Digital Chemical Engineering 9 (2023) 100133

Contents lists available at ScienceDirect

Digital Chemical Engineering

journal homepage: www.elsevier.com/locate/dche

Original article

Encrypted distributed model predictive control with state estimation for nonlinear processes

Yash A. Kadakia [a], Aisha Alnajdi [b], Fahim Abdullah [a], Panagiotis D. Christofides [a,b,*]

[a] Department of Chemical and Biomolecular Engineering, University of California, Los Angeles, CA, 90095-1592, USA
[b] Department of Electrical and Computer Engineering, University of California, Los Angeles, CA 90095-1592, USA

ARTICLE INFO

Keywords:
Model predictive control
Encrypted control
State estimation
Distributed control
Cybersecurity
Process control

ABSTRACT

This research focuses on encrypted distributed control architectures, aimed at enhancing the operational safety, cybersecurity and computational efficiency of large-scale nonlinear systems, where only partial state measurements are available. In this setup, a distributed model predictive controller (DMPC) is utilized to partition the process into multiple subsystems, each controlled by a distinct Lyapunov-based MPC (LMPC). To consider the interactions among different subsystems, each controller receives and shares with the other controllers control inputs computed for its particular subsystem. As full state feedback is not available, we integrate an extended Luenberger observer with each LMPC, initializing the LMPC model with complete state estimate information provided by the observer. Furthermore, to enhance cybersecurity, wireless signals received and transmitted by the controllers are encrypted. Guidelines are established to implement this proposed control structure in any large-scale nonlinear chemical process network. Simulation results, conducted on a specific nonlinear chemical process network, demonstrate the effective closed-loop performance of the encrypted DMPC with state estimation, utilizing partial state feedback with sensor noise. This is followed by a comprehensive comparison of the closed-loop performance, control input computational time, and suitability of encrypted centralized, decentralized, and distributed MPC frameworks.

1. Introduction

Industrial control systems for large-scale processes have been subject to extensive research over the past decades, with the primary objectives of enhancing operational safety, promoting environmental sustainability, optimizing profitability, and economizing on utility costs. Nonetheless, the evolution of technology has led to the integration and interlinking of industrial control systems with corporate networks and the internet, to create cyber–physical systems that have streamlined monitoring, control, and automation of complex processes, enhancing productivity and operational efficiency. However, the increased connectivity and linking of these systems have made them vulnerable to cyberthreats, given their extensive reliance on networked communication. A breach or compromise in these systems can have severe consequences, including the disruption of essential services, physical damage, financial losses, and are even a threat to public safety. As a result, the past few years have witnessed a surge in research efforts directed towards enhancing the cybersecurity of industrial control systems.

Recent developments in cyberattack techniques underscore the need to establish robust cybersecurity (Gandhi et al., 2011). Dealing with cybersecurity issues within industrial control systems is mainly within the realm of operational technology (OT). While there have been notable advancements in improving cybersecurity in the information technology (IT) sector, which centers on the software elements of systems, including aspects like network architecture and data management, cybersecurity within the OT domain is currently trailing behind (Conklin, 2016). Numerous real-world examples highlight the need of cybersecurity in networked cyber–physical systems and SCADA (Supervisory Control and Data Acquisition) systems. These include the 2015 cyberattacks on SCADA controls responsible for managing the power grid in Ukraine, leading to widespread power outages (Khan et al., 2016). Likewise, in the 2021 DarkSide ransomware attack on Colonial Pipeline, cyberattackers encrypted its networked communication and demanded a ransom for the decryption keys. As a result, Colonial Pipeline was compelled to suspend its operations, resulting in interruptions to fuel distribution and financial losses (Tsvetanov and Slaria, 2021).

Traditional control systems, like proportional–integral–derivative (PID) control, have long been used in chemical plants to control processes with a decentralized structure. In this setup, each controller uses

* Corresponding author at: Department of Chemical and Biomolecular Engineering, University of California, Los Angeles, CA, 90095-1592, USA.
E-mail address: pdc@seas.ucla.edu (P.D. Christofides).

https://doi.org/10.1016/j.dche.2023.100133
Received 26 October 2023; Received in revised form 4 November 2023; Accepted 4 November 2023
Available online 7 November 2023

one process measurement and calculates actions to control that specific state at its desired set point. However, PIDs do not consider how the controlled variable interacts with other states, limiting their ability to optimize control inputs for a multi-input-multi-output (MIMO) system. To address this limitation, model predictive controllers (MPCs) have been applied to manage complex processes. MPCs employ models, derived from first principles, data, or mathematical representations, to predict future states within a specified horizon. They then optimize control inputs using real-time sensor feedback while accounting for interactions among all the process states and inputs. This approach not only elevates control precision but also mitigates utility expenses, ultimately enhancing overall process performance.

However, acquiring sensor measurements for all states in large processes can incur significant costs. Furthermore, installing the necessary instrumentation and equipment to collect and transmit measurements may not always be feasible, particularly in specific areas of the plant or process. As a result, extensive research has been carried out on state estimation techniques, enabling real-time prediction of unmeasured states though deterministic and stochastic estimation methods. Notably, the extended Kalman Filter (EKF) and extended Luenberger observer (ELO) stand out as commonly used state estimators for nonlinear processes. The EKF utilizes a stochastic approach, employing a linearized approximation from continuous time to a discrete-time system to estimate the state. This method can account for system and measurement noise using probabilistic approaches. Conversely, the ELO adopts a deterministic approach, using nonlinear model dynamics to estimate states without explicitly addressing stochastic disturbances or measurement noise. While the EKF can account for sensor noise better, the ELO handles nonlinearity by directly incorporating it in the observer equations. More details on the advantages, drawbacks, and similarities about various state estimation methods has been discussed in the works of Radke and Gao (2006) and Ali et al. (2015). To attain the desired performance using these methods, a mathematical model for the specific system is typically required to describe process dynamics within a defined operating range. Nevertheless, when integrated with MPC, the MPC model can be extended for use by the state estimator, and vice versa, enabling a collaborative and effective solution.

Since MPCs employ nonlinear optimization for control input optimization in nonlinear processes, in large-scale systems, where numerous control inputs must be calculated, the control problem can become too extensive and intricate to be solved within the given sampling time. As a response, decentralized and distributed MPC strategies have been introduced to break down the complex problem into smaller segments, handled by different computing units. In such arrangements, the system to be controlled is partitioned into smaller subsystems, where the control input of each subsystem is computed separately. Decentralized MPCs compute control inputs for their respective subsystems without any knowledge of the control inputs being applied by other subsystems. This limits the controller from taking into account interactions among different process subsystems and only considers interactions within its specific subsystem. In contrast, distributed controllers share information about the control inputs computed for their subsystem, enabling other controllers to optimize their control inputs accordingly. This collaborative approach improves the handling of interdependencies among various process subsystems.

Significant research efforts have been devoted to various domains of cybersecurity, and process control, including the development of machine learning-based cyberattack detectors (Al-Abassi et al., 2020; Dutta et al., 2020), the implementation of nonlinear encrypted centralized MPCs (Suryavanshi et al., 2023), the utilization of sequential and iterative DMPCs (Liu et al., 2010), and the application of nonlinear state estimators (Zeitz, 1987; Kazantzis and Kravaris, 1998). However, to the best of our knowledge, the development of distributed control systems that employ encrypted networked communication for large-scale nonlinear processes with partial state feedback remains an unexplored area, prompting our proposal for a novel control structure to address this challenge. Specifically, we propose a distributed control structure comprising a set of Lyapunov-based MPCs, integrated with an extended Luenberger observer, utilizing encrypted networked communication. In this configuration, we assume the presence of secure edge computers responsible for computing control inputs and receiving and transmitting encrypted signals. Integrating observer-based state estimation within this setup serves to provide each LMPC with complete state information in real-time. To address interactions within different subsystems in large processes and reduce the complexity associated with centralized control problems, we employ a distributed MPC. Further, the incorporation of encryption within the networked communication channels enhances cybersecurity as each edge computing unit receives and transmits encrypted wireless signals.

The remainder of this paper is structured as follows: In Section 2, we provide an overview of various aspects, including notation, the considered class of nonlinear systems, system stabilizability assumptions, the formulation of the extended Luenberger observer, the employed encryption cryptosystem, and the implications of quantization. Section 3 delves into the design of the encrypted distributed MPC, outlining the formulation of sequential and iterative DMPCs utilizing state estimates from the observer, and further detailing the extended Luenberger observer. In Section 4, we present and discuss closed-loop simulations for a nonlinear chemical process network with partial state feedback in the presence of sensor noise with the encrypted sequential and iterative DMPCs. In Section 5, we conduct a comparative assessment of the encrypted control strategies, encompassing centralized, decentralized, and distributed MPCs.

2. Preliminaries

2.1. Notation

The symbol $\|\cdot\|$ represents the Euclidean norm of a vector. $x^\top$ denotes the transpose of a vector x. $\mathbb{R}$, $\mathbb{Z}$, and $\mathbb{N}$ represent the sets of real numbers, integers, and natural numbers, respectively. $\mathbb{Z}_M$ denotes the additive groups of integers modulo M. Set subtraction is indicated by the symbol "\", where $A\backslash B$ represents the set of elements that are in set A but not in set B. A function, $f(\cdot)$, falls under the class C^1 if it is continuously differentiable within its defined domain. The term $\mathrm{lcm}(i,j)$ denotes the least common multiple of the integers i and j, while $\gcd(i,j)$ signifies the greatest common divisor, that divides i and j without any remainder.

2.2. Class of systems

This study is centered on multi-input multi-output (MIMO) systems, which are characterized by a category of continuous-time nonlinear systems represented in state-space form as follows:

$$\dot{x} = F(x,u) = f(x) + g(x)u \tag{1a}$$

$$y = h(x) + w \tag{1b}$$

The state vector is denoted by $x = [x_1, \ldots, x_n] \in \mathbb{R}^n$, while $u \in \mathbb{R}^m$ represents the control input vector bounded by the set, $U \subset \mathbb{R}^m$. The output vector consisting of the state measurements that are continuously sampled is $y = [y_1, \ldots, y_q] \in \mathbb{R}^q$, and $w \in \mathbb{R}^q$ is the measurement noise vector. $F(x,u)$ is a nonlinear function with respect to x and u, rendering the origin as a steady state of Eq. (1). Without loss of generality, we assume the initial time as zero ($t_0 = 0$). The functions $f(\cdot)$, $g(\cdot)$, and $h(\cdot)$ are matrices of dimension $n \times 1$, $n \times m$, and $q \times 1$, respectively. Additionally, we define the set $S(\Delta)$ as the set of piecewise constant functions characterized by a period of Δ. We consider $j = 1, \ldots, N_{sys}$ sub-systems, where each subsystem j is regulated only by inputs u_j but potentially impacted by inputs of other subsystems due to coupling between subsystems. The control input vector for the j^{th} subsystem is $u_j \in \mathbb{R}^{m_j}$. $u = [u_1^\top \ldots u_{N_{sys}}^\top]^\top \in \mathbb{R}^m$ is the control input vector

for the entire system, with $m = \sum_{j=1}^{N_{sys}} m_j$. The control input vector constraints are $u_j \in U_j := \{u_{\min,j_i} \le u_{j_i} \le u_{\max,j_i}, \forall i = 1, 2, \dots, m_j\} \in \mathbb{R}^{m_j}$, $\forall j = 1, \dots, N_{sys}$. Hence, the set U that constrains the control input vector for the entire system is formed by the union of sets U_j, where $j = 1, \dots, N_{sys}$.

2.3. Extended luenberger observer

The extended Luenberger observer (ELO) was introduced as a natural extension of the Luenberger observer, originally developed based on a linear approximation of processes (Zeitz, 1987; Dochain, 2003). The primary objective of a state observer, such as the ELO, is to estimate the unmeasured internal states of a given system. This estimation is achieved by leveraging the available measured states from the process, in combination with the applied inputs. The formulation of the Extended Luenberger observer for a nonlinear system is expressed through Eq. (2), presenting a means to capture and estimate the system's unmeasured internal states in the following manner:

$$\dot{\bar{x}} = F(\bar{x}, u) + K(y - h(\bar{x})) \tag{2}$$

where $\bar{x} \in \mathbb{R}^n$ represents the estimated state vector, and the observer gain matrix is $K \in \mathbb{R}^{n \times q}$. Eq. (2) comprises two key components: the initial term corresponds to the process model dependent on the estimated states and applied control inputs, while the final term serves as the output prediction error, functioning as a correction term.

The objective of the ELO is to minimize the estimation error, $e = x - \bar{x}$, in which the time-derivative of the error is determined by the following equation (Dochain, 2003):

$$\dot{e} = F(\bar{x} + e, u) - F(\bar{x}, u) - K(h(\bar{x} + e) - h(\bar{x})) \tag{3}$$

For the estimation error, e, to decay to zero, the time-derivative of the error (shown in Eq. (3)) must also decay to zero. Therefore, the observer gain matrix K must be designed accordingly. To design K, Eq. (3) can be simplified to the following equation by linearizing the process model at a fixed point:

$$\dot{e} = (A - KL)e \tag{4}$$

where $A = \frac{\partial F(x,u)}{\partial x}\Big|_{x=\bar{x}}$ and $L = \frac{\mathrm{d}h(x)}{\mathrm{d}x}\Big|_{x=\bar{x}}$ are linearized terms of the nonlinear system evaluated at a specific reference point (in general, $L = \partial h(x,u)/\partial x|_{x=\bar{x}}$), typically the operating steady state of the system. Subsequently, the selection of the observer gain matrix K is conducted in a manner that ensures that all the eigenvalues of the matrix $A - KL$ have strictly negative real components.

2.4. Stability assumptions

Based on how the overall large-scale system is partitioned, there may exist interacting dynamics between the subsystems, as the states and control inputs of one subsystem may impact the states and control inputs of other subsystems. Accounting for these interactions, we assume the existence of an observer and feedback stabilizing control law $u = \Phi(\bar{x})$ for the overall system with $u_j = \Phi_j(\bar{x}) \in U_j$, which regulate the individual subsystems $j = 1, \dots, N_{sys}$, such that the origin of the overall system of Eq. (1) is rendered exponentially stable. This signifies the presence of a C^1 control Lyapunov function $V(x)$ for which the following inequalities hold for all $x, \bar{x} \in \mathbb{R}^n$ within an open region D surrounding the origin:

$$c_1|x|^2 \le V(x) \le c_2|x|^2, \tag{5a}$$

$$\frac{\partial V(x)}{\partial x} f(x, \Phi(\bar{x})) \le -c_3|x|^2, \tag{5b}$$

$$\left|\frac{\partial V(x)}{\partial x}\right| \le c_4|x| \tag{5c}$$

where c_1, c_2, c_3, and c_4 are positive constants. $\Phi(\bar{x}) = [\Phi_1(\bar{x})^\top, \dots, \Phi_{N_{sys}}(\bar{x})^\top]^\top$ is the vector concatenating the stabilizing feedback control laws for all N_{sys} subsystems. For the nonlinear system described by Eq. (1), the region of closed-loop stability can be defined as a level set, Ω_ρ, of the control Lyapunov function V, such that $\Omega_\rho := \{x \in D | V(x) \le \rho\}$, where $\rho > 0$. Hence, originating from any initial condition within Ω_ρ, the control input, $\Phi(\bar{x})$, guarantees that the state trajectory of the closed-loop system remains within Ω_ρ.

Remark 1. The assumption of an output feedback controller satisfying Eq. (5) involves two key requirements. First, it mandates that the observer states remain bounded within the region Ω_ρ. Second, it necessitates that the estimated error, denoted as e and defined as the difference between x and $\bar{x}$, converges to zero within a finite timeframe, regardless of the initial condition within Ω_ρ. To ensure the fulfillment of these prerequisites, a series of random closed-loop trajectories are generated for the nonlinear system described in Eq. (1) under the observer and state feedback controller, and it is ensured that all trajectories converge within Ω_ρ in a finite number of sampling periods with $e \to 0$. More details regarding the observer and controller tuning are presented in Section 4.

2.5. Paillier cryptosystem

In this research, we employ the Paillier cryptosystem (Paillier, 1999) to encrypt signals, specifically state measurements (x) and control inputs (u), transmitted to and from the controllers. Although we do not make use of the semi-homomorphic property of additive homomorphism within the Paillier cryptosystem, we employ it so that traditional controllers, such as proportional-integral controllers, which can conduct computations in an encrypted space, can be integrated into the overall control architecture if required. The encryption procedure is initiated by generating the public and private key. The public key is used to encrypt integer messages into ciphertexts, and the private key is employed to decrypt ciphertexts and retrieve the original integer messages. The process of generating the public and private key can be outlined as follows:

1. Choose two large prime integers (p and q) randomly, ensuring $\gcd(pq, (p-1)(q-1)) = 1$.
2. Compute $M = pq$.
3. Choose an arbitrary integer $\bar{g}$ such that $\bar{g} \in \mathbb{Z}_{M^2}$, which is the multiplicative group of integers modulo M^2.
4. Compute $\lambda = \mathrm{lcm}(q-1, p-1)$.
5. Specify $\bar{L}(x) = (x-1)/M$.
6. Verify the existence of the subsequent modular multiplicative inverse, $u = (\bar{L}(\bar{g}^\lambda \bmod M^2))^{-1} \bmod M$.
7. If the inverse does not exist, revisit step 3 and select an alternate value of $\bar{g}$. If the inverse exists, $(M, \bar{g})$ is the public key and (λ, u) is the private key.

Once the keys are acquired, the public and private keys are distributed to authorized recipients for encryption and decryption, respectively. The encryption process is as follows:

$$E_M(m, r) = c = \bar{g}^m r^M \bmod M^2 \tag{6}$$

where r is a randomly selected integer from the set $\mathbb{Z}_M$, and c represents the ciphertext achieved through the encryption of m. The decryption procedure is as follows:

$$D_M(c) = m = \bar{L}(c^\lambda \bmod M^2)u \bmod M \tag{7}$$

Remark 2. The significance of encryption lies in safeguarding data privacy against potential cyberattacks, particularly sophisticated attacks that might go undetected by traditional cybersecurity measures. In scenarios where constant values are transmitted during steady-state operations, conventional methods might result in the transmission of

the same values after data transformations, mathematical operations or mapping of data to a certain set. However, in encryption, the generation of a random number each time data is encrypted ensures that identical numbers, when encrypted, yield distinct ciphertexts, bolstering cybersecurity measures significantly.

Remark 3. Various encryption methods, such as symmetric encryption, fully homomorphic encryption, and partially homomorphic encryption, can be employed to secure data. Symmetric encryption, like AES (Advanced Encryption Standard), is a non-homomorphic encryption technique that does not allow mathematical operations within an encrypted space. In contrast, fully homomorphic encryption, exemplified by schemes like BGV (Brakerski-Gentry-Vaikuntanathan), permits both addition and multiplication operations within an encrypted environment. Meanwhile, partially homomorphic encryption enables either multiplication or addition operations within the encrypted domain. For instance, the Paillier cryptosystem allows addition operations in an encrypted space. While our work does not utilize the semi-homomorphic property of the Paillier cryptosystem, it has been recently considered for integrating linear controllers like Proportional-Integral (PI) control in large-scale systems alongside nonlinear controllers such as MPCs. This integration allows for control input computations for the linear controller within an encrypted space without decryption, as demonstrated in the work of Kadakia et al. (2024).

2.6. Quantization

To use the Paillier cryptosystem, data to be encrypted must be in the form of natural numbers in $\mathbb{Z}_M$. However, the signal values before encryption are in floating-point. Consequently, we employ quantization, mapping the floating-point numbers into $\mathbb{Z}_M$ (Darup et al., 2017). Using a signed fixed-point binary representation, we create a set, $\mathbb{Q}_{l_1,d}$, with parameters l_1 and d. These parameters define the total bit count (integer and fractional) and the fractional bits, respectively. The $\mathbb{Q}_{l_1,d}$ set encompasses rational numbers from -2^{l_1-d-1} to $2^{l_1-d-1}-2^{-d}$, separated by 2^{-d}. A rational number q in $\mathbb{Q}_{l_1,d}$ can be expressed as $q \in \mathbb{Q}_{l_1,d}$, where $\exists \beta \in \{0,1\}^{l_1}$, and $q = -2^{l_1-d-1}\beta_{l_1} + \sum_{i=1}^{l_1-1} 2^{i-d-1}\beta_i$. To map a real number data point a to the $\mathbb{Q}_{l_1,d}$ set, we use the function $g_{l_1,d}$, defined by the equation,

$$\begin{aligned} g_{l_1,d} &: \mathbb{R} \to \mathbb{Q}_{l_1,d} \\ g_{l_1,d}(a) &:= \arg\min_{q \in \mathbb{Q}_{l_1,d}} |a-q| \end{aligned} \tag{8}$$

Next, the quantized data is transformed into a set of integers through a one-to-one (bijective) mapping known as $f_{l_2,d}$, as outlined in Darup et al. (2017). The following mapping ensures that the quantized data is transformed into a subset of the message space $\mathbb{Z}_M$:

$$\begin{aligned} f_{l_2,d} &: \mathbb{Q}_{l_1,d} \to \mathbb{Z}_{2^{l_2}} \\ f_{l_2,d}(q) &:= 2^d q \bmod 2^{l_2} \end{aligned} \tag{9}$$

During the encryption process, integer plaintext messages from the set $Z_{2^{l_2}}$ are converted to ciphertexts, which can be decrypted back into the same set $Z_{2^{l_2}}$. To recover the original data from the set $\mathbb{Q}_{l_1,d}$, an inverse mapping, denoted as $f^{-1}_{l_2,d}$, is defined as follows:

$$f^{-1}_{l_2,d} : \mathbb{Z}_{2^{l_2}} \to \mathbb{Q}_{l_1,d} \tag{10}$$

$$f^{-1}_{l_2,d}(m) := \frac{1}{2^d} \begin{cases} m - 2^{l_2} & \text{if } m \geq 2^{l_2-1} \\ m & \text{otherwise} \end{cases} \tag{11}$$

3. Development of the encrypted distributed control architectures with state estimation

In this section, we describe the design and formulation of the encrypted distributed control architectures, both encrypted sequential and iterative distributed LMPCs with state estimation, provide additional details on the extended Luenberger observer.

3.1. Design of the encrypted sequential distributed LMPC

The control architecture of the encrypted sequential distributed LMPC is depicted in Fig. 1. In a sequential distributed framework involving various LMPCs, communication is unidirectional. Specifically, the optimal control trajectory derived from solving the optimization problem for one LMPC is transmitted to another LMPC. This information is subsequently utilized by the receiving LMPC to proceed with its own optimization problem. The control strategy adheres to the following sequence of steps:

1. At time $t = t_k$, where k represents the sampling instance, signals $y(t_k)$ from sensors are encrypted to ciphertext c using the public key and transmitted to each control subsystem, within its respective edge computing unit.
2. Within each unit, the encrypted signals are decrypted using the private key, and the quantized states $\hat{y}(t_k)$ are used by the state estimator along with the control inputs computed at the previous sampling instance $u(t_{k-1})$ to estimate the current value of the states $\bar{x}(t_k)$.
3. The LMPC of the N^{th}_{sys} subsystem evaluates the optimal control trajectory $u^*_{N_{sys}}$ using the estimated states $\bar{x}$ at $t = t_k$, and the stabilizing control law for the other $N_{sys}-1$ subsystems, encrypts the control action of the first sampling period $u^*_{N_{sys}}(t_k)$ using the public key, transmits the ciphertext to the corresponding actuator, and transmits the entire optimal trajectory $u^*_{N_{sys}}(t|t_k)$, $t \in [t_k, t_{k+N})$ to the $N_{sys}-1^{\text{th}}$ LMPC through the Ethernet crossover cable connection established between the different computing units.
4. The $N_{sys}-1^{\text{th}}$ LMPC receives the entire optimal trajectory of the N^{th}_{sys} LMPC and evaluates the optimal trajectory $u_{N_{sys}-1}$ using the estimated states $\bar{x}(t_k)$ and the optimal input trajectory of the N^{th}_{sys} subsystem. It assumes the stabilizing control law for the remaining $N_{sys}-2$ subsystems. It then encrypts the optimal trajectory for its respective subsystem over the next sampling period using the public key and transmits the complete optimal trajectory of subsystems N_{sys} and $N_{sys}-1$ to the $N_{sys}-2^{\text{th}}$ LMPC.
5. This same process is repeated up to the 1^{st} LMPC, which receives the optimal control input trajectories of all the other subsystems and computes its own optimal trajectory using the estimated states $\bar{x}(t_k)$ and the optimal control input trajectories of all the other subsystems.
6. At the actuator, the ciphertext $\acute{c}$ is decrypted to the quantized input $\hat{u}(t_k)$ using the private key, which is then applied to the process.

Formulation of the optimization problem, its constraints, and additional details of the encrypted sequential LMPC is presented in Section 3.4.

3.2. Design of the encrypted iterative distributed LMPC

The control architecture of the encrypted iterative distributed LMPC is depicted in Fig. 2. In this framework, all controllers communicate with each other to cooperatively optimize the control actions. The controllers solve their respective optimization problems independently within a parallel framework, and solutions for each control problem are exchanged at the end of each iteration. The control strategy adheres to the subsequent sequence of steps:

1. At time $t = t_k$, where k represents the sampling instance, signals $y(t_k)$ from sensors are encrypted to ciphertext c using the public key and transmitted to each control subsystem, within its respective edge computing unit.
2. Within each unit, the encrypted signals are decrypted using the private key, and the quantized states $\hat{y}(t_k)$ are used by the state estimator along with the control inputs computed at the previous sampling instance $u(t_{k-1})$ to estimate the current value of all the system states $\bar{x}(t_k)$.

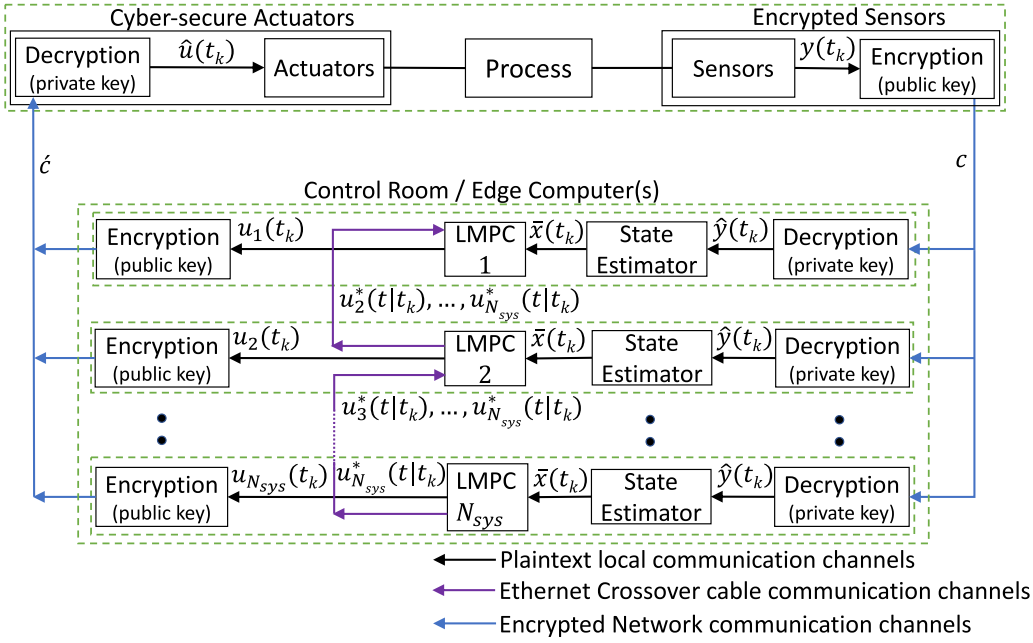

Fig. 1. Illustration of the encrypted sequential distributed control structure.

3. At iteration $z = 1$, the k^{th} LMPC in the k^{th} subsystem evaluates optimal control input trajectories $u_k^*(t)$, using the estimated states $\bar{x}(t_k)$, and assuming $u_j(t) = \Phi_j(\bar{x}(t))$ where $j \in \{1, \ldots, N_{sys}\}, \quad j \neq k$. At the end of the first iteration, each subsystem transmits its complete optimal control input trajectory to all N_{sys} subsystems through the Ethernet crossover cable connection established between the different control subsystems.
4. At iteration $z = 2$, the k^{th} LMPC in the k^{th} subsystem re-evaluates optimal control input trajectories $u_k^*(t)$ using the estimated states $\bar{x}(t_k)$, and the optimal control input trajectories $u_j^*(t)$ where $j \in \{1, \ldots, N_{sys}\}, \quad j \neq k$. At the end of the second iteration, each subsystem transmits its complete optimal control input trajectory to all N_{sys} subsystems. This process is continued until a termination criterion is satisfied. The termination criterion can be either that the number of iterations, denoted as z, must not exceed the maximum number of iterations, denoted as $z_{\max}$, or that the difference in the value of the cost function between two consecutive iterations is smaller than a threshold value.
5. After the termination criterion is satisfied, each LMPC encrypts its control input corresponding to the lowest cost function over the next sampling period (using the public key), and the encrypted ciphertext is transmitted to the corresponding actuators of that particular subsystem.
6. At the actuator, the ciphertext $\acute{c}$ is decrypted to the quantized input $\hat{u}(t_k)$ using the private key, which is then applied to the process.

Formulation of the optimization problem, its constraints, and additional details for the iterative encrypted DMPC are presented in Section 3.5.

Remark 4. In the closed-loop block diagrams shown in Figs. 1 and 2, Ethernet crossover cable connections facilitate communication between the computing units of different subsystems. This setup assumes a secure edge computer(s) within a protected control room, where encrypted signals from sensors at the process site are received and from where encrypted control inputs are transmitted to the actuators. However, communication between subsystems responsible for computing control inputs remains unencrypted. The rationale behind this decision is to minimize the communication overhead due to encryption–decryption in the control system. Complete control input trajectories must be communicated multiple times within a single sampling period in the case of iterative DMPC. Encrypting and decrypting these trajectories repeatedly within a single sampling period may not be feasible, particularly for very large systems. Such repetition could lead to increased communication overhead. Since the primary objective of a DMPC is to distribute the optimization problem among separate computing units and solve each one effectively, the assumption of having all responsible edge computing units in a secure room with secure cable connections between them is reasonable. Alternatively, the option to encrypt and decrypt inputs could be considered if the initial arrangement is not achievable. Further insights into the communication and computational implications associated with encryption and decryption are available in Kadakia et al. (2023).

The closed-loop design of Figs. 1 and 2 introduces two sources of error: one from state quantization in the sensor–controller link and another from control input quantization in the controller–actuator link. These errors are bounded by:

$$|y(t_k) - \hat{y}(t_k)| \leq 2^{-d-1} \tag{12a}$$

$$|u(t_k) - \hat{u}(t_k)| \leq 2^{-d-1} \tag{12b}$$

The state estimator, as expressed in Eq. (2), can be written as a function $\phi(\bar{x}, y, u)$. An additional error arises in the applied control input, as the state estimator receives $\hat{y}$ instead of the true state y to estimate all the system states. Using the local Lipschitz property, this error will be confined by the underlying equation, where $L_1' > 0$:

$$|\phi(\bar{x}, \hat{y}, u) - \phi(\bar{x}, y, u)| \leq L_1' |\hat{y} - y| \leq L_1' 2^{-d-1} \tag{13}$$

Remark 5. A quantization error occurs when the value to be quantized does not precisely match a member of the set $\mathbb{Q}_{l_1,d}$. The elements in this set are spaced apart by 2^{-d}, which represents the resolution of the set. Let us assume the value to be quantized is denoted as a, and it falls within the range of b to $b + 2^{-d}$. If the absolute difference between a and b is smaller than the difference between a and $b + 2^{-d}$, a is assigned to the value b. Otherwise, it is assigned to the value $b + 2^{-d}$. Consequently, the maximum potential discrepancy between the actual and quantized values is half of the resolution, which is equal to 2^{-d-1}. This limitation also implies that a greater value of d would lead to a reduced quantization error.

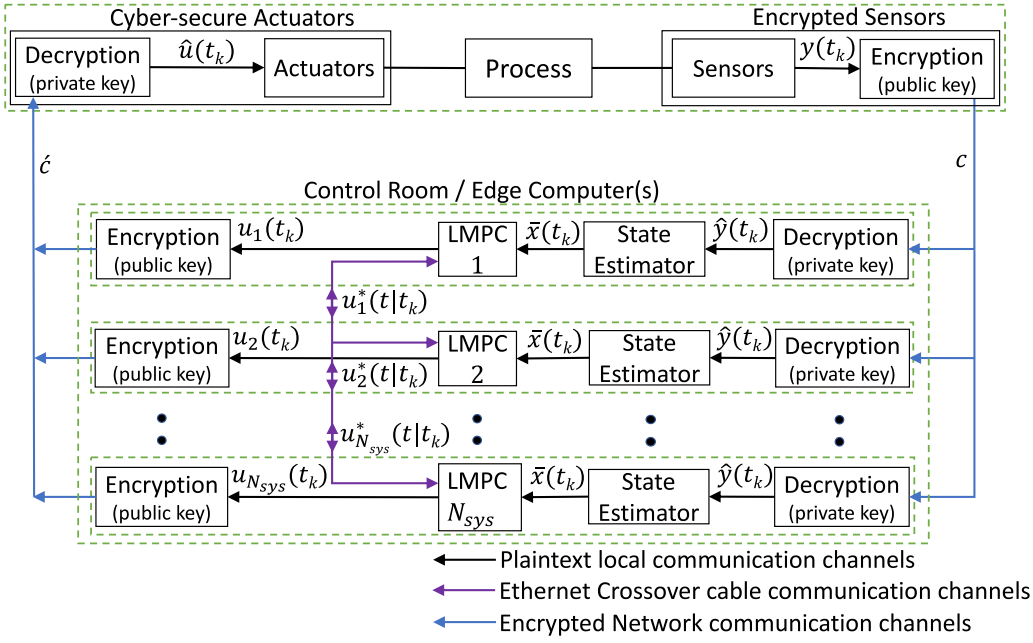

Fig. 2. Illustration of the encrypted iterative distributed control structure.

3.3. Extended luenberger observer-based state estimation

An extended Luenberger observer (ELO) is employed to estimate all the states of the nonlinear system, as detailed in Eq. (1). This estimation process relies on noisy partial state feedback obtained from sensors after decryption. Consequently, each subsystem's computing unit integrates an ELO, initializing the LMPC model of each subsystem with a complete state estimate (through the ELO) denoted as $\bar{x}$. In the design of Eq. (2), the observer necessitates a process model of the nonlinear system. Interestingly, the LMPC model can be extended and utilized within the observer, or vice versa. This dual utilization presents an effective approach for large-scale processes, reducing the number of required measured states through the ELO, and enhancing closed-loop performance and ensuring stability through the constraints of a Lyapunov-based MPC.

The typical sequence of actions executed by an ELO within the computing unit assigned to compute control inputs for a particular subsystem is as follows:

1. At time $t = t_k$, where k is the sampling instance, the ELO process model is initialized using all the estimated states of the system at the previous sampling instance $\bar{x}(t_{k-1})$, and all the control inputs computed at the previous sampling instance $u(t_{k-1})$.
2. The ELO process model predicts the state at the next integration time step $\bar{x}(t_{k-1} + h_c)$, where h_c represents the integration time step. A correction term $h_c \times K(\hat{y}(t_k) - h(\bar{x}(t_{k-1} + h_c)))$ is added to the estimated state $\bar{x}(t_{k-1} + h_c)$. Here, $\hat{y}(t_k)$ is the quantized measured state vector after decryption at time $t = t_k$.
3. The above step is reiterated Δ/h_c times, with Δ representing the sampling period, in order to compute the final estimated state at t_k, denoted as $\bar{x}(t_k)$. It is important to note that the control input $u(t_{k-1})$ remains constant within a single sampling period, as it is applied in a sample-and-hold manner and does not undergo any change during this interval.

During the initial sampling period, denoted as t_0, we make the assumption that the control inputs and the initial estimated states are set to their steady-state values. This assumption is necessary as no prior data is accessible for this specific sampling instance.

Remark 6. The procedure outlined in Remark 1 involves linearizing the nonlinear system described in Eq. (1) around its steady state. The observer gains are adjusted in such a way that the matrix $A - KL$ in Eq. (4) possesses eigenvalues with negative real components. However, since the observer is intended to be applied to a nonlinear system, further fine-tuning of the gains might be required. To achieve this, multiple simulations of the observer integrated within the nonlinear system, along with the state feedback controller, are conducted. These simulations encompass random initial conditions within the set Ω_ρ. During this process, the observer gains are refined to ensure that the error $e = x - \bar{x}$ tends to zero or a sufficiently small threshold, within a finite number of iterations for each randomly initialized simulation. Each iteration corresponds to a sampling period. Furthermore, with these newly fine-tuned observer gains, it is ensured that the matrix $A-KL$ continues to possess eigenvalues with negative real components. This adjustment is particularly necessary for nonlinear systems because the assumptions and properties of a linear system cannot be directly extrapolated to nonlinear systems.

3.4. Encrypted sequential distributed LMPC

In order to mitigate the computational time and complexity associated with a centralized control problem, especially in the context of large-scale systems featuring multiple states and control inputs, we propose the establishment of a sequential distributed LMPC system, where the optimization problem for the j^{th} LMPC is delineated as follows:

$$\mathcal{J} = \min_{u_{d_j} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{x}(t), \boldsymbol{\Phi}_m(\tilde{x}(t)), u_{d_n}(t))\, \mathrm{d}t,$$

$$\text{where } m = 1, \ldots, j-1 \text{ and } n = j, \ldots, N_{sys} \tag{14a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = F(\tilde{x}(t), \boldsymbol{\Phi}_m(\tilde{x}(t)), u_{d_n}(t)) \tag{14b}$$

$$\dot{\tilde{x}}(t) = F(\tilde{x}(t), \boldsymbol{\Phi}_m(\tilde{x}(t)), u_{d_n}(t)) + K(\hat{y}(t) - h(\tilde{x}(t))) \tag{14c}$$

$$u_{d_j}(t) \in U_j, \ \forall\ t \in [t_k, t_{k+N}) \tag{14d}$$

$$\tilde{x}(t_k) = \bar{x}(t_k) \tag{14e}$$

$$\dot{V}(\tilde{x}(t_k), \boldsymbol{\Phi}_m(\tilde{x}(t_k)), u_{d_n}(t_k)) \leq \dot{V}(\tilde{x}(t_k), \boldsymbol{\Phi}_m(\tilde{x}(t_k)), \boldsymbol{\Phi}_n(\tilde{x}(t_k)))$$

$$\text{if } \bar{x}(t_k) \in \Omega_\rho \backslash \Omega_{\rho_{\min}} \tag{14f}$$

$$V(\bar{x}(t)) \leq \rho_{\min},\ \forall\ t \in [t_k, t_{k+N}),$$
$$\text{if } \bar{x}(t_k) \in \Omega_{\rho_{\min}} \tag{14g}$$

At time $t = t_k$, where k represents the sampling instance, the ELO in the computing unit corresponding to the j^{th} LMPC decrypts the ciphertext c to receive the quantized state measurements $\hat{y}(t_k)$. The ELO uses these along with the computed control inputs at the previous sampling instance, i.e., $\boldsymbol{\Phi}_m(\bar{x}(t_{k-1}))$ and $u_{d_n}(t_{k-1})$, where $m = 1, \ldots, j-1$ and $n = j, \ldots, N_{sys}$, and the estimated states at the previous sampling instance $\bar{x}(t_{k-1})$ to predict the states at the current sampling instance, $\bar{x}(t_k)$, through Eq. (14c). The j^{th} LMPC then receives the complete state estimate $\bar{x}(t_k)$ from the ELO, but only computes the control input of its subsystem, u_{d_j}, which is to be applied by the corresponding actuators. It assumes the stabilizing control law for control inputs of subsystems 1 to $j-1$, and receives the optimal control input trajectories $u_{d_{n'}}$ from the remaining n' subsystems where $n' = j+1, \ldots, N_{sys}$. $\tilde{x}$ represents the predicted state trajectory of the process model of the j^{th} LMPC. The estimated states, $\bar{x}$, serve as the initial conditions for the LMPC process model to predict the state trajectory as per Eq. (14b), which is used to integrate the cost function of Eq. (14a) to calculate optimized control inputs, $u^*_{d_j}(t)$, $t \in [t_k, t_{k+N})$, for the entire prediction horizon. However, the LMPC transmits only the first input of this sequence, $u^*_{d_j}(t_k)$ to the actuator for application to the system within the interval $t \in [t_k, t_{k+1})$ and transmits the entire control input trajectory $u^*_{d_j}(t)$ along with the received control input trajectory, $u^*_{d_{n'}}$ where $n' = j+1, \ldots, N_{sys}$ to the $j-1^{\text{th}}$ LMPC. This process is repeated at each sampling period. N represents the number of sampling periods within the prediction horizon. Eq. (14d) represents the constraints imposed on the control inputs, and Eq. (14e) uses the quantized states to initialize the plant model described in Eq. (14b). The Lyapunov constraint in Eq. (14f) ensures that, if the state $\bar{x}(t_k)$ at time t_k lies within the set $\Omega_\rho \setminus \Omega_{\rho_{\min}}$, where $\rho_{\min}$ represents a level set of V in proximity to the origin, the time-derivative of the control Lyapunov function of the closed-loop subsystem j under the j^{th} LMPC, and stabilizing control law for the other control inputs, is less than or equal to the time-derivative of the control Lyapunov function when the subsystem is controlled by the stabilizing controller $\boldsymbol{\Phi}(\bar{x})$. When the closed-loop state $\bar{x}(t_k)$ enters $\Omega_{\rho_{\min}}$, the constraint of Eq. (14g) ensures that this state remains within $\Omega_{\rho_{\min}}$.

Remark 7. Within the proposed framework, a secure edge computer receives the encrypted partial state feedback. This computer then decrypts the received encrypted partial state feedback and employs the extended Luenberger observer within the same unit to compute all states, using the quantized partial state feedback values. Following this process, the LMPC utilizes the estimated states received from the observer, all within the same computing unit. Since these operations occur internally in the same unit, they are not encrypted. However, the control input computed by the LMPC, to be sent to and applied by the actuator, is encrypted before transmission. This configuration ensures that all wireless networked communications remain encrypted, thereby enhancing the cybersecurity of the control system.

Remark 8. Although state constraints have not been explicitly utilized in our LMPC formulations (only input constraints are considered), they can still be integrated if required for both the LMPC and extended Luenberger observer. In the case of the ELO, one feasible approach could involve utilizing the estimated states from the observer and subsequently applying a post-processing technique (like modifying the observer gain and re-running the observer) to ensure adherence to the defined constraints. Future research can be conducted to identify other methods to account for state constraints within the observer.

3.5. Encrypted iterative distributed LMPC

An alternative approach to a sequential DMPC is the iterative DMPC, in which the controllers responsible for computing control inputs for each subsystem of the overall process share their control inputs at the end of each iteration until a termination criterion is met. The optimization problem for the j^{th} LMPC within the iterative distributed LMPC structure, for the first iteration, $z = 1$, is described as follows:

$$\mathcal{J} = \min_{u_{d_j} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\bar{x}(t), \boldsymbol{\Phi}_m(\bar{x}(t)), u_{d_j}(t))\, \mathrm{d}t,$$
$$\text{where } m = 1, \ldots, N_{sys} \text{ and } m \neq j \tag{15a}$$
$$\text{s.t.}\quad \dot{\tilde{x}}(t) = F(\tilde{x}(t), \boldsymbol{\Phi}_m(\bar{x}(t)), u_{d_j}(t)) \tag{15b}$$
$$\dot{\bar{x}}(t) = F(\bar{x}(t), u_{d_m}(t), u_{d_j}(t)) + K(\hat{y}(t) - h(\bar{x}(t))) \tag{15c}$$
$$u_{d_j}(t) \in U_j,\ \forall\ t \in [t_k, t_{k+N}) \tag{15d}$$
$$\tilde{x}(t_k) = \bar{x}(t_k) \tag{15e}$$
$$\dot{V}(\bar{x}(t_k), \boldsymbol{\Phi}_m(\bar{x}(t_k)), u_{d_j}(t_k)) \leq \dot{V}(\bar{x}(t_k), \boldsymbol{\Phi}_m(\bar{x}(t_k)), \boldsymbol{\Phi}_j(\bar{x}(t_k))),$$
$$\text{if } \bar{x}(t_k) \in \Omega_\rho \setminus \Omega_{\rho_{\min}} \tag{15f}$$
$$V(\bar{x}(t)) \leq \rho_{\min},\ \forall\ t \in [t_k, t_{k+N}),$$
$$\text{if } \bar{x}(t_k) \in \Omega_{\rho_{\min}} \tag{15g}$$

At the iteration $z > 1$ following the exchange of the optimized input trajectories $u^*_{d_m}(t)$ with the rest of the LMPCs, the optimization problem of j^{th} LMPC is modified as follows:

$$\mathcal{J} = \min_{u_{d_j} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{x}(t), u_{d_m}(t), u_{d_j}(t))\, \mathrm{d}t,$$
$$\text{where } m = 1, \ldots, N_{sys} \text{ and } m \neq j \tag{16a}$$
$$\text{s.t.}\quad \dot{\tilde{x}}(t) = F(\tilde{x}(t), u_{d_m}(t), u_{d_j}(t)) \tag{16b}$$
$$\dot{\bar{x}}(t) = F(\bar{x}(t), u_{d_m}(t), u_{d_j}(t)) + K(\hat{y}(t) - h(\bar{x}(t))) \tag{16c}$$
$$u_{d_j}(t) \in U_j,\ \forall\ t \in [t_k, t_{k+N}) \tag{16d}$$
$$\tilde{x}(t_k) = \bar{x}(t_k) \tag{16e}$$
$$\dot{V}(\bar{x}(t_k), u_{d_m}(t_k), u_{d_j}(t_k)) \leq \dot{V}(\bar{x}(t_k), \boldsymbol{\Phi}_m(\bar{x}(t_k)), \boldsymbol{\Phi}_j(\hat{x}(t_k))),$$
$$\text{if } \bar{x}(t_k) \in \Omega_\rho \setminus \Omega_{\rho_{\min}} \tag{16f}$$
$$V(\bar{x}(t)) \leq \rho_{\min},\ \forall\ t \in [t_k, t_{k+N}),$$
$$\text{if } \bar{x}(t_k) \in \Omega_{\rho_{\min}} \tag{16g}$$

The j^{th} LMPC receives the complete state estimate $\bar{x}(t_k)$ from the ELO, but only computes the control input of its specific subsystem, denoted as u_{d_j}, which is to be applied by the corresponding actuators. Initially, for the first iteration, $z = 1$, it assumes the stabilizing control law for control inputs of m subsystems, where $m = 1, \ldots, N_{sys}$, and $m \neq j$. Subsequently, for iterations $z > 1$, the j^{th} LMPC transmits its computed control input at the previous iteration to all other LMPCs, and receives the control inputs computed by all other LMPCs at the previous iteration over the entire prediction horizon. The j^{th} LMPC then recalculates the control inputs for its respective subsystem, assuming the received control input trajectories for the other subsystems. At the end of the current iteration, it transmits the updated control input trajectory of its subsystem to the other subsystems. This is repeated until a termination criterion is satisfied. The formulation of the optimization problems presented in Eqs. (15), and (16) is very similar to Eq. (14), which was elaborated in detail in Section 3.4.

Remark 9. In the context of the stability analysis for the introduced encrypted distributed LMPC architectures in this study, the bounds related to encryption-induced errors have been established in Section 3.2. Additionally, each LMPC in the distributed structure incorporates a constraint stipulating that the value of the time-derivative of the control

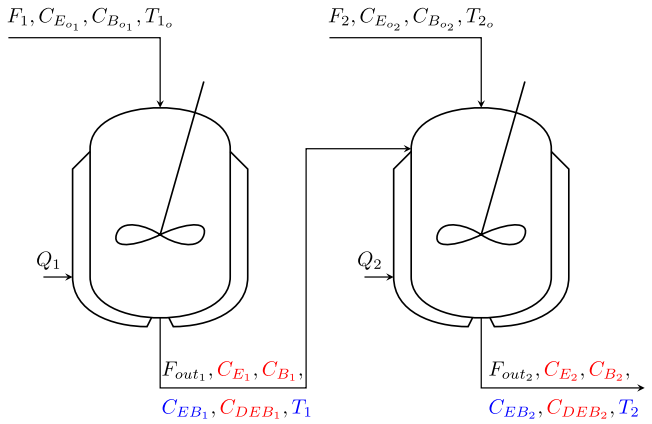

Fig. 3. Process schematic featuring two CSTRs connected in series. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Lyapunov function under the LMPC should be more negative than that of the observer-based stabilizing control law. A comprehensive stability analysis has previously been conducted for a nonlinear centralized encrypted system in Suryavanshi et al. (2023). Building on this foundation, a similar stability analysis can be carried out for the encrypted distributed LMPC, incorporating an observer. It is important to note that, given our assumption of an observer-based stabilizing control law, the stability analysis is simplified and does not require elaborate demonstration; thus, it has been omitted.

4. Application to a nonlinear chemical process network operating at an unstable steady state

This section demonstrates the proposed encrypted distributed control architectures, both sequential and iterative distributed LMPCs with state estimation, on a nonlinear chemical process network with noisy partial state feedback, operating at an unstable steady state. A nonlinear dynamical model based on first-principles modeling fundamentals is developed for the state estimator and the LMPCs. Guidelines are established to implement the encrypted distributed LMPC systems in any nonlinear process with partial state feedback. We then conduct closed-loop simulations, employing the distributed LMPCs with state estimators, and analyze the results.

4.1. Process description and model development

The process considered is the synthesis of ethylbenzene (EB) by reacting ethylene (E) and benzene (B) within two non-isothermal, well-mixed continuous stirred tank reactors (CSTRs) as depicted in Fig. 3. The primary reaction, termed as "primary", is characterized as a second-order, exothermic, and irreversible reaction, in conjunction with two supplementary side reactions. The chemical reactions taking place are articulated as follows:

$$C_2H_4 + C_6H_6 \rightarrow C_8H_{10} \quad \text{(primary)} \tag{17a}$$

$$C_2H_4 + C_8H_{10} \rightarrow C_{10}H_{14} \tag{17b}$$

$$C_6H_6 + C_{10}H_{14} \rightarrow 2C_8H_{10} \tag{17c}$$

Details of the steady-state values and model parameter values can be obtained from Kadakia et al. (2023). The dynamic model of the first CSTR is described by the following mass and energy balance equations:

$$\dot{C}_{E_1} = \frac{F_1 C_{Eo_1} - F_{out_1} C_{E_1}}{V_1} - r_{1,1} - r_{1,2} \tag{18a}$$

$$\dot{C}_{B_1} = \frac{F_1 C_{Bo_1} - F_{out_1} C_{B_1}}{V_1} - r_{1,1} - r_{1,3} \tag{18b}$$

$$\dot{C}_{EB_1} = \frac{-F_{out_1} C_{EB_1}}{V_1} + r_{1,1} - r_{1,2} + 2r_{1,3} \tag{18c}$$

$$\dot{C}_{DEB_1} = \frac{-F_{out_1} C_{DEB_1}}{V_1} + r_{1,2} - r_{1,3} \tag{18d}$$

$$\dot{T}_1 = \frac{T_{1_o} F_1 - T_1 F_{out_1}}{V_1} + \sum_{j=1}^{3} \frac{-\Delta H_j}{\rho_1 C_p} r_{1,j} + \frac{Q_1}{\rho_1 C_p V_1} \tag{18e}$$

The dynamic model of the second CSTR is represented by the following equations:

$$\dot{C}_{E_2} = \frac{F_2 C_{Eo_2} + F_{out_1} C_{E_1}}{V_2} - \frac{F_{out_2} C_{E_2}}{V_2} - r_{2,1} - r_{2,2} \tag{19a}$$

$$\dot{C}_{B_2} = \frac{F_2 C_{Bo_2} + F_{out_1} C_{B_1}}{V_2} - \frac{F_{out_2} C_{B_2}}{V_2} - r_{2,1} - r_{2,3} \tag{19b}$$

$$\dot{C}_{EB_2} = \frac{F_{out_1} C_{EB_1} - F_{out_2} C_{EB_2}}{V_2} + r_{2,1} - r_{2,2} + 2r_{2,3} \tag{19c}$$

$$\dot{C}_{DEB_2} = \frac{F_{out_1} C_{DEB_1} - F_{out_2} C_{DEB_2}}{V_2} + r_{2,2} - r_{2,3} \tag{19d}$$

$$\dot{T}_2 = \frac{T_{2_o} F_2 + T_1 F_{out_1} - T_2 F_{out_2}}{V_2} + \sum_{j=1}^{3} \frac{-\Delta H_j}{\rho_2 C_p} r_{2,j} + \frac{Q_2}{\rho_2 C_p V_2} \tag{19e}$$

where the reaction rates are calculated by the following expressions:

$$r_{i,1} = k_1 e^{\frac{-E_1}{RT_i}} C_{E_i} C_{B_i} \tag{20a}$$

$$r_{i,2} = k_2 e^{\frac{-E_2}{RT_i}} C_{E_i} C_{EB_i} \tag{20b}$$

$$r_{i,3} = k_3 e^{\frac{-E_3}{RT_i}} C_{DEB_i} C_{B_i} \tag{20c}$$

where $i = \{1, 2\}$ is the reactor index. The state variables are the concentration of ethylene, benzene, ethylbenzene, di-ethylbenzene, and the reactor temperature for each CSTR in deviation terms, that is: $x^\top = [C_{E_1} - C_{E_{1s}},\ C_{B_1} - C_{B_{1s}},\ C_{EB_1} - C_{EB_{1s}},\ C_{DEB_1} - C_{DEB_{1s}},\ T_1 - T_{1s},\ C_{E_2} - C_{E_{2s}},\ C_{B_2} - C_{B_{2s}},\ C_{EB_2} - C_{EB_{2s}},\ C_{DEB_2} - C_{DEB_{2s}},\ T_2 - T_{2s}]$. The subscript "$s$" denotes the steady-state value. The desired product, ethyl benzene, and the CSTR temperature are the measured states corresponding to $y^\top = [C_{EB_1} - C_{EB_{1s}},\ T_1 - T_{1s},\ C_{EB_2} - C_{EB_{2s}},\ T_2 - T_{2s}]$. The measured states are visually represented in blue in Fig. 3. In contrast, the remaining states that are not measured are depicted in red within the same figure. Bounded white Gaussian noise is added to the measured states of both CSTRs. The mean of the noise is zero for both states, and the standard deviation is 0.003 kmol/m^3 for the measured concentration of ethylbenzene and 0.15 K for the measured CSTR temperature, in each CSTR. The noise is bounded by the closed sets [-0.01, 0.01] kmol/m^3 and [-0.5, 0.5] K for the measured ethylbenzene concentration and temperature states, respectively.

The rate of heat removal for the two reactors $[Q_1 - Q_{1s},\ Q_2 - Q_{2s}]$ and inlet feed concentrations for each reactor, $[C_{E_{o1}} - C_{E_{o1s}}, C_{B_{o1}} - C_{B_{o1s}}, C_{E_{o2}} - C_{E_{o2s}}, C_{B_{o2}} - C_{B_{o2s}}]$, are the manipulated inputs of the nonlinear system. These inputs are bounded by the closed sets, $[-10^4, 2 \times 10^3]$ kW, $[-1.5\times10^4, 5\times10^3]$ kW, $[-2.5, 2.5]$ kmol/m^3, $[-2.5, 2.5]$ kmol/m^3, $[-3, 3]$ kmol/m^3, and $[-3, 3]$ kmol/m^3, respectively. To assess the stability of the selected equilibrium state, an open-loop simulation was conducted. During this simulation, the control inputs remained fixed at their equilibrium values, and the initial conditions of the system were set near the operating equilibrium point, within the region $\Omega_{\rho_{\min}}$. After a finite period of time, the system's states departed from the stability region Ω_ρ, and eventually converged to an entirely different equilibrium state. This transition signifies the instability of the initial equilibrium. The rationale for choosing this particular state was its capability to achieve a significantly high steady-state concentration of the desired product, ethyl benzene of 4.22 kmol/m^3, at the outlet of reactor 2, under reasonable operating conditions.

The overall control of the system was partitioned into two LMPCs. Both LMPCs utilized a first-principles-based model, and received the estimated states $\bar{x}$ from the ELO. Further, LMPC 1 optimizes the control inputs $u_1 = [C_{E_{o1}} - C_{E_{o1s}}, C_{B_{o1}} - C_{B_{o1s}}, Q_1 - Q_{1s}]^\top$, while LMPC 2 optimizes the control inputs $u_2 = [C_{E_{o2}} - C_{E_{o2s}}, C_{B_{o2}} - C_{B_{o2s}}, Q_2 - Q_{2s}]^\top$. Thus, the partitioning of the overall systems is done such that LMPC 1 manipulates all the control inputs of CSTR 1, while LMPC 2 manipulates all the control inputs of CSTR 2. In case of the sequential distributed LMPC system, LMPC 2 assumes the control inputs for LMPC 1 as per the stabilizing control law, and accordingly computes the optimized control inputs for its subsystem, CSTR 2. It then transmits the control input trajectory over the complete prediction horizon to LMPC 1, which uses this information to compute the control inputs of its respective subsystem, CSTR 1. On the other hand, in the iterative distributed LMPC system, in the first iteration, both LMPCs compute the control inputs of their respective subsystems, assuming the stabilizing control law for the inputs of the other subsystem. At the second iteration, both LMPCs, share the control input trajectory over the prediction horizon, computed for their respective subsystems with each other. Based on the information received about the control inputs of the other subsystem, both LMPCs recompute the optimized control inputs of their respective subsystem. This exchange of information goes on until a termination criterion is satisfied. In the example demonstrated in this section, we have used a termination criterion of 2 iterations for the iterative distributed LMPC. The control objective is to operate both CSTRs at their unstable equilibrium point through the encrypted distributed control schemes, sequential and iterative, employing quantized partial state feedback with sensor noise for computation of the required control inputs.

4.2. Encrypting the distributed control architectures

Prior to integrating encryption and decryption into a process, the process of parameter selection, specifically involving the variables d, l_1, and l_2, takes place. By considering the extreme feasible states and inputs, the integer bit count $l_1 - d$ is determined. The upper limit within the set $\mathbb{Q}_{l_1,d}$ is calculated using the formula $2^{l_1-d-1} - 2^{-d}$, while the lower limit is established as -2^{l_1-d-1}. The selection of the quantization parameter d, which represents the fractional bit count, depends on the desired level of precision and the range of state and input values. Additionally, l_2 is chosen to be greater than l_1. In the context of the example presented in this section, a value of 16 is determined for $l_1 - d$, which in turn determines the values of l_1 and d. Within the set $\mathbb{Q}_{l_1,d}$, rational numbers are spaced apart by a resolution of 2^{-d}. For simulation purposes, we have opted for a value of $d = 8$. With $d = 8$, l_1 is set at 24, and l_2 is selected as 30. The implementation of the Paillier Encryption procedure is carried out using the Python "phe" module, specifically PythonPaillier (Data61, 2013). For solving the multi-constrained, non-convex optimization problem within the LMPCs operating within the distributed control framework, we utilize the Python module from the IPOPT software (Wächter and Biegler, 2006).

While deciding the sampling time (Δ) for an encrypted distributed system with state estimation, it is crucial to ensure that it exceeds the total time required for encryption–decryption of the states and control inputs, time required by the state estimator to estimate the states, and the time needed to compute all the control inputs at each sampling instance for the considered quantization parameter d. Encryption–decryption and state estimation is performed in parallel between different edge computing units. Hence, we select the maximum time from all the different subsystems across all sampling instances. As control input information is exchanged, control input computation time is the total time needed to compute all the control inputs, and not just inputs for a particular subsystem. Hence, we select the maximum time taken to compute all the control inputs at any sampling instance. Mathematically,

$$\begin{aligned}\Delta > &\max\left(\text{encryption–decryption time}\right)_j \\ &+ \max\left(\text{State-estimation time}\right)_j \\ &+ \max\left(\text{Control input computation time}\right)\end{aligned} \tag{21}$$

where $j = \{1, \ldots, N_{sys}\}$ represents the control subsystem. Details on how the control input computation time is calculated for the sequential and iterative DMPCs is provided in the next section. Considering the above criteria, the sampling time Δ is chosen as 30 s in the discussed example. In real-world scenarios, the state estimation computations in a process simulation (not the actual process) could take place on the specific type of computer intended for the actual usage of these calculations. The computational time across multiple sampling instances of the process simulation can be recorded, and the maximum duration among these instances can be chosen as the maximum state-estimation time. This same concept can be extended to obtain the maximum encryption-decryption time. Additionally, to account for precautionary measures, this value could be multiplied by a factor, such as 1.25.

To calculate the cost function of the LMPCs over the prediction horizon, an integration time step, $h_c = 10^{-2} \times \Delta$, is chosen. The positive definite matrix P in the control Lyapunov function $V = x^\top Px$ is selected as diag [200 200 400 1000 2.5 250 250 200 1000 0.5], from extensive simulations. The LMPCs employ a prediction horizon of $N = 2$ sampling periods. The stability criterion is defined as $\rho = 1800$, while $\rho_{\min} = 2$ is the smaller level set of the Lyapunov function where the state is desired to be confined. The weight matrix in the cost function of LMPCs is chosen as Q = diag [1000 1000 1500 5 8 1000 1000 3000 5 110], R = diag [2.1 1.95 1.5×10^{-5} 10 10 0.5×10^{-4}]. The cost function is defined as $L(x, u) = x^\top Qx + u^\top Ru$.

4.3. Simulation results of the encrypted distributed control architectures

The proposed encrypted distributed control architecture is applied to a nonlinear chemical process, and the control inputs are computed using partial state feedback with sensor noise. Figs. 4 to 6 depict the results for the encrypted sequential distributed LMPC system with state estimation from the first set of initial conditions, $x_0 = [-0.35\ \text{kmol/m}^3, -0.3\ \text{kmol/m}^3, 0.2\ \text{kmol/m}^3, 0\ \text{kmol/m}^3, -20\,\text{K}, 0.2\ \text{kmol/m}^3, 0.15\ \text{kmol/m}^3, -0.25\ \text{kmol/m}^3, 0\ \text{kmol/m}^3, -15\,\text{K}]^\top$. Figs. 7 to 9 depict the results for the encrypted iterative distributed LMPC system with state estimation from the second set of initial conditions, $x_0 = [0.5\ \text{kmol/m}^3, 0.35\ \text{kmol/m}^3, -0.2\ \text{kmol/m}^3, 0\ \text{kmol/m}^3, 20\,\text{K}, 0.45\ \text{kmol/m}^3, 0.5\ \text{kmol/m}^3, -0.8\ \text{kmol/m}^3, 0\ \text{kmol/m}^3, -30\,\text{K}]^\top$.

In Figs. 4, 5, 7 and 8, the blue solid line represents the true state value, while the red dashed line represents the state value estimated by the ELO. For both initial conditions, the state estimator (ELO) provides the distributed LMPCs with fairly accurate state estimates, using partial state feedback with sensor noise. Minor deviations between the estimated and predicted states are noticeable in Fig. 5. These deviations can be attributed to the observer receiving partial state feedback with sensor noise. Additionally, errors stemming from quantization can also play a role in this discrepancy. However, it is essential to note that both sources of error are bounded, as previously indicated, resulting in the observed deviations being minor in nature. The distributed LMPCs successfully stabilize the system within the desired closed-loop stability region $\Omega_{\rho_{\min}}$ in approximately 1.5 h for the first set of initial conditions and 1 h for the second set.

In the case of the first set of initial conditions, the normalized sum of the control cost function is 1 for the encrypted sequential distributed LMPC and 0.9907 for the encrypted iterative distributed LMPC. For the second set of initial conditions, it is 1 for the encrypted sequential LMPC and 0.9884 for the encrypted iterative LMPC. The iterative LMPC outperforms the sequential approach because, in the iterative framework, both LMPCs share and recalculate their control

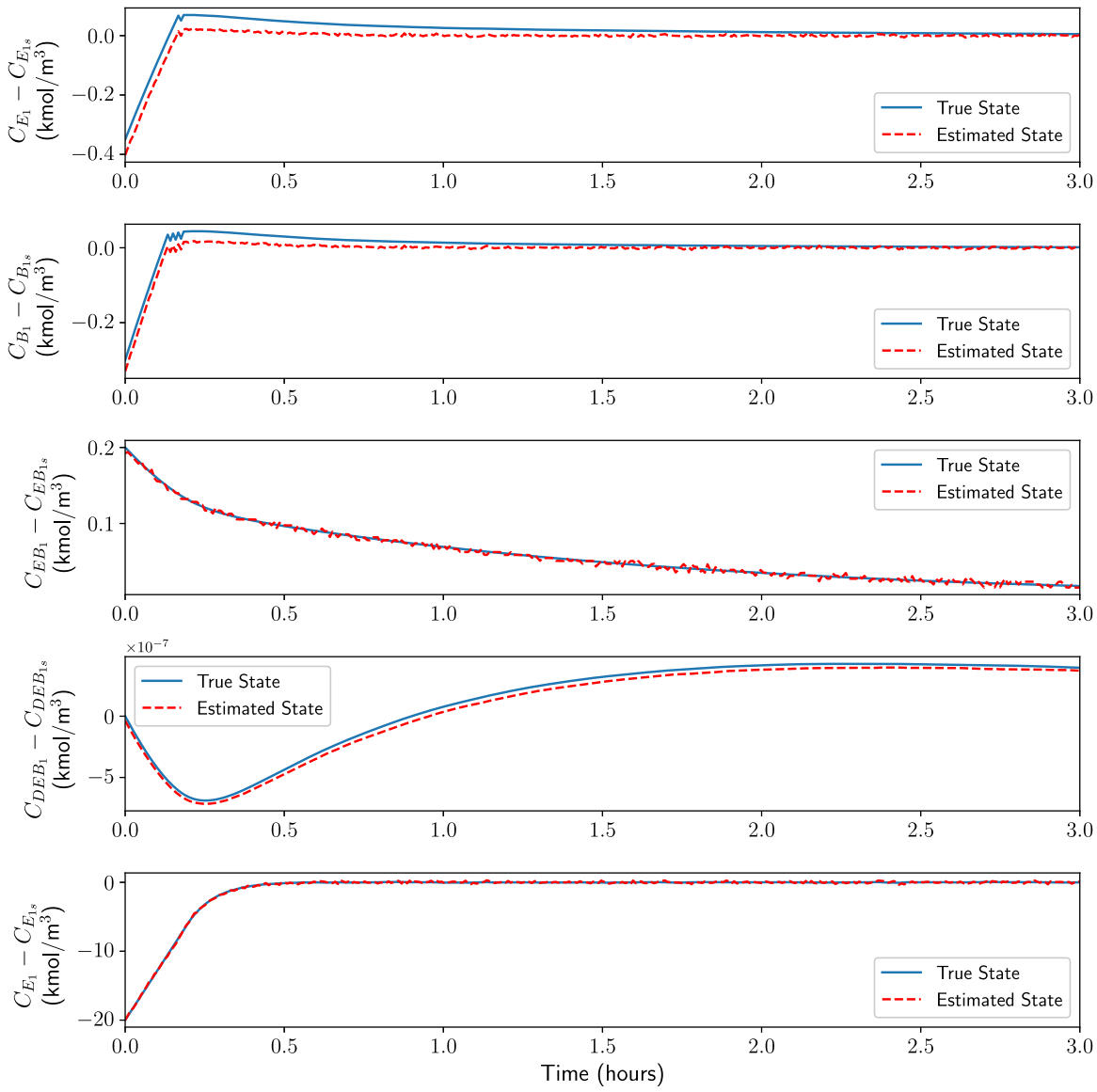

Fig. 4. True state profiles (blue solid line) and estimated state profiles (red dashed line) of CSTR 1 under the encrypted sequential distributed LMPC framework for the first set of initial conditions.

inputs, while, in the sequential framework, LMPC 2 computes control inputs based on an assumption of the stabilizing control law for LMPC 1. The following section provides a detailed comparative analysis of the sequential and iterative DMPCs. Visual results are provided exclusively for the encrypted sequential distributed LMPC demonstrating the performance under the first set of initial conditions and for the encrypted iterative distributed LMPC under the second set of initial conditions. Notably, when the alternative DMPC system was applied to both initial conditions, the differences in the closed-loop state trajectories were not significant, as evidenced by the close values of the normalized sum of the control cost functions in both scenarios.

Remark 10. The encrypted distributed LMPC systems explored in this study involved encrypting and decrypting data as outlined in Section 3, which can lead to errors due to quantization. Suryavanshi et al. (2023) demonstrated quantization effects in the context of a first-principles-based LMPC and process model. Additionally, Kadakia et al. (2023) highlighted the potential for quantization-induced errors to exceed model mismatch errors when different models are employed in the LMPC and in the controlled process. To minimize the quantization error, both works recommended using a higher quantization parameter d. With $d = 8$, both works reported almost identical closed-loop results with encryption compared to without encryption. Thus, we have used the quantization parameter $d = 8$ for all simulations in this work.

Remark 11. Although the LMPC and state estimation models used in this work are first-principles-based, data-based models employing artificial neural networks can also be used in the predictor and LMPC. Alhajeri et al. (2021) used machine-learning-based models for the ELO model and LMPC while simulating a first-principles-based process with partial state feedback, showcasing the effectiveness of the state estimator in the presence of plant/model mismatch.

5. Comparative analysis of encrypted centralized, decentralized, and distributed LMPC architectures

In this section, we provide a concise overview of the encrypted centralized and decentralized control architectures, both of which incorporate state estimation. Following this, we offer an in-depth comparative analysis that covers the encrypted centralized, decentralized, and distributed LMPCs.

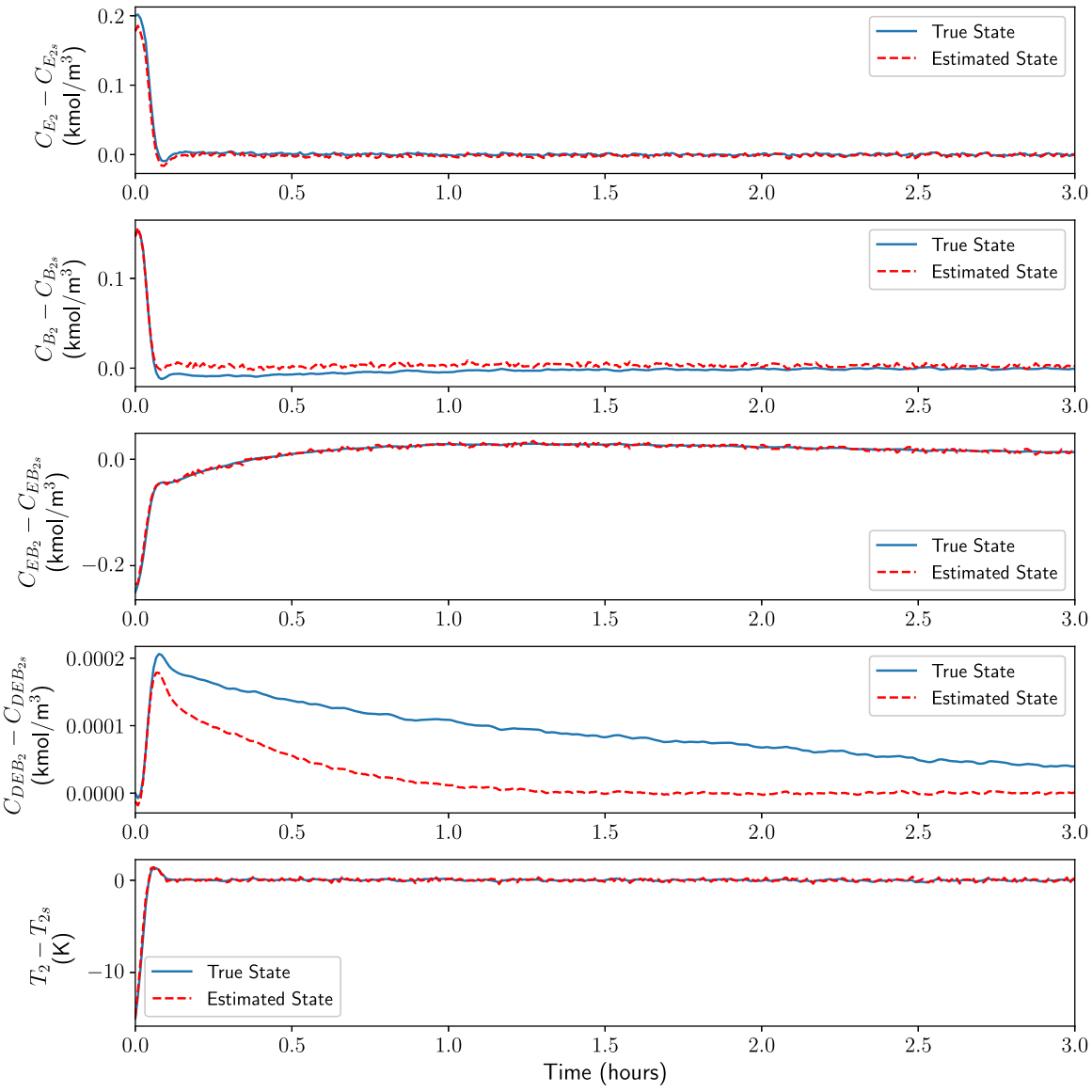

Fig. 5. True state profiles (blue solid line) and estimated state profiles (red dashed line) of CSTR 2 under the encrypted sequential distributed LMPC framework for the first set of initial conditions.

5.1. Encrypted centralized MPC with state estimation

In Fig. 10, the diagram illustrates the flow of information within an encrypted centralized LMPC system that incorporates state estimation. At time $t = t_k$, where k signifies the sampling instance, the sensors encrypt the measurements denoted as $y(t_k)$ and transmit the resulting ciphertext c to the computing unit responsible for computing all the control inputs. Upon arrival, the data is decrypted, and the quantized states $\hat{y}(t_k)$ are utilized by the state estimator to estimate all states of the system $\bar{x}(t_k)$. These estimated states initialize the LMPC model, enabling it to compute the control inputs, $u(t_k)$. Subsequently, these control inputs are encrypted into the ciphertext $\acute{c}$ and transmitted to the actuators, where it is decrypted to the quantized control inputs $\hat{u}(t_k)$ and applied to the process. Thus, in this approach, only a single computing unit that receives and transmits encrypted signals is utilized for all computations. Additional details and formulation of the LMPC equations of the centralized LMPC can be obtained in Kadakia et al. (2023).

5.2. Encrypted decentralized MPC with state estimation

In Fig. 11, the diagram illustrates the flow of information within an encrypted decentralized LMPC system with state estimation. Here, the overall system is divided into multiple subsystems, with each subsystem independently computing its control inputs in separate computing units. There is no information exchange of the control inputs between subsystems. At time $t = t_k$, where k is the sampling instance, the sensors encrypt the measurements represented as $y(t_k)$ and transmit the resulting ciphertext c to all the computing units responsible for calculating the control inputs of different subsystems. The ciphertext c is decrypted in each computing unit, and the quantized states $\hat{y}(t_k)$ are used by the state estimator in each subsystem to estimate all states of the entire system $\bar{x}(t_k)$. All the state estimators shown in Fig. 11 are identical, as each LMPC in the decentralized control framework receives full state feedback to compute the control inputs of its respective subsystem. The LMPC model in each subsystem is then initialized by these estimated states, and is used to compute the control inputs of its respective subsystem only, $u_j(t_k)$. Here j represents the j^{th} subsystem. Subsequently, all control inputs are encrypted and transmitted

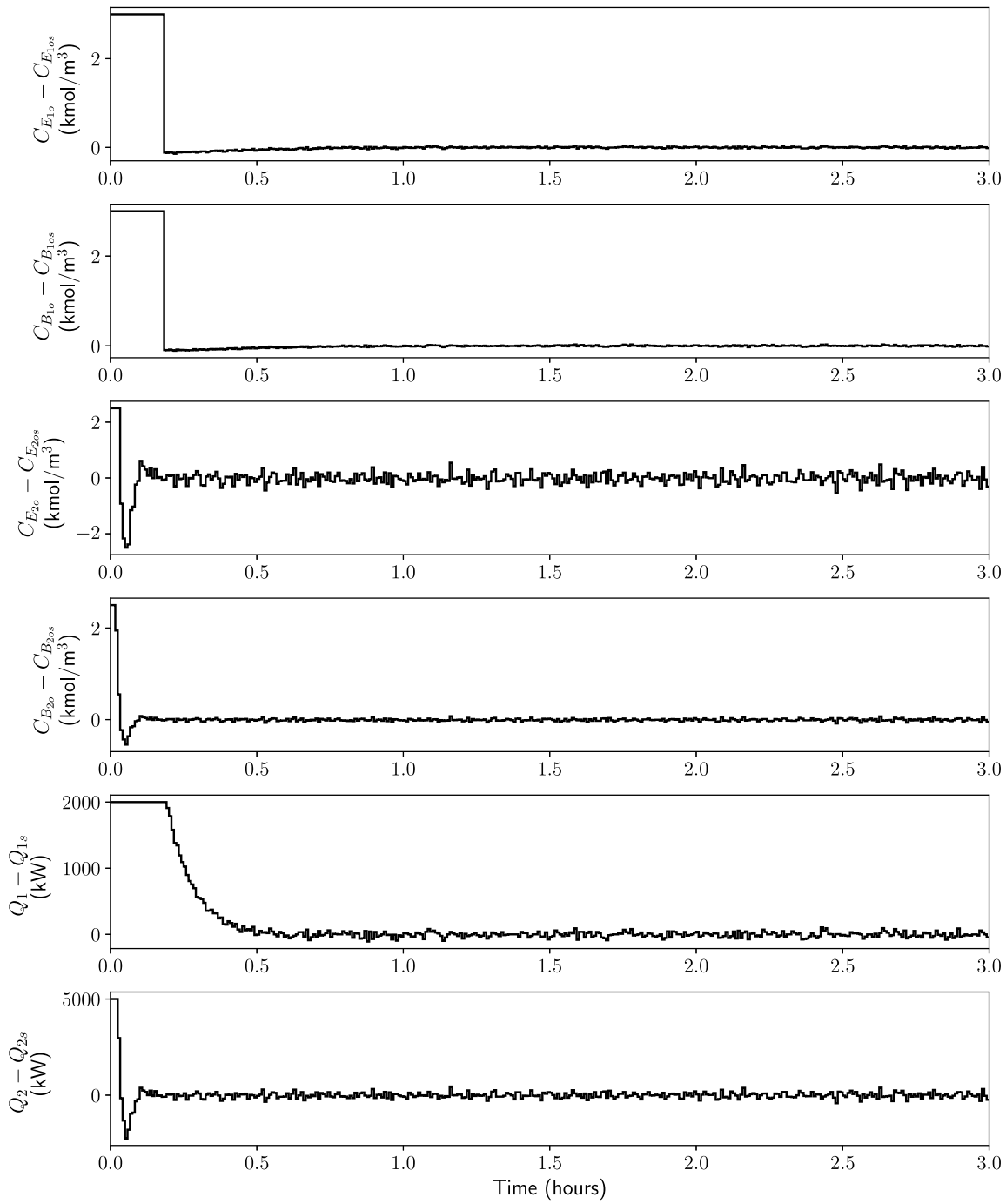

Fig. 6. Control input profiles under the encrypted sequential distributed LMPC framework for the first set of initial conditions.

to their respective actuators, where the ciphertext $\acute{c}$ is decrypted to the quantized control inputs $\hat{u}(t_k)$ and applied to the process. Thus, multiple computing units (equal to the number of subsystems) that receive and transmit encrypted signals are utilized for all computations, which are carried out in an independent and isolated manner. Additional details and formulation of the LMPC equations of the decentralized LMPC can be obtained in Chen et al. (2020).

5.3. Comparison of the encrypted centralized, decentralized, and distributed LMPCs with state estimation

In our analysis, we applied the same system as the example described in Section 4, using the second set of initial conditions mentioned in the preceding section. Our objective was to compare the computation time and performance of various control architectures in computing the control inputs. Table 1 provides a summary of the total computation time required for computing control inputs and the normalized sum of the control cost functions for the encrypted centralized, decentralized, and distributed LMPCs.

To determine the computation time of the centralized framework, we calculated the time spent by the LMPC in computing the control inputs for the system at each sampling instance. In the case of the decentralized framework, we recorded the longer of the two LMPC computation times at each sampling instance. For the sequential distributed LMPC, we summed the time spent by both LMPCs at each sampling instance to obtain the total control input computation time for that specific sampling instance. In case of the iterative distributed LMPC, we recorded the higher value of the time spent by the two LMPCs at each

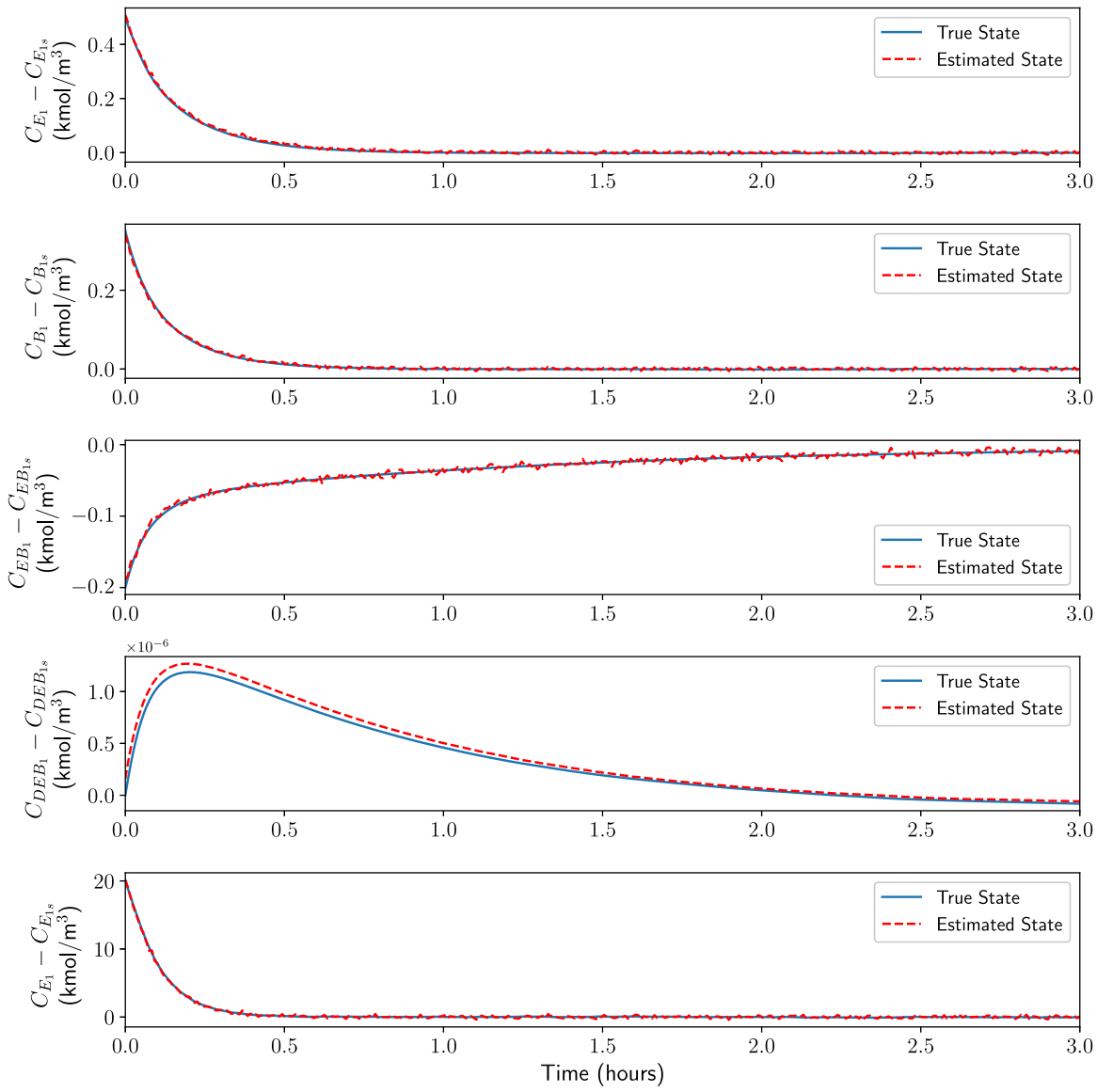

Fig. 7. True state profiles (blue solid line) and estimated state profiles (red dashed line) of CSTR 1 under the encrypted iterative distributed LMPC framework for the second set of initial conditions.

Table 1
Computational time and performance of the encrypted centralized, decentralized, sequential distributed, and iterative distributed LMPCs.

Control architecture	Average control input computation time	Normalized sum of the control cost function
Centralized MPC	15.28 s	1
Decentralized MPC	2.87 s	0.9751
Sequential DMPC	4.03 s	0.9817
Iterative DMPC	5.22 s	0.9703

iteration, and summed these values for the two iterations to calculate the total time spent for computing control inputs, at a particular sampling instance. It was ensured that the control input computation time at any interval was not only smaller than the sampling period of 30 s but also satisfied Eq. (21). Fig. 12 displays the computation times for all 4 cases at each sampling instance of process operation. Based on the results from Tables 1 and 12, we can conclude that the decentralized LMPC required the shortest computational time, while the iterative distributed LMPC exhibited the best performance. In contrast, the centralized LMPC not only had the longest computational time but also demonstrated inferior performance compared to the distributed and decentralized LMPC systems.

The reason behind the slightly improved performance observed with the decentralized LMPC can be attributed to the sequential flow sheet of the process network, featuring two CSTRs in series. This characteristic renders decentralized LMPC a more suitable and well-conditioned choice compared to centralized LMPC when addressing the optimization problem. Furthermore, the iterative distributed LMPC, which shares control input information with other subsystems during each iteration, demonstrates superior performance compared to both the decentralized LMPC and the sequential distributed LMPC. It is essential to note that this performance enhancement may not be universally applicable to all nonlinear systems, but the enhanced computational efficiency of decentralized and distributed frameworks over centralized ones can indeed be extended to other large-scale systems.

In general, the advantages and disadvantages of all 4 control schemes can be summarized in the following manner:

1. **Centralized MPC:** It offers the advantage of requiring only a single computing unit for all computations, simplifying information flow and reducing costs, which makes it suitable for

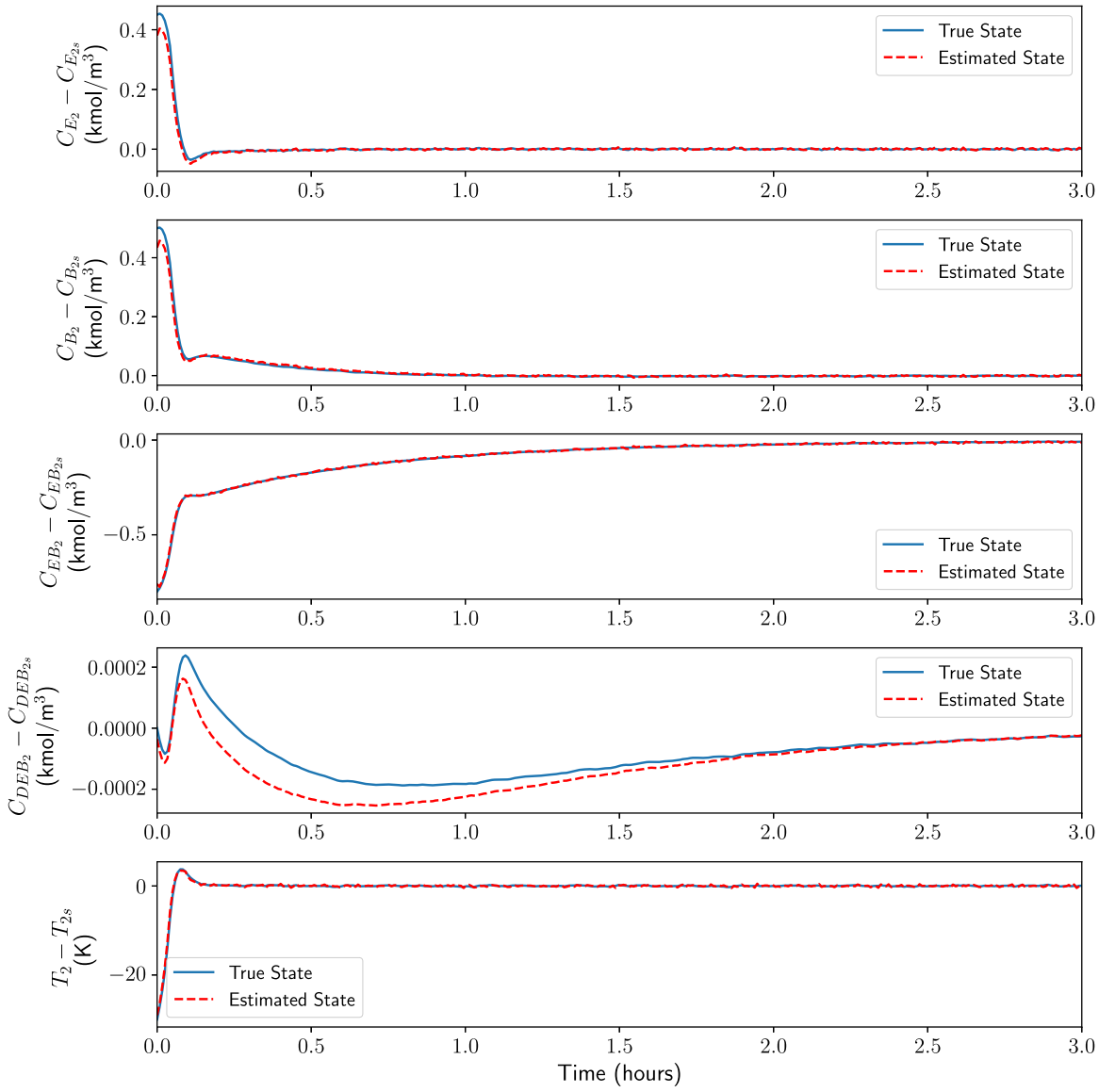

Fig. 8. True state profiles (blue solid line) and estimated state profiles (red dashed line) of CSTR 2 under the encrypted iterative distributed LMPC framework for the second set of initial conditions.

small systems. All signals transmitted to and received from the remote edge computing unit can be encrypted. However, it comes with a significantly higher computation time compared to the decentralized and distributed MPCs, making it less viable for large processes with numerous states and control inputs. It is, nevertheless, a suitable choice for small-scale systems, where only a single computing unit is needed.

2. **Decentralized MPC:** This approach stands out with the shortest computation time among the four systems, and can even outperform the closed-loop performance of the centralized MPC in specific cases. Furthermore, a decentralized MPC framework does not require communication between different computing units, making it particularly well-suited for large systems partitioned into many subsystems, where the coupling between subsystems is not very significant. Also, all signals transmitted to and received from the different computing units remain encrypted. However, its performance may deteriorate in cases where the overall system is partitioned into highly coupled subsystems.
3. **Sequential DMPC:** While a decentralized MPC can perform better than the centralized MPC in some cases, it calculates control inputs independently, without any information exchange among subsystems. The integration of information exchange between subsystems can be achieved through the use of a sequential DMPC. In the example discussed, the overall system was only partitioned into two subsystems, and the control inputs were computed in series for the two subsystems. However, this approach may be slower for very large systems partitioned into numerous subsystems, not making it a viable option in that case. The major advantage of a sequential DMPC over an iterative DMPC lies in its reduced communication among subsystems, as information flows only in one direction (from higher to lower subsystems). It is suitable for cases where communication between controllers is necessary, implementing iterative DMPC is not feasible, and the number of subsystems is not extensive. Further, it would be more suitable when Ethernet crossover communication cannot be established between different computing units (if they are placed in different locations) because the communication load of a sequential DMPC is much less compared to an iterative DMPC, and, hence, encrypted signals could be used for internal communication between subsystems,

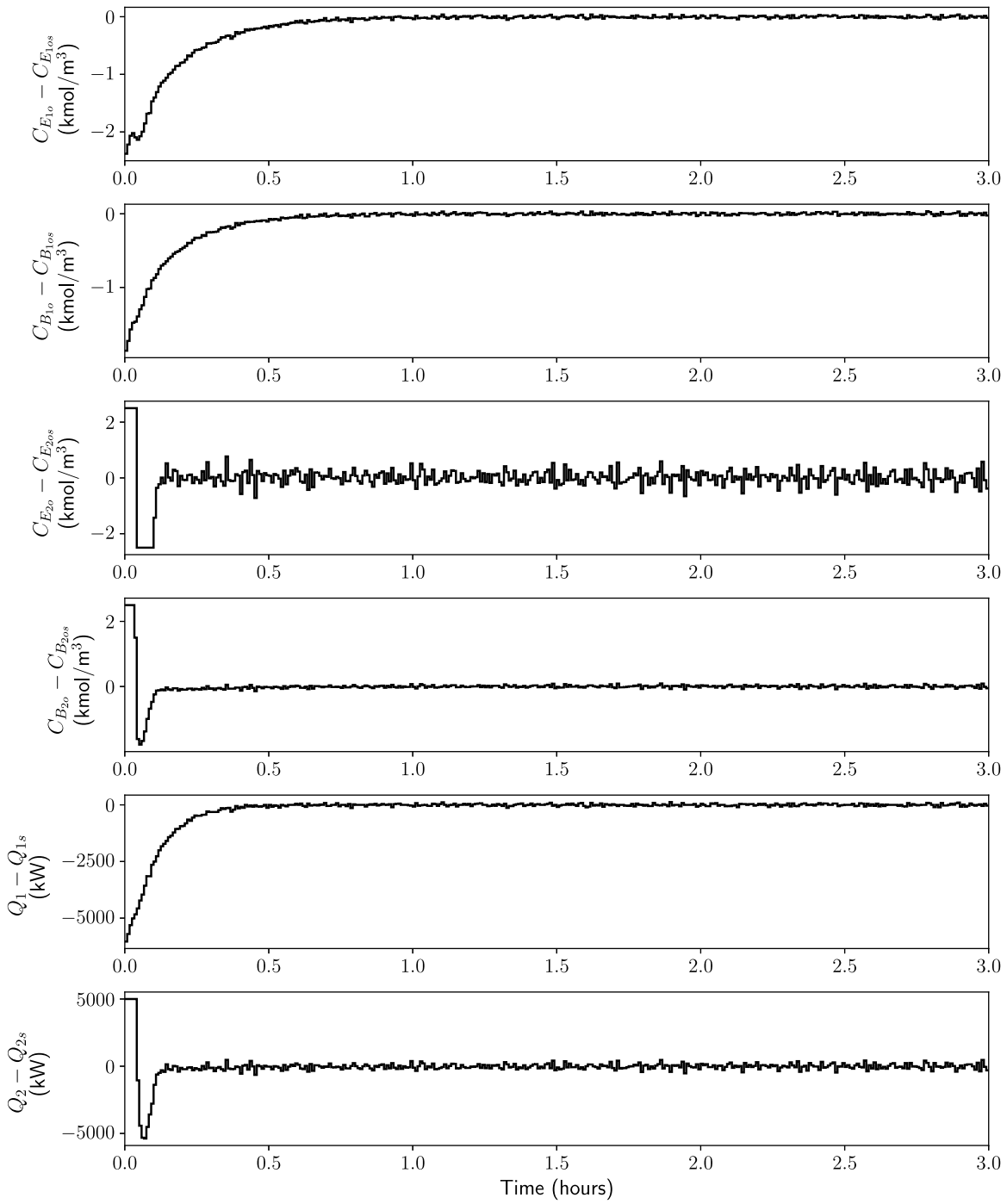

Fig. 9. Control input profiles under the encrypted iterative distributed LMPC framework for the second set of initial conditions.

as long as the overall system is not partitioned into a large number of subsystems.

4. **Iterative DMPC:** This approach delivers the best overall performance, although it entails longer computation times compared to decentralized and sequential DMPCs. In the example presented, we considered only two MPCs in the partition, but for systems with more partitions, it can outperform the sequential DMPC in terms of computation time. However, implementing this system requires multiple computing units compared to a single unit in a centralized MPC. Moreover, these units must be located in the same room to establish secure Ethernet crossover communication between them. On the other hand, a decentralized MPC allows for computing units to be located in different locations. Also, the communication load between subsystems in an iterative DMPC is higher than a sequential DMPC, as control input trajectories are shared with all other subsystems multiple times within a single sampling instance. Therefore, it is most suitable for very large systems partitioned into numerous subsystems, especially when these subsystems exhibit a substantial coupling effect with one another, and in situations where secure internal communication channels between different subsystems can be easily established.

To summarize, this section offered a general overview of the advantages, disadvantages, and the suitability of various control architectures

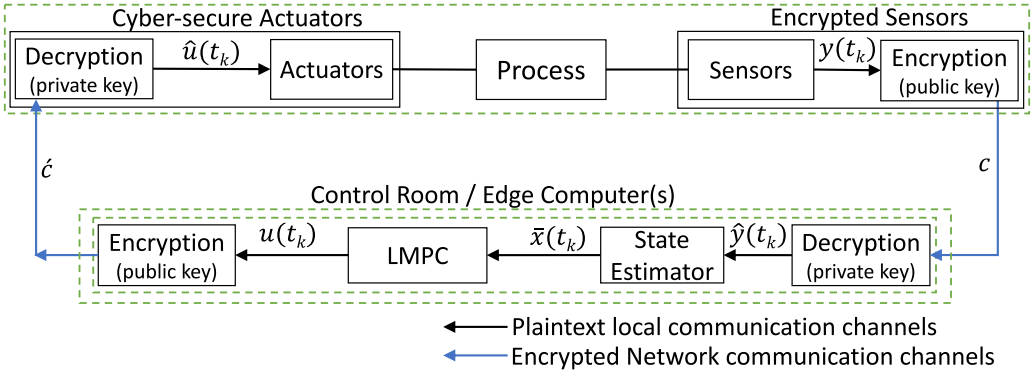

Fig. 10. Illustration of the encrypted centralized control structure.

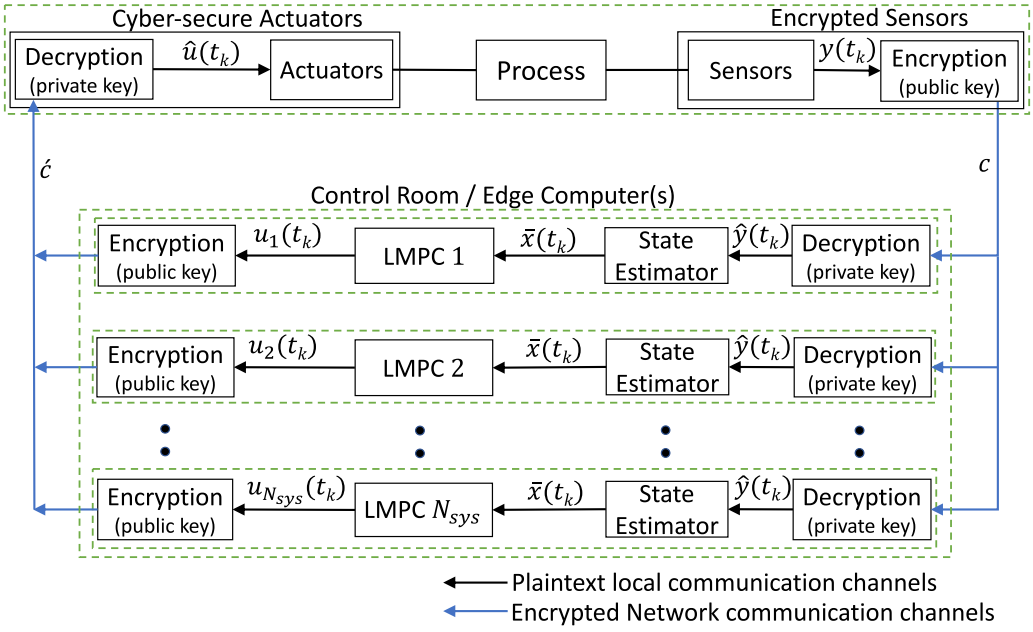

Fig. 11. Illustration of the encrypted decentralized control structure.

with encryption. The decision on which control framework to practically implement should be based on several factors, such as the specific characteristics (size, coupling effect, etc.) of the system to be controlled, the available resources, the desired level of control performance, the budget allocated for computing hardware, and other pertinent considerations.

6. Conclusion

In this study, we introduced and applied encrypted distributed control architectures, both sequential and iterative, with state estimation, to a large-scale nonlinear chemical process network utilizing partial state feedback with sensor noise. We established practical guidelines for implementing this control structure in any nonlinear process by including the selection of key parameters such as l_1, l_2, and d for quantization, and the criterion for setting the sampling time. Through closed-loop simulations, we demonstrated that both the sequential and iterative distributed LMPCs, with encrypted communication between the sensor–controller and controller–actuator links, could stabilize the system within the desired stability region using the extended Luenberger observer for state estimation, in a finite process simulation time. Furthermore, we conducted a comprehensive comparative analysis of various encrypted control strategies, including centralized, decentralized, and distributed approaches with state estimation. The computational time, closed-loop performance, and suitability of the different encrypted control architectures were discussed. In conclusion, our findings indicate that the encrypted iterative distributed LMPC emerges as the most suitable choice for enhancing the cybersecurity of large and complex systems, with highly coupled dynamics between states. This approach reduces the computational complexity associated with centralized control, leverages controller communication to improve closed-loop performance, and maintains a reasonable computation time, while enhancing the cybersecurity of the control system.

Declaration of competing 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 paper.

Acknowledgments

Financial support from the National Science Foundation, United States, CBET-2227241, is gratefully acknowledged.

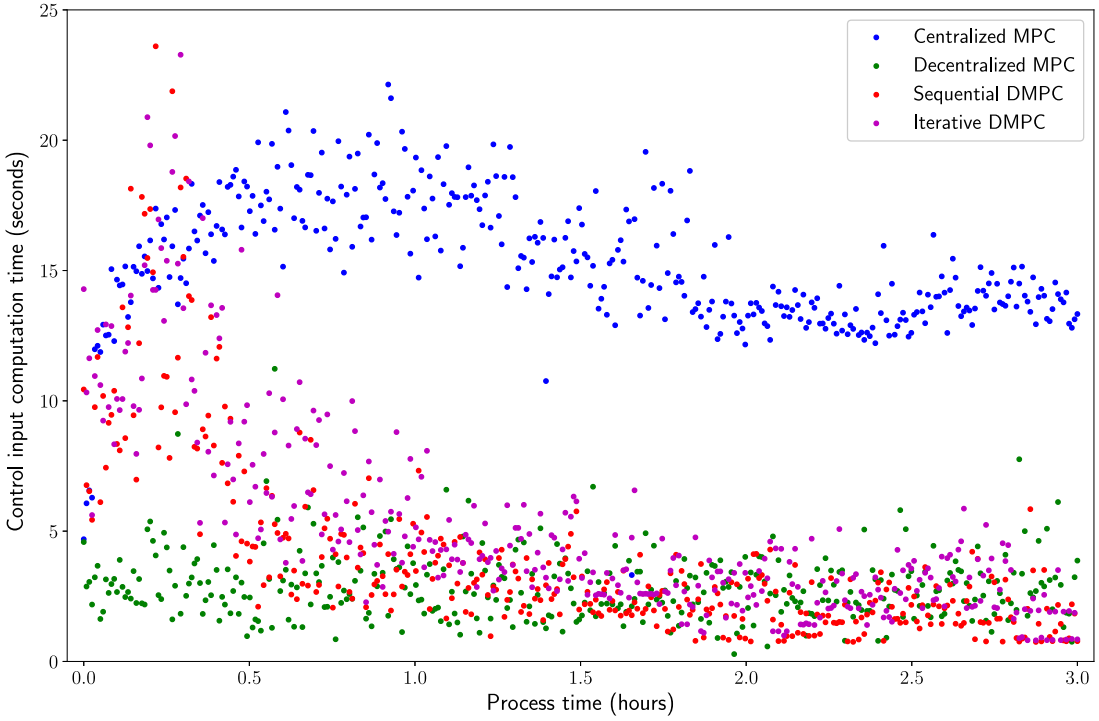

Fig. 12. Control input computation time for the encrypted centralized, decentralized, sequential distributed, and iterative distributed LMPCs at every sampling instance.

References

Al-Abassi, A., Karimipour, H., Dehghantanha, A., Parizi, R.M., 2020. An ensemble deep learning-based cyber-attack detection in industrial control system. IEEE Access 8, 83965–83973.

Alhajeri, M.S., Wu, Z., Rincon, D., Albalawi, F., Christofides, P.D., 2021. Machine-learning-based state estimation and predictive control of nonlinear processes. Chem. Eng. Res. Des. 167, 268–280.

Ali, J.M., Hoang, N.H., Hussain, M.A., Dochain, D., 2015. Review and classification of recent observers applied in chemical process systems. Comput. Chem. Eng. 76, 27–41.

Chen, S., Wu, Z., Christofides, P.D., 2020. Decentralized machine-learning-based predictive control of nonlinear processes. Chem. Eng. Res. Des. 162, 45–60.

Conklin, W.A., 2016. IT vs. OT security: A time to consider a change in CIA to include resilienc. In: Proceedings of 49th Hawaii International Conference on System Sciences. Koloa, Hawaii, pp. 2642–2647.

Darup, M.S., Redder, A., Shames, I., Farokhi, F., Quevedo, D., 2017. Towards encrypted MPC for linear constrained systems. IEEE Control Syst. Lett. 2, 195–200.

Data61, C., 2013. Python paillier library. https://github.com/data61/python-paillier.

Dochain, D., 2003. State and parameter estimation in chemical and biochemical processes: a tutorial. J. Process Control 13, 801–818.

Dutta, V., Choraś, M., Pawlicki, M., Kozik, R., 2020. A deep learning ensemble for network anomaly and cyber-attack detection. Sensors 20, 4583.

Gandhi, R., Sharma, A., Mahoney, W., Sousan, W., Zhu, Q., Laplante, P., 2011. Dimensions of cyber-attacks: Cultural, social, economic, and political. IEEE Technol. Soc. Mag. 30, 28–38.

Kadakia, Y.A., Suryavanshi, A., Alnajdi, A., Abdullah, F., Christofides, P.D., 2023. Encrypted model predictive control of a nonlinear chemical process network. Processes 11, 2501.

Kadakia, Y.A., Suryavanshi, A., Alnajdi, A., Abdullah, F., Christofides, P.D., 2024. Integrating machine learning detection and encrypted control for enhanced cybersecurity of nonlinear processes. Comput. Chem. Eng. 180, 108498.

Kazantzis, N., Kravaris, C., 1998. Nonlinear observer design using Lyapunov's auxiliary theorem. Systems Control Lett. 34 (5), 241–247.

Khan, R., Maynard, P., McLaughlin, K., Laverty, D., Sezer, S., 2016. Threat analysis of BlackEnergy malware for synchrophasor based real-time control and monitoring in smart grid. In: Proceedings of the 4th International Symposium for ICS & SCADA Cyber Security Research. Belfast, United Kingdom, pp. 1–11.

Liu, J., Chen, X., Muñoz de la Peña, D., Christofides, P.D., 2010. Sequential and iterative architectures for distributed model predictive control of nonlinear process systems. AIChE J. 56 (8), 2137–2149.

Paillier, P., 1999. Public-key cryptosystems based on composite degree residuosity classes. In: Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg, pp. 223–238.

Radke, A., Gao, Z., 2006. A survey of state and disturbance observers for practitioners. In: 2006 American Control Conference. IEEE, pp. 5183–5188.

Suryavanshi, A., Alnajdi, A., Alhajeri, M., Abdullah, F., Christofides, P.D., 2023. Encrypted model predictive control design for security to cyberattacks. AIChE J. 69, e18104.

Tsvetanov, T., Slaria, S., 2021. The effect of the colonial pipeline shutdown on gasoline prices. Econom. Lett. 209, 110122.

Wächter, A., Biegler, L.T., 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. 106, 25–57.

Zeitz, M., 1987. The extended luenberger observer for nonlinear systems. Systems Control Lett. 9, 149–156.