Received: 27 February 2021 | Revised: 13 July 2021 | Accepted: 19 August 2021

DOI: 10.1002/rnc.5772

SPECIAL ISSUE ARTICLE

WILEY

Discrete-time decentralized linear quadratic control for linear time-varying systems

Leonardo Pedroso | **Pedro Batista**

Institute for Systems and Robotics, Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal

Correspondence
Leonardo Pedroso, Institute for Systems and Robotics, Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal.
Email: leonardo.pedroso@tecnico.ulisboa.pt

Funding information
Fundação para a Ciência e a Tecnologia (FCT) through LARSyS, Grant/Award Number: UIDB/50009/2020; PIDDAC programs through the FCT project DECENTER, Grant/Award Number: LISBOA-01-0145-FEDER-029605; Programa Operacional Regional de Lisboa 2020 through the FCT project DECENTER, Grant/Award Number: LISBOA-01-0145-FEDER-029605

Abstract

This article addresses the problem of designing a decentralized control solution for a network of agents modeled by linear time-varying (LTV) dynamics, in a discrete-time framework. A general scheme is proposed, in which the problem is formulated as a classical linear quadratic regulator problem, for the global system, subject to a given sparsity constraint on the gain, which reflects the decentralized nature of the network. A method able to compute a sequence of well-performing stabilizing regulator gains is presented and validated resorting to simulations of two randomly generated LTV systems, one stable and the other unstable. Moreover, a tracking solution is developed, building on the solution to the regulator problem. Both methods rely on a closed-form solution, thus they can be computed very rapidly. Similarly to the centralized solution, both the presented methods require that a window of the future system dynamics is known. Both methods are validated resorting to simulations of: (i) a nonlinear network of four interconnected tanks; and (ii) a large-scale nonlinear network of interconnected tanks. When implemented to a nonlinear network, approximated by an LTV system, the proposed methods are able to compute well-performing gains that track the desired output. Finally, both algorithms are scalable, being adequate for implementation in large-scale networks.

KEYWORDS

decentralized control, decentralized linear quadratic regulator, decentralized linear quadratic tracker, interconnected systems, linear time-varying systems

1 | INTRODUCTION

Over the past decades, decentralized control and estimation has been a highly researched topic,[1,2] since it provides a solution to the control and estimation problems of large-scale systems of interconnected agents. 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 decentralized 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,[3-7] unmanned underwater formations,[8-12] satellite constellations,[13-16] automated highway control,[17-20] and irrigation networks,[21-24] which can be modeled as networks of interconnected systems.[25,26]

Although plenty of work has been carried out in decentralized control of linear time-invariant (LTI) systems, the problem of designing such controllers, which consists in solving an optimization problem subject to a constraint that arises from the decentralized nature of the configuration, is extremely difficult[27] and remains an open problem. In fact, the

optimal solution for a linear system may be nonlinear.[28] Furthermore, it has been shown that the solution of a decentralized design control problem is the result of a convex optimization problem if and only if quadratic invariance of the controller set is ensured.[29,30] For these reasons, the overwhelming majority of the approaches found in the literature attempt to find the optimal linear solution, which is also a difficult nonconvex optimization 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 optimization problem by a convex one, which allows to obtain an approximate solution to the original problem. However, such results seldom have stability or boundedness guarantees for the closed-loop system. Another approach is to solve the controller synthesis problem for particular cases, imposing constraints on the decentralized configuration and on the dynamics of the system.

One of the proposed approaches for the design of a decentralized controller for an arbitrary network of interconnected LTI systems is to design an $\mathcal{H}_2$-optimal control policy, which amounts to solving a bilinear matrix inequality.[31-33] Although there are well-known algorithms to solve these problems, their computational load render this solution unfeasible for large-scale systems. Another approach to design a control law for an arbitrary decentralized configuration of an LTI system is to apply convex relaxation to the controller synthesis problem and solving it using well known procedures. In fact, Viegas et al.[34] use this technique to devise two algorithms for the decentralized controller synthesis problems of LTI systems, one of which has a closed-form solution and, thus, is computationally efficient, being suitable for large-scale systems. Moreover, there are also authors that solve the control synthesis problem for directed networks and under constraints on the interconnected systems dynamics such as: (i) dynamically uncoupled systems;[35] and (ii) single and double integrator system dynamics.[36]

The research on decentralized control of linear time-varying (LTV) systems, which is naturally more challenging, has been undergone to a much lesser extent. Even though a large fraction of real-life systems can be modeled as LTI, there are plenty of engineering problems that require an LTV model.[37,38] Furthermore, a nonlinear system can also be approximated by an LTV system carrying out successive linearizations about the operation points.[39] This article aims to address the lack of research into this area. One of the few contributions in this topic is by Farhood et al.[40] In their paper, the finite-horizon regulator problem of an LTV system is reduced to a set of linear matrix inequalities (LMIs). Albeit computationally demanding, numerical algorithms for LMIs are well known.

The main goal of this article is to address the problem of designing a decentralized control solution for a network of agents modeled by LTV dynamics, in a discrete-time framework. The decentralized regulator problem is tackled, in a first instance, and then building on its solution a decentralized tracking controller is designed. A general scheme for the design of decentralized regulators is followed in this article, in which the problem is formulated as a classical optimal linear control problem, for the global system, with a given sparsity constraint on the regulator gain. Such sparsity constraints impose certain entries of the global gain matrix to be null, following a structure that reflects the decentralized nature of the network, necessary for the implementation of the decentralized regulator. It is also assumed that limited communication between agents is possible. This article introduces one method for the computation of decentralized regulator gains for an arbitrary LTV system with an arbitrary time-invariant network configuration, portrayed by a sparsity constraint, which generalizes the one-step method introduced in Reference 34 for LTI systems. Nevertheless, the LTI and LTV decentralized controller design problems have very different natures. On one hand, for the LTI formulation, a steady-state gain that minimizes a cost function over an infinite window is sough. On the other hand, for the LTV formulation, a sequence of gains is sought over a finite-window, which has to be, afterwards, properly extended to solve the infinite horizon problem. As a result, both the cost function and the variables of the optimization problem that arises are very distinct, thus it is not evident how to proceed to generalize this result to LTV systems. For these reasons, the derivation presented in this article is original and it is found to provide additional insight into the LTI formulation of the problem. The classical infinite-horizon optimization problem subject to a sparsity constraint is nonconvex. Hence, the methods that are proposed herein rely on conveniently defined convex relaxations of the original optimization problem to achieve a computationally efficient approximation to its solution. The first method, denoted as the one-step method in the sequel, follows a similar approach as the classical LQR, minimizing at each time step the quadratic cost-to-go. This method is computationally very efficient, exhibiting a closed-form solution, and it does not require any particular initialization. The proposed tracking method is derived relying on the solution to the regulator problem. Finally, both methods are validated resorting to extensive numerical simulations. In addition to two randomly generated systems, one stable and the other unstable, two nonlinear networks of tanks are also considered, one of which has a considerable dimension. A MATLAB implementation of the decentralized algorithms put forward in this article can be found in the *DECENTER* toolbox available at https://decenter2021.github.io (accessed on July 10, 2021).

This article is organized as follows. In Section 2, the control problem is formulated and the assumptions that are considered are introduced. In Sections 3 and 4, the one-step and decentralized tracking methods are derived, respectively. Section 5 details the implementation of the one-step method to a stable and an unstable LTV system, illustrating some details of its application. In Section 6, both methods are applied to a network of four interconnected tanks, which is generalized to N tanks and simulated for $N = 40$, in Section 7. In Section 8, the performance, communication needs, and computational cost of the decentralized solutions are analyzed thoroughly and compared with the centralized solution. Finally, Section 9 presents the main conclusions of this article.

1.1 | Notation

Throughout this article, $\text{sgn}(x)$ denotes the sign of a real number x. The identity and null matrices, both of appropriate dimensions, are denoted by $\mathbf{I}$ and $\mathbf{0}$, respectively. Alternatively, $\mathbf{I}_n$ and $\mathbf{0}_{n\times m}$ are also used to represent the $n \times n$ identity matrix and the $n \times m$ null matrix, respectively. The entry (i, j) of a matrix $\mathbf{A}$ is denoted by $[\mathbf{A}]_{ij}$. The ith component of a vector $\mathbf{v} \in \mathbb{R}^n$ is denoted by $[\mathbf{v}]_i$ and $\text{diag}(\mathbf{v})$ denotes the $n \times n$ square diagonal matrix whose diagonal is $\mathbf{v}$. The vectorization of a matrix $\mathbf{A}$, denoted herein by $\text{vec}(\mathbf{A})$, returns a vector composed of the concatenated columns of $\mathbf{A}$. Given a symmetric matrix $\mathbf{P}$, $\mathbf{P} \succ \mathbf{0}$ and $\mathbf{P} \succcurlyeq \mathbf{0}$ are used to point out that $\mathbf{P}$ is positive definite and positive semidefinite, respectively.

2 | PROBLEM STATEMENT

To model the global dynamics of a network of agents, consider a generic LTV system of the form

$$\begin{cases} \mathbf{x}(k+1) = \mathbf{A}(k)\mathbf{x}(k) + \mathbf{B}(k)\mathbf{u}(k), \\ \mathbf{z}(k) = \mathbf{H}(k)\mathbf{x}(k), \end{cases} \tag{1}$$

where $\mathbf{x}(k) \in \mathbb{R}^n$ is the state vector, $\mathbf{u}(k) \in \mathbb{R}^m$ is the input vector, and $\mathbf{z}(k) \in \mathbb{R}^o$ is the tracking output of the system. $\mathbf{A}(k)$, $\mathbf{B}(k)$, and $\mathbf{H}(k)$ are known time-varying matrices of appropriate dimensions, $\mathbf{A}(k)$ is assumed invertible for $k \in \mathbb{N}_0$, and the pair $(\mathbf{A}(k), \mathbf{B}(k))$ is assumed to be uniformly controllable. This article addresses the problem of designing a decentralized linear quadratic controller for (1). For that reason, the dynamic system (1), used to develop the methods put forward in this article, neither includes a model for the sensors nor takes into account process noise. Instead, the problem is formulated for (1) with initial state $\mathbf{x}(0) = \mathbf{x_0}$ and assuming that full state feedback is available, as in the standard linear quadratic regulator (LQR) formulation.

The first goal of this article is to find a decentralized solution to the LQR problem, that is, finding a sequence of input commands that drive the state of the network to zero in an optimal way, by decentralized linear state feedback. For that reason, the method put forward in this article is devised for the regulator formulation, considering the system dynamics

$$\mathbf{x}(k+1) = \mathbf{A}(k)\mathbf{x}(k) + \mathbf{B}(k)\mathbf{u}(k). \tag{2}$$

However, one is, oftentimes, interested in the tracking problem instead, that is, finding a sequence of input commands that drive the tracking output of the system to follow a reference signal. Nevertheless, such problem can be solved by building on the solution to the regulator problem. This is discussed in Section 4.

In this article, state regulation is proposed considering a controller based on the classical infinite-horizon LQR applied to the global dynamics of the network. Define

$$J_\infty(k) := \sum_{\tau=k}^{\infty} \left(\mathbf{x}^T(\tau)\mathbf{Q}(\tau)\mathbf{x}(\tau) + \mathbf{u}^T(\tau)\mathbf{R}(\tau)\mathbf{u}(\tau)\right)$$

as the infinite-horizon performance index at time instant k, where $\mathbf{Q}(k) \succcurlyeq \mathbf{0}$ and $\mathbf{R}(k) \succ \mathbf{0}$ are known time-varying matrices of appropriate dimensions. Given this quadratic performance index, it is well-known that, even though the optimal action is a linear full state feedback for the centralized formulation,[41] that it not necessarily the case for a decentralized design.[28] Nevertheless, given the extreme difficulty of this problem, the optimal linear command action of the form

$$\mathbf{u}(k) = -\mathbf{K}(k)\mathbf{x}(k) \tag{3}$$

is herein sought instead, where $\mathbf{K}(k) \in \mathbb{R}^{m\times n}$ is the regulator gain, to be determined. Designing a decentralized regulator for a network of agents, whose global dynamics are described by the LTV system (2), is equivalent to constraining the structure of the gain of the global regulator (3). This fact, which follows directly from the decentralized nature of the network, is illustrated for networks of interconnected tanks in Sections 6 and 7. Such constraints fall in a broader category designated by sparsity constraints. Let matrix $\mathbf{E} \in \mathbb{R}^{m\times n}$ denote a sparsity pattern. The set of matrices which obey the sparsity constraint determined by $\mathbf{E}$ is defined as

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

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

$$\begin{aligned} &\underset{\substack{\mathbf{K}(i) \in \mathbb{R}^{m\times n}\\ i \in \mathbb{N}_0}}{\text{minimize}} && J_\infty(0), \\ &\text{subject to} && \mathbf{K}(i) \in \text{Sparse}(\mathbf{E}), \quad i \in \mathbb{N}_0. \end{aligned} \tag{4}$$

Because of the sparsity constraint, the optimization problem (4) is nonconvex and its optimal solution is still an open problem. To overcome this difficulty, the optimization problem may be relaxed so that it becomes convex, allowing for the use of well known optimization 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 it is used to derive the methods put forward in this article.

3 | ONE-STEP METHOD FOR COMPUTATION OF DECENTRALIZED LQR GAINS

The one-step method for computation of the decentralized LQR gains considering a finite-horizon window is derived in this section. It is the LTV counterpart of the one-step method for the regulation of LTI systems, proposed in Reference 34. The proposed method consists of an approximation to the solution of the infinite-horizon problem (4) considering, in a first instance, a finite-horizon problem. The extension of this problem to an infinite-horizon is detailed in Section 3.2.

Define the finite-horizon performance index, over a given finite window $\{k, \dots, k+T\}$, where $T \in \mathbb{N}$, as

$$J(k) := \mathbf{x}^T(k+T)\mathbf{Q}(k+T)\mathbf{x}(k+T) + \sum_{\tau=k}^{k+T-1} \left(\mathbf{x}^T(\tau)\mathbf{Q}(\tau)\mathbf{x}(\tau) + \mathbf{u}^T(\tau)\mathbf{R}(\tau)\mathbf{u}(\tau)\right), \tag{5}$$

where $\mathbf{u}(\tau)$ satisfies (3). The goal is to compute the sequence of controller gains that minimizes $J(k)$. The finite-horizon decentralized LQR problem is thus given by

$$\begin{aligned} &\underset{\substack{\mathbf{K}(\tau) \in \mathbb{R}^{m\times n}\\ \tau = k, \dots, k+T-1}}{\text{minimize}} && J(k), \\ &\text{subject to} && \mathbf{K}(\tau) \in \text{Sparse}(\mathbf{E}), \quad \tau = k, \dots, k+T-1, \end{aligned} \tag{6}$$

for a fixed initial state $\mathbf{x}(k)$. Substituting (5) in the cost function of the relaxed optimization problem (6) yields a nonlinear expression. Even though the sparsity constraint is convex, the optimization problem (6) is nonconvex. Nevertheless, it is

possible to find a sub-optimal solution for this optimization problem, using techniques similar to those used to solve the unconstrained problem, as shown in the following result.

Theorem 1. *Let $\mathbf{l}_j$ denote a column vector whose entries are all set to zero except for the jth one, which is set to 1, and $\boldsymbol{\mathcal{L}}_j := \text{diag}(\mathbf{l}_j)$. Define a vector $\mathbf{m}_j \in \mathbb{R}^m$ to encode the nonzero entries in the jth column of $\mathbf{K}(\tau)$ as*

$$\begin{cases} \mathbf{m}_j(i) = 0, & [\mathbf{E}]_{ij} = 0, \\ \mathbf{m}_j(i) = 1, & [\mathbf{E}]_{ij} \neq 0, \end{cases} \quad i = 1, \ldots, m,$$

and let $\boldsymbol{\mathcal{M}}_j := \text{diag}(\mathbf{m}_j)$. Then, the gain of the one-step sub-optimal solution to (6) is given by

$$\mathbf{K}(\tau) = \sum_{j=1}^{n} \left(\mathbf{I} - \boldsymbol{\mathcal{M}}_j + \boldsymbol{\mathcal{M}}_j \mathbf{S}(\tau) \boldsymbol{\mathcal{M}}_j\right)^{-1} \boldsymbol{\mathcal{M}}_j \mathbf{B}^T(\tau) \mathbf{P}(\tau+1) \mathbf{A}(\tau) \boldsymbol{\mathcal{L}}_j, \tag{7}$$

$\tau = k, \ldots, k+T-1$, where

$$\mathbf{S}(\tau) := \mathbf{B}^T(\tau) \mathbf{P}(\tau+1) \mathbf{B}(\tau) + \mathbf{R}(\tau)$$

and $\mathbf{P}(\tau)$, $\tau = k, \ldots, k+T$, is a symmetric positive semidefinite matrix given by

$$\begin{cases} \mathbf{P}(k+T) = \mathbf{Q}(k+T), \\ \mathbf{P}(\tau) = \mathbf{Q}(\tau) + \mathbf{K}^T(\tau) \mathbf{R}(\tau) \mathbf{K}(\tau) + (\mathbf{A}(\tau) - \mathbf{B}(\tau)\mathbf{K}(\tau))^T \mathbf{P}(\tau+1) (\mathbf{A}(\tau) - \mathbf{B}(\tau)\mathbf{K}(\tau)). \end{cases} \tag{8}$$

Moreover, the one-step sub-optimal solution yields a sub-optimal performance index that follows

$$J^\star(k) = \mathbf{x}^T(k) \mathbf{P}(k) \mathbf{x}(k). \tag{9}$$

Proof. See Appendix A. ∎

Remark 1. As expected, (7) is similar to the solution of the unconstrained LQR, given by

$$\mathbf{K}(\tau) = \mathbf{S}(\tau)^{-1} \mathbf{B}^T(\tau) \mathbf{P}(\tau+1) \mathbf{A}(\tau). \tag{10}$$

The fundamental difference between them is that, imposing the sparsity constraint, the entries of the jth column of $\mathbf{K}(\tau)$ depend on $\boldsymbol{\mathcal{M}}_j \mathbf{S}(\tau) \boldsymbol{\mathcal{M}}_j$, instead of $\mathbf{S}(\tau)$. This reduced form of $\mathbf{S}(\tau)$ corresponds to one of its principal submatrices, in which the ith row and ith column are replaced by zeros if $[\mathbf{E}]_{ij} = 0$ for $i = 1, \ldots, m$. It is important to note that, due to the form of (7), this similarity does not mean that the global decentralized gain is obtained by setting to zero the entries of the unconstrained regulator gain corresponding to the null entries of the sparsity pattern. The solution is, in fact, much more intricate.

Remark 2. The computation of the closed-form solution (7) requires $\mathcal{O}(n^4)$ floating point operations, using Gaussian elimination. Instead of using it, the exact numeric algorithm proposed in Reference 42 can be, alternatively, applied to (A8) 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 decentralized control applications, $|\chi| \approx cn$, where $c \in \mathbb{N}$ is a constant, as it is the case for the networks of tanks simulated in Sections 6 and 7. It, thus, follows that a computational complexity of $\mathcal{O}(n^3)$ is achieved, which is identical to the computational complexity 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 July 10, 2021).

Remark 3. Although sub-optimal for the optimization problem (6), the one-step solution, presented in Theorem 1, is actually the closed-form solution to the following convex optimization problem

$$\begin{aligned} &\underset{\mathbf{K}(\tau) \in \mathbb{R}^{m \times n}}{\text{minimize}} && \text{tr}(\mathbf{P}(k)), \\ &\tau = k, \ldots, k+T-1 \\ &\text{subject to} && \mathbf{K}(\tau) \in \text{Sparse}(\mathbf{E}), \quad \tau = k, \ldots, k+T-1, \end{aligned} \tag{11}$$

as it is proved in Appendix B. Note that, even though (6) and (11) are not, in general, equivalent, (9) suggests that optimization problem (11) is a plausible convex relaxation of the original regulator problem.

Remark 4. It is also interesting to point out that, unlike the dual estimation problem, the sequence of gains that arises in Theorem 1 can only be computed backward in time. For this reason, this method is said to be noncausal, in the sense that, for each time instant, the gain computation requires a window of the future dynamic matrices of the system to be known *a priori*. This is identical to the centralized LQR solution for LTV systems. The application of this algorithm is, thus, 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 prediction algorithm. These approaches are explored: (i) in Section 5 for a synthetic system; and (ii) in Sections 6 and 7 for two nonlinear networks of interconnected tanks. Nonetheless, considering the closed-form solution, presented in Theorem 1, this method allows for the computation of a well-performing sequence of gains for each time window with reduced computational cost.

Remark 5. The proposed derivation of the one-step method for the computation of decentralized LQR gains, presented in Appendix A, follows the Lagrange-multiplier approach. Nevertheless, it is possible to obtain the same result following an approach based on dynamic programming, which offers additional insight into the anatomy of the one-step method. This alternative derivation is presented in Appendix C.

3.1 | Example of gain computation

In this section, the one-step method is applied to two LTV systems, with $n = 4$ and $m = 2$. For the first system, which was chosen to be stable, the matrices, whose constant parts were randomly generated, rounded to 3 decimal places, are given by

$$\mathbf{A}(k) = \begin{bmatrix} -0.348 & -0.422 & -0.495 & -0.416 \\ 0.326 & -0.057 & 0.275 & -0.100 \\ 0.038 & -0.393 & 0.317 & -0.240 \\ 0.496 & 0.462 & 0.369 & 0.300 \end{bmatrix} + \begin{bmatrix} 0 & 0 & \cos(k/10) & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sin^2(k/10) \\ \cos(k/20) & 0 & 0 & 0 \end{bmatrix}, \tag{12a}$$

$$\mathbf{B}(k) = \begin{bmatrix} 0.140 & -0.638 \\ 0.291 & -0.764 \\ 0.447 & -0.515 \\ 0.361 & -0.984 \end{bmatrix} + \begin{bmatrix} \cos(k/5) & 0 \\ \sin(k/10) & 0 \\ \cos(k/13) & 0 \\ 0 & \cos^2(k/20) \end{bmatrix}, \tag{12b}$$

$$\mathbf{Q}(k) = (5 + \sin(k/20))\,\mathbf{I}_n, \tag{12c}$$

$$\mathbf{R}(k) = (5 + \cos(k/20))\,\mathbf{I}_m, \tag{12d}$$

and

$$\mathbf{E} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}. \tag{12e}$$

The second was chosen to be an open-loop unstable system. Its matrices $\mathbf{B}(k)$, $\mathbf{Q}(k)$, $\mathbf{R}(k)$, and $\mathbf{E}$ are equal to those of the open-loop stable system, given, respectively, by (12b)–(12e), and $\mathbf{A}(k)$ by

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

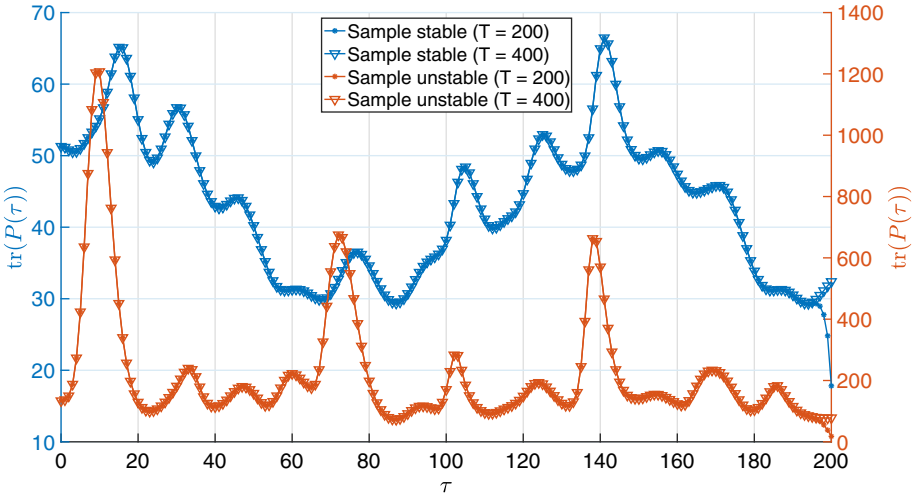


FIGURE 1 Evolution of $\mathrm{tr}(\mathbf{P}(\tau))$ for the one-step method applied to the stable and unstable systems

Figure 1 shows the evolution of $\mathrm{tr}(\mathbf{P}(\tau))$ for the minimization of $J(0)$, using the one-step method, for the stable and unstable systems for two different window lengths T. It is noticeable that none of them grows unbounded with time.

3.2 | Application to the infinite-horizon problem

Given that the gain computation is run backward in time, it is evident that it is unfeasible to make $T \to \infty$ to approximate the solution of the infinite-horizon problem (4), due to the increasing computational load as T becomes large and the fact that it is not possible, in general, to know the dynamics of the network very far into the future. Noticing that

$$J_\infty(k) = \mathbf{x}^T(k)\mathbf{Q}(k)\mathbf{x}(k) + \mathbf{u}^T(k)\mathbf{R}(k)\mathbf{u}(k) + J_\infty(k+1),$$

one may attempt to minimize $J_\infty(k)$ at each time step, instead of $J_\infty(0)$, which, of course, results in a sub-optimal solution. However, the problem corresponding to the minimization of $J_\infty(k)$ is also nonconvex, therefore, the one-step method, developed in this section, is used as an approximation to the original solution.

Given the characteristics of this method, it seems logical to make use of a scheme similar to model predictive control (MPC). One considers a finite window, $\{k, \ldots, k+T\}$, 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. To illustrate this idea, recall Figure 1, where it is possible to compare the evolution of $\mathrm{tr}(\mathbf{P}(\tau))$, for the minimization of $J(0)$, for window lengths $T = 200$ and $T = 400$, for both the stable and unstable systems. It is clear that, apart from the disparity at the end of the window, both solutions are very similar. The minimum window length for convergence varies depending on the system dynamics. For the stable and unstable systems that were considered here, the windows $T = 30$ and $T = 40$, respectively, are sufficient, as seen in Figure 2, in which the evolution of the difference between $\mathrm{tr}(\mathbf{P}(\tau))$ for the chosen window and $T = 400$ is depicted for both systems. With this in mind, at each time instant k, the gains that minimize $J(k)$ are computed for the appropriate window, using the one-step method, and only the first is actually used to compute the control action for that time instant, discarding the remaining gains. At the next time instant, $k+1$, a new finite window is considered and a new sequence of gains is computed to minimize $J(k+1)$, and so forth.

To reduce the computational load, as well as to approach the lower performance index, $J(k)$, that would be obtained if one were to make use of all the gains of the window, d gains may be used, instead of just one, defining a new window and computing the gains associated with it every d time steps. It is, nevertheless, necessary that the sequence of gains that are used in each window does not fall in the region where there is a sudden decrease of $\mathrm{tr}(\mathbf{P}(\tau))$. Although, at first sight, this may seem an advantageous characteristic, the use of the gains near the end of the window, responsible for such a sudden decrease, deteriorates the performance after the transition to the next time window, resulting in a sudden

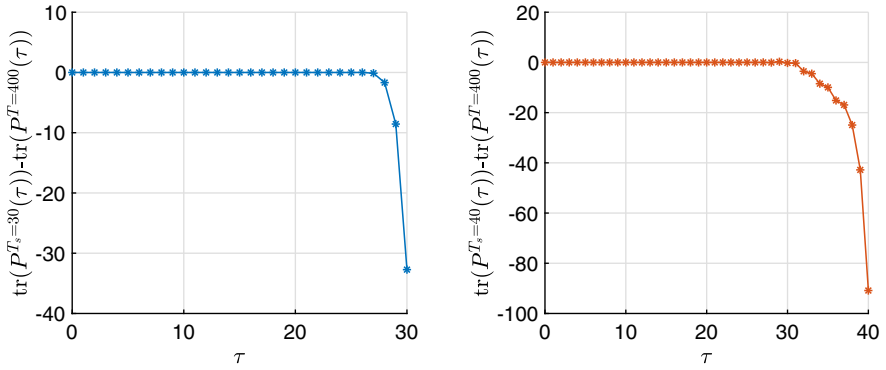


FIGURE 2 Evolution of the difference between $\mathrm{tr}(\mathbf{P}(k|k))$ for $T = 400$, $T_s = 30$ for the stable system (left), and $T_s = 40$ for the unstable system (right)

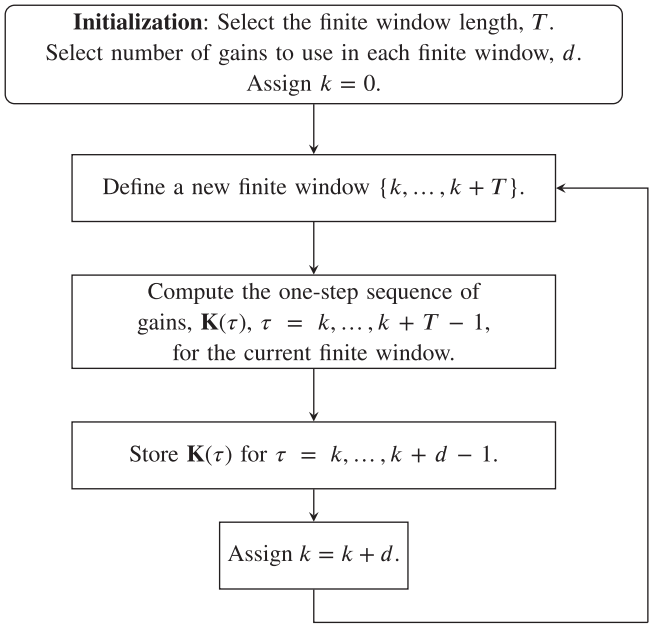


FIGURE 3 Flowchart of the proposed gain computation scheme for the infinite-horizon problem

spike of $\mathrm{tr}(\mathbf{P}(\tau))$. The advantages and drawbacks of the use of more than one gain of each finite window, as well as more details on the practical implementation of this algorithm, are discussed in Sections 5–7. The flowchart of the proposed approach, using more than one gain of each time window, is depicted in Figure 3. In fact, this approach is a very good approximation to the original infinite-horizon problem (4), requiring, nonetheless, a manageable computational load for real-time implementation.

4 | EXTENSION TO THE TRACKING PROBLEM

Although the main concern of this article is the regulator problem, one is, more often than not, interested in tracking a reference signal, $\mathbf{r}(k)$, with the output of the system, $\mathbf{z}(k)$, as given by (1), instead of driving the state of the system to zero using a regulator. This section details an approach to the design of a tracker, suitable both for centralized and decentralized configurations. For a decentralized configuration, the tracker is designed building on the results of the decentralized LQR, put forward in Section 3, that consists of a decentralized regulator of the tracking error dynamics.

4.1 | Tracker design

The proposed tracker consists of a combination of feedforward and feedback terms. For each time instant, consider an equilibrium point consistent with the reference signal as if the system were LTI. The feedforward terms are designed to maintain such equilibrium, for each time instant as if the system were LTI, as well as to ensure the transition to the succeeding equilibrium point, since the system is, in fact, LTV. Writing the dynamics of the system alongside the feedforward terms yields an LTV system for the dynamics of the tracking error. The decentralized LQR, presented in Section 3, is then applied to the error dynamics, from which a feedback term stems. The origin and purpose of each of the different terms is made clearer in the following derivation.

It is well known that for perfect tracking to be possible one needs, in general, as many inputs as the dimension of the vector to track,[43(Theorem3.14)] that is, $m = o$. For this reason, the tracker designed in this article assumes that the reference and input vectors have the same dimension, that is, $\mathbf{r}(k) \in \mathbb{R}^m$. Furthermore, it is assumed that, for $k \in \mathbb{N}_0$, $\mathbf{H}(k)$ has full rank, that is, $\text{rank}(\mathbf{H}(k)) = m$ which is necessary if one aims to follow an arbitrary reference of dimension m.

First, an equilibrium point for time instant k, if the system were LTI, that follows the reference signal $\mathbf{r}(k)$ is sought. To that purpose, define $\overline{\mathbf{x}}(k) \in \mathbb{R}^n$ and $\overline{\mathbf{u}}(k) \in \mathbb{R}^m$ such that

$$\begin{cases} \overline{\mathbf{x}}(k) = \mathbf{A}(k)\overline{\mathbf{x}}(k) + \mathbf{B}(k)\overline{\mathbf{u}}(k), \\ \mathbf{H}(k)\overline{\mathbf{x}}(k) = \mathbf{r}(k) \end{cases} \tag{13}$$

is satisfied for $k \in \mathbb{N}_0$. The augmented matrix of the linear system of equations (13) is given by

$$\left[\begin{array}{cc|c} \mathbf{A}(\tau) - \mathbf{I} & \mathbf{B}(\tau) & \mathbf{0} \\ \mathbf{H}(\tau) & \mathbf{0} & \mathbf{r}(\tau) \end{array}\right]. \tag{14}$$

It is possible to note that, provided that $\mathbf{H}(\tau)$ has full rank, then the rank of the augmented matrix is equal to the rank of the coefficient matrix, that is, the submatrix on the left of (14). It follows directly from Rouch–Capelli theorem[44(Theorem2.38)] that (13) has, at least, one solution $(\overline{\mathbf{x}}(\tau), \overline{\mathbf{u}}(\tau))$. However, the tracker design that is proposed herein encompasses a more integrated approach, as it will be seen shortly, and the solution of (14) is not explicitly shown at this point. The proposed approach aims to drive the system to follow the sequence of pairs $(\overline{\mathbf{x}}(k), \overline{\mathbf{u}}(k))$, penalizing only the error in the tracking space. Define the tracking error in the space of the system state as $\mathbf{e}(k) := \mathbf{x}(k) - \overline{\mathbf{x}}(k)$. Using (1) and (13) allows to write the dynamics of the tracking error as

$$\mathbf{e}(k+1) = \mathbf{A}(k)\mathbf{e}(k) + \mathbf{B}(k)(\mathbf{u}(k) - \overline{\mathbf{u}}(k)) - \left(\overline{\mathbf{x}}(k+1) - \overline{\mathbf{x}}(k)\right). \tag{15}$$

The regulator cannot be applied to (15) because of the presence of the last term. For that reason, one may attempt to write the difference $\overline{\mathbf{x}}(k+1) - \overline{\mathbf{x}}(k)$ as the combination of an additional feedforward control action, $\mathbf{u_a}(k)$, and a disturbance, $\mathbf{d}(k)$, that is,

$$\overline{\mathbf{x}}(k+1) - \overline{\mathbf{x}}(k) = \mathbf{B}(k)\mathbf{u_a}(k) + \mathbf{d}(k), \tag{16}$$

which allows to rewrite the tracking error dynamics (15) as

$$\mathbf{e}(k+1) = \mathbf{A}(k)\mathbf{e}(k) + \mathbf{B}(k)(\mathbf{u}(k) - \overline{\mathbf{u}}(k) - \mathbf{u_a}(k)) - \mathbf{d}(k). \tag{17}$$

Note that the tracking error, $\mathbf{e}(k+1)$, defined in this formulation, is given by the difference between the actual state of the system and $\overline{\mathbf{x}}(k+1)$, not just the difference between the output of the system and the reference signal. For that reason, the main concern is the minimization of the component of the error in the tracking space, that is, the column space of $\mathbf{H}(k+1)$. Therefore, instead of choosing $\mathbf{u_a}(k)$ such that the norm of the disturbance is minimal, it is selected such that the component of the disturbance in the tracking space is minimized. The design of the reference and feedforward terms then takes the form of a quadratic optimization problem with linear equality constraints, for a finite window, given by

$$\begin{aligned} &\underset{\substack{\overline{\mathbf{x}}(\tau),\overline{\mathbf{u}}(\tau),\tau=k,\ \dots\ ,k+T\\ \mathbf{u_a}(\tau),\tau=k,\ \dots\ ,k+T-1}}{\text{minimize}} \quad \sum_{\tau=k}^{k+T-1} ||\mathbf{H}(\tau+1)\mathbf{d}(\tau)||^2, \\ &\text{subject to} \quad \begin{cases} \overline{\mathbf{x}}(\tau)=\mathbf{A}(\tau)\overline{\mathbf{x}}(\tau)+\mathbf{B}(\tau)\overline{\mathbf{u}}(\tau), \\ \mathbf{H}(\tau)\overline{\mathbf{x}}(\tau)=\mathbf{r}(\tau), \end{cases} \quad \tau=k,\ \dots\ ,k+T. \end{aligned} \tag{18}$$

Using (16) to expand the cost function of the optimization problem (18) yields a quadratic expression. Thus, it is possible to find the optimal solution for this optimization problem using well-known optimization techniques, as shown in the following result.

Theorem 2. *There is either one or infinitely many solutions, all globally optimal, to the optimization problem (18), which are given by the solutions to the system of linear equations*

$$\mathbf{G}\overline{\chi}=\overline{\mathbf{r}}, \tag{19}$$

where $\overline{\chi}\in\mathbb{R}^{((3m+2n)T+2m+2n)}$ *corresponds to*

$$\overline{\chi}=\left[\overline{\mathbf{x}}^T(k)\ \overline{\mathbf{u}}^T(k)\ \mathbf{u}_{\mathbf{a}}^T(k)\ \lambda^T(k)\ \boldsymbol{\gamma}^T(k)\ \ \dots\ \ \lambda^T(k+T-1)\ \boldsymbol{\gamma}^T(k+T-1)\ \overline{\mathbf{x}}^T(k+T)\ \overline{\mathbf{u}}^T(k+T)\ \lambda^T(k+T)\ \boldsymbol{\gamma}^T(k+T)\right]^T,$$

where $\lambda(\tau)\in\mathbb{R}^n$ *and* $\boldsymbol{\gamma}(\tau)\in\mathbb{R}^m$ *are Lagrange multipliers. The vector* $\overline{\mathbf{r}}\in\mathbb{R}^{((3m+2n)T+2m+2n)}$ *is written as*

$$\overline{\mathbf{r}}=\left[\mathbf{0}_{1\times(2m+2n)}\quad \mathbf{r}^T(k)\quad \dots\quad \mathbf{0}_{1\times(2m+2n)}\quad \mathbf{r}^T(k+T-1)\quad \mathbf{0}_{1\times(m+2n)}\quad \mathbf{r}^T(k+T)\right]^T,$$

and $\mathbf{G}\in\mathbb{R}^{((3m+2n)T+2m+2n)\times((3m+2n)T+2m+2n)}$ *is a symmetric block tridiagonal matrix given by*

$$\mathbf{G}=\begin{bmatrix} \overline{\boldsymbol{\alpha}}_k & \overline{\boldsymbol{\beta}}_k & & & \mathbf{0} \\ \overline{\boldsymbol{\beta}}_k^T & \overline{\boldsymbol{\alpha}}_{k+1} & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \overline{\boldsymbol{\alpha}}_{k+T-1} & \overline{\boldsymbol{\beta}}_{k+T-1} \\ \mathbf{0} & & & \overline{\boldsymbol{\beta}}_{k+T-1}^T & \overline{\boldsymbol{\alpha}}_{k+T} \end{bmatrix}, \tag{20}$$

with

$$\overline{\boldsymbol{\alpha}}_k=\begin{bmatrix} \mathbf{H}^T(k+1)\mathbf{H}(k+1) & \mathbf{0}_{n\times m} & \mathbf{H}^T(k+1)\mathbf{H}(k+1)\mathbf{B}(k) & \mathbf{A}^T(k)-\mathbf{I} & \mathbf{H}^T(k) \\ \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times m} & \mathbf{B}^T(k) & \mathbf{0}_{m\times m} \\ \mathbf{B}^T(k)\mathbf{H}^T(k+1)\mathbf{H}(k+1) & \mathbf{0}_{m\times m} & \mathbf{B}^T(k)\mathbf{H}^T(k+1)\mathbf{H}(k+1)\mathbf{B}(k) & \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} \\ \mathbf{A}(k)-\mathbf{I} & \mathbf{B}(k) & \mathbf{0}_{n\times m} & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times m} \\ \mathbf{H}(k) & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} \end{bmatrix},$$

$$\overline{\boldsymbol{\alpha}}_\tau=\begin{bmatrix} \mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1)+\mathbf{H}^T(\tau)\mathbf{H}(\tau) & \mathbf{0}_{n\times m} & \mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1)\mathbf{B}(\tau) & \mathbf{A}^T(\tau)-\mathbf{I} & \mathbf{H}^T(\tau) \\ \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times m} & \mathbf{B}^T(\tau) & \mathbf{0}_{m\times m} \\ \mathbf{B}^T(\tau)\mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1) & \mathbf{0}_{m\times m} & \mathbf{B}^T(\tau)\mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1)\mathbf{B}(\tau) & \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} \\ \mathbf{A}(\tau)-\mathbf{I} & \mathbf{B}(\tau) & \mathbf{0}_{n\times m} & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times m} \\ \mathbf{H}(\tau) & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} \end{bmatrix},$$

for $\tau=k+1,\ \dots\ ,k+T-1$,

$$\overline{\boldsymbol{\alpha}}_{k+T} = \begin{bmatrix} \mathbf{H}^T(k+T)\mathbf{H}(k+T) & \mathbf{0}_{n\times m} & \mathbf{A}^T(k+T)-\mathbf{I} & \mathbf{H}^T(k+T) \\ \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} & \mathbf{B}^T(k+T) & \mathbf{0}_{m\times m} \\ \mathbf{A}(k+T)-\mathbf{I} & \mathbf{B}(k+T) & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times m} \\ \mathbf{H}(k+T) & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} \end{bmatrix},$$

and

$$\overline{\boldsymbol{\beta}}_{\tau} = \begin{bmatrix} -\mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1) & \mathbf{0}_{n\times m} & \mathbf{0}_{m\times m} & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times m} \\ \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} \\ -\mathbf{B}^T(\tau)\mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1) & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} \\ \mathbf{0}_{n\times n} & \mathbf{0}_{n\times m} & \mathbf{0}_{n\times m} & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times m} \\ \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times m} & \mathbf{0}_{m\times n} & \mathbf{0}_{m\times m} \end{bmatrix},$$

for $\tau = k, \ldots, k+T-1$.

Proof. See Appendix D. ∎

Remark 6. Note that (13) is a system of linear equations with $n+m$ constraints and $n+m$ unknowns. If $\mathbf{H}(\tau)\mathbf{B}(\tau)$ is invertible, multiplying the first equation of (13) by $\mathbf{H}(\tau)$ and making use of the second equation yields $\overline{\mathbf{u}}(\tau) = (\mathbf{H}(\tau)\mathbf{B}(\tau))^{-1}\left(\mathbf{r}(\tau) - \mathbf{H}(\tau)\mathbf{A}(\tau)\overline{\mathbf{x}}(\tau)\right)$. Substituting this for $\overline{\mathbf{u}}(\tau)$ in the first equation of (13) and solving for $\overline{\mathbf{x}}(\tau)$ it is possible to conclude that, for the particular case for which $\text{rank}(\mathbf{H}(\tau)\mathbf{B}(\tau)) = m$ and $\text{rank}(\mathbf{I} - \mathbf{A}(\tau) + \mathbf{B}(\tau)(\mathbf{H}(\tau)\mathbf{B}(\tau))^{-1}\mathbf{H}(\tau)\mathbf{A}(\tau)) = n$, the solution of (13) is unique and given by

$$\begin{cases} \overline{\mathbf{x}}(\tau) = \left[\mathbf{I} - \mathbf{A}(\tau) + \mathbf{B}(\tau)(\mathbf{H}(\tau)\mathbf{B}(\tau))^{-1}\mathbf{H}(\tau)\mathbf{A}(\tau)\right]^{-1}\mathbf{B}(\tau)(\mathbf{H}(\tau)\mathbf{B}(\tau))^{-1}\mathbf{r}(\tau), \\ \overline{\mathbf{u}}(\tau) = (\mathbf{H}(\tau)\mathbf{B}(\tau))^{-1}\left(\mathbf{r}(\tau) - \mathbf{H}(\tau)\mathbf{A}(\tau)\overline{\mathbf{x}}(\tau)\right). \end{cases} \tag{21}$$

For this reason, the linear equality constraint fully defines $(\overline{\mathbf{x}}(\tau), \overline{\mathbf{u}}(\tau))$ for the time instant τ. Moreover, if $\text{rank}(\mathbf{H}(\tau+1)\mathbf{B}(\tau)) = m$, it is possible to achieve $\mathbf{H}(\tau+1)\mathbf{d}(k) = 0$. The feedforward term $\mathbf{u_a}(k)$ is, thus, computed as follows

$$\begin{aligned} \mathbf{H}(k+1)\left(\overline{\mathbf{x}}(k+1) - \overline{\mathbf{x}}(k)\right) &= \mathbf{H}(k+1)\mathbf{B}(k)\mathbf{u_a}(k) + \mathbf{H}(k+1)\mathbf{d}(k) \\ \Leftrightarrow\ \mathbf{u_a}(k) &= (\mathbf{H}(k+1)\mathbf{B}(k))^{-1}\left(\mathbf{r}(k+1) - \mathbf{H}(k+1)\overline{\mathbf{x}}(k)\right) \end{aligned} \tag{22}$$

Therefore, for every time instant τ for which one or both of these particular cases are verified, the system of linear equations (19) may be reduced, allowing for a decrease in computational load.

Remark 7. The matrix $\mathbf{G}$ in the system of linear equations (19) is a symmetric block tridiagonal matrix. There are plenty of algorithms to solve systems of linear equations featuring such sort of matrices based on cyclic reduction,[45] which allows parallelization.[46]

An LQR, either decentralized using the method put forward in Section 3, or centralized, may be applied to the tracking error dynamics, given by (17), with the weighting matrices selected as $\mathbf{Q}(k) = \mathbf{H}^T(k)\mathbf{Q}_T(k)\mathbf{H}(k)$, to penalize only the error in the tracking space, where $\mathbf{Q}_T(k) \in \mathbb{R}^{m\times m}$ is a positive semidefinite matrix selected to weight the actual output tracking error, $\mathbf{H}(k)\mathbf{x}(k) - \mathbf{r}(k)$, and $\mathbf{R}(k) \in \mathbb{R}^{m\times m}$ is a positive definite matrix selected to weight the feedback term of the control input. The control action for the output tracking problem is, then, given by

$$\mathbf{u}(k) = -\mathbf{K}(k)(\mathbf{x}(k) - \overline{\mathbf{x}}(k)) + \overline{\mathbf{u}}(k) + \mathbf{u_a}(k), \tag{23}$$

where $\mathbf{K}(k)$ is the LQR feedback gain.

Remark 8. It is interesting to remark the effect of each of the three terms that make up the control action (23). First, $\overline{\mathbf{u}}(k)$ is a feedforward term that allows to maintain the output of the system constant and equal to the reference signal of the current time step, $\mathbf{r}(k)$, if the system were LTI, with dynamics defined by $\mathbf{A}(k)$ and $\mathbf{B}(k)$. Second, $\mathbf{u_a}(k)$ is another

feedforward term that compensates the change in the reference signal, which is time-varying, in the following time step. Third, $-\mathbf{K}(k)(\mathbf{x}(k) - \overline{\mathbf{x}}(k))$ is a feedback term that drives to zero the component in the tracking space of the difference between the state of the system and $\overline{\mathbf{x}}(k)$. It is also interesting to point out that, if there is a feasible trajectory that follows the reference signal, then there exists $\mathbf{u_a}(k)$ for which the disturbance $\mathbf{d}(k)$ in (17) is null. Therefore, the feedback term vanishes as time goes to infinity and the actuation tends to be given by both feedforward terms. On top of that, if the reference is a step function, then the term $\mathbf{u_a}(k)$ vanishes as well.

4.2 | Addition of integral action

If the model of the system is exact and the reference signal is a feasible trajectory, then the proposed output tracking design achieves null steady-state error. However, in the overwhelming majority of real scenarios that is not the case. Having that in mind, to improve steady-state performance and add robustness to the controller, an integral action feedback term may be included.

Consider new state variables that correspond to the integral of the tracking error, given by

$$\mathbf{x}_I(k) = \sum_{\tau=0}^{k} (\mathbf{z}(\tau) - \mathbf{r}(\tau)) = \sum_{\tau=0}^{k} \mathbf{H}(\tau)\mathbf{e}(\tau).$$

Then, making use of the error dynamics (17), one can define the augmented system

$$\begin{bmatrix} \mathbf{e}(k+1) \\ \mathbf{x}_I(k+1) \end{bmatrix} = \boldsymbol{\mathcal{A}}(k) \begin{bmatrix} \mathbf{e}(k) \\ \mathbf{x}_I(k) \end{bmatrix} + \boldsymbol{\mathcal{B}}(k) \left(\mathbf{u}(k) - \overline{\mathbf{u}}(k) - \mathbf{u_a}(k) \right) - \begin{bmatrix} \mathbf{I}_n \\ \mathbf{H}(k+1) \end{bmatrix} \mathbf{d}(k), \tag{24}$$

where

$$\boldsymbol{\mathcal{A}}(k) = \begin{bmatrix} \mathbf{A}(k) & \mathbf{0} \\ \mathbf{H}(k+1)\mathbf{A}(k) & \mathbf{I} \end{bmatrix} \quad \text{and} \quad \boldsymbol{\mathcal{B}}(k) = \begin{bmatrix} \mathbf{B}(k) \\ \mathbf{H}(k+1)\mathbf{B}(k) \end{bmatrix}.$$

One can, now, add the integral action feedback term applying the LQR to the augmented system (24) with weighting matrices $\boldsymbol{\mathcal{Q}}(k)$ and $\boldsymbol{\mathcal{R}}(k)$ given by

$$\boldsymbol{\mathcal{Q}}(k) = \begin{bmatrix} \mathbf{H}^T(k)\mathbf{Q}_T(k)\mathbf{H}(k) & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_I(k) \end{bmatrix} \quad \text{and} \quad \boldsymbol{\mathcal{R}}(k) = \mathbf{R}(k),$$

where $\mathbf{Q}_I(k) \in \mathbb{R}^{m \times m}$ is the positive semidefinite matrix that weights the integral of the tracking error. Note, again, that this procedure is suitable both for a centralized configuration and for a decentralized configuration using the LQR design synthesis method put forward in Section 3. The resulting gain matrix has the form $\boldsymbol{\mathcal{K}}(k) = \begin{bmatrix} \mathbf{K}(k) & \mathbf{K}_I(k) \end{bmatrix} \in \mathbb{R}^{m \times (n+m)}$, where $\mathbf{K}(k)$ and $\mathbf{K}_I(k) \in \mathbb{R}^{m \times m}$ are the error and integral action feedback gains, respectively. The control input of the proposed approach is, thus, given by

$$\mathbf{u}(k) = -\mathbf{K}(k)(\mathbf{x}(k) - \overline{\mathbf{x}}(k)) - \mathbf{K}_I(k)\mathbf{x}_I(k) + \overline{\mathbf{u}}(k) + \mathbf{u_a}(k). \tag{25}$$

If there is a sudden significant change in the reference signal that cannot be attained by the system, the tracking error is significant until the controller has enough time to drive the output of the system to the reference. During that period, the integral states keep accumulating the tracking error, which leads to what is known as integral windup, resulting in a significant overshoot until the integral states decrease, in absolute value, to normal operation values. In this design, the integral action term is only included to provide for better performance after the transient state, rejecting disturbances. Thus, a very simple and effective anti-windup technique for this formulation is to saturate the integral state variables and disable the integrators while they are saturated. That is, the integral state is subject to an entrywise saturation

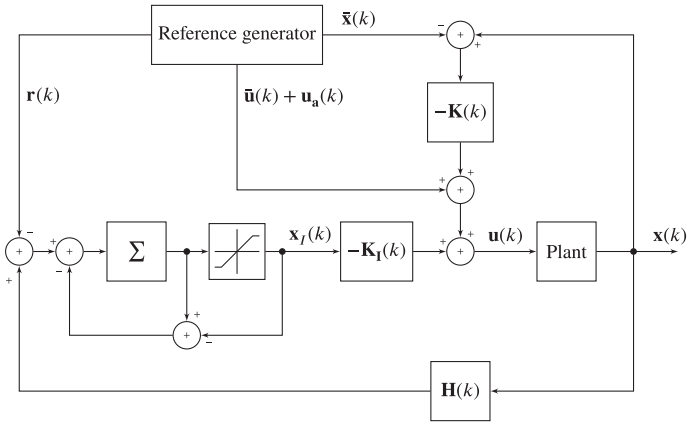


FIGURE 4 Block diagram of the proposed tracking system, with anti-windup integral action

$$\left[\mathbf{x}_I(k)\right]_i = \begin{cases} \left[\mathbf{x}_I(k)\right]_i, & \left|\left[\mathbf{x}_I(k)\right]_i\right| \leq \left[\mathbf{x}_I^{\text{sat}}\right]_i, \\ \left[\mathbf{x}_I^{\text{sat}}\right]_i \operatorname{sgn}\left(\left[\mathbf{x}_I(k)\right]_i\right), & \left|\left[\mathbf{x}_I(k)\right]_i\right| > \left[\mathbf{x}_I^{\text{sat}}\right]_i, \end{cases} \quad i = 1, \ldots, m,$$

where $\mathbf{x}_I^{\text{sat}} \in \mathbb{R}^m$ is a constant vector that holds the saturation limits of each of the corresponding entries of $\mathbf{x}_I(k)$. To disable the integrator whose state is saturated, one can subtract the difference between the integral state before and after the saturation from the corresponding integral state. The drawback of this anti-windup technique is that the magnitude of the perturbations that the controller is able to reject is limited by the chosen saturation limits. For this reason, the entries of $\mathbf{x}_I^{\text{sat}}$ are selected to be the minimum possible that still allow for the rejection of the perturbations with the expected highest magnitude the system is subject to. Figure 4 depicts the block diagram of the proposed tracking system.

5 | SIMULATION RESULTS FOR THE SYNTHETIC STABLE AND UNSTABLE SYSTEMS

In this section, the decentralized LQR proposed in Section 3.1 is simulated and its performance is compared with the centralized solution. For the stable and unstable systems, a finite window was considered, as discussed in Section 3.2, with $T = 30$ and $T = 40$, respectively. Given that, in this case, the dynamics of the systems are known in the future without uncertainty, more than one of the computed gains of each window may be used without a significant loss of performance. For that reason, both systems are also simulated making use of $d = 20$ and $d = 25$ gains of each window, respectively, for the stable and unstable systems. This choice allows for a difference, in relation to the use of an arbitrarily large window, that is at least 4 orders of magnitude below the magnitude of $\text{tr}(\mathbf{P}(\tau))$. Performing the same analysis as in Section 3.2 to the centralized method, one concludes that the use of a window $T = 20$ of which $d = 13$ and $d = 12$ gains computed for each window are used, for the stable and unstable systems, respectively, allows for a difference, in relation to the use of an arbitrarily large window, that is at least 4 orders of magnitude below the magnitude of $\text{tr}(\mathbf{P}(\tau))$.

Monte Carlo simulations were carried out to assess the performance of the proposed solution. For each run, the initial state, $\mathbf{x_0}$, was randomly selected from a zero-mean normal distribution with covariance $\mathbf{I}_n$. Tables 1 and 2 show the mean performance index, $J(0)$, for 20,000 Monte Carlo runs, for the stable and unstable system, respectively. First, as expected, the performance of the one-step method is inferior to the centralized solution. Second, one can conclude that both the centralized and one-step methods, applied to an infinite-horizon, using d of the gains computed for each window, yield similar performance to the simulations using just one. Given that for each time instant, the dynamics of the system are known without any uncertainty in the future and that the parameter d was chosen thoughtfully, the use of more than one gain does not introduce any significant loss of performance, thereby allowing for a considerable decrease in computational load. It is interesting to remark that for the stable system, unlike the unstable system, the use of more than one gain actually improves slightly the performance index. In fact, in this case the loss of performance caused by the approximation

TABLE 1 Comparison of the performance index for the centralized and decentralized LQR, for the stable system

	Centralized ($d = 1$)	Centralized ($d = 13$)	Decentralized ($d = 1$)	Decentralized ($d = 20$)
Mean($J(0)$)	38.64	37.97	41.28	41.16

TABLE 2 Comparison of the performance index for the centralized and decentralized LQR, for the unstable system

	Centralized ($d = 1$)	Centralized ($d = 12$)	Decentralized ($d = 1$)	Decentralized ($d = 25$)
Mean($J(0)$)	72.32	72.57	84.75	85.15

of the infinite-horizon problem using an finite-horizon window is compensated by the use of more gains of the solution to the finite-horizon problem, which allows for a decrease of the performance index.

6 | SIMULATION RESULTS FOR A QUADRUPLE-TANK NETWORK

In this section, the methods put forward in this article are applied to a network of four tanks, as a means of assessing their performance. Given that the dynamics of the projected system are nonlinear, to employ the methods devised one can approximate its behavior by an LTV system, linearizing and discretizing its dynamics about successive equilibrium points. For this reason, it allows to assess the performance of the proposed decentralized estimation method when implemented in nonlinear time-varying systems. The quadruple-tank network introduced in Reference 47 inspired the example showed herein.

6.1 | Quadruple-tank network dynamics

Consider four interconnected tanks as shown in Figure 5. The water levels of tank 1 to tank 4 are denoted by h_1, h_2, h_3, and h_4. The network is controlled by two pumps, whose inputs are denoted by u_1 and u_2, which are controlled by tank 1 and tank 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. Making use of mass balances and Bernoulli's law, the system dynamics, in the absence of noise, are given by

$$\begin{cases} \dot{h}_1(t) = -\frac{a_1}{A_1}\sqrt{2gh_1(t)} + \frac{a_3}{A_1}\sqrt{2gh_3(t)} + \frac{\gamma_1 k_1}{A_1}u_1(t), \\ \dot{h}_2(t) = -\frac{a_2}{A_2}\sqrt{2gh_2(t)} + \frac{a_4}{A_2}\sqrt{2gh_4(t)} + \frac{\gamma_2 k_2}{A_2}u_2(t), \\ \dot{h}_3(t) = -\frac{a_3}{A_3}\sqrt{2gh_3(t)} + \frac{(1-\gamma_2)k_2}{A_3}u_2(t), \\ \dot{h}_4(t) = -\frac{a_4}{A_4}\sqrt{2gh_4(t)} + \frac{(1-\gamma_1)k_1}{A_4}u_1(t), \end{cases} \tag{26}$$

where A_i and a_i are the cross sections of tank i and of its outlet hole, respectively; constants γ_1 and γ_2 represent the fraction of the flow that passes through the valves to the lower tanks; k_1 and k_2 are the constants of proportionality between the mass flow and the input for each pump; and g denotes the acceleration of gravity. Furthermore, the input of each pump is subject to a hard constraint $u_{1,2} \in [0, u^{\text{sat}}]$, where $u^{\text{sat}} \in \mathbb{R}^+$.

The nonlinear dynamics (26) can be linearized about a given equilibrium point, characterized by equilibrium water levels, h_1^0, h_2^0, h_3^0, h_4^0, and corresponding inputs, u_1^0 and u_2^0. Writing the state and control vectors, respectively, as

$$\mathbf{x_c}(t) = \begin{bmatrix} h_1(t) - h_1^0 \\ h_2(t) - h_2^0 \\ h_3(t) - h_3^0 \\ h_4(t) - h_4^0 \end{bmatrix} \quad \text{and} \quad \mathbf{u_c}(t) = \begin{bmatrix} u_1(t) - u_1^0 \\ u_2(t) - u_2^0 \end{bmatrix},$$

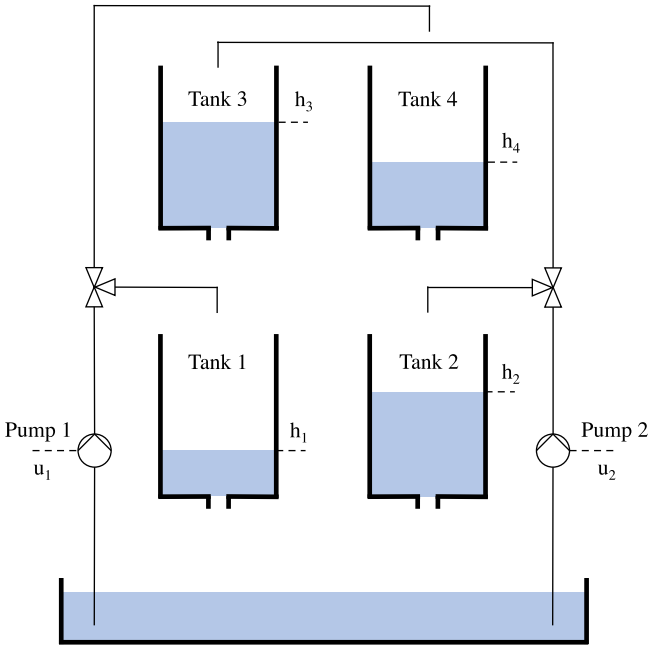


FIGURE 5 Schematic of a quadruple-tank network

the continuous-time linearized system dynamics are modeled by

$$\dot{\mathbf{x}}_\mathbf{c}(t) = \mathbf{A}_\mathbf{c}(t)\mathbf{x}_\mathbf{c}(t) + \mathbf{B}_\mathbf{c}(t)\mathbf{u}_\mathbf{c}(t), \tag{27}$$

with

$$\mathbf{A}_\mathbf{c}(t) = \begin{bmatrix} -\frac{1}{T_1} & 0 & \frac{A_3}{A_1 T_3} & 0 \\ 0 & -\frac{1}{T_2} & 0 & \frac{A_4}{A_2 T_4} \\ 0 & 0 & -\frac{1}{T_3} & 0 \\ 0 & 0 & 0 & -\frac{1}{T_4} \end{bmatrix}$$

and

$$\mathbf{B}_\mathbf{c}(t) = \begin{bmatrix} \frac{\gamma_1 k_1}{A_1} & 0 \\ 0 & \frac{\gamma_2 k_2}{A_2} \\ 0 & \frac{(1-\gamma_2)k_2}{A_3} \\ \frac{(1-\gamma_1)k_1}{A_4} & 0 \end{bmatrix},$$

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

$$T_i = \frac{A_i}{a_i}\sqrt{\frac{2h_i^0}{g}}. \tag{28}$$

Provided that this system is slow, one can assume that both the water level measurements and control inputs are updated with a constant period T. Under this assumption, the discretization of (27) yields

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

where

$$\mathbf{A}(k) = e^{\mathbf{A_c}(kT)T}, \tag{29a}$$

$$\mathbf{B}(k) = \left(\int_0^T e^{\mathbf{A_c}(kT)\tau} d\tau \right) \mathbf{B_c}(kT), \tag{29b}$$

and

$$\mathbf{u}(k) = \mathbf{u_c}(kT). \tag{29c}$$

It is important to remark that, to perform the linearization, each local controller ought to access the necessary variables of the network that define the equilibrium point, through communication. Provided that this network varies slowly, it is not necessary to perform the linearization at every time instant, thereby reducing the computational load and communication needs. Instead, it may be updated with a given periodicity, $T_{lin} = qT$, where q is an integer number.

6.2 | Controller implementation

The problem considered for this network is the design of a decentralized solution to control the water level of the lower tanks. For that reason, the output of the network is computed as in (1), with

$$\mathbf{H}(k) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

Analyzing the system dynamics one notices that, for any time instant k, an equilibrium point corresponds to the solution of a system of four equations with six unknowns, $h_1(k), h_2(k), h_3(k), h_4(k), u_1(k)$, and $u_2(k)$. It is, thus, necessary to select two of these variables to define an equilibrium point, which, for this simulation, were chosen to be the water levels of the lower tanks, $h_1(k)$ and $h_2(k)$, as a means of ensuring that the system dynamics used for their control are as accurate as possible. For this reason, each time a new linearization is performed, every local controller has to receive through communication the water level in the lower tanks, compute the remaining variables that define the equilibrium point, and linearize the relevant entries of the matrix $\mathbf{A}(k)$ about that point. Although measurements of the water level of the upper tanks could be available to the lower tanks via a communication link between them, to reduce the need for such links, a fully decentralized network is considered instead. In fact, the approach followed consists of a local controller in each of the lower tanks, which computes the control action of the associated pump, making use of the measurement of its own water level only. The control action of the decentralized controllers is computed using the tracker design proposed in Section 4, that makes use of the one-step method developed in Section 3 to compute the feedback gains. Given that the reference signal one desires to track in the simulations is not a feasible trajectory, an integral feedback term is also included alongside an anti-windup technique, as detailed in Section 4.2, to improve the steady-state performance. Thus, for the linearized system, each local controller computes an input of the form

$$\begin{cases} \left[\mathbf{u}(k)\right]_1 = -\left[\mathbf{K}(k)\right]_{11} \left(\left[\mathbf{x}(k)\right]_1 - \left[\overline{\mathbf{x}}(k)\right]_1 \right) - \left[\mathbf{K}_I(k)\right]_{11} \left[\mathbf{x}_I(k)\right]_1 + \left[\overline{\mathbf{u}}(k)\right]_1 + \left[\mathbf{u_a}(k)\right]_1, \\ \left[\mathbf{u}(k)\right]_2 = -\left[\mathbf{K}(k)\right]_{22} \left(\left[\mathbf{x}(k)\right]_2 - \left[\overline{\mathbf{x}}(k)\right]_2 \right) - \left[\mathbf{K}_I(k)\right]_{22} \left[\mathbf{x}_I(k)\right]_2 + \left[\overline{\mathbf{u}}(k)\right]_2 + \left[\mathbf{u_a}(k)\right]_2, \end{cases}$$

in such a way that each tank has only access to its measured water level and to the integral of its tracking error. Note that $u_i(k)$ and $\left[\mathbf{u}(k)\right]_i$ are distinct. Indeed, the former is the input to pump i and the latter is the ith component of $\mathbf{u}(k)$. It is important to remark that the dynamics of the quadruple-tank network verify the two conditions explored in Remark 6 for every time instant. Thus, the reference values $(\overline{\mathbf{x}}(k), \overline{\mathbf{u}}(k), \mathbf{u_a}(k))$ can be computed independently using (21) and (22) for each time instant. Furthermore, $(\overline{\mathbf{x}}(k), \overline{\mathbf{u}}(k))$ does not have, necessarily, to be calculated with (21) using the linearized system. In fact, the analogous nonlinear equation is

$$\begin{cases} -\frac{a_1}{A_1}\sqrt{2g\ \ \overline{h}_1(k)} + \frac{a_3}{A_1}\sqrt{2g\ \ \overline{h}_3(k)} + \frac{\gamma_1 k_1}{A_1}\bar{u}_1(k) = 0, \\ -\frac{a_2}{A_2}\sqrt{2g\ \ \overline{h}_2(k)} + \frac{a_4}{A_2}\sqrt{2g\ \ \overline{h}_4(k)} + \frac{\gamma_2 k_2}{A_2}\bar{u}_2(k) = 0, \\ -\frac{a_3}{A_3}\sqrt{2g\ \ \overline{h}_3(k)} + \frac{(1-\gamma_2)k_2}{A_3}\bar{u}_2(k) = 0, \\ -\frac{a_4}{A_4}\sqrt{2g\ \ \overline{h}_4(k)} + \frac{(1-\gamma_1)k_1}{A_4}\bar{u}_1(k) = 0, \\ \begin{bmatrix} \overline{h}_1(k) & \overline{h}_2(k) \end{bmatrix}^T = \mathbf{r}(k), \end{cases} \tag{30}$$

with

$$\overline{\mathbf{x}}(k) = \begin{bmatrix} \overline{h}_1(k) - h_1^0(k) \\ \overline{h}_2(k) - h_2^0(k) \\ \overline{h}_3(k) - h_3^0(k) \\ \overline{h}_4(k) - h_4^0(k) \end{bmatrix}, \quad \overline{\mathbf{u}}(k) = \begin{bmatrix} \bar{u}_1(k) - u_1^0(k) \\ \bar{u}_2(k) - u_2^0(k) \end{bmatrix},$$

where $h_i^0(k)$ and $u_j^0(k)$ are the equilibrium water level of tank i and equilibrium input of pump j, respectively, computed in the last linearization prior to the time instant k. The unique closed-form solution to (30) is given by

$$\begin{cases} \begin{bmatrix} \overline{h}_1(k) & \overline{h}_2(k) \end{bmatrix}^T = \mathbf{r}(k), \\ \begin{bmatrix} \overline{h}_3(k) & \overline{h}_4(k) \end{bmatrix}^T = \boldsymbol{\alpha} \begin{bmatrix} \left[\mathbf{r}(k)\right]_1 & \left[\mathbf{r}(k)\right]_2 & \sqrt{\left[\mathbf{r}(k)\right]_1 \left[\mathbf{r}(k)\right]_2} \end{bmatrix}^T, \\ \begin{bmatrix} \bar{u}_1(k) & \bar{u}_2(k) \end{bmatrix}^T = \boldsymbol{\beta} \begin{bmatrix} \sqrt{\left[\mathbf{r}(k)\right]_1} & \sqrt{\left[\mathbf{r}(k)\right]_2} \end{bmatrix}^T, \end{cases} \tag{31}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^{2\times 3}$ and $\boldsymbol{\beta} \in \mathbb{R}^{2\times 2}$ are constant matrices, whose entries are a function of the physical parameters of the network exclusively. It is also interesting to point out that (31) can also be used to compute the equilibrium point for each linearization, setting $\mathbf{r}(k) = \begin{bmatrix} h_1(k) & h_2(k) \end{bmatrix}^T$, given that the water levels of the lower tanks are the variables chosen to define the equilibrium point. Note that the use of the nonlinear equilibrium solution (31) to find $\overline{\mathbf{x}}(k)$ and $\overline{\mathbf{u}}(k)$ is not only computationally efficient but also much more accurate because it does not rely on the linearization, which is only updated every q time steps. The actual pump input is, thus, given by

$$\begin{cases} u_1(k) = -\left[\mathbf{K}(k)\right]_{11} \left(h_1(k) - \overline{h}_1(k)\right) - \left[\mathbf{K}_I(k)\right]_{11} \left[\mathbf{x}_I(k)\right]_1 + \left[\overline{\mathbf{u}}(k)\right]_1 + \left[\mathbf{u}_\mathbf{a}(k)\right]_1 + u_1^0(k), \\ u_2(k) = -\left[\mathbf{K}(k)\right]_{22} \left(h_2(k) - \overline{h}_2(k)\right) - \left[\mathbf{K}_I(k)\right]_{22} \left[\mathbf{x}_I(k)\right]_2 + \left[\overline{\mathbf{u}}(k)\right]_2 + \left[\mathbf{u}_\mathbf{a}(k)\right]_2 + u_2^0(k). \end{cases} \tag{32}$$

Although the controller design put forward is not projected to handle hard input constraints, as this network requires, to meet those constraints on the control inputs of the pumps, the computed inputs, given by (32), are saturated.

Comparing the local controller (32) with the global controller of the network (25), it follows that the feedback gain of the augmented system, $\boldsymbol{\mathcal{K}}(k) = \begin{bmatrix} \mathbf{K}(k) & \mathbf{K}_I(k) \end{bmatrix}$, must follow the sparsity pattern defined by

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

Given that both the centralized and one-step methods require 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 linearized dynamics. For that reason, a very simple iterative LQR smoothing (iLQR) scheme, based on References 48 and 49, is used. This approach is an iterative algorithm, applied to a finite window T, as

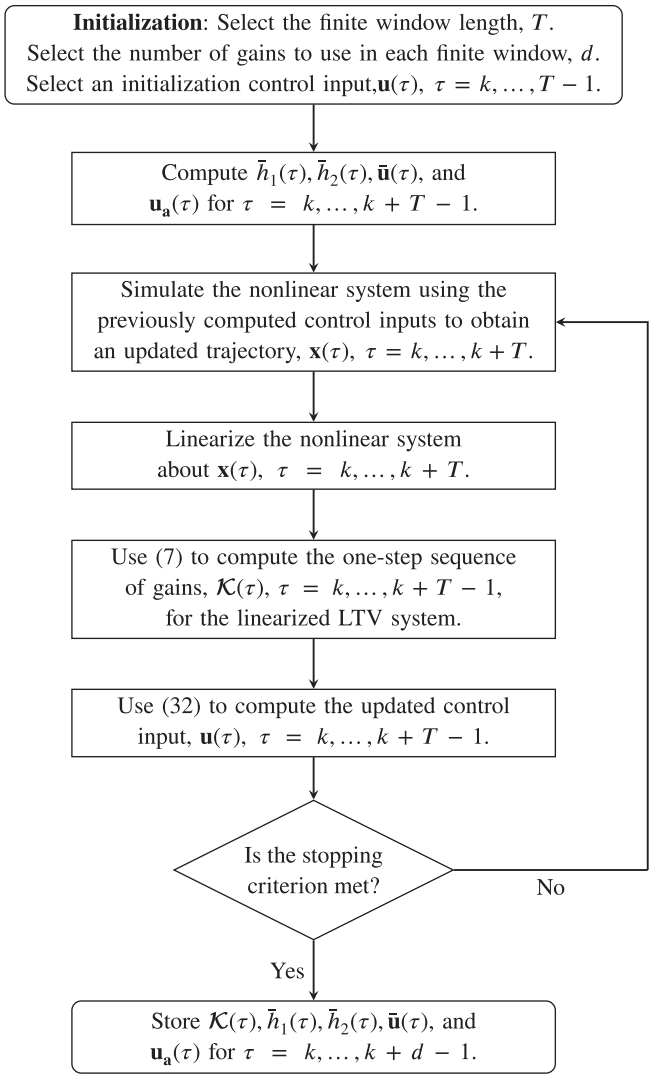


FIGURE 6 Flowchart of the iLQR scheme used for the gain computation for a finite window

described in Section 3.2, which consists of backward and forward passes carried out in turns, as a means of computing the control action for the nonlinear system. For the initialization of the algorithm, start with an initial control input and the resulting trajectory for the window considered, that is, $\mathbf{u}(\tau)$ for $\tau = k, \dots, k+T-1$ and $\mathbf{x}(\tau)$ for $\tau = k, \dots, k+T$. From this initial trajectory, one can linearize the system dynamics around successive equilibrium points and find an LTV system that approximates the nonlinear system throughout the window. In this particular case, for the initialization, an initial control input was chosen such that the water level in the lower tanks remains constant throughout the window, which approximates the nonlinear dynamics by an LTI system, due to the way the linearization was defined. Then run the backward and forward passes, in this order, in turns, until convergence is reached. The backward pass consists of the computation of the control action, using (32), considering the approximate LTV system given by the previous forward pass. Note that $\bar{h}_1(\tau), \bar{h}_2(\tau), \bar{\mathbf{u}}(\tau)$, and $\mathbf{u_a}(\tau)$ for $\tau = k, \dots, k+T-1$, only have to be computed once, since they do not depend on the evolution of the system. The forward pass is the simulation of the nonlinear system using the control action computed in the previous backward pass, whose trajectory is used to update the LTV system that approximates the dynamics of the nonlinear system throughout the window. The algorithm stops when the maximum relative difference of the norm of the actuation throughout the window, computed in two consecutive iterations, is below 10^{-4}. The linearizations performed in each iteration are also carried out with a periodicity T_{lin}. After convergence has been reached, one can make use of the first d gains of the window. A new window $\{k+d, \dots, k+d+T-1\}$ is then defined and so forth. A flowchart of the iLQR scheme used for the gain computation for a finite window is presented in Figure 6.

TABLE 3 Values of the physical constants of the quadruple-tank network

Constant	Value
A_1, A_3	28 cm^2
A_2, A_4	32 cm^2
a_1	0.071 cm^2
a_2	0.057 cm^2
a_3, a_4	0.040 cm^2
g	981 $cm\,s^{-2}$
k_1, k_2	3.33 $cm^3\,s^{-1}\,V^{-1}$
γ_1	0.7
γ_2	0.6
u^{sat}	12 V

6.3 | Simulation results

The values of the physical constants of the quadruple-tank network are presented in Table 3. For those parameters, the constant matrices that are used to compute the equilibrium points for each linearization, as well as for part of the feedforward control action, using (31), rounded to 3 decimal places, are given by

$$\boldsymbol{\alpha} = \begin{bmatrix} 0.5041 & 1.769 & -1.889 \\ 1.134 & 0.3249 & -1.214 \end{bmatrix} \quad \text{and} \quad \boldsymbol{\beta} = \begin{bmatrix} 1.889 & -1.011 \\ -0.9444 & 1.769 \end{bmatrix}.$$

The sampling time was set to $T = 1\,\text{s}$ and the linearization period to $T_{lin} = 10T$. The initial level of the tanks is set to $h_1 = h_2 = h_3 = h_4 = 20\,\text{cm}$. The reference signal is given by

$$[\mathbf{r}(t)]_1 = \begin{cases} 30, & t \in [0{,}200\text{ s}[\quad \vee \quad t \geq 400\text{ s}, \\ 20, & t \in [200\text{ s}, 400\text{ s}[, \end{cases} \qquad [\mathbf{r}(t)]_2 = 30 + 10\cos(t/35),$$

which is sampled with a periodicity of T.

As detailed in Section 3.2, the infinite-horizon solution is approximated by the solution of several finite-horizon problems on a chosen window. Its length was selected to be $T = 30$, roughly twice the time constant of the slowest pole of the system, which is enough for the centralized and one-step methods to approximate the infinite-horizon solution. The proposed method is compared with the analogous centralized solution, that is, using the unconstrained LQR solution to compute the LQR feedback gains of (32). Due to the nonlinear dynamics of the network, the controller parameters were tuned empirically. The weighting matrices were set to $\mathbf{R}(k) = \mathbf{I}_2$, $\mathbf{Q}_T = 20\mathbf{I}_2$, and $\mathbf{Q}_I = 0.05\mathbf{I}_2$, and the integral state saturation limits vector was chosen to be $\mathbf{x}_I^{\text{sat}} = [10 \;\; 10]^T\text{cm}$, for both methods.

Figure 7 depicts the nonlinear simulation of the water level of the four tanks, as well the reference signal, for the centralized and one-step methods. The simulation was also carried out making use of $d = 15$ and $d = 10$ of the gains computed for each window instead of just one, for the centralized and decentralized solutions, respectively. Figure 8 depicts the evolution of the input to the pumps. First, it is clear that none of the solutions diverge and all drive the system state very close to the reference signal, yielding very good performance. Note that the system is nonlinear and the control inputs are subject to hard constraints, none of which were taken into account in the design of the decentralized controller. Moreover, the variation of water level in the tanks is considerable, which allows for a significant variation in the dynamics of the system throughout the simulation. Second, it is interesting to notice that, despite having considered a fully decentralized network design, the performance obtained with the centralized solution is not considerably better, which makes the use of a decentralized solution very compelling for this network. Third, regarding the MPC scheme used, it is clear that the use of more than one gain of each finite window does not introduce a noticeable loss of performance,

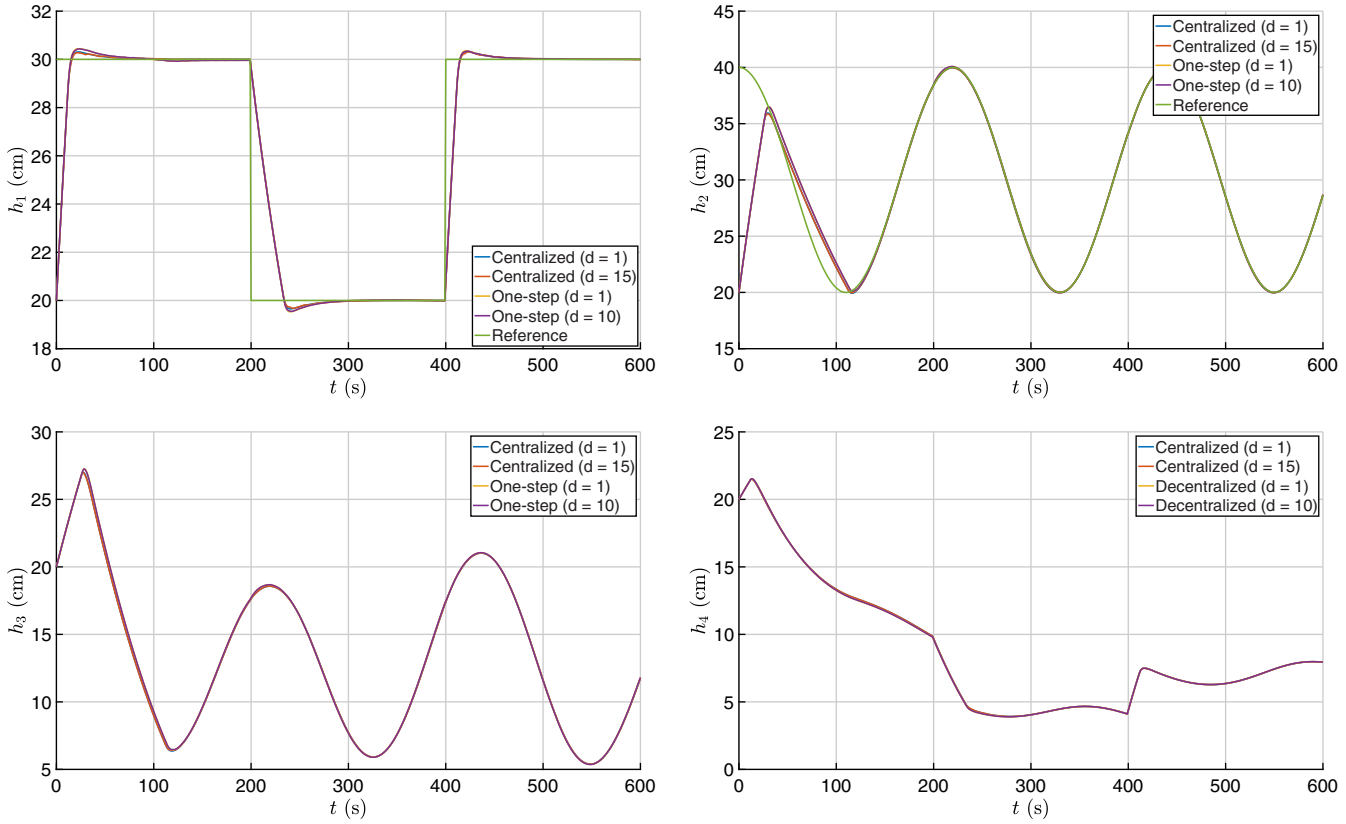


FIGURE 7 Evolution of the water levels of the quadruple-tank network

yielding results very similar to those which make use of just the first gains, even though the dynamics of the system used for each window are predicted. The centralized solution allows, nevertheless, to make use of a larger fraction of the gains of each finite window, which is consistent with the results obtained for the synthetic systems studied in Section 5. For this reason, it is possible to reduce the computational cost of this control approach substantially, without any significant loss of performance. Fourth, the reference signal used cannot be followed by a feasible state trajectory, which means the tracking error does not converge to zero. However, the use of integral action alongside an anti-windup technique allow for very good steady-state performance, achieving very reduced tracking error. Furthermore, note that despite the sudden significant variation, for $t = 400$ s, of the reference signal for tank 1, the tracking performance of tank 2 is almost unaffected. Fifth, it is possible to point out the effects of the saturation of the input of the pumps. In fact, it is visible that, for roughly between $t = 40$ s and $t = 120$ s, the evolution of the water level in tank 2 yields a significant tracking error. In fact, analyzing Figure 8, it is possible to see that such error is due to the saturation of pump 2, since it is not possible to pump water out of the tanks.

7 | SIMULATION RESULTS FOR A NETWORK OF N TANKS

In this section, the methods put forward in this article are applied to a network of N tanks, which corresponds to the generalization of the quadruple-tank network introduced in the previous section. This network is also nonlinear, thus to employ the method devised in this article one can approximate its behavior by an LTV system, linearizing and discretizing its dynamics about successive equilibrium points, as in the previous section. This network is presented to show the scalability of the proposed methods.

7.1 | N tanks network dynamics

Consider N interconnected tanks, as shown in Figure 9, 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

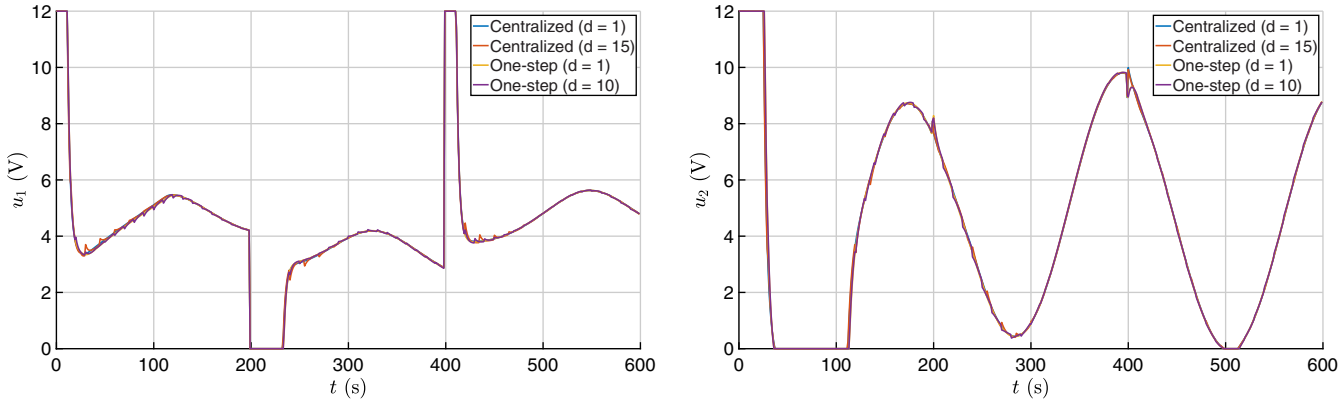


FIGURE 8 Inputs of pump 1 (left) and pump 2 (right) of the quadruple-tank network

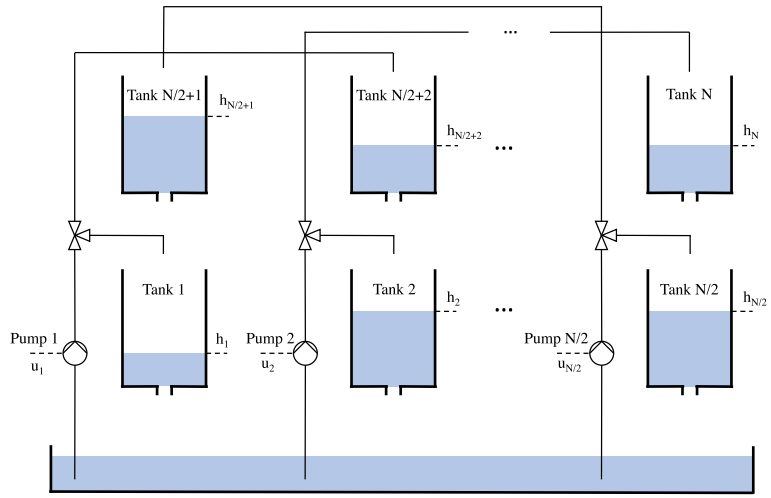


FIGURE 9 Schematic of the N tanks network

for $i = 1, \dots, N/2$, 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. Making use of mass balances and Bernoulli's law, the system dynamics, in the absence of noise, are given by

$$\begin{cases} A_i \dot{h}_i(t) = -a_i\sqrt{2gh_i(t)} + a_{\frac{N}{2}+i}\sqrt{2gh_{\frac{N}{2}+i}(t)} + \gamma_i k_i u_i(t), & i = 1, \dots, N/2, \\ A_i \dot{h}_i(t) = -a_i\sqrt{2gh_i(t)} + (1-\gamma_{i-\frac{N}{2}-1})k_{i-\frac{N}{2}-1}u_{i-\frac{N}{2}-1}(t), & i = \frac{N}{2}+2, \dots, N, \\ A_{\frac{N}{2}+1}\dot{h}_{\frac{N}{2}+1}(t) = -a_{\frac{N}{2}+1}\sqrt{2gh_{\frac{N}{2}+1}(t)} + (1-\gamma_{\frac{N}{2}})k_{\frac{N}{2}}u_{\frac{N}{2}}(t), \end{cases} \tag{33}$$

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; and g denotes the acceleration of gravity. Furthermore, the input of each pump is subject to a hard constraint $u_i \in [0, u^{\text{sat}}]$, where $u^{\text{sat}} \in \mathbb{R}^+$.

Similarly to the procedure followed for the quadruple-tank network, the nonlinear dynamics are linearized about a given equilibrium point, characterized by equilibrium water levels, $h_i^0, i = 1, \dots, N$; and inputs $u_i^0, i = 1, \dots, N/2$. Writing the state and control vectors, respectively, as

$$\mathbf{x_c}(t) = \begin{bmatrix} h_1(t) - h_1^0 \\ \vdots \\ h_N(t) - h_N^0 \end{bmatrix} \quad \text{and} \quad \mathbf{u_c}(t) = \begin{bmatrix} u_1(t) - u_1^0 \\ \vdots \\ u_{\frac{N}{2}}(t) - u_{\frac{N}{2}}^0 \end{bmatrix},$$

the continuous-time linearized system is given by

$$\dot{\mathbf{x}}_\mathbf{c}(t) = \mathbf{A_c}(t)\mathbf{x_c}(t) + \mathbf{B_c}(t)\mathbf{u_c}(t), \tag{34}$$

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

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

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

where T_i is the time constant of tank i, given by (28).

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 discretization of (34) yields

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

where $\mathbf{A}(k)$ and $\mathbf{B}(k)$ are discretized using (29). It is important to remark that, similarly to the quadruple tank network, to perform the linearization, each local controller ought to access the necessary 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, $T_{lin} = qT$, where q is an integer number, thereby reducing the computational load and communication needs.

7.2 | Controller implementation

The problem considered for this network is the design of a decentralized solution to control the water level of the lower tanks. For that reason, the output of the network is computed as in (1), using $\mathbf{H}(k) = \begin{bmatrix} \mathbf{I}_{N/2} & \mathbf{0}_{N/2} \end{bmatrix}$. Similarly to the quadruple tank network, the water level of the lower tanks are the variables used to define each equilibrium point, as a means of ensuring that the system dynamics used for their control are as accurate as possible. For this reason, each time a new linearization is performed, every local controller has to receive through communication the water level of the lower tanks, compute the remaining variables that define the equilibrium point, and linearize the relevant entries of matrix $\mathbf{A}(k)$ about that point. The controller design approach, which is very similar to the one proposed for the controller design of the quadruple tank network in Section 6.2, consists of a local controller in each of the lower tanks, which computes the control action of the associated pump, making use of the measurement of its own water level only. The control action of the decentralized controllers is computed using the tracker design proposed in Section 4, that makes use of the one-step method developed in Section 3 to compute the feedback gains. Given that the reference signal one desires to track in the simulations is not a feasible trajectory, an integral feedback term is also included alongside an anti-windup technique, as detailed in Section 4.2, to improve the steady-state performance. Thus, for the linearized system, each local controller computes an input of the form

$$\left[\mathbf{u}(k)\right]_i = -\left[\mathbf{K}(k)\right]_{ii}\left(\left[\mathbf{x}(k)\right]_i - \left[\bar{\mathbf{x}}(k)\right]_i\right) - \left[\mathbf{K}_I(k)\right]_{ii}\left[\mathbf{x}_I(k)\right]_i + \left[\bar{\mathbf{u}}(k)\right]_i + \left[\mathbf{u_a}(k)\right]_i$$

for $i = 1, \dots, N/2$, in such a way that each tank has only access to its measured water level and to the integral of its tracking error. Note that $u_i(k)$ and $\left[\mathbf{u}(k)\right]_i$ are distinct, the former is the input to pump i and the latter is the ith component of $\mathbf{u}(k)$. To compute $(\overline{\mathbf{x}}(k), \overline{\mathbf{u}}(k))$, an analogous approach to the one presented in Section 6.2 is followed. The actual pump input is, thus, computed making use of

$$u_i(k) = -\left[\mathbf{K}(k)\right]_{ii}\left(h_i(k) - \overline{h}_i(k)\right) - \left[\mathbf{K}_I(k)\right]_{ii}\left[\mathbf{x}_I(k)\right]_i + \left[\overline{\mathbf{u}}(k)\right]_i + \left[\mathbf{u}_\mathbf{a}(k)\right]_i + u_i^0(k), \tag{35}$$

where $\overline{h}_i(k)$ and $\bar{u}_i(k)$ are defined by

$$\overline{\mathbf{x}}(k) = \begin{bmatrix} \overline{h}_1(k) - h_1^0(k) \\ \vdots \\ \overline{h}_N(k) - h_N^0(k) \end{bmatrix} \quad \text{and} \quad \overline{\mathbf{u}}(k) = \begin{bmatrix} \bar{u}_1(k) - u_1^0(k) \\ \vdots \\ \bar{u}_{N/2}(k) - u_{N/2}^0(k) \end{bmatrix}, \tag{36}$$

where $h_i^0(k)$ and $u_j^0(k)$ are the equilibrium water level of tank i and equilibrium input of pump j computed in the last linearization prior to time instant k, respectively. Although the controller design put forward is not projected to handle hard input constraints, as this network requires, to meet those constraints on the control inputs of the pumps, the computed inputs, given by (35), are saturated. Comparing the local controller (35) with the global controller (25), it follows that the feedback gain of the augmented system, $\boldsymbol{\mathcal{K}}(k) = \begin{bmatrix}\mathbf{K}(k) & \mathbf{K}_I(k)\end{bmatrix}$, must follow the sparsity pattern defined by $\mathbf{E} = \begin{bmatrix}\mathbf{I}_{N/2} & \mathbf{0}_{N/2} & \mathbf{I}_{N/2}\end{bmatrix}$.

Both the centralized and one-step methods require the dynamics of the system in a time window that spans future instants. Considering that the dynamics of the network vary with its state vector, a mechanism that predicts the future evolution of the state vector to obtain the linearized dynamics is necessary. For that reason, the iLQR scheme described in Section 6.2 is used.

7.3 | Simulation results

The network was simulated for $N = 40$ tanks and the values of its physical constants are presented in Table 4. The sampling time was set to $T = 1\,\text{s}$ and the linearization period to $T_{lin} = 10T$. The initial level of the tanks is set to $h_i = 20\,\text{cm}$ for $i = 1, \dots, N$. The reference signal is given by

$$[\mathbf{r}(t)]_i = \begin{cases} 25 + 5\ \text{sgn}\left(\sin\left(2\pi t/100\right)\right), & i = 4n+1, \\ 30 + 10\cos\left(t/35\right), & i = 4n+2, \\ 25 + 5\ \text{sgn}\left(\sin\left(2\pi t/200\right)\right), & i = 4n+3, \\ 30 + 10\cos\left(t/50\right), & i = 4n+4, \end{cases} \quad n \in \{0, \dots, N/4-1\},$$

which is sampled with a periodicity of T.

As detailed in Section 3.2, the infinite-horizon solution is approximated by the solution of several finite-horizon problems on a chosen window. Its length was selected to be $T = 30$, roughly twice the time constant of the slowest pole of the system, which is enough for the centralized and one-step methods to approximate the infinite-horizon solution. The proposed method is compared with the analogous centralized solution, that is, using the unconstrained LQR solution to compute the LQR feedback gains of (32). Due to the nonlinear dynamics of the network, the controller parameters were tuned empirically. The weighting matrices were set to $\mathbf{R}(k) = \mathbf{I}_{N/2}$, $\mathbf{Q}_T = 20\mathbf{I}_{N/2}$, and $\mathbf{Q}_I = 0.05\mathbf{I}_{N/2}$. The integral state saturation limits vector was selected as $\left[\mathbf{x}_\mathbf{I}^{\text{sat}}\right]_i = 10\,\text{cm}$ for $i = 1, \dots, N/2$, for both methods.

Figure 10 depicts the nonlinear simulation of the water level of four tanks, one for each of the four types of reference signals, for the centralized and one-step methods. The simulation was also carried out making use of $d = 15$ and $d = 10$ of the gains computed for each window instead of just one, for the centralized and decentralized solutions, respectively. Figure 11 depicts the evolution of the input to the pumps associated with the tanks whose water level is plotted in Figure 10. First, it is clear that none of the solutions diverge and all drive the system state very close to the reference signal, yielding a very good performance. Note that the system is nonlinear and the control inputs are subject to hard

TABLE 4 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
g	981 $cm\,s^{-2}$
k_i	3.33 $cm^3\,s^{-1}\,V^{-1}$
γ_i, i odd	0.7
γ_i, i even	0.6
u^{sat}	12 V

constraints, none of which were taken into account in the design of the decentralized controller. Moreover, the variation of the water level in the tanks is considerable, which allows for a significant variation in the dynamics of the system throughout the simulation. Second, the fact that the proposed method for the computation of the decentralized feedback gains has a closed-form solution allows for its use in large-scale networks, as in this example, showing the scalability of the methods put forward. Third, it is interesting to notice that, similarly to the quadruple tank network, despite having considered a fully decentralized design, the performance obtained with the centralized solution is not considerably better. In fact, given that the use of a decentralized solution reduces tremendously the communication needs, particularly for a large-scale network as the case being considered, the use of a decentralized solution is even more compelling than for the quadruple tank network. Forth, regarding the proposed MPC scheme, it is clear that the use of more than one gain of each finite window does not introduce a noticeable loss of performance, yielding results very similar to those which make use of just the first gains. For this reason, it is possible to reduce the computational cost of this control approach dramatically, without any significant loss of performance.

8 | COMPARISON BETWEEN THE CENTRALIZED AND DECENTRALIZED SOLUTIONS

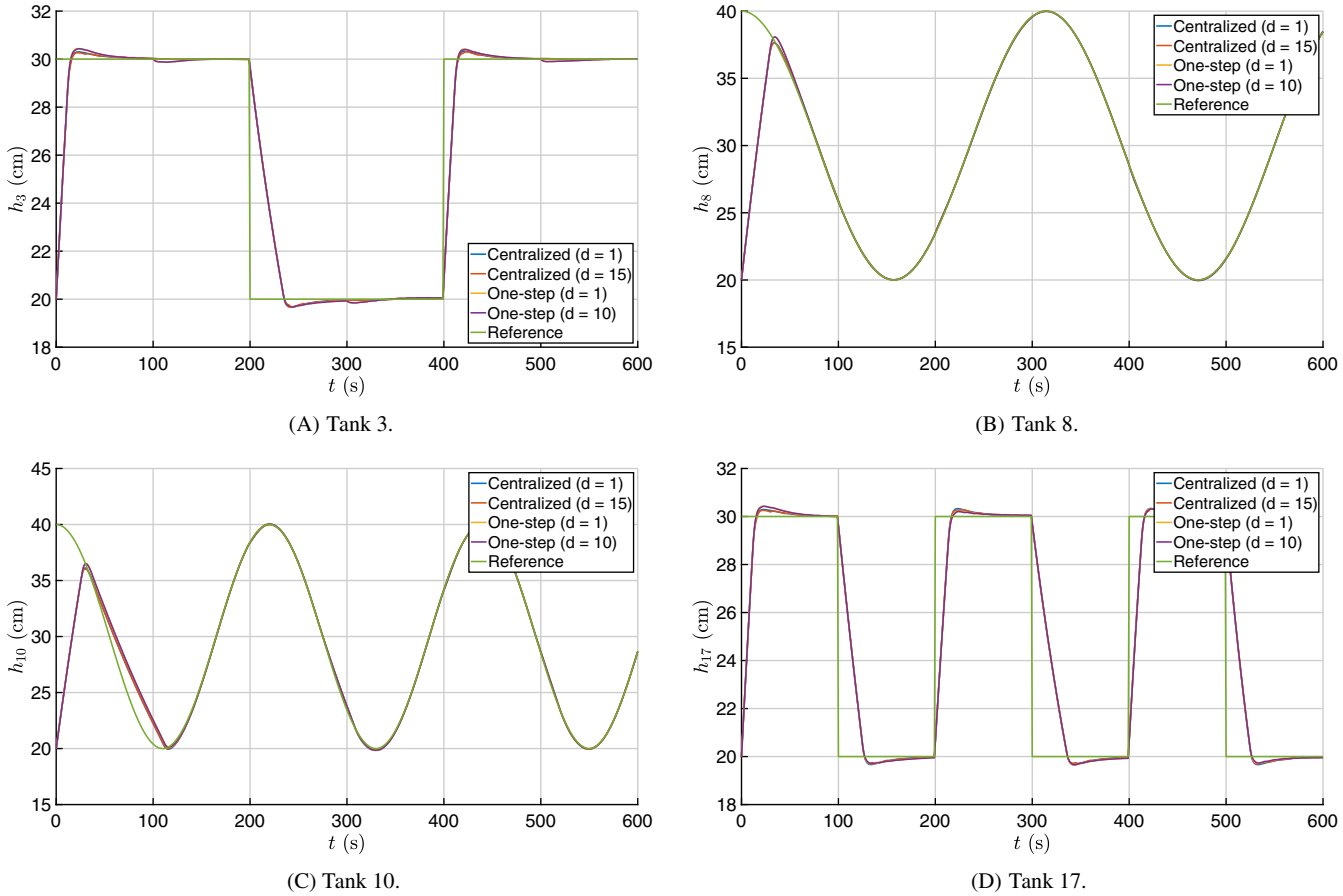
In this section, the results obtained for the networks of interconnected tanks simulated in Sections 6 and 7 for the decentralized solution proposed in this article are further compared with the centralized solution. This analysis regards: (i) performance; (ii) communication needs; and (iii) computational cost.

First, consider the tracking fitness function

$$J_t := \sum_{\tau=0}^{T_{\text{sim}}} (\mathbf{z}(\tau) - \mathbf{r}(\tau))^T (\mathbf{z}(\tau) - \mathbf{r}(\tau)),$$

where T_{sim} is the number of time steps of a given simulation. The tracking fitness values of the centralized and decentralized solutions presented in Sections 6 and 7, with $T_{\text{sim}} = 600$, are shown in Table 5. These values show that the use of one gain of each window or more offer an almost indistinguishable performance. Furthermore, it is possible to notice that there is a relative increase roughly of only 2.0% and 0.81% for the decentralized solutions in relation to the centralized results, for the network of $N = 4$ and $N = 40$ tanks, respectively.

The centralized solution makes use of $(N-1)N/2$ directional communication links, for a network of N tanks, which are necessary to connect every lower tank to every other tank of the network. On the other hand, the decentralized solution would not need any communication links between tanks to compute the control action, if the system was LTV. However, since the network is nonlinear, each lower tank has to receive, via a communication link, the water level of all the other tanks of the network to compute the linearized dynamics of the network, requiring the same number of communication



(A) Tank 3.

(B) Tank 8.

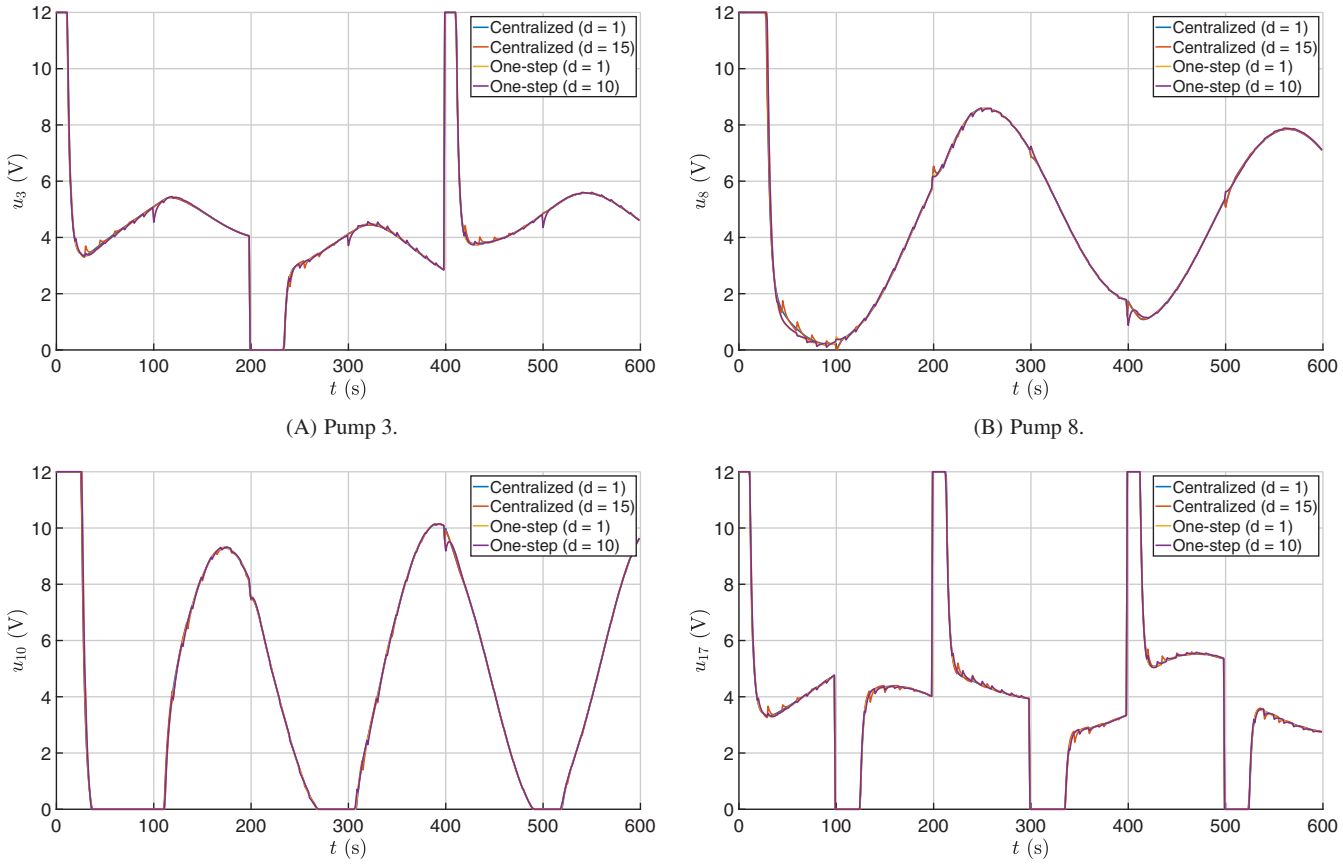
(C) Tank 10.

(D) Tank 17.

FIGURE 10 Evolution of the water levels of the network of N tanks

links as the centralized solution. Nevertheless, the transmission rate is much lower in a decentralized configuration, given that it only requires one transmission of the water level of each tank to the other tanks every q time-steps. In particular, for the simulations performed in Sections 6 and 7, it was set $q = 10$, which allows for a reduction of 90% in the transmission rate of the decentralized design in comparison with the centralized configuration. Nevertheless, the number of such communication links can be lessened neglecting the water level in some tanks for the computation of the equilibrium water level in each tank, whose influence is not very significant.

Some computational performance parameters are shown in Tables 6 and 7, regarding, respectively, the simulation of the network with $N = 4$ and $N = 40$ tanks: (i) the number of times that the iLQR method is performed; (ii) the total time spent during all the calls to the iLQR algorithm; (iii) the average time spent in each iLQR call; (iv) the number of finite windows for which either the centralized or one-step gains were computed within all the iLQR calls; (v) the average number of finite window computations used in each LQR call; (vi) the total time spent computing the centralized or one-step gains for all finite windows; and (vii) the average time spent in the computation of the gains for each finite window. The elapsed times shown in these tables are wall-clock times, resulting from the average of five simulations of a MATLAB implementation on a 1.60 GHz Dual-Core Intel Core i5 with 4 GB RAM. First, the number of iLQR calls is equal to T_{sim}/d, which suggests that to reduce the computational load one should use the highest d possible that does not lower the performance significantly. In fact, for the centralized and decentralized solutions, the use of $d = 15$ and $d = 10$, respectively, allowed for a decrease of roughly one order of magnitude on the time of the iLQR gain computation. Second, it was assumed in the simulations presented in Sections 6 and 7 that the gain computation is instantaneous and thus there are no delays. However, it is possible to notice that the use of the iLQR algorithm, which is necessary because of the nonlinearity of the network, introduces a delay at every instant that corresponds to the beginning of an iLQR call, whose duration is the time taken per iLQR call. Albeit small, it is not negligible compared with the time constant of the network for $N = 40$ tanks, for this particular implementation. Nevertheless, if the system were LTV, it would only be necessary to compute the one-step gains once instead of iterating until convergence, which would introduce a negligible



(A) Pump 3.

(B) Pump 8.

(C) Pump 10.

(D) Pump 17.

FIGURE 11 Inputs of the pumps of the network of N tanks

TABLE 5 Tracking fitness values of the centralized and decentralized solutions presented in Sections 6 and 7 for a network of N interconnected tanks

	$N = 4$	$N = 40$
Centralized ($d = 1$)	0.6300 m^2	7.406 m^2
Centralized ($d = 10$)	0.6294 m^2	7.407 m^2
One-step ($d = 1$)	0.6423 m^2	7.466 m^2
One-step ($d = 15$)	0.6419 m^2	7.466 m^2

TABLE 6 Computational cost parameters for the gain computation of the simulation presented in Section 6 for the quadruple-tank network

	iLQR calls	iLQR time	Average time per iLQR call	Computed windows	Average windows per iLQR call	Time computing window gains	Average time per window
Centralized ($d = 1$)	600	119.4 s	0.1990 s	1860	3.100	2.624 s	1.411 ms
Centralized ($d = 10$)	40	10.30 s	0.2575 s	124	3.100	0.2247 s	1.812 ms
One-step ($d = 1$)	600	91.59 s	0.1526 s	1329	2.215	2.164 s	1.629 ms
One-step ($d = 15$)	60	11.11 s	0.1852 s	133	2.217	0.2740 s	2.061 ms

TABLE 7 Computational cost parameters for the gain computation of the simulation presented in Section 7 for a network of $N = 40$ interconnected tanks

	iLQR calls	iLQR time	Average time per iLQR call	Computed windows	Average windows per iLQR call	Time computing window gains	Average time per window
Centralized ($d = 1$)	600	284.8 s	0.4747 s	1822	3.037	11.69 s	6.414 ms
Centralized ($d = 10$)	40	21.76 s	0.5441 s	121	3.025	0.8847 s	7.311 ms
One-step ($d = 1$)	600	404.7 s	0.6745 s	1460	2.433	176.2 s	120.7 ms
One-step ($d = 15$)	60	43.53 s	0.7255 s	148	2.467	18.75 s	126.7 ms

delay even for the network of $N = 40$ tanks. Third, it is interesting to notice that, since the centralized gain has more degrees of freedom, it is necessary to compute more finite window gains, on average, for the convergence of the iLQR iterations. Fourth, while, for the quadruple-tank network, the gain computation on the one-step method for each window takes roughly as much time as the centralized solution, for the network of $N = 40$ tanks it is almost 20 times slower. The computational complexity comparison between the one-step method and centralized solution, detailed in Remark 2, is not verified in this case because the values of N in these simulations are too low for an asymptotic comparison.

Figure 12 depicts the time to compute a window of $T = 30$ gains averaged over 5 finite window computations as a function of the dimension of the network. First, both solutions appear to asymptotically approach a linear relation between the logarithms of both variables, thus both algorithms are of polynomial time. Note that it takes a large value of N to be possible to distinguish the apparent asymptotic evolution. In fact, the transient is a consequence of terms of lower polynomial complexity that arise from memory allocation and less intensive computations of the algorithm. Thus, it may take a large N for those transient terms to become insignificant compared with the highest polynomial complexity of the operations performed in the algorithm. Moreover, the apparent asymptotic linear relation is estimated using a linear regression, shown in Figure 12 as well, which was obtained making use of the data points with $N \geq 212$ for the centralized solution and $N \geq 120$ for the one-step method, achieving high correlation coefficients. Second, as detailed in Remark 2, the average time per window for the centralized solution and one-step method grow asymptotically at a similar rate of $\mathcal{O}(N^3)$. Finally, this analysis has only qualitative significance, since it depends greatly on the implementation and on the hardware used to run it.

Taking everything analyzed in this section into consideration, it is possible to conclude that the use of a decentralized configuration comes with little performance penalty, but achieves a significant reduction in communication needs. Furthermore, the use of more than one gain of each finite window does not penalize performance and allows for a significant reduction in computational cost. Depending on the dimension of the network and on the implementation of the algorithm, it may be necessary to account for delays in the computation of the gains at the beginning of each finite window. Finally, it is important to make clear that if the system were LTV, there would be no need for the use of the

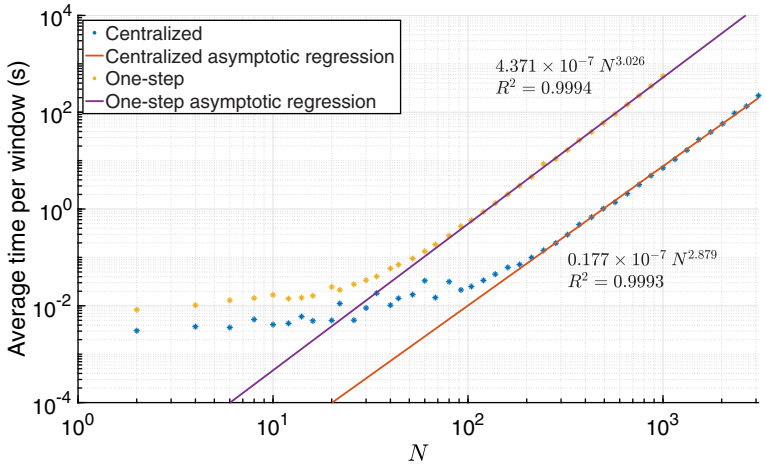


FIGURE 12 Average time to compute a window of $T = 30$ gains averaged over 5 finite window computations as a function of the dimension of the network of interconnected tanks

iLQR algorithm which is responsible for increasing the computational load significantly, as it is noticeable in the analysis carried out in this section.

9 | CONCLUSION

Very little work has been carried out regarding the design of decentralized control solutions for arbitrary LTV systems. In this article, a method for the computation of decentralized LQR gains for arbitrary LTV systems with arbitrary time-invariant network configurations, portrayed by the sparsity constraint that is imposed, is proposed. Moreover, a method for the tracking problem is also put forward, building on the solution to the regulator problem. These methods are obtained for a finite-horizon formulation and then extended to the infinite-horizon problem by making use of an MPC-like scheme. Nevertheless, the methods put forward require that a window of the future dynamics of the system is known, similarly to the corresponding centralized methods. First, it was shown that the method for the computation of regulator gains proposed in this article is able to compute a sequence of well-performing stabilizing gains subject to an arbitrary sparsity constraint. Second, the regulator gains, as well as the control action for the tracking problem, have a closed-form solution, thus they can be computed very rapidly and efficiently. Third, regarding the MPC-like scheme that was proposed, it was possible to conclude that the use of a considerable fraction of the gains computed for each window yields a performance identical to the simulations using just one, which allows for a significant decrease in computational load. Fourth, both algorithms put forward in this article were applied to a nonlinear system of interconnected tanks with hard constraints on the inputs, whose dynamics were approximated by an LTV system corresponding to successive linearizations about the operations points. To predict a window of the future dynamics of the system, an iLQR algorithm was used. Even though the system is nonlinear and the control inputs are subject to hard constraints, none of which were taken into account in the design of the decentralized controller, the proposed solution drives the system state to the reference signal, yielding very good performance. Fifth, despite having considered a fully decentralized network design, the performance obtained with the centralized solution is not considerably better, which makes the use of a decentralized solution very compelling, allowing for a significant decrease in the communication needs of the network. Sixth, it was possible to show the scalability of both methods, with the successful application to a large-scale system. Finally, the average time per window for the one-step method and the centralized solution, applied a network of N interconnected tanks, grow asymptotically roughly with $\mathcal{O}(N^3)$.

ACKNOWLEDGMENTS

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.

CONFLICT OF INTEREST

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this article.

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.

ORCID

Leonardo Pedroso https://orcid.org/0000-0002-1508-496X
Pedro Batista https://orcid.org/0000-0001-6079-0436

REFERENCES

1. Bakule L. Decentralized control: an overview. *Annu Rev Control*. 2008;32(1):87-98.
2. Bakule L. Decentralized control: status and outlook. *Annu Rev Control*. 2014;38(1):71-80.
3. Wolfe J, Chichka D, Speyer J. Decentralized controllers for unmanned aerial vehicle formation flight. Proceedings of the Guidance, Navigation, and Control Conference. American Institute of Aeronautics and Astronautics; 1996; Reston, Virigina.
4. Singh SN, Zhang R, Chandler P, Banda S. Decentralized nonlinear robust control of UAVs in close formation. *Int J Robust Nonlinear Control*. 2003;13(11):1057-1078.
5. Raffard R, Tomlin C, Boyd S. Distributed optimization for cooperative agents: application to formation flight. Proceedings of the 2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601); Vol. 3, 2004:2453-2459.
6. Bereg S, Dıaz-Bánez JM, Lopez MA, Rozario T, Valavanis K. A decentralized geometric approach for the formation keeping in unmanned aircraft navigation. Proceedings of the 2015 International Conference on Unmanned Aircraft Systems (ICUAS); 2015:989-997.
7. Thien RT, Kim Y. Decentralized formation flight via PID and integral sliding mode control. *Aerosp Sci Technol*. 2018;81:322-332.
8. Curtin T, Bellingham J, Catipovic J, Webb D. Autonomous oceanographic sampling networks. *Oceanography*. 1993;6(3):86-94. https://doi.org/10.5670/oceanog.1993.03
9. Healey A. Application of formation control for multi-vehicle robotic minesweeping. *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No.01CH37228)*. IEEE; 2001;2:1497-1502. https://doi.org/10.1109/CDC.2001.981106
10. Viegas D, Batista P, Oliveira P, Silvestre C. Decentralized observers for position and velocity estimation in vehicle formations with fixed topologies. *Syst Control Lett*. 2012;61(3):443-453. https://doi.org/10.1016/j.sysconle.2011.12.004
11. Yuan C, Licht S, He H. Formation learning control of multiple autonomous underwater vehicles with heterogeneous nonlinear uncertain dynamics. *IEEE Trans Cybern*. 2017;99:1-15.
12. Enayat M, Khorasani K. Semi-decentralized nonlinear cooperative control strategies for a network of heterogeneous autonomous underwater vehicles. *Int J Robust Nonlinear Control*. 2017;27(16):2688-2707.
13. Shaw GB. The generalized information network analysis methodology for distributed satellite systems; 1999.
14. Russell CJ. Decentralized control of satellite formations. *Int J Robust Nonlinear Control*. 2002;12(2-3):141-161. https://doi.org/10.1002/rnc.680
15. Wu B, Wang D, Poh EK. Decentralized sliding-mode control for attitude synchronization in spacecraft formation. *Int J Robust Nonlinear Control*. 2013;23(11):1183-1197.
16. Ivanov D, Monakhova U, Ovchinnikov M. Nanosatellites swarm deployment using decentralized differential drag-based control with communicational constraints. *Acta Astronaut*. 2019;159:646-657.
17. Bender J. An overview of systems studies of automated highway systems. *IEEE Trans Veh Technol*. 1991;40(1):82-99. https://doi.org/10.1109/25.69977
18. Yanakiev D, Kanellakopoulos I. A simplified framework for string stability analysis in AHS 1. *IFAC Proc Vol*. 1996;29(1):7873-7878. https://doi.org/10.1016/S1474-6670(17)58959-4
19. Mu J, Yan XG, Spurgeon SK, Zhao D. Nonlinear sliding mode control for interconnected systems with application to automated highway systems. *IEEE Trans Control Netw Syst*. 2016;5(1):664-674.
20. Alam A, Mårtensson J, Johansson KH. Experimental evaluation of decentralized cooperative cruise control for heavy-duty vehicle platooning. *Control Eng Pract*. 2015;38:11-25.
21. Gómez M, Rodellar J, Mantecón JA. Predictive control method for decentralized operation of irrigation canals. *Appl Math Model*. 2002;26(11):1039-1056. https://doi.org/10.1016/S0307-904X(02)00059-8
22. Cantoni M, Weyer E, Li Y, Ooi SK, Mareels I, Ryan M. Control of large-scale irrigation networks. *Proc IEEE*. 2007;95(1):75-91. https://doi.org/10.1109/JPROC.2006.887289
23. Li Y. Offtake feedforward compensation for irrigation channels with distributed control. *IEEE Trans Control Syst Technol*. 2014;22(5):1991-1998.
24. Prodan I, Lefevre L, Genon-Catalot D. Distributed model predictive control of irrigation systems using cooperative controllers. *IFAC-PapersOnLine*. 2017;50(1):6564-6569.

25. D'Andrea R, Dullerud GE. Distributed control design for spatially interconnected systems. *IEEE Trans Automat Contr*. 2003;48(9):1478-1495.
26. Rice JK, Verhaegen M. Distributed control: a sequentially semi-separable approach for spatially heterogeneous linear systems. *IEEE Trans Automat Contr*. 2009;54(6):1270-1283.
27. Blondel VD, Tsitsiklis JN. A survey of computational complexity results in systems and control. *Automatica*. 2000;36(9):1249-1274.
28. Witsenhausen HS. A counterexample in stochastic optimum control. *SIAM J Control*. 1968;6(1):131-147.
29. Lessard L, Lall S. Internal quadratic invariance and decentralized control. *Proceedings of the 2010 American Control Conference*. IEEE; 2010:5596-5601. https://doi.org/10.1109/ACC.2010.5531024
30. Lessard L, Lall S. Convexity of decentralized controller synthesis. *IEEE Trans Automat Contr*. 2015;61(10):3122-3127.
31. Zuo L, Nayfeh SA. Structured H2 optimization of vehicle suspensions based on multi-wheel models. *Veh Syst Dyn*. 2003;40(5):351-371.
32. Zhai G, Yoshida M, Imae J, Kobayashi T. Decentralized H2 controller design for descriptor systems: an LMI approach. *Nonlinear Dyn Syst Theory*. 2006;6(1):98-108.
33. Shah P, Parrilo PA. H2-optimal decentralized control over posets: a state-space solution for state-feedback. *IEEE Trans Automat Contr*. 2013;58(12):3084-3096.
34. Viegas D, Batista P, Oliveira P, Silvestre C. Distributed controller design and performance optimization for discrete-time linear systems. *Opt Control Appl Methods*. 2021;42(1):126-143. https://doi.org/10.1002/oca.2669
35. Zhao Y, Liu Y, Wen G, Ren W, Chen G. Designing distributed specified-time consensus protocols for linear multiagent systems over directed graphs. *IEEE Trans Automat Contr*. 2018;64(7):2945-2952.
36. Zhao Y, Zhou Y, Liu Y, Wen G, Huang P. Fixed-time bipartite synchronization with a pre-appointed settling time over directed cooperative–antagonistic networks. *Automatica*. 2021;123:109301.
37. Kulkarni J, Campbell M. An approach to magnetic torque attitude control of satellites via H-infinity control for LTV systems. *Proceedings of the 2004 43rd IEEE Conference on Decision and Control (CDC)(IEEE Cat. No. 04CH37601)*. IEEE; 2004;1:273-277. https://doi.org/10.1109/CDC.2004.1428642
38. Vazquez R, Gavilan F, Camacho EF. Pulse-width predictive control for LTV systems with application to spacecraft rendezvous. *Control Eng Pract*. 2017;60:199-210.
39. Marinescu B. Output feedback pole placement for linear time-varying systems with application to the control of nonlinear systems. *Automatica*. 2010;46(9):1524-1530.
40. Farhood M, Di Z, Dullerud GE. Distributed control of linear time-varying systems interconnected over arbitrary graphs. *Int J Robust Nonlinear Control*. 2015;25(2):179-206.
41. Anderson B, Moore JB. *Optimal Control: Linear Quadratic Methods*. Prentice-Hall, Inc. 1990.
42. Pedroso L, Batista P. Efficient algorithm for the computation of the solution to a sparse matrix equation in distributed control theory. *Mathematics*. 2021;9(13):1497. https://doi.org/10.3390/math9131497
43. Kwakernaak H, Sivan R. *Linear Optimal Control Systems*. Vol 1. Wiley-Interscience; 1972.
44. Shafarevich IR, Remizov AO. *Linear Algebra and Geometry*. Springer Science & Business Media; 2012.
45. Heller D. Some aspects of the cyclic reduction algorithm for block tridiagonal linear systems. *SIAM J Numer Anal*. 1976;13(4):484-496.
46. Hirshman SP, Perumalla KS, Lynch VE, Sanchez R. BCYCLIC: a parallel block tridiagonal matrix cyclic solver. *J Comput Phys*. 2010;229(18):6392-6404.
47. Johansson KH. The quadruple-tank process: a multivariable laboratory process with an adjustable zero. *IEEE Trans Control Syst Technol*. 2000;8(3):456-465. https://doi.org/10.1109/87.845876
48. Li W, Todorov E. Iterative linear quadratic regulator design for nonlinear biological movement systems. ICINCO; Vol. 1; 2004:222-229.
49. Todorov E, Li W. A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems. *Proceedings of the 2005, American Control Conference*. IEEE; 2005;1:300-306. https://doi.org/10.1109/ACC.2005.1469949
50. Lewis FL, Vrabie D, Syrmos VL. *Optimal Control*. John Wiley & Sons; 2012.
51. Viegas D, Batista P, Oliveira P, Silvestre C. Discrete-time distributed Kalman filter design for formations of autonomous vehicles. *Control Eng Pract*. 2018;75:55-68. https://doi.org/10.1016/j.conengprac.2018.03.014
52. Bertsekas DP. *Dynamic Programming and Optimal Control*. Vol 1. 3rd ed. Athena Scientific; 2005.

How to cite this article: Pedroso L, Batista P. Discrete-time decentralized linear quadratic control for linear time-varying systems. *Int J Robust Nonlinear Control*. 2021;1–35. https://doi.org/10.1002/rnc.5772

APPENDIX A. DERIVATION OF THE CLOSED-FORM ONE-STEP SUB-OPTIMAL SOLUTION TO THE DECENTRALIZED FINITE-HORIZON LINEAR QUADRATIC REGULATOR PROBLEM

The proposed derivation of the one-step method for the computation of decentralized LQR gains, which correspond to a sub-optimal solution to the finite-horizon decentralized LQR problem, follows the Lagrange-multiplier approach detailed, for instance, in Reference 50. The goal of using this approach is to ease the inclusion of the sparsity constraint

$\mathbf{K}(k) \in \text{Sparse}(\mathbf{E})$, the state equation (2), and the linear feedback action (3), which allows to write (6) as an unconstrained optimization problem.

Writing an augmented performance index, $J'(k)$, that takes into account the linear feedback action (3), as well as the state equation (2), yields

$$\begin{aligned} J'(k) = \mathbf{x}^T(k+T)\mathbf{Q}(k+T)\mathbf{x}(k+T) + \sum_{\tau=k}^{k+T-1} \mathbf{x}^T(\tau)\left(\mathbf{Q}(\tau) + \mathbf{K}^T(\tau)\mathbf{R}(\tau)\mathbf{K}(\tau)\right)\mathbf{x}(\tau) \\ + \sum_{\tau=k}^{k+T-1} \lambda^T(\tau+1)\left[(\mathbf{A}(\tau) - \mathbf{B}(\tau)\mathbf{K}(\tau))\,\mathbf{x}(\tau) - \mathbf{x}(\tau+1)\right], \end{aligned} \tag{A1}$$

where $\lambda(\tau+1) \in \mathbb{R}^n$ is the Lagrange-multiplier associated with each of the constraints that arise from the state equation. The augmented performance index (A1) is often written, for convenience, as a function of the Hamiltonian, defined, in this case, as

$$H(k) = \mathbf{x}^T(k)\left(\mathbf{Q}(k) + \mathbf{K}^T(k)\mathbf{R}(k)\mathbf{K}(k)\right)\mathbf{x}(k) + \lambda^T(k+1)\left(\mathbf{A}(k) - \mathbf{B}(k)\mathbf{K}(k)\right)\mathbf{x}(k),$$

which yields

$$J'(k) = \mathbf{x}^T(k+T)\mathbf{Q}(k+T)\mathbf{x}(k+T) - \lambda^T(k+T)\mathbf{x}(k+T) + H(k) + \sum_{\tau=k+1}^{k+T-1}\left(H(\tau) - \lambda^T(\tau)\mathbf{x}(\tau)\right). \tag{A2}$$

Taking the differential of the augmented performance index (A2), one obtains

$$\begin{aligned} dJ'(k) = {} & \left(2\mathbf{Q}(k+T)\mathbf{x}(k+T) - \lambda(k+T)\right)^T d\mathbf{x}(k+T) \\ & + \left(\frac{\partial H(k)}{\partial \mathbf{x}(k)}\right)^T d\mathbf{x}(k) + \sum_{\tau=k+1}^{k+T}\left(\frac{\partial H(\tau-1)}{\partial \lambda(\tau)} - \mathbf{x}(\tau)\right)^T d\lambda(\tau) \\ & + \left(\frac{\partial H(k)}{\partial \text{vec}\,(\mathbf{K}(k))}\right)^T d\text{vec}\,(\mathbf{K}(k)) + \sum_{\tau=k+1}^{k+T-1}\left[\left(\frac{\partial H(\tau)}{\partial \text{vec}\,(\mathbf{K}(\tau))}\right)^T d\text{vec}\,(\mathbf{K}(\tau)) + \left(\frac{\partial H(\tau)}{\partial \mathbf{x}(\tau)} - \lambda(\tau)\right)^T d\mathbf{x}(\tau)\right]. \end{aligned} \tag{A3}$$

Define the set χ of integer pairs of the form (i,j) to index the nonzero entries of $\mathbf{K}(k)$ as

$$\begin{cases} (i,j) \in \chi & \text{if } [\mathbf{E}]_{ij} \neq 0, \\ (i,j) \notin \chi & \text{otherwise,} \end{cases} \quad i = 1, \ldots, m, \quad j = 1, \ldots, n. \tag{A4}$$

The necessary conditions for the constrained minimum follow from (A3) and from the sparsity constraint. For a fixed initial state $\mathbf{x}(k)$ and free final state $\mathbf{x}(k+T)$, the constrained minimum requires that $dJ'(k) = 0$ holds for any: (i) $d\mathbf{x}(\tau)$, with $\tau = k+1, \ldots, k+T$; (ii) $d\lambda(\tau)$, with $\tau = k+1, \ldots, k+T$; and (iii) $\mathbf{l}_i^T d\mathbf{K}(\tau)\mathbf{l}_j$, with $\tau = k, \ldots, k+T-1$ and $(i,j) \in \chi$. Hence, it follows that

$$\mathbf{x}(\tau+1) = \frac{\partial H(\tau)}{\partial \lambda(\tau+1)}, \quad \tau = k, \ldots, k+T-1, \tag{A5a}$$

$$\lambda(\tau) = \frac{\partial H(\tau)}{\partial \mathbf{x}(\tau)}, \quad \tau = k+1, \ldots, k+T-1, \tag{A5b}$$

$$\mathbf{l}_i^T \frac{\partial H(\tau)}{\partial \mathbf{K}(\tau)}\mathbf{l}_j = 0, \quad \tau = k, \ldots, k+T-1, \ (i,j) \in \chi, \tag{A5c}$$

$$\mathbf{l}_i^T \mathbf{K}(\tau)\mathbf{l}_j = 0, \quad \tau = k, \ldots, k+T-1, \ (i,j) \notin \chi, \tag{A5d}$$

and

$$\lambda(k+T) = 2\mathbf{Q}(k+T)\mathbf{x}(k+T), \tag{A5e}$$

where $\mathbf{l}_i$ is defined as in Theorem 1. (A5a) is the state equation, (A5b) is the costate equation, (A5c) is the stationary condition, (A5d) is the sparsity constraint, and (A5e) is the boundary condition. It is interesting to remark the usefulness of the Hamiltonian function, which allows to write the constraints of the optimization problem as neat identities involving its partial derivatives. As the form of the boundary condition suggests, the Lagrange-multipliers can possibly be written as $\lambda(k) = 2\mathbf{P}(k)\mathbf{x}(k)$, where $\mathbf{P}(k)$ is a symmetric positive semidefinite matrix. In that case, from the boundary condition (A5e), it follows that $\mathbf{P}(k+T) = \mathbf{Q}(k+T)$. In fact, making use of the costate equation (A5b), this hypothesis yields

$$\mathbf{P}(\tau)\mathbf{x}(\tau) = \left(\mathbf{Q}(\tau) + \mathbf{K}^T(\tau)\mathbf{R}(\tau)\mathbf{K}(\tau)\right)\mathbf{x}(\tau) + (\mathbf{A}(\tau) - \mathbf{B}(\tau)\mathbf{K}(\tau))^T\mathbf{P}(\tau+1)\mathbf{x}(\tau+1),$$

$\tau = k, \ldots, k+T-1$, which holds for every $\mathbf{x}(\tau)$ if and only if

$$\mathbf{P}(\tau) = \mathbf{Q}(\tau) + \mathbf{K}^T(\tau)\mathbf{R}(\tau)\mathbf{K}(\tau) + (\mathbf{A}(\tau) - \mathbf{B}(\tau)\mathbf{K}(\tau))^T\mathbf{P}(\tau+1)\left(\mathbf{A}(\tau) - \mathbf{B}(\tau)\mathbf{K}(\tau)\right).$$

For this reason, the hypothesis on the form of the Lagrange multipliers, $\lambda(k) = 2\mathbf{P}(k)\mathbf{x}(k)$, is valid, and $\mathbf{P}(k)$ is given by the recursive closed-form expression (8). Making use of (A5c), and using also the closed-loop system dynamics

$$\mathbf{x}(\tau+1) = (\mathbf{A}(\tau) - \mathbf{B}(\tau)\mathbf{K}(\tau))\mathbf{x}(\tau), \tag{A6}$$

one can write

$$\mathbf{l}_i^T\left[\mathbf{R}(\tau)\mathbf{K}(\tau)\mathbf{x}(\tau)\mathbf{x}^T(\tau) - \mathbf{B}^T(\tau)\mathbf{P}(\tau+1)\left(\mathbf{A}(\tau) - \mathbf{B}(\tau)\mathbf{K}(\tau)\right)\mathbf{x}(\tau)\mathbf{x}^T(\tau)\right]\mathbf{l}_j = 0, \tag{A7}$$

for all $(i,j) \in \chi$ and $\tau = k, \ldots k+T-1$. Note that (A7) depends on $\mathbf{x}(\tau)$, $\tau = k, \ldots k+T-1$, which is not readily available in a decentralized formulation. For that reason, unlike the centralized finite-horizon problem, finding all the solutions to (A7), being the global minimum among them, is not possible without the knowledge of $\mathbf{x}(\tau)$, $\tau = k, \ldots, k+T-1$. For that reason, it is only possible to compute one sub-optimal solution using this equation, designated herein by the one-step solution. For more details on the properties of this solution, see Remark 3. Introducing the sparsity constraint (A5d), this solution satisfies

$$\begin{cases} \mathbf{l}_i^T\left[\mathbf{S}(\tau)\mathbf{K}(\tau) - \mathbf{B}^T(\tau)\mathbf{P}(\tau+1)\mathbf{A}(\tau)\right]\mathbf{l}_j = 0, & (i,j) \in \chi, \\ \mathbf{l}_i^T\mathbf{K}(\tau)\mathbf{l}_j = 0, & (i,j) \notin \chi, \end{cases} \quad \tau = k, \ldots, k+T-1, \tag{A8}$$

where $\mathbf{S}(\tau)$ is defined as in Theorem 1. The sub-optimal gain is, then, given by the solution of (A8), which, taking its transpose, can also be written as

$$\begin{cases} \mathbf{l}_j^T\left[\mathbf{K}^T(\tau)\mathbf{S}(\tau) - \mathbf{A}^T(\tau)\mathbf{P}(\tau+1)\mathbf{B}(\tau)\right]\mathbf{l}_i = 0, & (i,j) \in \chi, \\ \mathbf{l}_j^T\mathbf{K}^T(\tau)\mathbf{l}_i = 0, & (i,j) \notin \chi, \end{cases} \quad \tau = k, \ldots, k+T-1. \tag{A9}$$

Note that (A9) has the same form, for the transpose of the gain, as the equation that arises in the LTI formulation of the one-step method for the decentralized estimation problem, put forward in Reference 51 (Theorem 4.1). The closed-form solution (7) follows from that result.

One can also prove, by induction, that

$$J(i) = \mathbf{x}^T(i)\mathbf{P}(i)\mathbf{x}(i), \tag{A10}$$

for $i = k, \ldots, k+T$. First, note that $J(k+T) = \mathbf{x}^T(k+T)\mathbf{P}(k+T)\mathbf{x}(k+T)$, which follows directly from the definition of the finite-horizon performance index (5) and the fact that $\mathbf{P}(k+T) = \mathbf{Q}(k+T)$. Moreover, for $i = k, \ldots, k+T-1$, it follows from (5) and the command action (3) that

$$J(i) = J(i+1) + \mathbf{x}^T(i)\left(\mathbf{Q}(i) + \mathbf{K}^T(i)\mathbf{R}(i)\mathbf{K}(i)\right)\mathbf{x}(i). \tag{A11}$$

Substituting the inductive hypothesis (A10) in (A11) and making use of the closed-loop system dynamics (A6) yields

$$J(i) = \mathbf{x}^T(i)\left(\mathbf{Q}(i) + \mathbf{K}^T(i)\mathbf{R}(i)\mathbf{K}(i) + (\mathbf{A}(i) - \mathbf{B}(i)\mathbf{K}(i))^T\mathbf{P}(i+1)\,(\mathbf{A}(i) - \mathbf{B}(i)\mathbf{K}(i))\right)\mathbf{x}(i),$$

which by comparison with (8) concludes the proof by induction. Hence, for the particular case of $J(k)$, the sub-optimal cost, which is attained using the sub-optimal sequence of gains (7), is given by (9).

APPENDIX B. DERIVATION OF THE ONE-STEP METHOD AS THE CLOSED-FORM SOLUTION TO A RELAXED DECENTRALIZED FINITE-HORIZON LINEAR QUADRATIC REGULATOR PROBLEM

The one-step gain presented in Theorem 1 can be shown to be the closed-form solution to (11). For a time-instant τ, $\mathbf{P}(\tau)$ is given by (8). Taking the derivative of its trace with respect to $\mathbf{K}(\tau)$ yields

$$\frac{\partial}{\partial \mathbf{K}(\tau)}\mathrm{tr}(\mathbf{P}(\tau)) = -2\mathbf{B}^T(\tau)\mathbf{P}(\tau+1)\mathbf{A}(\tau) + 2\mathbf{S}(\tau)\mathbf{K}(\tau). \tag{B1}$$

Define χ to index the nonzero entries of $\mathbf{E}$ as in (A4). Equaling the nonzero entries $(i,j) \in \chi$ of (B1) to zero and introducing the sparsity constraint on the gain matrix yields (A8). The solution to optimization problem (11) is, thus, given by (7).

APPENDIX C. ALTERNATIVE DERIVATION OF THE ONE-STEP SOLUTION TO THE DECENTRALIZED FINITE-HORIZON LINEAR QUADRATIC REGULATOR PROBLEM

Consider the finite-horizon performance index (5). Recall that it can be written as (A10), in which $\mathbf{P}(i)$ is given by the recursive backward-time closed-form expression (8), as shown in Appendix A. To make use of the optimality principle, on which dynamic programming is based, assume the optimal cost-to-go at time instant $i+1$, $J^\star(i+1)$, is known. Note that such optimal cost-to-go depends on the state of the system at time instant $i+1$, $\mathbf{x}(i+1)$. Therefore to put emphasis on this dependence, it is represented henceforth by $J^\star(i+1, \mathbf{x}(i+1))$. By the definition of the finite horizon-performance index (5)

$$J(i, \mathbf{x}(i)) = \mathbf{x}^T(i)\mathbf{Q}(i)\mathbf{x}(i) + \mathbf{u}^T(i)\mathbf{R}(i)\mathbf{u}(i) + J\left(i+1, \mathbf{A}(i)\mathbf{x}(i) + \mathbf{B}(i)\mathbf{u}(i)\right). \tag{C1}$$

Applying the principle of optimality[52(Proposition1.3.1)] to (C1)

$$\begin{aligned} J^\star(i, \mathbf{x}(i)) = & \min_{\substack{\mathbf{K}(\tau) \in \mathrm{Sparse}(\mathbf{E}) \\ \tau = i, \ldots, k+T-1}} J(i, \mathbf{x}(i)) \\ = & \mathbf{x}^T(i)\mathbf{Q}(i)\mathbf{x}(i) + \min_{\mathbf{K}(i) \in \mathrm{Sparse}(\mathbf{E})} \left(\mathbf{u}^T(i)\mathbf{R}(i)\mathbf{u}(i) + \min_{\substack{\mathbf{K}(\tau) \in \mathrm{Sparse}(\mathbf{E}) \\ \tau = i+1, \ldots, k+T-1}} J\left(i+1, \mathbf{A}(i)\mathbf{x}(i) + \mathbf{B}(i)\mathbf{u}(i)\right) \right). \end{aligned}$$

Therefore, the decentralized finite-horizon LQR problem (6) can be written as $T-1$ optimization problems, one for each gain of the finite window,

$$\min_{\mathbf{K}(i) \in \mathrm{Sparse}(\mathbf{E})} \left(\mathbf{u}^T(i)\mathbf{R}(i)\mathbf{u}(i) + J^\star\left(i+1, \mathbf{A}(i)\mathbf{x}(i) + \mathbf{B}(i)\mathbf{u}(i)\right)\right), \tag{C2}$$

$i = k, \ldots, k+T-1$. Making use of the closed-loop system dynamics (A6), as well as of (A10), one can rewrite (C2) as

$$\min_{\mathbf{K}(i)\in \text{Sparse}(\mathbf{E})} \left[\mathbf{x}^T(i)\left(\mathbf{K}^T(i)\mathbf{R}(i)\mathbf{K}(i) + (\mathbf{A}(\tau) - \mathbf{B}(i)\mathbf{K}(i))^T\mathbf{P}(i+1)(\mathbf{A}(i) - \mathbf{B}(i)\mathbf{K}(i))\right)\mathbf{x}(i)\right]. \tag{C3}$$

Note that $\mathbf{P}(i+1)$ does not depend on $\mathbf{K}(i)$, thus, it is possible to solve (C3) for each $\mathbf{K}(i)$ backward in time. Taking the derivative of the objective function of (C3) w.r.t. the unconstrained entries of $\mathbf{K}(i)$ yields

$$\mathbf{l}_l^T\left[\mathbf{R}(i)\mathbf{K}(i)\mathbf{x}(i)\mathbf{x}^T(i) - \mathbf{B}^T(\tau)\mathbf{P}(i+1)\left(\mathbf{A}(i) - \mathbf{B}(i)\mathbf{K}(i)\right)\mathbf{x}(i)\mathbf{x}^T(i)\right]\mathbf{l}_j = 0, \tag{C4}$$

for all $(l,j) \in \chi$, where the set χ is defined as in (A4), and $i = k, \ldots, k+T-1$. Note that (C4) is analogous to the constrained condition on each gain that arises using the Lagrange multiplier approach (A7). Thus it can be solved likewise. It is important to remark that, similarly to the Lagrange multiplier approach, it is only possible to compute a sub-optimal solution to (C4), which is designated by one-step solution, which is shown to be the optimal solution to a relaxed version of the original regulator problem (see Remark 3). Having this in mind, the one-step method attempts to minimize $J(k)$ by computing a sequence of gains backward in time. Starting at $i = k+T-1$, for each backward step i, $\mathbf{K}(i)$ is computed such that it attempts to minimize $J(i)$ and follows the imposed sparsity constraint.

APPENDIX D. DERIVATION OF THE LINEAR QUADRATIC TRACKER FEEDFORWARD TERMS

The following derivation makes use of the Lagrange multiplier approach. The Lagrangian function of the optimization problem can be written as

$$\begin{aligned}\mathcal{L}(k) = &\frac{1}{2}\sum_{\tau=k}^{k+T-1}\left(\overline{\mathbf{x}}(\tau+1) - \overline{\mathbf{x}}(\tau) - \mathbf{B}(\tau)\mathbf{u_a}(\tau)\right)^T\mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1)\left(\overline{\mathbf{x}}(\tau+1) - \overline{\mathbf{x}}(\tau) - \mathbf{B}(\tau)\mathbf{u_a}(\tau)\right)\\ &+\sum_{\tau=k}^{k+T}\left(\lambda^T(\tau)\left((\mathbf{A}(\tau)-\mathbf{I})\overline{\mathbf{x}}(\tau) + \mathbf{B}(\tau)\overline{\mathbf{u}}(\tau)\right) + \boldsymbol{\gamma}^T(\tau)\left(\mathbf{H}(\tau)\overline{\mathbf{x}}(\tau) - \mathbf{r}(\tau)\right)\right).\end{aligned} \tag{D1}$$

The necessary conditions for the constrained minimum follow from (D1)

$$\begin{aligned}\frac{\partial\mathcal{L}(k)}{\partial\lambda(\tau)} &= (\mathbf{A}(\tau)-\mathbf{I})\overline{\mathbf{x}}(\tau) + \mathbf{B}(\tau)\overline{\mathbf{u}}(\tau) = \mathbf{0}, \quad \tau = k, \ldots, k+T,\\
\frac{\partial\mathcal{L}(k)}{\partial\boldsymbol{\gamma}(\tau)} &= \mathbf{H}(\tau)\overline{\mathbf{x}}(\tau) - \mathbf{r}(\tau) = \mathbf{0}, \quad \tau = k, \ldots, k+T,\\
\frac{\partial\mathcal{L}(k)}{\partial\overline{\mathbf{x}}(k)} &= \mathbf{H}^T(k+1)\mathbf{H}(k+1)\overline{\mathbf{x}}(k) - \mathbf{H}^T(k+1)\mathbf{H}(k+1)\overline{\mathbf{x}}(k+1) + \mathbf{H}^T(k+1)\mathbf{H}(k+1)\mathbf{B}(k)\mathbf{u_a}(k)\\
&\quad + (\mathbf{A}(k)-\mathbf{I})^T\lambda(k) + \mathbf{H}^T(k)\boldsymbol{\gamma}(k) = \mathbf{0},\\
\frac{\partial\mathcal{L}(k)}{\partial\overline{\mathbf{x}}(\tau)} &= \mathbf{H}^T(\tau)\mathbf{H}(\tau)\overline{\mathbf{x}}(\tau) + \mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1)\overline{\mathbf{x}}(\tau) - \mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1)\overline{\mathbf{x}}(\tau+1) - \mathbf{H}^T(\tau)\mathbf{H}(\tau)\overline{\mathbf{x}}(\tau-1)\\
&\quad + \mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1)\mathbf{B}(\tau)\mathbf{u_a}(\tau) - \mathbf{H}^T(\tau)\mathbf{H}(\tau)\mathbf{B}(\tau-1)\mathbf{u_a}(\tau-1) + (\mathbf{A}(\tau)-\mathbf{I})^T\lambda(\tau)\\
&\quad + \mathbf{H}^T(\tau)\boldsymbol{\gamma}(\tau) = \mathbf{0}, \quad \tau = k+1, \ldots, k+T-1,\\
\frac{\partial\mathcal{L}(k)}{\partial\overline{\mathbf{x}}(k+T)} &= \mathbf{H}^T(k+T)\mathbf{H}(k+T)\overline{\mathbf{x}}(k+T) - \mathbf{H}^T(k+T)\mathbf{H}(k+T)\overline{\mathbf{x}}(k+T-1)\\
&\quad - \mathbf{H}^T(k+T)\mathbf{H}(k+T)\mathbf{B}(k+T-1)\mathbf{u_a}(k+T-1) + (\mathbf{A}(k+T)-\mathbf{I})^T\lambda(k+T)\\
&\quad + \mathbf{H}^T(k+T)\boldsymbol{\gamma}(k+T) = \mathbf{0},\\
\frac{\partial\mathcal{L}(k)}{\partial\overline{\mathbf{u}}(\tau)} &= \mathbf{B}^T(\tau)\lambda(\tau) = \mathbf{0}, \quad \tau = k, \ldots, k+T,\\
\frac{\partial\mathcal{L}(k)}{\partial\boldsymbol{u_a}(\tau)} &= \mathbf{B}^T(\tau)\mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1)\mathbf{B}(\tau)\mathbf{u_a}(\tau) - \mathbf{B}^T(\tau)\mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1)\overline{\mathbf{x}}(\tau+1)\\
&\quad + \mathbf{B}^T(\tau)\mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1)\overline{\mathbf{x}}(\tau) = \mathbf{0}, \quad \tau = k, \ldots, k+T-1,\end{aligned}$$

which can be written has the system of linear equations (19). As shown in Section 4.1, the linear equality constraint (13) has, at least, one solution $(\overline{\mathbf{x}}(\tau), \overline{\mathbf{u}}(\tau))$. Furthermore, let f_k denote the objective function of the optimization problem, that is, $f_k := \sum_{\tau=k}^{k+T-1} \mathbf{d}(\tau)^T \mathbf{H}^T(\tau+1)\mathbf{H}(\tau+1)\mathbf{d}(\tau)$. Note that f_k features positive semidefinite matrices $\mathbf{H}^T(\tau)\mathbf{H}(\tau)$, for $\tau = k+1, \ldots, k+T$, since $\mathbf{H}(\tau)$ has full rank for $\tau \in \mathbb{N}_0$. For that reason, there is at least one global constrained minimum for the minimization problem (18). Thus, at least one of the solutions to (19) corresponds to the global minimum.

Let p be the dimension and $\{\mathbf{v}_1, \ldots, \mathbf{v}_p\}$ a basis of the null space of matrix $\mathbf{G}$. The solutions to (19) are of the form

$$\overline{\chi}^\star = \overline{\chi}^* + \sum_{i=1}^{p} t_i \mathbf{v}_i \,,$$

where $\overline{\chi}^*$ is a particular solution and $t_1, \ldots, t_p \in \mathbb{R}$. All the solutions to (19) are critical points, that is,

$$\left.\frac{\partial f_k}{\partial \overline{\chi}}\right|_{\overline{\chi}=\overline{\chi}^\star} = \mathbf{0}.$$

Therefore, using the chain rule,

$$\frac{\partial f_k}{\partial t_i} = \left.\frac{\partial f_k}{\partial \overline{\chi}}\right|_{\overline{\chi}=\overline{\chi}^\star} \cdot \frac{\partial \overline{\chi}}{\partial t_i} = \mathbf{0} \cdot \mathbf{v}_i = 0,$$

for $i = 1, \ldots, p$. Hold every t_j constant, with $j \in \{1, \ldots, p\} \setminus \{i\}$. Then, given that f_k is continuous and differentiable for $t_i \in \mathbb{R}$, f_k is constant over the set of solutions to (19), one of which is the global minimum. Therefore all of the solutions to (19) achieve global optimality.