Discrete-time distributed Kalman filter design for networks of interconnected systems with linear time-varying dynamics

Leonardo Pedroso[1], Pedro Batista[1], Paulo Oliveira[2], and Carlos Silvestre[3]

[1] *Institute for Systems and Robotics, Instituto Superior Técnico, Universidade de Lisboa, Portugal*
[2] *Department of Mechanical Engineering, Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal*
[3] *University of Macau, Macau, People's Republic of China, on leave from the Instituto Superior Técnico of the University of Lisbon, Portugal*

Accepted version[1]

Accepted on 30 Oct 2021

Cite as

L. Pedroso, P. Batista, P. Oliveira, and C. Silvestre, 'Discrete-time distributed Kalman filter design for networks of interconnected systems with linear time-varying dynamics', *International Journal of Systems Science*, vol. 53, no. 6, pp. 1334–1351, 2022. doi: 10.1080/00207721.2021.2002461.

ORIGINAL PAPER

Discrete-time distributed Kalman filter design for networks of interconnected systems with linear time-varying (LTV) dynamics

Leonardo Pedroso[a], Pedro Batista[a], Paulo Oliveira[b], Carlos Silvestre[c]

[a]Institute for Systems and Robotics, Instituto Superior Técnico, Universidade de Lisboa, Portugal; [b]Department of Mechanical Engineering, Instituto Superior Técnico, Universidade de Lisboa, Portugal; [c]University of Macau, Macau, China, on leave from the Instituto Superior Técnico of the University of Lisbon, Portugal

ARTICLE HISTORY
Compiled December 29, 2021

ABSTRACT
This paper addresses the problem of designing distributed state estimation solutions for a network of interconnected systems modelled by linear time-varying (LTV) dynamics, in a discrete-time framework. The problem is formulated as a classical optimal estimation problem, for the global system, subject to a given sparsity constraint on the filter gain, which reflects the distributed nature of the network. Two methods are presented, both of them able to compute a sequence of well performing stabilising gains. Moreover, both methods are validated resorting to simulations of: i) a randomly generated synthetic LTV system; and ii) a large-scale nonlinear network of interconnected tanks. One of the proposed methods relies on a computationally efficient solution, thus it is computed very rapidly. The other achieves better performance, but it is computationally more expensive and requires that a window of the future dynamics of the system is known. When implemented to a nonlinear network, approximated by an LTV system, the proposed methods are able to compute well performing gains that stabilise the estimation error dynamics. Both algorithms are scalable, being adequate for implementation in large-scale networks.

KEYWORDS
Linear time-varying systems; distributed Kalman filter; distributed estimation; interconnected systems

1. Introduction

Over the past decades, distributed estimation and control has been a highly researched topic, since it provides a solution to the estimation and control problems of large-scale networks of interconnected systems. In fact, it emerges as an alternative to the use of well known centralized solutions, which become unfeasible to implement as the dimension of the network increases. The popularity of distributed solutions is also increasing with the widening of its applications to a broad range of engineering fields. Examples of such applications are unmanned aircraft formation flight (Bereg, Díaz-Báñez, Lopez, Rozario, and Valavanis (2015); Raffard, Tomlin, and Boyd (2004); Thien and Kim (2018); Wolfe, Chichka, and Speyer (1996)), unmanned underwater formations (Curtin, Bellingham, Catipovic, and Webb (1993); Healey (2001); Viegas, Batista,

CONTACT Leonardo Pedroso Email: leonardo.pedroso@tecnico.ulisboa.pt

Oliveira, and Silvestre (2012); Yuan, Licht, and He (2017)), satellite constellations (Ivanov, Monakhova, and Ovchinnikov (2019); Russell Carpenter (2002); Shaw (1999)), automated highway control (Alam, Mårtensson, and Johansson (2015); Bender (1991); Mu, Yan, Spurgeon, and Zhao (2016); Yanakiev and Kanellakopoulos (1996)), and irrigation networks (Cantoni et al. (2007); Gómez, Rodellar, and Mantecón (2002); Li (2014); Prodan, Lefevre, Genon-Catalot, et al. (2017)).

Although plenty of work has been carried out in distributed control of linear time-invariant systems (LTI), the problem of designing such controllers, which consists in solving an optimisation problem subject to a constraint that arises from the distributed nature of the configuration, is extremely difficult, as discussed by Blondel and Tsitsiklis (2000), and remains an open problem. In fact, the optimal solution for a linear system with Gaussian noise may be nonlinear, as discussed by Witsenhausen (1968). Furthermore, it has been shown that the solution of a distributed design control problem is the result of a convex optimisation problem if and only if quadratic invariance of the controller set is ensured (Lessard and Lall (2010, 2015)). For those reasons, the overwhelming majority of the approaches found in the literature attempt to find the optimal linear solution, which is also a difficult nonconvex optimisation problem that remains unsolved. On top of that, given the difficulty in finding the optimal linear solution, the most common approach found in the literature is to approximate the nonconvex optimisation problem by a convex one, which allows to obtain an approximate solution to the original problem. This is the approach followed in this paper. However, this paper and most of the research that follows this approach do not have stability or boundedness guarantees for the closed-loop system. Having said that, the research on distributed estimation of linear time-varying (LTV) systems, which is naturally more challenging, has been undergone to a much lesser extent. For instance, one of the few papers in this matter is Heydari and Demetriou (2016), for a consensus based protocol, in which an adaptive strategy is developed for a network of agents that collaboratively estimate the state of an LTV system. In this context, this paper addresses the problem of designing a distributed state estimation solution for a network of systems modelled by LTV dynamics, in a discrete-time framework. A general scheme for the design of distributed filters is followed in this paper, in which the problem is formulated as a classical optimal estimation problem, for the global system, with a given sparsity constraint on the filter gain. Such sparsity constraints impose certain entries of the global gain matrix to be null, following a structure that reflects the distributed nature of the network, necessary for the implementation of the distributed state estimator. It is also assumed that limited communication between agents is possible.

This paper introduces two methods for the computation of distributed filter gains for an arbitrary network of interconnected LTV systems with an arbitrary time-invariant network configuration. The configuration of the network is shaped by the available directed communication links between agents, which allow for sharing information. In this paper, it is shown that it can be portrayed by a sparsity constraint on the filter gains, as put forward in Section 2. Both methods consist of the generalisation of those introduced in Viegas, Batista, Oliveira, and Silvestre (2018) for the particular problem of decentralized relative navigation for LTI systems. In this paper, the generic decentralized estimation problem for networks of interconnected systems with dynamic couplings, as well as output measurement couplings, is addressed with emphasis on the limited data transmissions between agents. Albeit straightforward, this generalisation requires attention to some details on its implementation, given that one seeks a sequence of stabilising filter gains instead of a single steady-state gain. It is also important to consider the computational complexity of the proposed methods,

since the computations have to be carried out online. This generalisation not only allows for a significant widening of the application of both methods, even to nonlinear systems, but it is also presented in a generic framework, allowing for its application to a broad range of fields. The classical infinite-horizon optimisation problem subject to a sparsity constraint is nonconvex. Hence, the methods that are proposed herein rely on conveniently defined convex relaxations of the original optimisation problem to achieve a computationally efficient approximation to its solution. The first method proposed, denoted as the one-step method in the sequel, follows a similar approach as the classical Kalman filter, minimising at each time step the trace of the covariance of the estimation error. This method is computationally very efficient, exhibits a closed-form solution, and it does not require any particular initialisation. However, its solution is sub-optimal. The second method, denoted as the finite-horizon algorithm in the sequel, is used to compute an approximation to the finite-horizon problem instead, which is then extended to thse original infinite-horizon formulation. Albeit iterative, each iteration of this algorithm can be computed in closed-form and, although it requires a sequence of gains for its initialisation, they do not need to be stabilising or even to follow the sparsity constraint. Finally, both methods are validated resorting to numerical simulations. In addition to a randomly generated synthetic system, a large-scale nonlinear network of tanks is also considered. A MATLAB implementation of the decentralized algorithms put forward in this paper can be found in the *DECENTER* toolbox available at `https://decenter2021.github.io` (accessed on 10 July 2021).

This paper is organised as follows. In Section 2, the estimation problem is formulated and the assumptions that are considered are introduced. In Sections 3 and 4, the one-step and finite-horizon methods are presented, respectively. Section 5 details the implementation of both methods to a synthetic LTV system, comparing their performance and illustrating some details of their application. In Section 6, both methods are applied to a large-scale nonlinear network of interconnected tanks. Finally, Section 7 presents the main conclusions of this paper.

1.1. Notation

Throughout this paper, the identity, null, and ones matrices, all of proper dimensions, are denoted by $\mathbf{I}$, $\mathbf{0}$, and $\mathbf{1}$, respectively. Alternatively, $\mathbf{I}_n$ is also used to represent the $n \times n$ identity matrix. The entry (i, j) of a matrix $\mathbf{A}$ is denoted by $[\mathbf{A}]_{ij}$. The i-th component of a vector $\mathbf{v} \in \mathbb{R}^n$ is denoted by $\mathbf{v}_i$, and $\mathrm{diag}(\mathbf{v})$ denotes the $n \times n$ square diagonal matrix, whose diagonal is $\mathbf{v}$. Similarly, $\mathrm{diag}(\mathbf{A}_1, ..., \mathbf{A}_N)$ denotes the square block diagonal matrix whose diagonal blocks are given by matrices $\mathbf{A}_1, ..., \mathbf{A}_N$. The vectorisation of a matrix $\mathbf{A}$, denoted herein by $\mathrm{vec}(\mathbf{A})$, returns a vector composed of the concatenated columns of $\mathbf{A}$. Given a symmetric matrix $\mathbf{P}$, $\mathbf{P} \succ \mathbf{0}$ and $\mathbf{P} \succeq \mathbf{0}$ are used to point out that $\mathbf{P}$ is positive definite and positive semidefinite, respectively. The Kronecker product of two matrices $\mathbf{A}$ and $\mathbf{B}$ is denoted by $\mathbf{A} \otimes \mathbf{B}$.

2. Problem statement

2.1. Network dynamics model

Consider a network of N interconnected systems, $\mathcal{S}_i$, with $i = 1, \ldots, N$. The topology of the network, which is assumed to be time invariant, is defined by the dynamic and output measurement couplings between systems. Such coupling topologies may

be represented by directed graphs, or digraphs, $\mathcal{G} := (\mathcal{V}, \mathcal{E})$, composed of a set $\mathcal{V}$ of vertices and a set $\mathcal{E}$ of directed edges. An edge e incident on vertices i and j, directed from j towards i, is denoted by $e = (j, i)$. For a vertex i, its in-degree, ν_i^-, is the number of edges directed towards it, and its in-neighborhood, $\mathcal{D}_i^-$, is the set of indices of the vertices from which such edges originate. Conversely, for a vertex i, its out-degree, ν_i^+, is the number of edges directed from it, and its out-neighborhood, $\mathcal{D}_i^+$, is the set of indices of the vertices towards which such edges are directed. For a more detailed overview of the elements of graph theory used to model this network, see Wallis (2010); West et al. (1996). In this framework, each system is represented by a vertex, *i.e.* system $\mathcal{S}_i$ is represented by node i, and the dynamics couplings and output measurement couplings are represented by the directed graphs ${}^d\mathcal{G}$ and ${}^m\mathcal{G}$, respectively. In this configuration, if the dynamics of $\mathcal{S}_i$ depend on the dynamics of system $\mathcal{S}_j$, then this coupling is represented by an edge directed from vertex j towards vertex i, *i.e.* edge $e = (j, i)$ in the directed graph ${}^d\mathcal{G}$. Conversely, if the output of $\mathcal{S}_i$ depends on the dynamics of system $\mathcal{S}_j$, then this coupling is represented by an edge directed from vertex j towards vertex i, *i.e.* edge $e = (j, i)$ in the directed graph ${}^m\mathcal{G}$. It is important to stress that the direction of the edge matters. Note, for instance, that the fact that the dynamics of $\mathcal{S}_i$ depend on the dynamics of system $\mathcal{S}_j$ does not, necessarily, imply the converse.

The dynamics of system $\mathcal{S}_i$ are modelled by the following discrete-time LTV system

$$\begin{cases} \mathbf{x}_i(k+1) = \mathbf{A}_{i,i}(k)\mathbf{x}_i(k) + \mathbf{B}_{i,i}(k)\mathbf{u}_i(k) + \mathbf{w}_{i,{}^d\mathcal{D}_i^-}(k) + \\ \qquad\qquad + \sum_{j \in {}^d\mathcal{D}_i^-} \left(\mathbf{A}_{i,j}(k)\mathbf{x}_j(k) + \mathbf{B}_{i,j}(k)\mathbf{u}_j(k)\right), \\ \mathbf{y}_i(k) = \mathbf{C}_i(k)\mathbf{x}_i(k) + \mathbf{v}_{i,{}^m\mathcal{D}_i^-}(k) + \sum_{j \in {}^m\mathcal{D}_i^-} \mathbf{C}_{i,j}(k)\mathbf{x}_j(k), \end{cases} \tag{1}$$

where $\mathbf{x}_i(k) \in \mathbb{R}^{n_i}$ is the state vector, $\mathbf{u}_i(t) \in \mathbb{R}^{m_i}$ is the input vector, which is assumed to be known, and $\mathbf{y}_i(t) \in \mathbb{R}^{o_i}$ is the output vector, all of system $\mathcal{S}_i$; matrices $\mathbf{A}_{i,j}(k) \in \mathbb{R}^{n_i \times n_j}(k)$, $\mathbf{B}_{i,j}(k) \in \mathbb{R}^{n_i \times m_j}(k)$, and $\mathbf{C}_{i,j}(k) \in \mathbb{R}^{o_i \times n_i}(k)$ are known time-varying matrices that model the dynamics of system $\mathcal{S}_i$ and its couplings with the other systems in its in-neighborhood; vectors $\mathbf{v}_{i,{}^m\mathcal{D}_i^-}(k) \in \mathbb{R}^{o_i}$ and $\mathbf{w}_{i,{}^d\mathcal{D}_i^-}(k) \in \mathbb{R}^{n_i}$ are the observation and process noise, respectively, whose models are defined in the sequel.

The global dynamics of the network are, then, modelled by a generic LTV system of the form

$$\begin{cases} \mathbf{x}(k+1) = \mathbf{A}(k)\mathbf{x}(k) + \mathbf{B}(k)\mathbf{u}(k) + \mathbf{w}(k) \\ \mathbf{y}(k) = \mathbf{C}(k)\mathbf{x}(k) + \mathbf{v}(k) \end{cases}, \tag{2}$$

where $\mathbf{x}(k) := \text{col}(\mathbf{x}_1(k), \ldots, \mathbf{x}_N(k)) \in \mathbb{R}^n$ is the global state vector, $\mathbf{u}(k) := \text{col}(\mathbf{u}_1(k), \ldots, \mathbf{u}_N(k)) \in \mathbb{R}^m$ is the global input vector, and $\mathbf{y}(k) := \text{col}(\mathbf{y}_1(k), \ldots, \mathbf{y}_N(k)) \in \mathbb{R}^o$ is the global output vector; $\mathbf{v}(k) := \text{col}(\mathbf{v}_{1,{}^m\mathcal{D}_1^-}(k), \ldots, \mathbf{v}_{N,{}^m\mathcal{D}_N^-}(k))$ is the observation noise, modelled as a zero-mean white

Gaussian process with associated covariance matrix given by the block matrix

$$\mathbf{R}(k) := \begin{bmatrix} \mathbf{R}_{1,1}(k) & \mathbf{R}_{1,2}(k) & \dots & \mathbf{R}_{1,N}(k) \\ \mathbf{R}_{1,2}^T(k) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \mathbf{R}_{1,N}^T(k) & \dots & \dots & \mathbf{R}_{N,N}(k) \end{bmatrix},$$

where $\mathbf{R}_{i,j}(k) \in \mathbb{R}^{o_i \times o_j}$ models the correlation between $\mathbf{v}_{i,{}^m\mathcal{D}_i^-}(k)$ and $\mathbf{v}_{j,{}^m\mathcal{D}_j^-}(k)$; $\mathbf{w}(k) := \text{col}(\mathbf{w}_{1,{}^d\mathcal{D}_1^-}(k), \dots, \mathbf{w}_{N,{}^d\mathcal{D}_N^-}(k))$ is the process noise, modelled as a zero-mean white Gaussian process with associated covariance matrix given by the block matrix

$$\mathbf{Q}(k) := \begin{bmatrix} \mathbf{Q}_{1,1}(k) & \mathbf{Q}_{1,2}(k) & \dots & \mathbf{Q}_{1,N}(k) \\ \mathbf{Q}_{1,2}^T(k) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \mathbf{Q}_{1,N}^T(k) & \dots & \dots & \mathbf{Q}_{N,N}(k) \end{bmatrix},$$

where $\mathbf{Q}_{i,j}(k) \in \mathbb{R}^{n_i \times n_j}$ models the correlation between $\mathbf{w}_{i,{}^d\mathcal{D}_i^-}(k)$ and $\mathbf{w}_{j,{}^d\mathcal{D}_j^-}(k)$; $\mathbf{A}(k) \in \mathbb{R}^{n \times n}$, $\mathbf{B}(k) \in \mathbb{R}^{n \times m}$, and $\mathbf{C}(k) \in \mathbb{R}^{o \times n}$ are block matrices given by

$$\mathbf{A}(k) := \begin{bmatrix} \mathbf{A}_{1,1}(k) & \mathbf{A}_{1,2}(k) & \dots & \mathbf{A}_{1,N}(k) \\ \mathbf{A}_{2,1}(k) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \mathbf{A}_{N,1}(k) & \dots & \dots & \mathbf{A}_{N,N}(k) \end{bmatrix},$$

$$\mathbf{B}(k) := \begin{bmatrix} \mathbf{B}_{1,1}(k) & \mathbf{B}_{1,2}(k) & \dots & \mathbf{B}_{1,N}(k) \\ \mathbf{B}_{2,1}(k) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \mathbf{B}_{N,1}(k) & \dots & \dots & \mathbf{B}_{N,N}(k) \end{bmatrix},$$

and

$$\mathbf{C}(k) := \begin{bmatrix} \mathbf{C}_{1,1}(k) & \mathbf{C}_{1,2}(k) & \dots & \mathbf{C}_{1,N}(k) \\ \mathbf{C}_{2,1}(k) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \mathbf{C}_{N,1}(k) & \dots & \dots & \mathbf{C}_{N,N}(k) \end{bmatrix}.$$

Note that some of the block entries of matrices $\mathbf{R}(k), \mathbf{Q}(k), \mathbf{A}(k)$, $\mathbf{B}(k)$, and $\mathbf{C}(k)$ may be null due of the nonexistence of couplings between every pair of systems. In fact, if: i) $(j,i) \notin {}^d\mathcal{E}$ with $i \neq j$, then $\mathbf{A}_{i,j} = \mathbf{Q}_{i,j} = \mathbf{0}_{n_i \times n_j}$ and $\mathbf{B}_{i,j} = \mathbf{0}_{n_i \times m_j}$; and ii) $(j,i) \notin {}^m\mathcal{E}$ with $i \neq j$, then $\mathbf{C}_{i,j} = \mathbf{0}_{o_i \times n_j}$ and $\mathbf{R}_{i,j} = \mathbf{0}_{o_i \times o_j}$. The pair $(\mathbf{A}(k), \mathbf{C}(k))$ is assumed to be uniformly completely observable.

Before proceeding with the statement of the estimation problem, it is worth pointing out that virtually all large-scale networks have sparse dynamics. In particular, matrices $\mathbf{A}(k), \mathbf{B}(k), \mathbf{C}(k), \mathbf{R}(k)$, and $\mathbf{Q}(k)$ are generally sparse. In the limit scenario

of i) fully decoupled dynamics, $\mathbf{A}(k)$, $\mathbf{B}(k)$, and $\mathbf{Q}(k)$ are block diagonal; and ii) fully decoupled outputs, $\mathbf{C}(k)$ and $\mathbf{R}(k)$ are block diagonal. Most of the time, a large-scale network is fully decoupled either regarding the dynamics or the outputs of the systems. For that reason, various works focus on only one of these couplings. Viegas et al. (2018), for instance, focus only on networks with output coupling between systems, to solve the decentralized relative navigation problem. In this paper, the network is treated generically, at no point making any assumptions on the sparsity of matrices $\mathbf{A}(k), \mathbf{B}(k), \mathbf{C}(k), \mathbf{R}(k)$, or $\mathbf{Q}(k)$.

2.2. Filter design

The goal is to design a distributed filter for a network of N interconnected systems, whose global dynamics are described by the LTV system (2), under limited communication between systems in a distributed configuration. In this paper, state estimation is assumed to be achieved by a dynamical filter based on prediction-update steps employed in a Kalman filter. The prediction step is, thus, given by

$$\hat{\mathbf{x}}_i(k+1|k) = \mathbf{A}_{i,i}(k)\hat{\mathbf{x}}_i(k|k) + \mathbf{B}_{i,i}(k)\mathbf{u}_i(k) + \sum_{j \in {}^{d}\mathcal{D}_i^-} \left(\mathbf{A}_{i,j}(k)\hat{\mathbf{x}}_j(k|k) + \mathbf{B}_{i,j}(k)\mathbf{u}_j(k)\right) , \quad (3)$$

where $\hat{\mathbf{x}}_i(k+1|k)$ denotes the predicted state estimate of system $\mathcal{S}_i$ at instant $k+1$ and $\hat{\mathbf{x}}_i(k|k)$ the updated state estimate of system $\mathcal{S}_i$ at instant k. The global prediction step can be written as

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{A}(k)\hat{\mathbf{x}}(k|k) + \mathbf{B}(k)\mathbf{u}(k) , \quad (4)$$

where $\hat{\mathbf{x}}(k+1|k) := \mathrm{col}(\hat{\mathbf{x}}_1(k+1|k), \ldots, \hat{\mathbf{x}}_N(k+1|k))$ denotes the global predicted state estimate at instant $k+1$ and $\hat{\mathbf{x}}(k|k) := \mathrm{col}(\hat{\mathbf{x}}_1(k|k), \ldots, \hat{\mathbf{x}}_N(k|k))$ the global updated state estimate at instant k. Note that, to perform the prediction step according to (3), each system $\mathcal{S}_i$ ought to receive, through communication, the updated state estimates $\hat{\mathbf{x}}_j(k|k)$ with $j \in {}^{d}\mathcal{D}_i^-$. Thus, the directed communication links required to perform the update step are represented by the directed graph ${}^{d}\mathcal{G}$.

In a centralized configuration, each system has access to the global output measurement, at the expense of all-to-all communication via a central system. In a decentralized configuration, that is not the case. Each system $\mathcal{S}_i$ has only access to a subset of the measurement outputs, which is defined by another directed graph ${}^{o}\mathcal{G}$. It is important to remark that, unlike directed graphs ${}^{d}\mathcal{G}$ and ${}^{m}\mathcal{G}$, which are defined by the physical system under study, ${}^{o}\mathcal{G}$ can be freely selected during the filter design stage. In this framework, each system is represented by a vertex, *i.e.* system $\mathcal{S}_i$ is represented by node i, and if system $\mathcal{S}_i$ has access to the output measured by system $\mathcal{S}_j$, then this link is represented by an edge directed from vertex j towards vertex i, *i.e.* edge $e = (j, i)$ in the directed graph ${}^{o}\mathcal{G}$. The update step is, thus, given by

$$\hat{\mathbf{x}}_i(k|k) = \hat{\mathbf{x}}_i(k|k-1) + \mathbf{K}_{i,i}(k)\left(\mathbf{y}_i(k) - \hat{\mathbf{y}}_i(k)\right) + \sum_{j \in {}^{o}\mathcal{D}_i^-} \left(\mathbf{K}_{i,j}(k)\left(\mathbf{y}_j(k) - \hat{\mathbf{y}}_j(k)\right)\right) , \quad (5)$$

where $\hat{\mathbf{y}}_i(k)$ denotes the predicted output of system $\mathcal{S}_i$ at time instant k, which is

given by

$$\hat{\mathbf{y}}_i(k) := \sum_{j \in {}^m\mathcal{D}_i^-} \mathbf{C}_{i,j}(k)\hat{\mathbf{x}}_j(k|k-1)\,, \tag{6}$$

and $\mathbf{K}_{i,j}(k)$, $i = 1, \ldots, N$, $j \in {}^o\mathcal{D}_i^-$, are the distributed filter gains. It is also important to remark that, to perform: i) the update step according to (5), each system $\mathcal{S}_i$ ought to receive, through communication, the output $\mathbf{y}_j(k)$ and predicted output $\hat{\mathbf{y}}_j(k)$ with $j \in {}^o\mathcal{D}_i^-$; and ii) the computation of the predicted output according to (6), each system $\mathcal{S}_i$ ought to receive, through communication, the predicted state estimates $\hat{\mathbf{x}}_j(k|k-1)$ with $j \in {}^m\mathcal{D}_i^-$. Thus, the directed communication links required to perform the update step are represented by the union of the directed links of the directed graphs ${}^m\mathcal{G}$ and ${}^o\mathcal{G}$.

The global update step can be written as

$$\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \mathbf{K}(k)\left(\mathbf{y}(k) - \mathbf{C}(k)\hat{\mathbf{x}}(k|k-1)\right)\,, \tag{7}$$

where $\mathbf{K}(k) \in \mathbb{R}^{n \times o}$ is the global filter gain, which must follow a sparsity pattern imposed by ${}^o\mathcal{G}$. Let $\mathbf{E} \in \mathbb{R}^{n \times o}$ denote a sparsity pattern of the form

$$\mathbf{E} = \begin{bmatrix} \mathbf{E_{11}} & \ldots & \mathbf{E_{1N}} \\ \vdots & \ddots & \vdots \\ \mathbf{E_{N1}} & \ldots & \mathbf{E_{NN}} \end{bmatrix}\,,$$

where $\mathbf{E_{ij}} \in \mathbb{R}^{n_i \times o_j}$ with

$$\mathbf{E_{ij}} = \begin{cases} \mathbf{1} & ,\ (j,i) \in {}^o\mathcal{E} \\ \mathbf{0} & ,\ (j,i) \notin {}^o\mathcal{E} \end{cases}\,. \tag{8}$$

The set of matrices which obey the sparsity constraint determined by $\mathbf{E}$ is defined as

$$\text{Sparse}(\mathbf{E}) = \left\{ [\mathbf{K}]_{ij} \in \mathbb{R}^{n \times o} : [\mathbf{E}]_{ij} = 0 \implies [\mathbf{K}]_{ij} = 0;\ i = 1, ..., n,\ j = 1, ..., o \right\}.$$

Note that the global update step (7) is equivalent to the concatenation of the local update steps (5) if and only if $\mathbf{K}(k) \in \text{Sparse}(\mathbf{E})$. An example of a sparsity pattern is given in Section 6 for a network of interconnected tanks. It is worth remarking that, if there is all-to-all communication, then $\mathbf{E} = \mathbf{1}$, which corresponds to a centralized configuration. It is also important to stress that, although the choice of the output communication links between systems can take advantage of the known dynamic dependences between systems, which makes use of the communication links that are already in place and yields better performance, a generic and predefined sparsity pattern is considered.

Defining $\mathbf{P}(k|k-1) \succeq \mathbf{0} \in \mathbb{R}^{n \times n}$ and $\mathbf{P}(k|k) \succeq \mathbf{0} \in \mathbb{R}^{n \times n}$ as the global covariance of the estimation error at instant k after the prediction and update steps, respectively, one can write

$$\mathbf{P}(k|k-1) = \mathbf{A}(k-1)\mathbf{P}(k-1|k-1)\mathbf{A}^T(k-1) + \mathbf{Q}(k-1)\,, \tag{9}$$

and

$$\mathbf{P}(k|k) = \mathbf{K}(k)\mathbf{R}(k)\mathbf{K}^T(k) + (\mathbf{I} - \mathbf{K}(k)\mathbf{C}(k))\mathbf{P}(k|k-1)(\mathbf{I} - \mathbf{K}(k)\mathbf{C}(k))^T\,, \qquad (10)$$

for the global system. Note that designing a distributed filter for a network of interconnected systems, whose local dynamics are described by the LTV system (1), is equivalent to designing a global filter (4),(7), whose gain must follow a sparsity pattern.

With the definition of a sparsity pattern, it is now possible to formulate the problem of designing a distributed filter for the global LTV system (2). One aims to optimally compute a sequence of filter gains that follow the sparsity pattern required by the structure of the network of systems, which is assumed to be time-invariant. For an infinite-horizon and a known and time-invariant sparsity pattern $\mathbf{E}$, solve the optimisation problem

$$\begin{aligned} \underset{\substack{\mathbf{K}(i)\in\mathbb{R}^{n\times o}\\ i\in\mathbb{N}}}{\text{minimise}} \quad & \lim_{T\to+\infty} \frac{1}{T}\sum_{k=1}^{T} \operatorname{tr}(\mathbf{P}(k|k)) \\ \text{subject to} \quad & \mathbf{K}(i) \in \operatorname{Sparse}(\mathbf{E}),\ i \in \mathbb{N}\,. \end{aligned} \qquad (11)$$

It is assumed, throughout the paper, that the distributed state observer is stabilisable. The assumption on the uniform observability of the pair $(\mathbf{A}(k), \mathbf{C}(k))$ is only a necessary condition for the existence of a stabilising sequence of sparse gains.

Because of the sparsity constraint, the optimisation problem (11) is nonconvex and its optimal solution is still an open problem. To overcome this difficulty the optimisation problem may be relaxed so that it becomes convex, allowing for the use of well known optimisation techniques. Albeit optimal for the modified problem, the relaxed solution is only an approximation to the solution of the original problem. For this reason, careful relaxation is necessary to ensure that the separation between both solutions is minimal. This approach is designated convex relaxation and will be used to derive the methods put forward in this paper.

Each of the following two sections presents a method for computing approximate solutions to the optimisation problem (11). The proposed methods are the time-varying counterparts of the one-step and finite-horizon methods presented in Viegas et al. (2018) for the relative navigation problem of LTI systems.

2.3. Communication requirements

As discussed in the previous subsection, there are various communication requirements associated with each step of the distributed filter. To sum up, each system $\mathcal{S}_i$ has to receive through communication from $\mathcal{S}_j$: i) the updated state estimates $\hat{\mathbf{x}}_j(k-1|k-1)$ with $j \in {}^d\mathcal{D}_i^-$; ii) the predicted state estimates $\hat{\mathbf{x}}_j(k|k-1)$ with $j \in {}^m\mathcal{D}_i^-$; and iii) the output $\mathbf{y}_j(k)$ and predicted output $\hat{\mathbf{y}}_j(k)$ with $j \in {}^o\mathcal{D}_i^-$. Thus, the required directed links correspond to the union of the directed edges that make up the directed graphs ${}^d\mathcal{G}$,${}^m\mathcal{G}$, and ${}^o\mathcal{G}$. Note that each system $\mathcal{S}_i$ is required to communicate with other systems, with which it is coupled either dynamically or through the output measurement. For that reason, generally speaking, it is feasible to establish these links in practice. While we maintain full generality for the structure of different networks, in a typical application either ${}^d\mathcal{D}_i^- = \emptyset\ \forall i$ or ${}^m\mathcal{D}_i^- = \emptyset\ \forall i$, and ${}^o\mathcal{G}$ is chosen according

to the physical structure of the system.

Another very important aspect to take into account is the synchronisation of the data transmissions. On one hand, the updated state estimates $\hat{\mathbf{x}}_j(k-1|k-1)$ with $j \in {}^{d}\mathcal{D}_i^-$, the predicted state estimates $\hat{\mathbf{x}}_j(k|k-1)$ with $j \in {}^{m}\mathcal{D}_i^-$, and the predicted output $\hat{\mathbf{y}}_j(k)$ with $j \in {}^{o}\mathcal{D}_i^-$ can be transmitted to system $\mathcal{S}_i$ at any time since time-instant $k-1$ until time instant k. For that reason no complex synchronisation protocols are required. On the other hand, the transmission of the output $\mathbf{y}_j(k)$ with $j \in {}^{o}\mathcal{D}_i^-$ is required at time instant k in system $\mathcal{S}_i$, but it is only available at time instant k in system $\mathcal{S}_j$. For this reason, very complex synchronisation algorithms are required to handle these transmission without causing prohibitively large delays. Furthermore, for large-scale networks, this synchronisation becomes unfeasible to implement in practice, so it is usual to adopt a fully distributed configuration instead, *i.e.*, setting ${}^{o}\mathcal{V} = \emptyset$ and ${}^{o}\mathcal{E} = \emptyset$ in the filter design stage. In such configuration, according to the definition of the sparsity pattern (8), the gain of the global system must follow a block diagonal structure.

Often, for dynamically coupled networks of interconnected systems, the initial model of the network is continuous-time. After discretisation, which is required to implement estimation and control techniques in a digital computer, the number of dynamical couplings between agents increases. An example of such phenomenon is presented in Section 6 for a network of interconnected tanks. The additional dynamical couplings, that arise with the discretisation of the continuous-time dynamics of the network, are generally very weak. For that reason, in applications where the cost of establishing communication links is high, it is usually a good practice to consider only the dominant couplings in ${}^{m}\mathcal{G}$ and find a compromise between the accuracy of the prediction step and the communication burden.

3. One-step method for computation of filter gains

For the derivation of the one-step method, an approach similar to the one used for the unconstrained Kalman filter is used. In fact, the gain in each instant is computed so that the trace of the covariance of the estimation error for that same instant is minimised. The optimisation problem (11) is, thus, modified to

$$\begin{aligned} \underset{\mathbf{K}(k)\in\mathbb{R}^{n\times o}}{\text{minimise}} \quad & \operatorname{tr}(\mathbf{P}(k|k)) \\ \text{subject to} \quad & \mathbf{K}(k) \in \operatorname{Sparse}(\mathbf{E}) \,, \end{aligned} \tag{12}$$

for $k \in \mathbb{N}$, given the predicted estimation error covariance, $\mathbf{P}(k|k-1)$, at each time step. Substituting (10) in the objective function of the relaxed optimisation problem (12) yields a quadratic function in relation to $\mathbf{K}(k)$. Given that the sparsity constraint is convex, the relaxed optimisation problem (12) is convex, allowing for the use of techniques similar to those used to solve the unconstrained problem. However, using this formulation, the computation of each gain does not take into account its influence on the estimation error covariance of future time steps, yielding a sub-optimal solution to the original problem. In spite of that, its solution can be obtained in closed-form.

Theorem 3.1. *Let* $\mathbf{l}_i$ *denote a column vector whose entries are all set to zero except for the* i*-th one, which is set to* 1*, and define* $\boldsymbol{\mathcal{L}}_i := \operatorname{diag}(\mathbf{l}_i)$*. Define a vector* $\mathbf{m}_i \in \mathbb{R}^o$

to encode the non-zero entries in the i-th row of $\mathbf{K}(k)$ following

$$\begin{cases} \mathbf{m}_i(j) = 0, & [\mathbf{E}]_{ij} = 0 \\ \mathbf{m}_i(j) = 1, & [\mathbf{E}]_{ij} \neq 0 \end{cases}, \; j = 1, ..., o \,,$$

and let $\boldsymbol{\mathcal{M}}_i = \text{diag}(\mathbf{m}_i)$. Then, the optimal one step gain that solves (12) is given by

$$\mathbf{K}(k) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}_i \mathbf{P}(k|k-1)\mathbf{C}^T(k)\mathcal{M}_i \left(\mathbf{I} - \boldsymbol{\mathcal{M}}_i + \boldsymbol{\mathcal{M}}_i \mathbf{S}(k) \boldsymbol{\mathcal{M}}_i\right)^{-1} \,, \tag{13}$$

which can be solved efficiently (see Remark 2), where $\mathbf{S}(k)$ is the innovation covariance at step k, given by

$$\mathbf{S}(k) = \mathbf{C}(k)\mathbf{P}(k|k-1)\mathbf{C}^T(k) + \mathbf{R}(k) \,. \tag{14}$$

Proof. See A. □

Remark 1. Note that, using the result above, the sequence of gains that solves the optimisation problem (12) can be computed forward in time. This computation takes turns propagating the predicted error covariance using (10) and (9), in this order, and computing the optimal gain making use of (13). For this reason, this method is said to be causal, in the sense that, for each instant, the gain computation does not require the future dynamics of the system to be known *a priori*. Allied with the fact that it is computationally efficient, this method allows for the online computation of the gain for each time step.

Remark 2. The closed-form solution (13) has a computational complexity of $\mathcal{O}(n^4)$. Instead of using it, the exact numeric algorithm proposed in Pedroso and Batista (2021) can be, alternatively, applied to (A1) to compute each gain with a computational complexity of $\mathcal{O}(|\chi|^3)$, where $|\chi|$ denotes the number of nonzero entries of $\mathbf{E}$. Usually, in distributed control applications, $|\chi|$ is given by $|\chi| \approx cn$, where $c \in \mathbb{N}$ is a constant. It, thus, follows that a computational complexity of $\mathcal{O}(n^3)$ is achieved, which is equal to the one of the centralised solution. An efficient MATLAB implementation of this method can be found in the *DECENTER* toolbox available at `https://decenter2021.github.io` (accessed on 10 July 2021).

4. Finite-horizon method for the computation of filter gains

The finite-horizon method, presented in this section, seeks to find an approximation to the solution of the infinite-horizon problem (11) considering, in a first instance, the equivalent finite-horizon problem, *i.e.* to compute a sequence of filter gains that minimise the sum of the trace of the covariance of the estimation error over a given finite window $W \in \mathbb{N}$. The optimisation problem is, thus, given by

$$\begin{aligned} \underset{\substack{\mathbf{K}(i) \in \mathbb{R}^{n \times o} \\ i=1,...,W}}{\text{minimise}} \quad & \sum_{k=1}^{W} \text{tr}(\mathbf{P}(k|k)) \\ \text{subject to} \quad & \mathbf{K}(i) \in \text{Sparse}(\mathbf{E}), \; i = 1, ..., W \,. \end{aligned} \tag{15}$$

However, similarly to the infinite-horizon problem, this problem is nonconvex. Therefore, to make use of well known optimisation techniques, convex relaxation is employed. In fact, instead of minimising the sequence of gains as a whole, one may iteratively minimise each gain of the sequence individually, while taking into account its effect on the whole finite window. The solution of the original finite-horizon optimisation problem (15) can, thus, be approximated by the solutions of the relaxed optimisation problem

$$\begin{aligned} \underset{\mathbf{K}(k)\in\mathbb{R}^{n\times o}}{\text{minimise}} \quad & \sum_{i=1}^{W} \text{tr}(\mathbf{P}(i|i)) \\ \text{subject to} \quad & \mathbf{K}(k) \in \text{Sparse}(\mathbf{E}) \,, \end{aligned} \tag{16}$$

for $k = 1, ..., W$. Expanding the objective function of the optimisation problem above, one readily concludes, after some algebraic manipulation using (9) and (10), that, for each k, it is quadratic in relation to $\mathbf{K}(k)$. Given that the sparsity constraint is also convex, not only is the relaxed optimisation problem (16) convex for each k, but it also has a closed-form solution, as detailed in the following result.

Theorem 4.1. *Define a matrix* $\mathbf{Z}$ *such that the vector* $\mathbf{Z}\text{vec}(\mathbf{K}(k))$ *contains the non-zero entries of* $\mathbf{K}(k)$ *according to the desired sparsity pattern. The closed-form solution of* (16) *is given by*

$$\text{vec}(\mathbf{K}(k)) = \mathbf{Z}^T \left(\mathbf{Z}(\mathbf{S}(k) \otimes \mathbf{\Lambda}(k+1))\mathbf{Z}^T\right)^{-1} \mathbf{Z}\text{vec}\left(\mathbf{\Lambda}(k+1)\mathbf{P}(k|k-1)\mathbf{C}(k)\right) \,, \tag{17}$$

which can be computed efficiently (see Remark 5), where $\mathbf{S}(k)$ *is given by* (14) *and*

$$\mathbf{\Lambda}(k+1) = \mathbf{I}_n + \sum_{i=k+1}^{W} \mathbf{\Gamma}^T(k+1, i)\mathbf{\Gamma}(k+1, i) \,,$$

with

$$\mathbf{\Gamma}(k_i, k_f) = \prod_{j=k_i}^{k_f} (\mathbf{I}_n - \mathbf{K}(k_i + k_f - j)\mathbf{C}(k_i + k_f - j))\mathbf{A}(k_i + k_f - j - 1) \,, \tag{18}$$

for $k_i \leq k_f$ *and* $\mathbf{\Gamma}(k_i, k_f) = \mathbf{I}_n$ *for* $k_i > k_f$.

Proof. See B. □

For an example on how to compute matrix $\mathbf{Z}$ for a given sparsity pattern, see (Viegas et al., 2018, Section 5). Each time a gain is modified, the sequence of error covariance matrices needs to be updated, which can be computationally expensive. Nevertheless, analysing the closed-form solution for the computation of $\mathbf{K}(k)$, given by (17), one readily notices it only makes use of the error covariance of instants up to k. For this reason, the gains can be computed in reverse order, *i.e.* from the last time step of the window to the first, updating the covariances when all the gains of the window have already been computed. Repeating this process, that is, taking turns computing the sequence of gains backwards in time and recomputing the covariance matrices forward in time, the sequence of gains converges to a near-optimal solution

of the finite-horizon optimisation problem (15). This process is referred to, herein, as an outer loop iteration. This algorithm is presented in Table 1.

Table 1. Algorithm for the computation of the sequence of gains using the finite-horizon method.

(1) **Initialisation:** Select a window size, W, an initial covariance $\mathbf{P}(0|0) \succeq \mathbf{0}$, and compute a set of initial filter gains $\mathbf{K}(k)$, $k = 1, ..., W$, using, *e.g.*, the one-step method. See Remark 3 for more details on the initialisation. Compute the resulting covariances $\mathbf{P}(k|k)$, $k = 1, ..., W$.
Select a stopping criterion, *e.g.*, a minimum improvement on the objective function of the finite-horizon optimisation problem (15) or a fixed number of iterations.
(2) **While:** stopping criterion is not met
  (a) **For:** $i = W, ..., 1$
    (i) Recompute $\mathbf{K}(i)$ using (17).
  (b) Recompute the covariances $\mathbf{P}(k|k)$, $k = 1, ..., W$, for the new filter gains, using (9) and (10).
(3) **Return:** the sequence of gains $\mathbf{K}(k)$, for $k = 1, ..., W$.

Remark 3. As pointed out in Table 1, the finite-horizon algorithm requires a sequence of gains and error covariance matrices for its initialisation. This issue is addressed similarly to the LTI counterpart of the finite-horizon algorithm. For more details see (Viegas et al., 2018, Remark 2).

Remark 4. Note that the finite-horizon algorithm, unlike the one-step method, is not causal, in a sense that, for each instant, the gain computation requires a window of the future dynamics of the system to be known. For this reason, the application of this algorithm is possible either if one has a model of the evolution of the system with time or if it is used in combination with an online system identification algorithm.

Remark 5. The closed-form solution (17) has a computational complexity of $\mathcal{O}(|\chi|(no)^2)$, where $|\chi|$ denotes the number of nonzero entries of $\mathbf{E}$. Instead of using it, the exact numeric algorithm proposed in Pedroso and Batista (2021) can be, alternatively, applied to (B2) to compute each iteration with a computational complexity of $\mathcal{O}(|\chi|^3)$. Usually, in distributed control applications, $|\chi|$ is given by $|\chi| \approx cn$, where $c \in \mathbb{N}$ is a constant. It, thus, follows that a computational complexity of $\mathcal{O}(n^3)$ is achieved for each iteration, which is equal to the one of the centralized solution. An efficient MATLAB implementation of this method can be found in the *DECENTER* toolbox available at `https://decenter2021.github.io` (accessed on 10 July 2021).

It is evident that it is unfeasible to make $W \to \infty$ to approximate the solution of the infinite-horizon problem (11) due to the increasing computational load as W becomes large. Instead, one considers a finite window, W, that is large enough so that the gains computed within that window converge to those that are obtained if an arbitrarily large window is used. The appropriate length for this finite window varies depending on the system dynamics. The gains may, then, be computed for the appropriate window and used until the end is reached, instant when a new window is defined and new gains are computed using as initial covariance, $\mathbf{P_0}^{new}$, the covariance

at the end of the previous window, $\mathbf{P}^{prev}(W|W)$.

However, an important characteristic of the finite-horizon algorithm is that it is greedy. In fact, given that the estimation error covariance in the time instants after the window is not taken into account in the cost function of the finite-horizon algorithm, it is possible to select gains that allow for a sudden decrease of the trace of the error covariance at the end of the window. This effect can be noticed in Section 5, when this procedure is applied to a synthetic system. At first sight, this may seem an advantageous characteristic, however, the use of the gains near the end of the window, responsible for the sudden decrease of the trace, deteriorates the performance after the transition to the next window, resulting in a sudden spike of the trace. Although, in this example, the sudden decrease may seem small and negligible at first sight, its impact is significant and it is important to take it into account, as it is exemplified in Section 5. Assume the sudden decrease of the trace occurs d time steps before the end of the window. Instead of using the gains computed for each window until its end, one should only use them until the instant $W - d$. At this instant a new window is defined using as initial covariance, $\mathbf{P_0}^{new}$, the covariance of the previous window before the sudden decrease, $\mathbf{P}^{prev}(W - d|W - d)$. Following this approach it is possible to approximate the evolution that would be obtained setting $W \to \infty$, with a manageable computational load and requiring the dynamics of the system to be known only until W time steps ahead, when computing the sequence of gains to be used for each window. This will be exemplified in Sections 5 and 6.

5. Simulation results for a synthetic system

In this section, the implementation of the one-step method and of the finite-horizon algorithm to a synthetic system is simulated. The matrices of the synthetic system, whose constant parts were randomly generated, rounded to 3 decimal places, are given by:

$$\mathbf{A}(k) = \begin{bmatrix} -0.695 & -0.844 & -0.991 & -0.831 \\ 0.652 & -0.115 & 0.550 & -0.200 \\ 0.0767 & -0.787 & 0.635 & -0.480 \\ 0.992 & 0.924 & 0.737 & 0.600 \end{bmatrix} + \begin{bmatrix} 0 & 0 & \cos(k/10) & 0 \\ 0 & \mathrm{e}^{-k/100} & 0 & 0 \\ 0 & 0 & 0 & \sin^2(k/10) \\ \cos(k/20) & 0 & 0 & 0 \end{bmatrix},$$

$$\mathbf{C}(k) = \begin{bmatrix} 0.096 & 0.956 & 0.235 & 0.015 \\ 0.132 & 0.575 & 0.353 & 0.043 \\ 0.942 & 0.060 & 0.821 & 0.169 \end{bmatrix} + \begin{bmatrix} 0 & 0 & \mathrm{e}^{-k/50} & 0 \\ 0 & 0 & \cos(k/10) & 0 \\ 0 & \sin(k/10) & 0 & 0 \end{bmatrix},$$

$$\mathbf{Q}(k) = \begin{bmatrix} 7.535 & -3.023 & -2.421 & 1.356 \\ -3.023 & 6.342 & 5.537 & -0.074 \\ -2.421 & 5.537 & 7.363 & 1.753 \\ 1.356 & -0.074 & 1.753 & 1.549 \end{bmatrix} + \begin{bmatrix} \sin^2(k/13) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \cos^2(k/10) & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\mathbf{R}(k) = \begin{bmatrix} 1.057 & 2.918 & -1.093 \\ 2.918 & 8.792 & -5.366 \\ -1.093 & -5.366 & 8.921 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & \cos^2(k) & 0 \\ 0 & 0 & \sin^2(k) \end{bmatrix},$$

and

$$\mathbf{E} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}.$$

The finite-horizon was applied to approximate an infinite-horizon window, as discussed in Section 4, with $W = 45$, $d = 15$, and performing 32 outer loop iterations. Fig. 1 depicts the evolution of the trace of the covariance of the estimation error for 20000 Monte Carlo simulations. The vertical dashed lines in this plot indicate the transitions between finite windows used in the finite-horizon algorithm. First, from Fig. 1, one can conclude that the finite-horizon method applied to approximate the infinite-horizon consistently outperforms the one-step method. Moreover, the transition between windows of the finite-horizon appears to be smooth, not having been affected by the sudden decrease in error covariance at the end of each finite window. Fig. 2 depicts the projected evolution of the trace of the covariance of the estimation error if one were to use $d = 0$. Analysing this plot, one can readily point out a sudden increase at the beginning of each new window, resulting in a significantly poorer estimation performance compared to the implementation described in Section 4 and depicted in Fig. 1. This practical example clearly demonstrates the advantage of using overlapping windows in the finite-horizon algorithm.

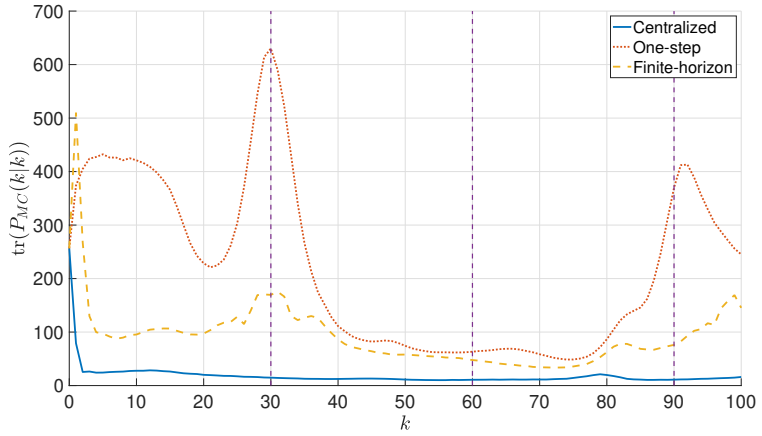


Figure 1. Evolution of the trace of the covariance of the estimation error for 20000 Monte Carlo simulations.

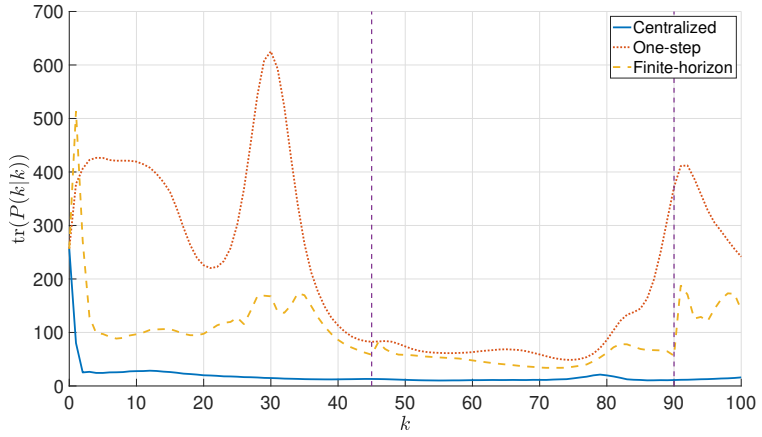


Figure 2. Projected evolution of the trace of the covariance of the estimation error with $d = 0$ in the implementation of the finite-horizon algorithm.

6. Simulation results for a network of N tanks

In this section, both methods put forward in this paper are applied to a large-scale network of N tanks, as a means of assessing their performance. Given that the dynamics of the projected network are nonlinear, to employ the methods devised one can approximate its behaviour by an LTV system, linearising and discretising its dynamics about successive equilibrium points. For this reason, it allows to assess the performance of the proposed distributed estimation methods when implemented in nonlinear time-varying systems. The quadruple-tank network introduced in Johansson (2000) inspired the example shown herein.

6.1. N tanks network dynamics

Consider N interconnected tanks, as shown in Fig. 3, where N is an even integer. The water level of tank i is denoted by h_i. The network is actuated by $N/2$ pumps, which are controlled by the lower tanks, whose inputs are denoted by u_i for $i = 1, ..., N/2$, in accordance with the schematic. Each pump is connected to a three-way valve that regulates the fraction of the flow, held constant, that goes to each of the tanks supplied by the pump. Each tank has a sensor, which measures its water level, with output y_i for tank i. Making use of mass balances and Bernoulli's law, the system dynamics, in the absence of noise, are given by

$$\begin{cases} A_i\dot{h}_i(t) = -a_i\sqrt{2gh_i(t)} + a_{\frac{N}{2}+i}\sqrt{2gh_{\frac{N}{2}+i}(t)} + \gamma_i k_i u_i(t), \;\; i = 1, ..., N/2 \\ A_i\dot{h}_i(t) = -a_i\sqrt{2gh_i(t)} + (1-\gamma_{i-\frac{N}{2}-1})k_{i-\frac{N}{2}-1}u_{i-\frac{N}{2}-1}(t), i = \frac{N}{2}+2, ..., N, \\ A_{\frac{N}{2}+1}\dot{h}_{\frac{N}{2}+1}(t) = -a_{\frac{N}{2}+1}\sqrt{2gh_{\frac{N}{2}+1}(t)} + (1-\gamma_{\frac{N}{2}})k_{\frac{N}{2}}u_{\frac{N}{2}}(t) \\ y_i(t) = k_c h_i(t), \;\; i = 1, ..., N \end{cases} \tag{19}$$

where A_i and a_i are the cross sections of tank i and of its outlet hole, respectively; the constant γ_i represents the fraction of the flow that passes through the valve i to the lower tanks; k_i is the constant of proportionality between the mass flow and the input of pump i; g denotes the acceleration of gravity; and k_c is the constant of proportionality between the water level and the output of each sensor.

The nonlinear dynamics are linearised about a given equilibrium point, characterised by equilibrium water levels, h_i^0, $i = 1, ..., N$; inputs u_i^0, $i = 1, ..., N/2$; and outputs y_i^0, $i = 1, ..., N$. Writing the state, control, and output vectors, respectively, as

$$\mathbf{x_c}(t) = \begin{bmatrix} h_1(t) - h_1^0 \\ \vdots \\ h_N(t) - h_N^0 \end{bmatrix}, \qquad \mathbf{u_c}(t) = \begin{bmatrix} u_1(t) - u_1^0 \\ \vdots \\ u_{\frac{N}{2}}(t) - u_{\frac{N}{2}}^0 \end{bmatrix}, \qquad \mathbf{y_c}(t) = \begin{bmatrix} y_1(t) - y_1^0 \\ \vdots \\ y_N(t) - y_N^0 \end{bmatrix},$$

the continuous-time linearised system is given by

$$\begin{cases} \dot{\mathbf{x_c}}(t) = \mathbf{A_c}(t)\mathbf{x_c}(t) + \mathbf{B_c}(t)\mathbf{u_c}(t) + \mathbf{w_c}(t) \\ \mathbf{y_c}(t) = \mathbf{C_c}(t)\mathbf{x_c}(t) + \mathbf{v_c}(t) \end{cases}, \tag{20}$$

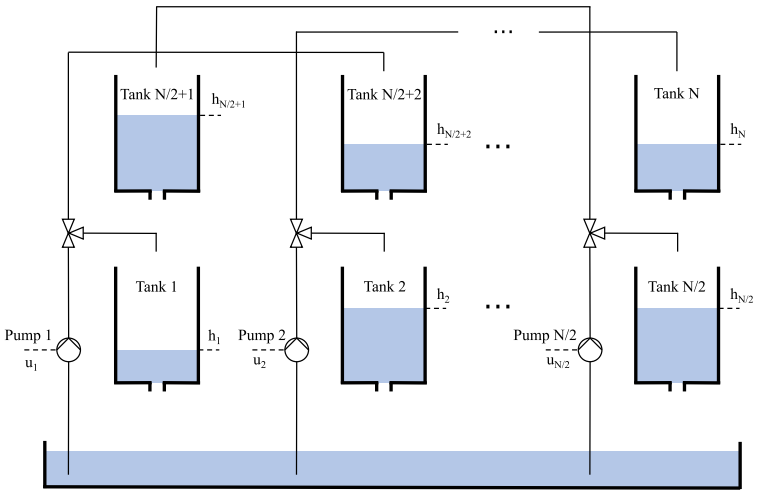


Figure 3. Schematic of the N tanks network.

with $\mathbf{A_c}(t) \in \mathbb{R}^{N\times N}$, $\mathbf{B_c}(t) \in \mathbb{R}^{N\times N/2}$, and $\mathbf{C_c}(t) \in \mathbb{R}^{N\times N}$ given by

$$[\mathbf{A_c}(t)]_{ij} = \begin{cases} -1/T_i & , i = j \\ \frac{A_j}{A_i T_j} & , j = i + N/2 \\ 0 & , \text{otherwise} \end{cases},$$

$$[\mathbf{B_c}(t)]_{ij} = \begin{cases} \gamma_i k_i / A_i & , i = j \\ (1-\gamma_j) k_j / A_i & , j = i - N/2 - 1 \\ (1-\gamma_j) k_j / A_i & , i = N/2+1,\ \ j = N/2 \\ 0 & , \text{otherwise} \end{cases},$$

and

$$\mathbf{C}_c(t) = k_c \mathbf{I}_N \,,$$

where T_i is the time constant of tank i, given by

$$T_i = \frac{A_i}{a_i}\sqrt{\frac{2h_i^0}{g}} \,.$$

Vectors $\mathbf{w_c}(t) \in \mathbb{R}^N$ and $\mathbf{v_c}(t) \in \mathbb{R}^N$ are the process and observation noise, modelled as zero-mean uncorrelated white Gaussian processes, with associated covariance matrices $\mathbf{Q_c}(t) \succeq \mathbf{0} \in \mathbb{R}^{N\times N}$ and $\mathbf{R_c}(t) \succ \mathbf{0} \in \mathbb{R}^{N\times N}$, respectively.

Provided that this system is slow, one can assume that the water level measurements and control inputs are updated with a constant period T. Under this assumption, the discretisation of (20) yields

$$\begin{cases} \mathbf{x}(k+1) = \mathbf{A}(k)\mathbf{x}(k) + \mathbf{B}(k)\mathbf{u}(k) + \mathbf{w}(k) \\ \mathbf{y}(k) = \mathbf{C}(k)\mathbf{x}(k) + \mathbf{v}(k) \end{cases},$$

where $\mathbf{w}(k)$ and $\mathbf{v}(k)$ are also zero-mean uncorrelated white Gaussian processes with associated covariance matrices $\mathbf{Q}(k) \succeq \mathbf{0} \in \mathbb{R}^{N\times N}$ and $\mathbf{R}(k) \succ \mathbf{0} \in \mathbb{R}^{N\times N}$, and $\mathbf{A}(k)$, $\mathbf{B}(k)$, $\mathbf{C}(k)$, $\mathbf{Q}(k)$, $\mathbf{R}(k)$, and $\mathbf{u}(k)$ are the result of the discretisation using

$$\begin{aligned}
\mathbf{A}(k) &= e^{\mathbf{A_c}(kT)T} \\
\mathbf{B}(k) &= \left(\int_0^T e^{\mathbf{A_c}(kT)\tau} d\tau\right) \mathbf{B_c}(kT) \\
\mathbf{C}(k) &= \mathbf{C_c}(kT) \\
\mathbf{Q}(k) &= \int_0^T e^{\mathbf{A_c}(kT)\tau} \mathbf{Q_c}(kT) e^{\mathbf{A_c}^T(kT)\tau} d\tau \\
\mathbf{R}(k) &= \mathbf{R_c}(kT) \\
\mathbf{u}(k) &= \mathbf{u_c}(kT)\,.
\end{aligned}$$

It is important to remark that to perform the linearisation, each local filter ought to access the estimates of the variables of the network that define the equilibrium point, through communication. Provided that the water levels change slowly, it may be carried out with a given periodicity, $T_{lin} = qT$, thereby reducing the computational load and communication needs.

6.2. Filter implementation

The problem considered for this network is the design of a distributed state estimation solution in which each tank has i) only access to the measurement of its water level, *i.e., a fully distributed configuration is considered*; and ii) may receive state estimates of other tanks through communication, to perform the prediction step of each local filter. The approach followed consists in the implementation of a local filter in each of the tanks, which estimates exclusively its own water level, relying on a communication link with some of the other tanks. The directed communication links that are implemented between tanks are carefully chosen to allow for an accurate prediction step of the local filters, requiring only estimates of the other tanks already available at the previous time-step. On the other hand, the update step is undergone in a fully distributed framework, not requiring the transmission of sensor outputs between tanks. Unlike previous state estimates, sensor outputs of each tank are only available at the same instant the communication is performed, thus their transmission is harder to implement using slow communication links without introducing a delay. Thus, the low coupling between agents in this particular example is exploited. For more details, see Section 2.3.

In continuous-time, the analysis of matrices $\mathbf{A_c}(t)$ and $\mathbf{B_c}(t)$ suggests that each lower tank i is only dynamically coupled to the tank above, tank $i + N/2$, through $\mathbf{A_c}(t)$. However, when this network is analysed in discrete-time, the tank whose pump supplies water to tank $i + N/2$, alters, during each discretisation interval, the rate at which water flows from tank $i+N/2$ to tank i. For that reason, after the discretisation of matrix $\mathbf{B_c}(t)$, one notices that each lower tank i is also dynamically coupled to the control input of the lower tank, whose pump supplies the tank $i + N/2$, through $\mathbf{B}(k)$. Thus, in discrete-time, the predicted state estimate for the lower tanks is, for $i = 2, ..., N/2$, given by

$$\hat{\mathbf{x}}_i(k+1|k) = [\mathbf{A}(k)]_{ii}\, \hat{\mathbf{x}}_i(k|k) + [\mathbf{A}(k)]_{i,i+\frac{N}{2}}\, \hat{\mathbf{x}}_{i+\frac{N}{2}}(k|k) + [\mathbf{B}(k)]_{ii}\, \mathbf{u}_i(k) + [\mathbf{B}(k)]_{i,i-1}\, \mathbf{u}_{i-1}(k)$$

and, for tank 1, by

$$\hat{\mathbf{x}}_1(k+1|k) = [\mathbf{A}(k)]_{11}\,\hat{\mathbf{x}}_1(k|k)+[\mathbf{A}(k)]_{1,1+\frac{N}{2}}\,\hat{\mathbf{x}}_{1+\frac{N}{2}}(k|k)+[\mathbf{B}(k)]_{11}\,\mathbf{u}_1(k)+[\mathbf{B}(k)]_{1,\frac{N}{2}}\,\mathbf{u}_{\frac{N}{2}}(k).$$

Note that $u_i(k)$ and $\mathbf{u}_i(k)$ are distinct, the former is the input to pump i and the latter is the i-th component of $\mathbf{u}(k)$, which is the input relative to the equilibrium point. The updated estimate follows

$$\hat{\mathbf{x}}_i(k+1|k+1) = \hat{\mathbf{x}}_i(k+1|k) + K_i(k+1)\left(\mathbf{y}_i(k) - [\mathbf{C}(k+1)]_{ii}\,\hat{\mathbf{x}}_i(k+1|k)\right)\,,$$

where $K_i(k+1)$ is the filter gain. Again, $y_i(k)$ and $\mathbf{y}_i(k)$ are distinct, the former is the output of the water level sensor in tank i and the latter is the i-th component of $\mathbf{y}(k)$. The equations above reflect the communication needs for the lower tanks. Each lower tank i has to receive an estimate of the water level in the tank above it, and also the control input given by the tank whose pump supplies water to the tank above tank i.

Either in continuous-time or discrete-time, each upper tank i is only dynamically coupled to the control input of the tank whose pump supplies tank i, through $\mathbf{B}(k)$. Thus, the predicted state estimate for an upper tank i is, in discrete-time, given by

$$\hat{\mathbf{x}}_i(k+1|k) = [\mathbf{A}(k)]_{ii}\,\hat{\mathbf{x}}_i(k|k) + [\mathbf{B}(k)]_{i,i-N/2-1}\,\mathbf{u}_{i-N/2-1}(k)\,,$$

for $i = N/2+2, ..., N$, and by

$$\hat{\mathbf{x}}_{\frac{N}{2}+1}(k+1|k) = [\mathbf{A}(k)]_{\frac{N}{2}+1,\frac{N}{2}+1}\,\hat{\mathbf{x}}_{\frac{N}{2}+1}(k|k) + [\mathbf{B}(k)]_{\frac{N}{2}+1,\frac{N}{2}}\,\mathbf{u}_{\frac{N}{2}}(k)\,,$$

for $i = N/2+1$. The updated estimate follows

$$\hat{\mathbf{x}}_i(k+1|k+1) = \hat{\mathbf{x}}_i(k+1|k) + K_i(k+1)\left(\mathbf{y}_i(k) - [\mathbf{C}(k+1)]_{ii}\,\hat{\mathbf{x}}_i(k+1|k)\right)\,, \quad (22)$$

where $K_i(k+1)$ is the filter gain. Analysing (22), each upper tank i has to receive, through communication, the input computed by the tank whose pump supplies tank i. A schematic of the communication links necessary for the proposed distributed solution is depicted in Fig. 4, which shows that only $3N/2$ directed communication links are used out of a possible $N(N-1)$ links. As an example, for the network of $N = 40$ tanks, simulated in Section 6.3, less than 4% of the possible directed communication links are used. It is important to remark that, in this particular example, given that the couplings between agents are weak, it is possible to implement few communication links without compromising estimation performance significantly. However, in networks with stronger couplings, there is a greater degradation of performance when compared with a centralized solution, which is the price to pay for an easier or feasible implementation.

Grouping the local filters, it is possible to write the dynamics of the global filter, whose gain is subject to a sparsity constraint. The predicted estimate of the global filter is given by

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{A}(k)\hat{\mathbf{x}}(k|k) + \mathbf{B}(k)\mathbf{u}(k)\,,$$

and the updated state estimate by

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \mathbf{K}(k+1)\left(\mathbf{y}(k+1) - \mathbf{C}(k+1)\hat{\mathbf{x}}(k+1|k)\right)\,,$$

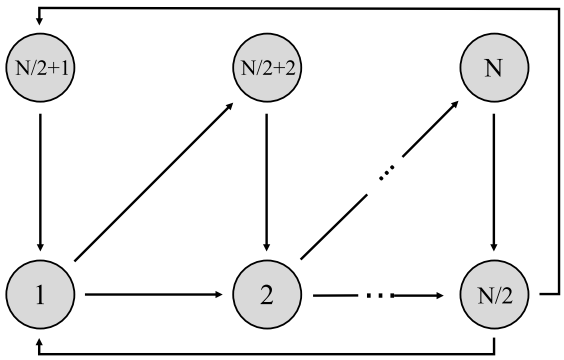


Figure 4. Schematic of the communication links for the N tanks network.

where $\mathbf{K}(k+1)$ is the global filter gain. Comparing the global and local filter dynamics, it follows that $\mathbf{K}(k) = \text{diag}(K_1(k), ..., K_N(k))$, which is equivalent to setting a sparsity constraint on the global gain corresponding to the sparsity pattern $\mathbf{E} = \mathbf{I}_N$. It is interesting to remark that the sparsity constraint allows only for $1/N$ of the total entries of the gain to be non-null, which alongside the imposed diagonal structure results in a very simple update step. One may reckon, at first sight, that such a severe constraint would result in a significant loss of performance when compared with the unconstrained solution. However, as the simulation results of Section 6.3 show, due to the weak couplings between agents, the performance loss is actually small.

6.3. Simulation results

The network was simulated for $N = 40$ tanks and the values of its physical constants are presented in Table 2. The process noise covariance matrix was randomly generated taking into consideration the dynamical dependencies between the tanks, and the observation noise covariance matrix was set to $\mathbf{R_c} = \mathbf{I}_N$, in accordance with the noise of the measurements of the water levels. The sampling time was set to $T = 1\,\text{s}$ and the linearisation period to $T_{lin} = 10T$. Analysing the system dynamics, given by (19), one notices that an equilibrium point corresponds to the solution of a system of N equations with $3N/2$ unknowns. It is, thus, necessary to select $N/2$ of these variables to define an equilibrium point, which, for this simulation, were chosen to be the estimates of the water levels of the lower tanks, $h_1, ..., h_{\frac{N}{2}}$. For this reason, each time a new linearisation is performed every local filter has to receive through communication the water level in the lower tanks, compute the remaining variables that define the equilibrium point, and linearise the relevant entries of matrix $\mathbf{A}$ about that point. The initial level of the tanks is set to $h_i = 20\,\text{cm}$ for $i = 1, ..., N$ and the initial covariance matrix to $\mathbf{P_0} = 10\mathbf{I}_N\ \text{cm}^2$. Considering that the only goal is to assess the estimation performance, an open-loop control law is chosen, as given by

$$u_i(t) = \begin{cases} 4\text{V} & t < 400 \\ 6\text{V} & t \geq 400 \end{cases}, i = 1, ..., \frac{N}{2}\,.$$

The finite-horizon algorithm was initialised with the gains computed using the one-step method. Moreover, 5 outer loop iterations were used, which proves to be enough for the convergence of the finite-horizon solution within 10^{-5} of the limit solution.

Table 2. Values of the physical constants of the N tanks network.

Constant	Value
A_i, i odd	28 cm^2
A_i, i even	32 cm^2
a_i, i odd $\leq N/2$	0.071 cm^2
a_i, i even $\leq N/2$	0.057 cm^2
a_i, $i > N/2$	0.040 cm^2
k_c	0.5 Vcm^{-1}
g	981 $\text{cm}\,\text{s}^{-2}$
k_i	3.33 $\text{cm}^3\,\text{s}^{-1}\,\text{V}^{-1}$
γ_i, i odd	0.7
γ_i, i even	0.6

Given that the finite-horizon method requires the dynamics of the system in a time window that spans future instants, and considering that the dynamics of the network vary with its state vector, it is not possible to simulate this method online without the use of a mechanism that predicts the future evolution of the state vector, thus allowing to obtain the linearised dynamics. To allow for the comparison of both methods put forward in this paper, at the beginning of each window, the finite-horizon method receives, through communication, the estimated water-level in all the tanks, which allows for the computation of all outer loop iterations without requiring additional communication. Given that the actuation is known, it simulates the linear evolution of the network until the end of the window, linearising the system with the same periodicity T_{lin}.

Fig. 5 depicts the evolution of the difference between the trace of the estimation error covariance of the finite-horizon algorithm for a window $W = 40$ and $W = 300$. It reveals that the sudden decrease due to the greediness of the method, when compared with the trace of the covariance of the estimation error, is not significant. In fact, the use of a window $W = 40$, of which none of the gains computed near the end of the window is ignored, *i.e.* setting $d = 0$, allows for a sudden decrease that is roughly 6 orders of magnitude below the magnitude of the trace of the covariance of the estimation error, which proves to be adequate for the application of the finite-horizon algorithm to approximate the solution of the infinite-horizon problem, as detailed in Section 4.

Figs. 6 depicts the nonlinear simulation of the water level in tanks 13 and 31, as well as the estimates for the centralized solution, one-step method, and finite-horizon algorithm. It is clear that none of the solutions diverge and all provide estimates that are close to the true water level in the tank. Fig. 7 depicts the evolution of the trace of the covariance of the estimation error, obtained with 5000 Monte Carlo simulations. Analysing this plot, one can readily note that there are no significant differences between the one-step and finite-horizon solutions. Moreover, both achieve a performance close to the centralized solution.

7. Conclusion

Very little work has been carried out regarding the design of distributed state estimation solutions for networks of interconnected systems modelled by LTV dynamics. In this paper, two methods for the computation of distributed filter gains for an arbitrary

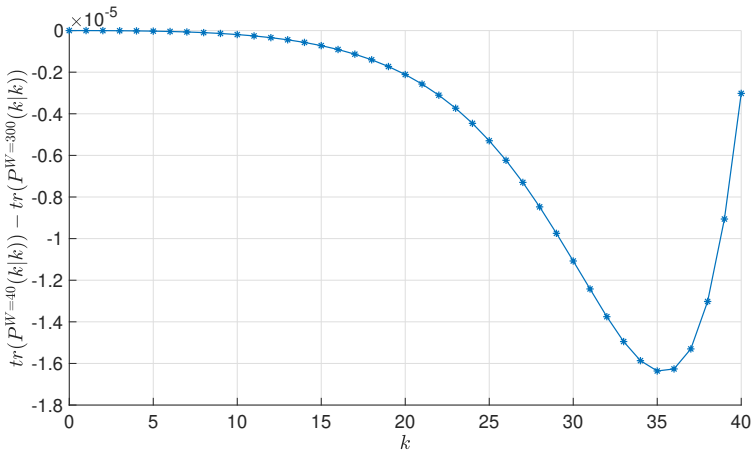
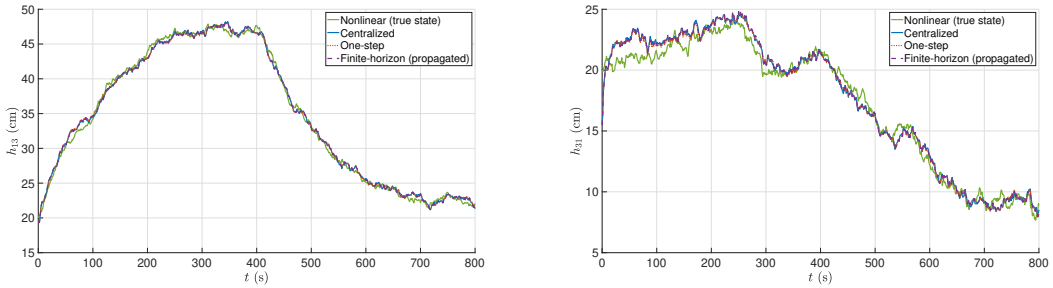


Figure 5. Evolution of the difference between the trace of the covariance of the estimation error of the finite-horizon algorithm for a window $W = 40$ and $W = 300$, for the $N = 40$ tanks network.



(a) Tank 13.

(b) Tank 31.

Figure 6. Evolution of the water level and of the estimates of the different filters, for the $N = 40$ tanks network.

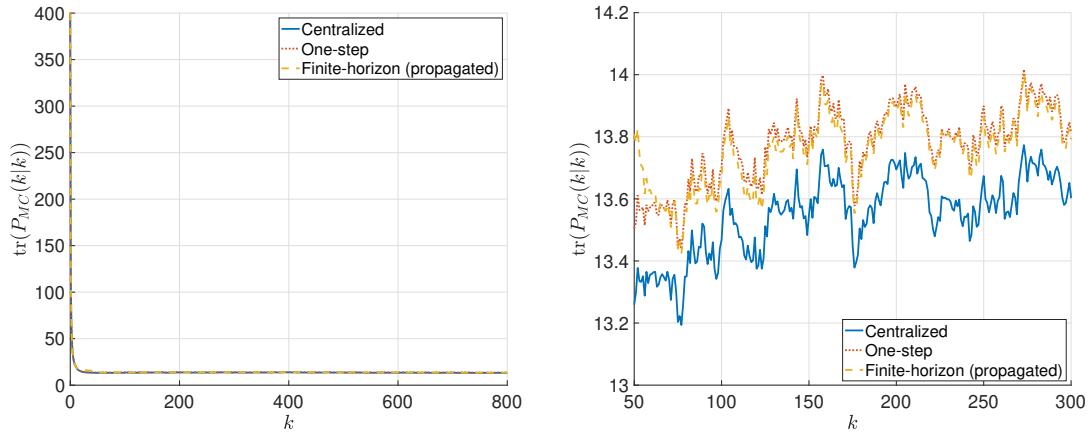


Figure 7. Evolution of the trace of the covariance of the estimation error, obtained with 5000 Monte Carlo simulations.

LTV network with arbitrary time-invariant network configurations, portrayed by the

sparsity constraint that is imposed, were proposed, as a generalisation of previously obtained results for LTI systems. This generalisation allows for a significant widening of the application of both these methods, even to nonlinear systems. First, it was shown that both methods put forward in this paper are able to compute a sequence of well performing stabilising gains subject to an arbitrary sparsity constraint. Second, the one-step method can be computed very efficiently and does not require that the future dynamics of the system are known. Third, the finite-horizon algorithm achieves better performance but it is computationally more expensive and requires that a window of the future dynamics of the system is known. Fourth, both algorithms put forward in this paper were applied to a nonlinear network, whose dynamics were approximated by an LTV network corresponding to successive linearisations about the operations points. It was possible to conclude that, even though the original system is nonlinear, the proposed methods are able to compute well performing gains that stabilise the estimation error dynamics. Fifth, for networks with weak couplings between agents, the one-step and finite-horizon solutions offer a similar performance, even though a very sparse and simple filter configuration was implemented. Sixth, it was possible to show the scalability of both methods, having been implemented to a large-scale system. Finally, note that, when applied to a nonlinear network, the proposed methods require the establishment of additional communication links to perform the linearisation of the system, compared to the application to an LTV network. Further work on this topic should address this drawback.

Acknowledgements

This work was supported by the Fundação para a Ciência e a Tecnologia (FCT) through LARSyS - FCT Project UIDB/50009/2020 and through the FCT project DECENTER [LISBOA-01-0145-FEDER-029605], funded by the Programa Operacional Regional de Lisboa 2020 and PIDDAC programs, and by Macau Science and Technology Development Fund under Grant FDCT/0146/2019/A3, by the Project MYRG2018-00198-FST of the University of Macau and by LARSyS FCT Project UIDB/50009/2020.

Data availability statement

The data that support the findings of this study are openly available in DECENTER toolbox at `https://decenter2021.github.io`, reference number R20210710.

References

Alam, A., Mårtensson, J., & Johansson, K. H. (2015). Experimental evaluation of decentralized cooperative cruise control for heavy-duty vehicle platooning. *Control Engineering Practice*, *38*, 11–25.

Bender, J. (1991, feb). An overview of systems studies of automated highway systems. *IEEE Transactions on Vehicular Technology*, *40*(1), 82–99.

Bereg, S., Díaz-Báñez, J., Lopez, M., Rozario, T., & Valavanis, K. (2015). A decentralized geometric approach for the formation keeping in unmanned aircraft navigation. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)* (p. 989-997).

Blondel, V. D., & Tsitsiklis, J. N. (2000). A survey of computational complexity results in systems and control. *Automatica*, *36*(9), 1249–1274.

Cantoni, M., Weyer, E., Li, Y., Ooi, S. K., Mareels, I., & Ryan, M. (2007). Control of large-scale irrigation networks. *Proceedings of the IEEE*, *95*(1), 75–91.

Curtin, T., Bellingham, J., Catipovic, J., & Webb, D. (1993). Autonomous Oceanographic Sampling Networks. *Oceanography*, *6*(3), 86–94.

Gómez, M., Rodellar, J., & Mantecón, J. A. (2002, nov). Predictive control method for decentralized operation of irrigation canals. *Applied Mathematical Modelling*, *26*(11), 1039–1056.

Healey, A. (2001). Application of formation control for multi-vehicle robotic minesweeping. In *Proceedings of the 40th IEEE Conference on Decision and Control* (Vol. 2, p. 1497-1502).

Heydari, M., & Demetriou, M. A. (2016, nov). Distributed Kalman filters with adaptive strategy for linear time-varying interconnected systems. *International Journal of Adaptive Control and Signal Processing*, *30*(11), 1568–1582.

Ivanov, D., Monakhova, U., & Ovchinnikov, M. (2019). Nanosatellites swarm deployment using decentralized differential drag-based control with communicational constraints. *Acta Astronautica*, *159*, 646–657.

Johansson, K. H. (2000). The quadruple-tank process: A multivariable laboratory process with an adjustable zero. *IEEE Transactions on control systems technology*, *8*(3), 456–465.

Lessard, L., & Lall, S. (2010). Internal quadratic invariance and decentralized control. In *Proceedings of the 2010 American Control Conference* (p. 5596-5601).

Lessard, L., & Lall, S. (2015). Convexity of decentralized controller synthesis. *IEEE Transactions on Automatic Control*, *61*(10), 3122–3127.

Li, Y. (2014). Offtake feedforward compensation for irrigation channels with distributed control. *IEEE Transactions on Control Systems Technology*, *22*(5), 1991–1998.

Mu, J., Yan, X.-G., Spurgeon, S. K., & Zhao, D. (2016). Nonlinear sliding mode control for interconnected systems with application to automated highway systems. *IEEE Transactions on Control of Network Systems*, *5*(1), 664–674.

Pedroso, L., & Batista, P. (2021). Efficient algorithm for the computation of the solution to a sparse matrix equation in distributed control theory. *Mathematics*, *9*(13).

Prodan, I., Lefevre, L., Genon-Catalot, D., et al. (2017). Distributed model predictive control of irrigation systems using cooperative controllers. *IFAC-PapersOnLine*, *50*(1), 6564–6569.

Raffard, R., Tomlin, C., & Boyd, S. (2004). Distributed optimization for cooperative agents: application to formation flight. In *2004 43rd IEEE Conference on Decision and Control (CDC)* (pp. 2453–2459 Vol.3). IEEE.

Russell Carpenter, J. (2002). Decentralized control of satellite formations. *International Journal of Robust and Nonlinear Control*, *12*(2-3), 141–161.

Shaw, G. B. (1999). *The generalized information network analysis methodology for distributed satellite systems* (Unpublished doctoral dissertation). Massachusetts Institute of Technology.

Thien, R. T., & Kim, Y. (2018). Decentralized formation flight via pid and integral sliding mode control. *Aerospace Science and Technology*, *81*, 322–332.

Viegas, D., Batista, P., Oliveira, P., & Silvestre, C. (2012, mar). Decentralized observers for position and velocity estimation in vehicle formations with fixed topologies. *Systems & Control Letters*, *61*(3), 443–453.

Viegas, D., Batista, P., Oliveira, P., & Silvestre, C. (2018, jun). Discrete-time distributed Kalman filter design for formations of autonomous vehicles. *Control Engineering Practice*, *75*, 55–68.

Wallis, W. D. (2010). *A beginner's guide to graph theory*. Springer Science & Business Media.

West, D. B., et al. (1996). *Introduction to graph theory* (Vol. 2). Prentice hall Upper Saddle River, NJ.

Witsenhausen, H. S. (1968). A counterexample in stochastic optimum control. *SIAM Journal on Control*, *6*(1), 131–147.

Wolfe, J., Chichka, D., & Speyer, J. (1996, jul). Decentralized controllers for unmanned

aerial vehicle formation flight. In *Guidance, Navigation, and Control Conference*. Reston, Virigina: American Institute of Aeronautics and Astronautics.

Yanakiev, D., & Kanellakopoulos, I. (1996, jun). A Simplified Framework for String Stability Analysis in AHS 1. *IFAC Proceedings Volumes*, *29*(1), 7873–7878.

Yuan, C., Licht, S., & He, H. (2017). Formation learning control of multiple autonomous underwater vehicles with heterogeneous nonlinear uncertain dynamics. *IEEE Transactions on Cybernetics*(99), 1–15.

Appendix A. Derivation of the closed-form optimal gain for the one-step method

The optimal gain for the one-step method is obtained by solving the optimisation problem

$$\begin{aligned} \underset{\mathbf{K}(k)\in\mathbb{R}^{n\times o}}{\text{minimise}} \quad & \text{tr}(\mathbf{P}(k|k)) \\ \text{subject to} \quad & \mathbf{K}(k) \in \text{Sparse}(\mathbf{E}) \,. \end{aligned}$$

The estimation error covariance for the instant k is given by (10). Taking the derivative of its trace with respect to $\mathbf{K}(k)$ yields

$$\frac{\partial}{\partial \mathbf{K}(k)} \text{tr}(\mathbf{P}(k|k)) = -2\mathbf{P}(k|k-1)\mathbf{C}^T(k) + 2\mathbf{K}(k)\mathbf{S}(k) \,, \tag{A1}$$

where $\mathbf{S}(k)$ is as defined in (14). For each k, (A1) is identical to its time-invariant counterpart, therefore the same techniques, found in Viegas et al. (2018), may be employed to solve $\partial \text{tr}(\mathbf{P}(k|k))/\partial \mathbf{K}(k) = \mathbf{0}$. The solution of this optimisation problem is, thus, given by (13).

Appendix B. Derivation of the closed-form optimal gain for the finite-horizon algorithm

One starts by noticing that the optimisation problem (16) is equivalent to

$$\begin{aligned} \underset{\mathbf{K}(k)\in\mathbb{R}^{n\times o}}{\text{minimise}} \quad & \sum_{i=k}^{W} \text{tr}(\mathbf{P}(i|i)) \\ \text{subject to} \quad & \mathbf{K}(k) \in \text{Sparse}(\mathbf{E}) \,, \end{aligned}$$

since $\mathbf{P}(i|i)$ does not depend on $\mathbf{K}(k)$, for $i < k$. Without loss of generality, assume $\mathbf{A}$ is invertible. Using (9) and (10), for $j \geq k$, the estimation error covariance can be written as

$$\begin{aligned} \mathbf{P}(j|j) = \boldsymbol{\Gamma}(k,j)\mathbf{P}(k-1|k-1)\boldsymbol{\Gamma}^T(k,j) + \sum_{i=k}^{j} \boldsymbol{\Gamma}(i+1,j)\mathbf{K}(i)\mathbf{R}(i)\mathbf{K}^T(i)\boldsymbol{\Gamma}^T(i+1,j) + \\ \sum_{i=k}^{j} \boldsymbol{\Gamma}(i,j)\mathbf{A}^{-1}(j-1)\mathbf{Q}(j-1)\mathbf{A}^{-T}(j-1)\boldsymbol{\Gamma}^T(i,j) \,, \end{aligned}$$

where $\mathbf{\Gamma}(k_i, k_f)$ is defined as in (18). Taking the derivative of the trace of $\mathbf{P}(j|j)$ with respect to $\mathbf{K}(k)$ yields

$$\frac{\partial}{\partial \mathbf{K}(k)} \mathrm{tr}(\mathbf{P}(j|j)) = 2\mathbf{\Gamma}^T(k+1, j)\mathbf{\Gamma}(k+1, j)\left(\mathbf{K}(k)\mathbf{S}(k) - \mathbf{P}(k|k-1)\mathbf{C}^T(k)\right) . \quad \text{(B1)}$$

For each k, (B1) is identical to its time-invariant counterpart, therefore the same techniques, found in Viegas et al. (2018), may be employed to solve

$$\sum_{j=k}^{W} \partial \mathrm{tr}(\mathbf{P}(j|j))/\partial \mathbf{K}(k) = \mathbf{0} . \quad \text{(B2)}$$

The solution of this optimisation problem is, thus, given, in closed-form, by (17), which can be computed efficiently.