameters values has been rased here, and will be treated in the next section.

V. Stability analysis

A. Preliminary stability analysis

This section deals with the stability analysis of (6b). The following lemma holds.

Lemma 2: The system defined in (6b) is constant for any sampling period T and any δ

$$\forall t, \quad z_2(t) = z_2(0) \tag{16}$$

Proof: Consider $k \geq 0$ and any $t \in [t_k \; t_{k+1}[$ and any parameters T, δ. The previous ordinary differential equation has known solutions of the form

$$z_2(t) = e^{-\delta\mu(t - t_k)} C_0 - z_2(t_k) \tag{17}$$

where $C_0 \in \mathbb{R}$ represent the initial condition of the ordinary differential equation. The initial condition is determined at time $t = t_k$. We then obtain $C_0 = 0$ and thus

$$\forall t \in [t_k \; t_{k+1}[, \quad z_2(t) = z_2(t_k) = z_2(0) \tag{18}$$

Then, we deduce that z_2 is constant ■

B. Stability analysis of the consensus algorithm

Consider the consensus algorithm (3) rewritten in the form of (6). We can establish

$$\dot{z_1}(t) = A(\delta)z_1(t) + A_d(\delta)z_1(t_k), \qquad (19)$$

with $A(\delta) = (-B + \delta(B + \mu I))$ and $A_d(\delta) = -\delta(B + \mu I)$. The following theorem holds

Theorem 1: Consider the proposed consensus algorithm (3) associated to a given Laplacian L representing a communication graph with a directed spanning tree, a given $\alpha > 0$, $\delta > 0$ and $T > 0$.

Assume that there exist $P > 0$, $R > 0$ and S_1 and $X \in \mathbb{S}^n$ and two matrices $S_2 \in \mathbb{R}^{n\times n}$ and $N \in \mathbb{R}^{2n\times n}$ that satisfy

$$\Pi_1 + f_\alpha(T,0)\Pi_2 + h_\alpha(T,0)\Pi_3 < 0, \qquad (20)$$

$$\begin{bmatrix} \Pi_1 + h_\alpha(T,T)\Pi_3 & g_\alpha(T,T)N \\ * & -g_\alpha(T,T)R \end{bmatrix} < 0, \qquad (21)$$

where

$$\begin{aligned} \Pi_1 &= 2He\{M_1^T P(M_0 + \alpha M_1)\} - M_3^T S_1 M_3 \\ &\quad -2He\{M_3^T S_2 M_2) - 2He\{N M_3\}, \\ \Pi_2 &= M_0^T R M_0 + 2He\{M_0^T (S_1 M_3 + S_2 M_2)\}, \\ \Pi_3 &= M_2^T X M_2, \end{aligned}$$

and $M_0 = \begin{bmatrix} A(\delta) & A_d(\delta) \end{bmatrix}$, $M_1 = \begin{bmatrix} I & 0 \end{bmatrix}$, $M_2 = \begin{bmatrix} 0 & I \end{bmatrix}$, $M_3 = \begin{bmatrix} I & -I \end{bmatrix}$. The functions f_α, g_α and h_α for all scalars T and $\tau \in [0\ T]$ are given by

$$\begin{aligned} f_\alpha(T,\tau) &= (e^{2\alpha(T-\tau)} - 1)/2\alpha, \\ g_\alpha(T,\tau) &= e^{2\alpha T}(1 - e^{-2\alpha\tau})/2\alpha, \\ h_\alpha(T,\tau) &= \frac{1}{\alpha}\left[\frac{e^{2\alpha T}-1}{2\alpha T} - e^{2\alpha\tau}\right]. \end{aligned} \qquad (22)$$

Then, the consensus algorithm (3) with the parameter δ and the sampling period T is thus $\alpha-$stable. Moreover the consensus equilibrium is given by

$$x(\infty) = U_2 x(0). \qquad (23)$$

Proof: The proof in based on the Lyapunov Theorem for discrete-time system using the continuous-time model of the multi-agent systems. For simplicity the proof is omitted but is presented in [18] and is similar to [12]. ■

VI. Examples

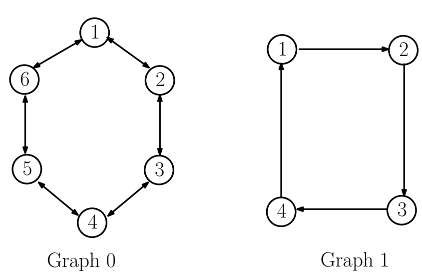


Fig. 2. Corresponding graphs of the matrices L_0 and L_1.

Consider a set of six and four agents connected through, respectively, the undirected and directed graphs shown in Figure 2. To each graph is associated a Laplacian matrix given by

$$L_0 = \begin{bmatrix} -1 & 0.5 & 0 & 0 & 0 & 0.5 \\ 0.5 & -1 & 0.5 & 0 & 0 & 0 \\ 0 & 0.5 & -1 & 0.5 & 0 & 0 \\ 0 & 0 & 0.5 & -1 & 0.5 & 0 \\ 0 & 0 & 0 & 0.5 & -1 & 0.5 \\ 0.5 & 0 & 0 & 0 & 0.5 & -1 \end{bmatrix},$$

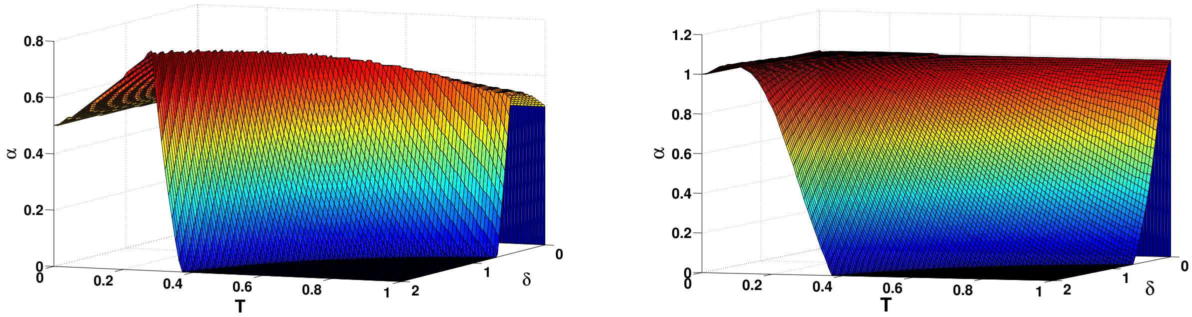
$$L_1 = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \\ 1 & 0 & 0 & -1 \end{bmatrix}.$$

For simulations, we took as initial conditions $x_0^T(0) = [30\ 25\ 15\ 0\ -10\ -30]$ and $x_1^T(0) = [30\ 25\ 15\ 0]$. Those two graphs are balanced, witch implies that consensus equilibrium value will be defined as the average of initial conditions presented just before.

The objective is to find the highest value for α (on the vertical axis) that guarantees algorithm (3) convergence. Figure 3, as a 3-D representation of α stability results, shows the maximum convergence rate satisfying Theorem 1 for several values of δ and T, and for L_0 and L_1, with $T \in [0,1]s$ and $\delta \in [0,2]$. We can identified a crest for specific values of (δ, T) meaning an improved behavior, and the best positive value of α is obtained when $(\delta, T) = (2, 0.32)$ and $(\delta, T) = (1.96, 0.09)$, for graph G_0 and graph G_1 respectively.

The stability conditions proposed in this article are sufficient but not necessary conditions. Best behavior/response is obtained for a certain value of (δ, T), and once it changes, this leads to a reduction of performances, as it will be shown in the following.

Figure 4 shows simulations from the classical algorithm (2) as well as the algorithm (3) considering L_0, L_1, and for several values of δ and T. The aim here is to compare systems performances with two different approaches and justify the interest of the proposed algorithm. Figure 4(a-b) show simulation results of the classical consensus algorithm. Figure 4(c-d) show simulation results using the optimal pair (δ, T) according to Theorem 1 and recovered on Figure 3.We can see that they correspond to a faster algorithm when compared with the trivial algorithm. In Figure 4(e-f), we kept the optimal value of T and changed δ value. Finally, for Figure 4(g-h), we kept the optimal value of δ and changed T value. In 4(c-d-e-f) we can then see that convergence rate decreases when compared to the others results. It's also possible to observe that the agreement value for the modified algorithm remains the average of the initial conditions. Consider now $\varepsilon = |x(t) - x_\infty|$, as the module of the error between agents states and the agreement value x_∞. Figure 5 shows the error ε evolution for graph G_0 and G_1. We consider the best values of (δ, T) retrieved before in Figure 3. Classical algorithm's performances correspond to the continuous line as the dotted line shows the behavior of the improved algorithm. We can than clearly observe that the algorithm proposed in this article converge more rapidly than the trivial simple integrator consensus. Analysis of Figure 4-5 strengthen the efficiency of the proposed approach.



(a) Exponential decay rate for G_0 (b) Exponential decay rate for G_1

Fig. 3. Convergence rate of the consensus algorithm for several values of (δ, T) and for the communication graphs G_0 and G_1.

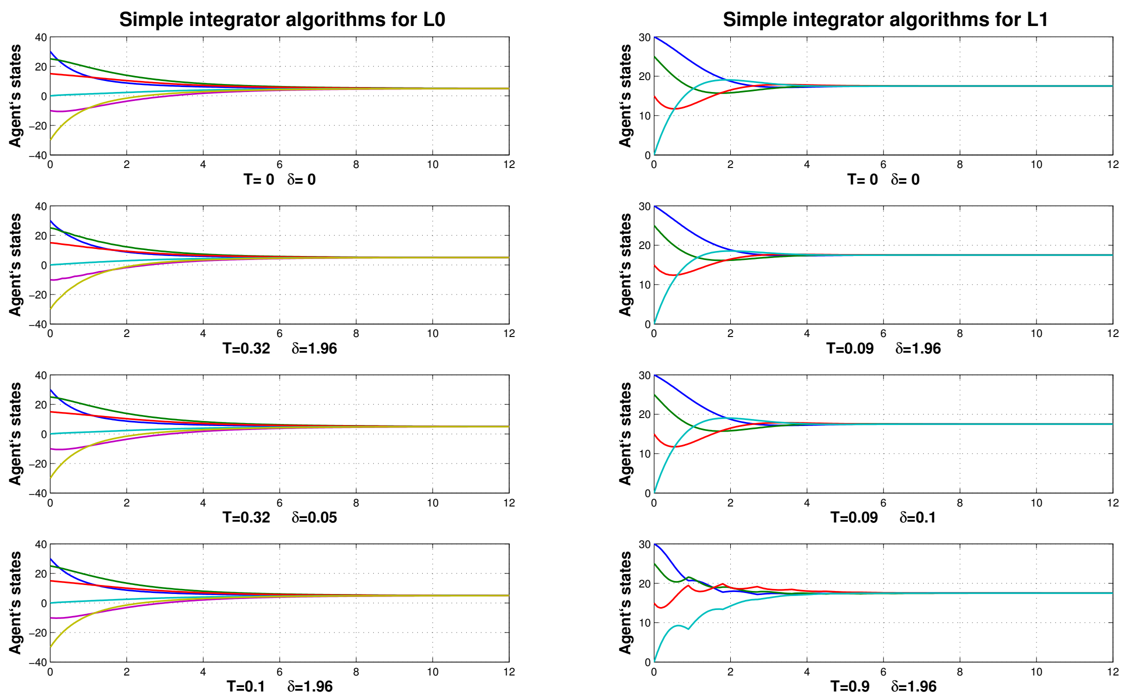


Fig. 4. System's behavior for different settings

VII. Conclusion

The influence of *local memory* in consensus algorithms for simple integrator agents have been studied. An optimization of controller parameters is proposed so that exponential stability of the solutions is achieved based on discrete-time Lyaponov Theorem and expressed in terms of LMI. Also, conditions for improved performances based on Laplacian's eigenvalues are provided here. Simulation results show the efficiency of the proposed algorithm, as well as the conservation of averaging properties. Further work might include robustness with respect to errors in the synchronisation clocks.

References

[1] P. Ogren, E. Fiorelli, and N. E. Leonard, "Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment," *In EEE Trans.Automat. Contr.*, vol. 49, no. 8, pp. 1292–1302, 2004.

[2] R. Olfati-Saber and R.M. Murray, "Consensus problems in network of agents with switching topology and time delays," *IEEE Trans. on Automatic Control*, vol. 49, no. 9, 2004.

[3] L. Xiao and S. Boyd, "Faster linear iterations for distributed averaging," *Systems and Control Letters*, vol. 53, pp. 65–78, 2004.

[4] D.V. Dimarogonas and K.J. Kyriakopoulos, "A connection between formation infeasibility and velocity alignment in kinematic multi-agent systems," *Automatica*, vol. 44, no. 10, pp. 2648–2654, 2008.

[5] R. Olfati-Saber, A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[6] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," in *Proceedings of IEEE*, vol. 95, 2007.

[7] W. Ren, R. W. Beard, and E. M. Atkins, "A survey of consensus problems in multi-agent coordination," *In American Control Conference*,

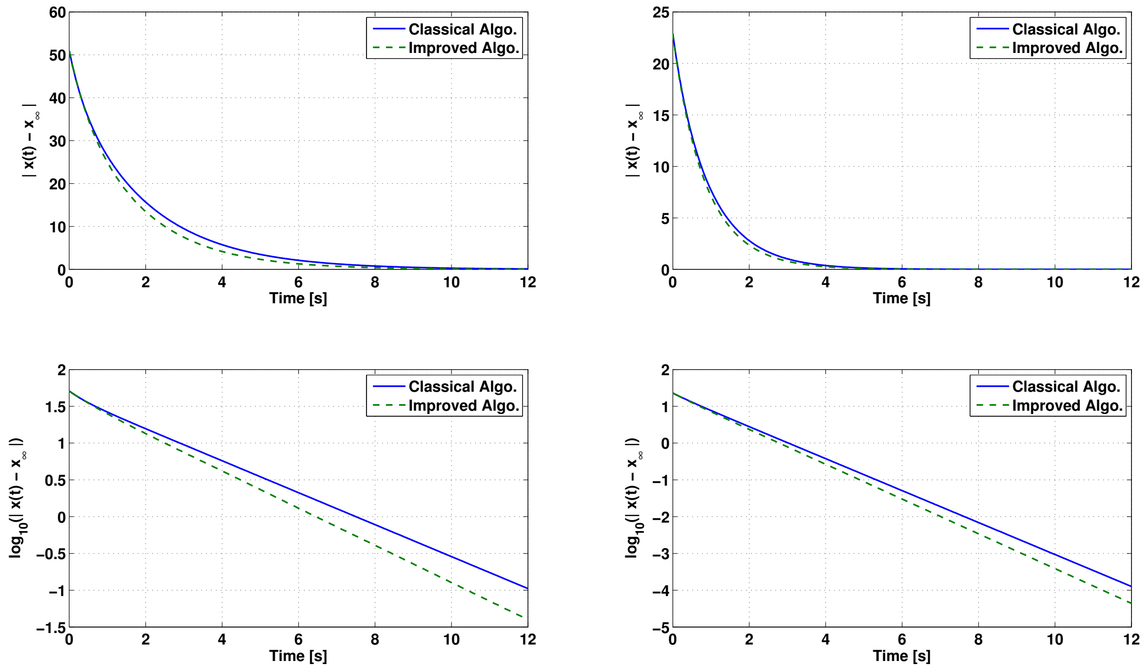


Fig. 5. Time evolution of error ε and $log_{10}(\varepsilon)$ for G_0 and G_1

2005.

[8] W. Ren and R. W. Beard, *Distributed Consensus in Multi-vehicle cooperative Control: theory and applications*. Springer, 2008.

[9] U. Muenz, A. Papachristodoulou, and F. Allgower, "Delay robustness in consensus problems," *To appear in Automatica*, 2010.

[10] K. Gu, V.-L. Kharitonov, and J. Chen, *Stability of time-delay systems*. Birkhauser, 2003.

[11] A. Seuret, C. Edwards, S.K. Spurgeon, and E. Fridman, "Static output feedback sliding mode control design via an artificial stabilizing delay," *IEEE Trans. on Automatic Control*, vol. 54, no. 2, pp. 256–265, 2007.

[12] G. Rodrigues de Campos and A. Seuret, "Continous-time double integrator consensus algortihms improved by an appropriate sampling," in 2^{nd} *IFAC Workshop on Distributed estimation and Control in Networked Systems*, 2010.

[13] B. N. Oreshkin, M. J. Coates, and Michael G. Rabbat, "Optimization and analysis of distributed averaging with memory," in 17^{th}*IFAC World Congress*, 2009.

[14] B. Johansson and M. Johansson, "Faster linear iterations for distributed averaging," in 47^{th} *Communication, Control, and Computing*, 2008.

[15] E. Fridman, A. Seuret, and J.-P. Richard, "Robust sampled-data stabilization of linear systems: An input delay approach," *Automatica*, vol. 40, no. 8, pp. 1141–1446, 2004.

[16] S.-I. Niculescu, C. E. De Souza, and L. Dugard and J.-M. Dion, "Robust exponential stability and stabilization of uncertain systems with time-varying delays," *IEEE Trans. on Automatic control*, vol. 43, no. 5, pp. 743–748, 1998.

[17] A. Seuret, D. V. Dimarogonas, and K.H. Johansson, "Consensus under communication delays," in *In 47th IEEE Conference on Decision and Control*, vol. 1, 2008, pp. 4922–4927.

[18] A. Seuret, "A novel stability analysis of sampled-data systems with applications to multi-rate sampling and packet loss," *Provisionnally accepted to Automatica*, 2011.