Contents lists available at ScienceDirect

Chemical Engineering Research and Design

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

Cyber-security of centralized, decentralized, and distributed control-detector architectures for nonlinear processes

Scarlett Chen[a], Zhe Wu[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

Article history:
Received 23 September 2020
Received in revised form 15 October 2020
Accepted 17 October 2020
Available online 27 October 2020

Keywords:
Feedforward neural networks
Model predictive control
Decentralized control
Cyber-security

ABSTRACT

Decentralized and distributed control systems provide an efficient solution to many challenges of controlling large-scale industrial processes. With the expansion in communication networks, vulnerability to cyber intrusions also increases. This work investigates the effect of different types of standard cyber-attacks on the operation of nonlinear processes under centralized, decentralized, and distributed model predictive control (MPC) systems. The robustness of the decentralized control architecture over distributed and centralized control architectures is analyzed. Moreover, a machine-learning-based detector is trained using sensor data to monitor the cyber security of the overall system. Specifically, detectors built using feed-forward neural networks are used to detect the presence of an attack or identify the subsystem being attacked. A nonlinear chemical process example is simulated to demonstrate the robustness of decentralized control architectures and the effectiveness of the neural-network detection scheme in maintaining the closed-loop stability of the system.

1. Introduction

Most existing methods for analyzing and controlling dynamical systems hinge on the common assumption of centrality, meaning that all the information available about the system is collected and received by a single controller to perform relevant calculations. Many industrial systems, such as chemical production plants, power distribution grids, and urban traffic networks, are considered large-scale systems in which the assumption of centrality cannot hold due to the absence of a centralized information-gathering platform and the lack of centralized computing capabilities. Challenges such as high dimensionality of the system, uncertainties and delays in the communication network, and geographical separation of components, combined with the rapid development of microprocessor technologies, are factors that push for the shift from centralized control to decentralized decision-making and distributed computations (Bemporad et al., 2010; Christofides et al., 2013).

Amongst many advanced control techniques for large-scale systems, model predictive control (MPC) is well known for its ability to handle multi-variable control problems with constraints. Centralized MPC is generally unsuited for the control of large-scale networked systems because of difficulties associated with scalability as well as the maintenance of global models (Bemporad et al., 2010). Therefore, the formulation of decentralized and distributed MPC algorithms has naturally emerged to address these challenges (Bakule, 2008; Christofides et al., 2013). The original optimization problem in the centralized controller is broken down into a number of smaller optimization problems, which are solved in separate decentralized or distributed local controllers. Thus, decentralized and distributed control structures provide a practical solution to decoupling large-scale processes and reducing the computational complexity of a centralized control problem by having multiple MPC controllers that work iteratively and cooperatively to achieve a common control objective applicable to the entire system (Chen et al., 2020c; Liu et al., 2009). The

* Corresponding author.
E-mail address: pdc@seas.ucla.edu (P.D. Christofides).
https://doi.org/10.1016/j.cherd.2020.10.014

advancement in computing and communication technologies has provided an adequate platform for networked control systems to handle the control problem of large-scale, complex processes. However, the existence of networked communication also implies inherent vulnerabilities to cyber-security risks. Past incidents of malicious cyber-attacks on cyber-physical systems, such as the Iranian nuclear plant attack in 2010 (Farwell and Rohozinski, 2011) and the Ukrainian electric power grid attack in 2015 (Case, 2016), demonstrate the sophistication and severity of cyber-security threats, as well as the importance of having suitable Operational Technology (OT) safe-guarding measures in place to prevent, monitor, and mitigate the occurrence of such attacks (Stouffer et al., 2011). Stealthy cyber-attacks may have knowledge of the operational details of the control system, and could target any sensors, actuators, or communication links in the system, with the intention of degrading control performance and destabilizing the process (Cárdenas et al., 2011). Many recent works have been developed, addressing the issue of cyber-security in networked systems from the perspective of system controls. In Teixeira et al. (2010), the cyber-security of linear and nonlinear state estimators was analyzed with respect to intelligent attacks on measurement data. Attack models defined based on their access to system knowledge, disclosure, and disruption resources were shown for a networked control architecture, and cyber-secure networked control was experimentally illustrated in Teixeira et al. (2012). Distributed event-triggered state estimators subject to deception attacks were presented in Liu et al. (2018) to realize the estimation of the decoupling output measurements and coupling intercommunication measurements. In Pang et al. (2016), a stealthy false data injection attack was proposed for both feedback and forward channels to disrupt closed-loop stability while avoiding the detection of a Kalman filter-based networked attack detector. As fundamental process safety can be compromised due to these intelligent cyber-attacks, safe-guarding measures must be considered and incorporated in plant-wide risk assessments (Stouffer et al., 2011). Moreover, as intelligent cyber-attacks may have knowledge of the process model and the operating system, conventional model-based detection methods may be rendered ineffective. With these considerations, in our present work, we propose a data-based detection and monitoring tool for cyber-attacks using sensor measurement data and machine-learning methods.

The applications of machine-learning and deep-learning methods, such as support vector machine, random forest, neural networks, particularly in systems engineering have increased in recent years with the development of available software packages as well as effective algorithms (Zhang et al., 2003; Farnaaz and Jabbar, 2016; Chen et al., 2020d). One prominent use of machine-learning algorithms is to solve classification problems. For example, in Shon and Moon (2007), a hybrid approach using support vector machines and genetic algorithms was implemented and compared to existing network intrusion detection systems in the industry. An overview of recent research directions for applying supervised and unsupervised machine learning techniques to address the problem of anomaly detection was presented in Omar et al. (2013). Neural network and many of its variances have demonstrated remarkable performance. For instance, long short-term memory recurrent neural network was used to build a classifier model for the intrusion detection system in Kim et al. (2016). Feed-forward neural networks were used to construct detectors in Chen et al. (2020a), which were implemented in an economic model predictive control system with combined open-loop and closed-loop control modes to monitor and stay resilient against cyber-attacks. Many variants of convolutional neural networks (CNN) with different topologies, parameters, and structures were analyzed for the task of intrusion detection in cyber-security of network traffic in Vinayakumar et al. (2017), which have shown significant improvement than conventional classifiers. At any large-scale chemical production plant, tremendous amount of data is being collected and archived daily in the historian. Using neural-network algorithms, these data can be put to use to train effective detection devices for monitoring and guarding the system against malicious cyber-attacks.

This work explores the impact of standard types of sensor cyber-attacks on centralized, decentralized, and distributed Lyapunov-based model predictive control (LMPC) systems, and shows the robustness of decentralized control system against certain cyber-attacks when compared to centralized and distributed control systems. We examine Lyapunov-based model predictive control systems in this study, however, the proposed control-detector architecture and detection methodology can be extended to other applications of model predictive control, or other methods of advanced control systems in general. Then, a neural-network-based detector is trained and implemented online to monitor sensor behaviors when the process is operated under the decentralized control system. Section 2 presents the preliminaries on notation, the general class of nonlinear systems considered, and the stabilizability assumptions. Section 3 includes the formulations of the centralized, decentralized, and distributed LMPCs. The design of intelligent cyber-attacks is shown in Section 4, and the development of the cyber-attack detector using neural networks is explained in Section 5. In Section 6, closed-loop simulations and analyses of a two-CSTR-in-series chemical process are presented.

2. Preliminaries

2.1. Notation

Throughout the manuscript, $|\cdot|$ is used to denote the Euclidean norm of a vector. The notation x^T is used to denote the transpose of x. Set subtraction is denoted by "\", i.e., $A \setminus B := \{x \in \mathbf{R}^n | x \in A,\ x \notin B\}$. A continuous function $\alpha : [0, a) \to [0, \infty)$ is said to belong to class $\mathcal{K}$ if it is strictly increasing and is zero only when evaluated at zero. The function $f(\cdot)$ is of class $\mathcal{C}^1$ if it is continuously differentiable in its domain. Lastly, $L_f V(x)$ denotes the standard Lie derivative $L_f V(x) := \frac{\partial V(x)}{\partial x} f(x)$.

2.2. Class of systems

Consider a general class of continuous-time nonlinear systems in which multiple distinct sets of manipulated inputs are used, with each set of manipulated inputs regulating a specific subsystem of the process. We consider $j = 1, \ldots, N_{sys}$ subsystems, with each subsystem j consisting of states x_j which are regulated by only u_j and potentially impacted by states in other subsystems due to coupling between subsystems. The continuous-time nonlinear dynamics of the subsystem j is described as follows:

$$\dot{x}_j = F_j(x, u_j) := f_j(x) + g_j(x)u_j, \quad x_j(t_0) = x_{j_0}, \quad \forall j = 1, \ldots, N_{sys} \qquad (1)$$

where N_{sys} represents the number of subsystems, $x_j \in \mathbf{R}^{n_j}$ represents the state vector for subsystem j, and x represents the vector of all states $x = [x_1^T \ \cdots \ x_{N_{sys}}^T]^T \in \mathbf{R}^n$, where $n = \sum_{j=1}^{N_{sys}} n_j$. $u_j \in \mathbf{R}^{m_j}$ is the set of manipulated input vectors for each subsystem j, which together constitute the vector of all manipulated inputs $u \in \mathbf{R}^m$ with $m = \sum_{j=1}^{N_{sys}} m_j$. The manipulated input vector constraints are defined by $u_j \in U_j := \{u_{j_i}^{\min} \leq u_{j_i} \leq u_{j_i}^{\max}, i = 1, \ldots, m_j\} \subset \mathbf{R}^{m_j}, \forall j = 1, \ldots, N_{sys}$. Therefore, the set that bounds the manipulated input vector u for the overall system is denoted by U, which is the union of all U_j, $j = 1, \ldots, N_{sys}$. $f_j(\cdot)$ and $g_j(\cdot)$ are sufficiently smooth vector and matrix functions of dimensions $n_j \times 1$ and $n_j \times m_j$, respectively. The initial time t_0 is taken to be zero ($t_0 = 0$). We assume that $f_j(0) = 0$, $\forall j = 1, \ldots, N_{sys}$, thus, the origin is a steady-state of the nominal system of Eq. (2a) (i.e., $u(t) \equiv 0$). Therefore, we have $(x_s, u_s) = (0, 0)$, where x_s and u_s are the steady-state state and input vectors for the overall system, respectively. The overall system is described as follows:

$$\dot{x} = F(x, u_1, \ldots, u_{N_{sys}}) := f(x) + \sum_{j=1}^{N_{sys}} g_j(x) u_j \tag{2a}$$

$$\bar{x} = h(x) \tag{2b}$$

where $f(\cdot)$ represents the vector function of dimension $n \times 1$ for all states of the two subsystems $f = [f_1^T \ \cdots \ f_{N_{sys}}^T]^T$. $\bar{x} \in \mathbf{R}^n$ denotes the vector of full state measurements from sensors, which may be compromised by sensor cyber-attacks. When no cyber-attacks are present in the system, $\bar{x} = x$.

2.3. *Stability assumptions*

Depending on the partitioning of the overall large-scale system, there may exist interacting dynamics between the subsystems, where the states of one subsystem may be impacted by the states of other subsystems. We assume that there exist stabilizing control laws $u_j = \Phi_j(x) \in U_j$ which regulate the individual subsystems $j = 1, \ldots, N_{sys}$ and will be applied to the control actuators in the respective subsystems such that the origin of the overall system of Eq. (2a) is rendered asymptotically stable. This implies that there exists a $\mathcal{C}^1$ control Lyapunov function $V(x)$ such that the following inequalities hold for all $x \in \mathbf{R}^n$ in an open neighborhood D around the origin:

$$c_1(|x|) \leq V(x) \leq c_2(|x|), \tag{3a}$$

$$\frac{\partial V(x)}{\partial x} F(x, \boldsymbol{\Phi}(x)) \leq -c_3(|x|), \tag{3b}$$

$$\left| \frac{\partial V(x)}{\partial x} \right| \leq c_4(|x|) \tag{3c}$$

where c_1, c_2, c_3 and c_4 are class $\mathcal{K}$ functions. $F(x, u)$ represents the overall nonlinear system of Eq. (2a). $\boldsymbol{\Phi}(x) = [\Phi_1(x)^T \cdots \Phi_{N_{sys}}(x)^T]^T$ represents the vector containing the candidate control laws for each subsystem j, i.e., $\Phi_j(x) \in \mathbf{R}^{m_j}$, for $j = 1, \ldots, N_{sys}$. The candidate controller for each subsystem j is given in the following form:

$$\phi_{j_i}(x) = \begin{cases} -\dfrac{p + \sqrt{p^2 + q^4}}{q^T q} q & \text{if } \ q \neq 0 \\ 0 & \text{if } \ q = 0 \end{cases} \tag{4a}$$

$$\Phi_{j_i}(x) = \begin{cases} u_{j_i}^{\min} & \text{if } \ \phi_{j_i}(x) < u_{j_i}^{\min} \\ \phi_{j_i}(x) & \text{if } \ u_{j_i}^{\min} \leq \phi_{j_i}(x) \leq u_{j_i}^{\max} \\ u_{j_i}^{\max} & \text{if } \ \phi_{j_i}(x) > u_{j_i}^{\max} \end{cases} \tag{4b}$$

Depending on whether the control system is decentralized or distributed, the candidate control laws of Eq. (4) may be calculated differently. In a decentralized control system, each controller has knowledge of the dynamics of the local subsystem that it regulates, but does not have access to the model dynamics of the entire process. Therefore, when the candidate controller of Eq. (4) is implemented, p in Eq. (4a) denotes $\frac{\partial V(x)}{\partial x_j} f_j(x)$ and q denotes $\frac{\partial V(x)}{\partial x_j} g_{j_i}(x)$. Here, $f_j \in \mathbf{R}^{n_j}$ and $g_{j_i} \in \mathbf{R}^{n_j \times m_j}$ for $j = 1, \ldots, N_{sys}$, and $i = 1, \ldots, m_j$ for subsystem j corresponding to the vector of control actions $\Phi_j(x) \in \mathbf{R}^{m_j}$. The control Lyapunov function $V(x)$ can be a linear combination of multiple control Lyapunov functions V_j. Each V_j is designated for the subsystem j and is a function of x_j only, i.e., $V(x) = \sum_{j=1}^{N_{sys}} V_j(x_j)$. Thus, $\frac{\partial V(x)}{\partial x_j} = \frac{\partial V_j(x_j)}{\partial x_j}, \quad \forall \ j = 1, \ldots, N_{sys}$, and the time-derivative of V can be represented as $\dot{V}(x) = L_f V + L_g V u = \sum_{j=1}^{N_{sys}} \frac{\partial V(x)}{\partial x_j} (f_j + \sum_{i=1}^{m_j} g_{j_i} u_{j_i})$. On the other hand, in a distributed control system, each controller has knowledge of the dynamics of the overall process, and calculates the control actions for each corresponding subsystem accordingly. Thus, when the candidate controller $\Phi_{j_i}(x)$ of Eq. (4) is calculated, p denotes $L_f V(x) = \frac{\partial V(x)}{\partial x} f(x) = \sum_{j=1}^{N_{sys}} \frac{\partial V(x)}{\partial x_j} f_j(x)$ and q denotes $L_{g_{j_i}} V(x) = \frac{\partial V(x)}{\partial x_j} g_{j_i}$ for subsystems $j = 1, \ldots, N_{sys}$, and $i = 1, \ldots, m_j$ number of inputs for subsystem j; here, we only consider g_{j_i} because $\Phi_j(x) \in \mathbf{R}^{m_j}$ regulates the states $x_j \in \mathbf{R}^{n_j}$ of the subsystem j only.

$\phi_{j_i}(x)$ of Eq. (4a) represents the ith component of the control law $\phi_j(x)$. $\Phi_{j_i}(x)$ of Eq. (4) represents the ith component of the saturated control law $\Phi_j(x)$ that accounts for the input constraints $u_j \in U_j$ for subsystem j. Note that the candidate control law $\Phi_j(x)$ is calculated based on the nonlinear dynamics of the subsystem j of Eq. (1), and the set of candidate control laws for the overall system is denoted as $\boldsymbol{\Phi}(x) = [\Phi_1(x)^T \cdots \Phi_{N_{sys}}(x)^T]^T \in U$, which together can render the overall system of Eq. (2a) asymptotically stable.

Based on Eq. (3), we first characterize a region where the time-derivative of the Control Lyapunov function V is rendered negative definite under the candidate control laws $\boldsymbol{\Phi}(x) \in U$ as $D = \{x \in \mathbf{R}^n | \dot{V}(x) = L_f V + L_g V u \leq -c_3(|x|), u = \boldsymbol{\Phi}(x) \in U\} \cup \{0\}$. Then, the closed-loop stability region Ω_ρ for the nonlinear system of Eq. (2a) is defined as a level set of V, which is an invariant set for the closed-loop system inside D: $\Omega_\rho := \{x \in D | V(x) \leq \rho\}$, where $\rho > 0$ and $\Omega_\rho \subset D$. Therefore, starting from any initial state $x_0 := x(t_0)$ in Ω_ρ, $\boldsymbol{\Phi}(x(t))$ guarantees that the state trajectory of the closed-loop system of Eq. (2a) remains within Ω_ρ and asymptotically converges to the origin. Thus, given that the sensor measurements received by the controller are secure and reliable (i.e., $\bar{x} = x$), the control law $\boldsymbol{\Phi}(x)$ is able to stabilize the process at the origin for any initial conditions $x_0 \in \Omega_\rho$.

3. Centralized, decentralized, and distributed Lyapunov-based model predictive control

3.1. Centralized LMPC

Traditionally, when the overall process is regulated by a centralized controller, the control problem that is solved incorporates all the manipulated inputs and state measurements of the process. Specifically, the centralized Lyapunov-based model predictive control (LMPC) is represented by the following optimization problem:

$$\mathcal{J} = \min_{u \in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{x}(t), u(t))dt \tag{5a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = F(\tilde{x}(t), u(t)) \tag{5b}$$

$$u(t) \in U, \quad \forall t \in [t_k, t_{k+N}) \tag{5c}$$

$$\tilde{x}(t_k) = \bar{x}(t_k) \tag{5d}$$

$$\dot{V}(\bar{x}(t_k), u) \leq \dot{V}(\bar{x}(t_k), \boldsymbol{\Phi}(\bar{x}(t_k))), \quad \text{if} \quad \bar{x}(t_k) \in \Omega_\rho \backslash \Omega_{\rho_s} \tag{5e}$$

$$V(\tilde{x}(t)) \leq \rho_s, \quad \forall t \in [t_k, t_{k+N}), \quad \text{if} \quad \bar{x}(t_k) \in \Omega_{\rho_s} \tag{5f}$$

where $\tilde{x}$ is the predicted state trajectory, $S(\Delta)$ is the set of piecewise constant functions with period Δ, and N is the number of sampling periods in the prediction horizon. $\dot{V}(x, u)$ is used to represent $\frac{\partial V(x)}{\partial x} F(x, u)$. The optimal input trajectory computed by the centralized LMPC is denoted by $u^*(t)$, which is calculated over the entire prediction horizon $t \in [t_k, t_{k+N})$. The control action computed for the first sampling period of the prediction horizon $u^*(t_k)$ is sent by the LMPC to be applied over the next sampling period in a sample-and-hold manner, and the centralized LMPC is solved again in a rolling horizon fashion. The LMPC of Eq. (5) is solved by minimizing the time integral of the objective function $L(\tilde{x}(t), u(t))$ of Eq. (5a) over the prediction horizon N, subject to the constraints of Eqs. (5b)–(5f). The objective function is generally in a quadratic form of $\tilde{x}^T Q \tilde{x} + u^T R u$ such that the states can be driven to the operating steady-state of the process without exhausting too much inputs; here, Q and R are the weighting matrices for the states and the inputs, respectively. Eq. (5c) defines the input constraints applied over the entire prediction horizon, and Eq. (5d) defines the initial condition $\tilde{x}(t_k)$ of Eq. (5b), which is the state measurement $\bar{x}(t)$ at $t = t_k$. The constraint of Eq. (5e) forces the closed-loop state to move towards the origin at a minimum rate characterized by the Lyapunov function V and the Lyapunov-based control law $\boldsymbol{\Phi}(\bar{x}(t_k))$, if $\bar{x}(t_k) \in \Omega_\rho \backslash \Omega_{\rho_s}$, where Ω_ρ is the stability region and Ω_{ρ_s} is the ultimate bounded region around the operating steady-state which the closed-loop states of Eq. (5b) will converge to. If $\bar{x}(t_k)$ enters Ω_{ρ_s}, Eq. (5f) ensures that the states predicted by Eq. (5b) will be maintained in Ω_{ρ_s} for the entire prediction horizon.

3.2. Decentralized LMPC

When the optimization problem of a centralized MPC is too complex to solve within a reasonable time period (i.e., the sampling period), the control problem may be decoupled into smaller local optimization problems that are solved in separate processors/controllers to achieve improved computational efficiency. In a decentralized LMPC system, no communication is established between the different local controllers, therefore each controller does not have any knowledge on the control actions calculated by the other controllers.

We design separate $j = 1, \ldots, N_{sys}$ LMPCs, each designated to regulate the states x_j of one subsystem j, and compute the respective control actions. The trajectories of control actions computed by LMPC j are denoted by u_{d_j}, which are applied to the corresponding control actuators in subsystem j. Each decentralized LMPC may receive full-state feedback measurements, but they only have information on the dynamic behavior of their respective subsystem. The mathematical formulation of each decentralized LMPC j, $j = 1, \ldots, N_{sys}$, is shown as follows:

$$\mathcal{J}_j = \min_{u_{d_j} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{x}_j(t), u_{d_j}(t))dt \tag{6a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}_j(t) = F_j(\hat{x}(t), u_{d_j}(t)) \tag{6b}$$

$$\hat{x}(t) = [\bar{x}_1(t_k)^T \ \cdots \ \bar{x}_{j-1}(t_k)^T \ \tilde{x}_j(t)^T \ \bar{x}_{j+1}(t_k)^T \ \cdots \ \bar{x}_{N_{sys}}(t_k)^T]^T \tag{6c}$$

$$u_{d_j}(t) \in U_j, \quad \forall t \in [t_k, t_{k+N}) \tag{6d}$$

$$\tilde{x}_j(t_k) = \bar{x}_j(t_k) \tag{6e}$$

$$\frac{\partial V(\bar{x}(t_k))}{\partial x_j}(F_j(\bar{x}(t_k), u_{d_j}(t_k))) \leq \frac{\partial V(\bar{x}(t_k))}{\partial x_j}(F_j(\bar{x}(t_k), \Phi_j(\bar{x}(t_k)))), \quad \text{if} \quad \bar{x}(t_k) \in \Omega_\rho \backslash \Omega_{\rho_s} \tag{6f}$$

$$V(\tilde{x}(t)) \leq \rho_s, \quad \forall t \in [t_k, t_{k+N}), \quad \text{if} \quad \bar{x}(t_k) \in \Omega_{\rho_s} \tag{6g}$$

where $\tilde{x}_j$ is the predicted state trajectory for subsystem j, $S(\Delta)$ is the set of piecewise constant functions with period Δ, and N is the number of sampling periods in the prediction horizon for subsystem j. The optimal input trajectory computed by this LMPC j is denoted by $u^*_{d_j}(t)$, which is calculated over the entire prediction horizon $t \in [t_k, t_{k+N})$. The control action computed for the first sampling period of the prediction horizon $u^*_{d_j}(t_k)$ is sent by LMPC j to its control actuators in subsystem j to be applied over the next sampling period. The objective function of Eq. (6a) is the integral of $L(\tilde{x}_j(t), u_{d_j}(t))$ over the prediction horizon, where $L(x_j, u_j)$ is typically in a quadratic form of $x_j^T Q_j x_j + u_j^T R_j u_j$, and Q_j and R_j are the weighting matrices of the states and the inputs of subsystem j respectively. The states of each subsystem will be driven towards the origin by minimizing this objective function. The constraint of Eq. (6b) is the first-principles model of Eq. (1) used to predict the states of the closed-loop subsystem j. $\hat{x}(t)$ is a vector containing the predicted states of subsystem j, $\tilde{x}_j(t)$, and the measured states of all other subsystems at $t = t_k$, $\bar{x}_i(t_k)$, $\forall i = 1, \ldots, N_{sys}$, and $i \neq j$. Eq. (6d) is the input constraints on u_{d_j} applied over the entire prediction horizon, and Eq. (6e) defines the initial conditions of Eq. (6b) for subsys-

tem j, which is the state measurement $\bar{x}_j(t)$ at $t = t_k$. Eq. (6f) is a constraint that forces the closed-loop state of subsystem j to move towards the origin at a minimum rate characterized by the Lyapunov function V and the Lyapunov-based control law $\Phi_j(\bar{x}(t_k))$ if $\bar{x}(t_k) \in \Omega_\rho \backslash \Omega_{\rho_s}$. If $\bar{x}(t_k)$ enters the terminal set region Ω_{ρ_s}, then the states $\tilde{x}_j$ predicted by Eq. (6b) will be maintained in Ω_{ρ_s} for the entire prediction horizon.

Since the decentralized LMPCs solve different optimization problems specific to their respective subsystems and are computed in separate processors in parallel, the computation time for solving one iteration of the entire decentralized LMPC design in one sampling period (assuming that the feedback measurements are available to both controllers at synchronous intervals) will be the maximum time out of the two LMPCs.

3.3. *Distributed LMPC*

To achieve better closed-loop control performance, some level of communication may be established between the different controllers. In this study, we consider an iterative distributed MPC system, which allows signal exchanges between all controllers, thereby allowing each controller to have full knowledge of the predicted state evolution along the prediction horizon and yielding better closed-loop performance via multiple iterations at the cost of more computational time. In the context that two LMPCs are designed, both controllers communicate with each other to cooperatively optimize the control actions. The controllers solve their respective optimization problems independently in a parallel structure, and solutions to each control problem are exchanged at the end of each iteration. More specifically, the following implementation strategy is used:

1. At each sampling instant t_k, each LMPC j, $j = 1, \ldots, N_{sys}$, receives the state measurement $\bar{x}(t)$ at $t = t_k$ from all the sensors.
2. At iteration $c = 1$, each LMPC j evaluates future trajectories of $u_{d_j}(t)$ assuming the control actions of all other subsystemsare calculated by the Lyapunov-based control law, $u_i(t) = \Phi_i(\bar{x}(t_k)), \forall t \in [t_k, t_{k+N}), i = 1, \ldots, N_{sys}, i \neq j$. The LMPCs then exchange their future input trajectories, calculate and store the value of their own objective function.
3. At iteration $c > 1$:
   (a) Each LMPC evaluates its own future input trajectory based on state measurement $\bar{x}(t_k)$ and the latest received input trajectories from the other LMPCs.
   (b) The LMPCs exchange their future input trajectories. Each LMPC calculates and stores the value of the cost function.
4. When a termination criterion is satisfied, each LMPC sends its entire future input trajectory corresponding to the smallest value of the cost function to its actuators. If the termination criterion is not satisfied, go to back Step 3 ($c \longleftarrow c + 1$).
5. When a new state measurement $\bar{x}$ is received, go to Step 1 ($k \longleftarrow k + 1$).

Following the same variables and constraints as defined in a decentralized LMPC design, the optimization problem of LMPC 1 in an iterative distributed LMPC at iteration $c = 1$ is presented as follows:

$$\mathcal{J} = \min_{u_{d_j} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{x}(t), u_{d_j}(t), \Phi_i(\tilde{x}(t)))dt \tag{7a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = F(\tilde{x}(t), u_{d_j}(t), \Phi_i(\tilde{x}(t))) \tag{7b}$$

$$u_{d_j}(t) \in U_j, \quad \forall t \in [t_k, t_{k+N}) \tag{7c}$$

$$\tilde{x}(t_k) = \bar{x}(t_k) \tag{7d}$$

$$\frac{\partial V(\bar{x}(t_k))}{\partial x}(F(\bar{x}(t_k), u_{d_j}(t_k), \Phi_i(\bar{x}(t_k)))) \leq \frac{\partial V(\bar{x}(t_k))}{\partial x}(F(\bar{x}(t_k), \boldsymbol{\Phi}(\bar{x}(t_k)))), \quad \text{if} \quad \bar{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s} \tag{7e}$$

$$V(\tilde{x}(t)) \leq \rho_s, \quad \forall t \in [t_k, t_{k+N}), \quad \text{if} \quad \bar{x}(t_k) \in \Omega_{\rho_s} \tag{7f}$$

where for each control action j corresponding to subsystem j, $i = 1, \ldots, N_{sys}$, $i \neq j$, which refers to the control actions of all other subsystems except for j. At iteration $c > 1$, after the exchange of the optimized input trajectories $u^*_{d_j}(t), \forall t \in [t_k, t_{k+N})$ between all LMPCs $j = 1, \ldots, N_{sys}$, the optimization problem of LMPC j is as follows:

$$\mathcal{J} = \min_{u_{d_j} \in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{x}(t), u_{d_j}(t), u^*_{d_i}(t))dt \tag{8a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = F(\tilde{x}(t), u_{d_j}(t), u^*_{d_i}(t)) \tag{8b}$$

$$u_{d_j}(t) \in U_j, \quad \forall t \in [t_k, t_{k+N}) \tag{8c}$$

$$\tilde{x}(t_k) = \bar{x}(t_k) \tag{8d}$$

$$\frac{\partial V(\bar{x}(t_k))}{\partial x}(F(\bar{x}(t_k), u_{d_j}(t_k), u^*_{d_i}(t_k)) \leq \frac{\partial V(\bar{x}(t_k))}{\partial x}(F(\bar{x}(t_k), \boldsymbol{\Phi}(\bar{x}(t_k)))), \quad \text{if} \quad \bar{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s} \tag{8e}$$

$$V(\tilde{x}(t)) \leq \rho_s, \quad \forall t \in [t_k, t_{k+N}), \quad \text{if} \quad \bar{x}(t_k) \in \Omega_{\rho_s} \tag{8f}$$

At each iteration $c \geq 1$, the two LMPCs can be solved simultaneously via parallel computing in separate processors. Therefore, the total computation time required for iterative distributed LMPC is the maximum solving time out of the two controllers accounting for all the iterations required before the termination criterion is met. The termination condition can be of many forms, e.g., c must not exceed a maximum iteration number c_{max} (i.e., $c \leq c_{max}$), the computational time for solving each LMPC must not exceed a maximum time period, or the difference in the cost function or of the solution trajectory between two consecutive iterations is smaller than a threshold value. During implementation, when one such criterion is met, the iterations will be terminated.

Once the optimization problems of all subsystems $j = 1 \ldots, N_{sys}$ are solved, the optimal control actions of the proposed decentralized or distributed LMPC systems are defined as follows:

$$u_j(t) = u^*_{d_j}(t_k), \quad j = 1, \ldots, N_{sys}, \quad \forall t \in [t_k, t_{k+1}) \tag{9}$$

4. Intelligent cyber-attacks

With the intention of destroying the control objectives of the system, cyber-attacks can jeopardize system stability. Intelligent cyber-attacks could target sensors, actuators, and/or the communication channels in the system (Cárdenas et al., 2011). In this work, we only consider sensor cyber-attacks. Falsified sensor feedback measurements that do not accurately report the true states of the process will result in incorrect control actions being calculated by the controllers, which may ultimately drive the true process states outside of the stability region. The cyber-attacks may have access to the stability region which the controllers are designed to operate the process within, and correspondingly they may have knowledge on any existing alarms configured on the input and output variables such that no alarms will be triggered to bring human attention and intervention, and the controllers are still able to compute feasible control actions based on the falsified sensor measurements.

4.1. *Design of cyber-attacks on sensors*

Sensor cyber-attacks can target any sensor in the overall system, regardless of the structure of the system being centralized, decentralized, or distributed. There are some standard types of cyber-attacks considered in literature (Singh and Nene, 2013), and the formulation details of three of them are shown in this section (Chen et al., 2020b). For simplicity, the following cyber-attacks will be introduced in the context of a decentralized LMPC system where the Lyapunov function of each subsystem V_j is assessed independently. However, the same form of cyber-attacks can be applied to systems under centralized and distributed control as well. Note that V_j is a function of x_j exclusively. Therefore, the design of the cyber-attacks is with respect to the stability region of each subsystem respectively.

Remark 1. In this work, we only consider sensor cyber-attacks. However, cyber-attacks can also happen in actuators or in any communication channels. Specifically, if the attacker has knowledge on the architecture of the control system being distributed, the attacks could be injected in the inter-controller communication channels. The controllers communicate their future input trajectories with each other, which gives information on the future predicted states of other subsystems. If such information is falsified, the controllers will calculate incorrect control actions based on the received wrongful information, and these incorrect control actions will get communicated to other controllers again through iterations, and the impact of the cyber-attack will propagate and magnify. Investigating the inter-controller communication channel attacks and detecting such attacks based on sensor data or other forms of metrics will be an interesting future research direction.

4.1.1. *Min–max cyber-attack*

Maximum disruption impact is achieved by a min–max attack within a short period of time without triggering any alarms. The falsified sensor measurement will be set to a value furthest away from the value of the current true state without exiting the stability region, and thus the resulting falsified measurements $\bar{x}_j$ will be on the boundary of the stability region, i.e., $V_j(\bar{x}_j) = \rho_j$. Depending on the value of x_j, the furthest value from the current state might be the maximum or the minimum as characterized by the boundary of the stability region. Therefore, the min–max attack on one particular subsystem j can be formulated as follows:

$$\bar{x}_j(t_i) = arg(\min / \max)_{x_j}\{x_j \in R^{n_j} | V_j(x_j(t_i)) = \rho_j\}, \quad \forall \ i \in [i_0, i_0 + L_a] \tag{10}$$

where i_0 is the time instant that the attack is injected, $\bar{x}_j$ denotes the compromised sensor measurement, L_a is the total duration of the attack in terms of sampling periods, ρ_j refers to the level set of the Lyapunov function $V_j(x_j)$ that characterizes the stability region of the closed-loop subsystem of Eq. (1) under the decentralized LMPC.

4.1.2. *Surge cyber-attack*

Surge attacks induce maximum deviation for an initial short period, and then the attacked value is set to a reduced value thereafter such that the cumulative deviation will not exceed a certain threshold, which is typically examined by conventional detection methods such as cumulative sum (Mohanty et al., 2007; Cárdenas et al., 2011). The initial surge value that causes maximum impact is also defined based on the stability region of the subsystem j, Ω_{ρ_j}. Designing the length of the initial surge period and the attacked sensor measurement after the surge can be of many forms, as long as the cumulative error between state measurements and their predicted true values (usually given by an estimation algorithm) for the entire attack duration does not exceed the statistics-based threshold defined in other detection methods (e.g., CUSUM). For our study, the attack after the initial surge is designed to act as a sufficiently small bounded noise imposed on the attacked sensor. The formulation of the surge attack is as follows:

$$\begin{aligned} &\bar{x}_j(t_i) = arg(\min / \max)_{x_j}\{x_j \in R^{n_j} | V_j(x_j(t_i)) \\ &= \rho_j\}, \quad \text{if} \ \ i_0 \le i \le i_0 + L_s \\ &\bar{x}_j(t_i) = x_j(t_i) + \eta(t_i), \quad V_j(\bar{x}_j(t_i)) < \rho_j, \quad \text{if} \ \ i_0 + L_s < i \le i_0 + L_a \end{aligned} \tag{11}$$

where L_s is the duration of the initial surge, and $\eta_l \le \eta(t_i) \le \eta_u$ is the bounded noise added on the sensor measurement after the initial surge period, where η_l and η_u are the lower and upper bounds of the noise, respectively. Here, the reduced attack value is designed to have much lesser magnitude and lower impact compared to the min/max value achieved during the initial surge so that the attack could last for a long time without being detected by conventional detection methods. Therefore, the Lyapunov function of the reduced attack measurement should not exceed the boundary of the stability region, i.e., $V_j(\bar{x}_j) < \rho_j$. In the case that the additive noise does generate $\bar{x}_j(t_i)$ such that $V_j(\bar{x}_j(t_i)) > \rho_j$ due to the true state x_j already outside of the stability region, $\bar{x}_j$ after the surge would be set to the closest point on the boundary of the stability region to the true state value until the true state returns inside the stability region, after which $\bar{x}_j$ would take the form of having a sufficiently small bounded noise added onto the true state measurement.

4.1.3. *Geometric cyber-attack*

Under geometric cyber-attacks, closed-loop system stability deteriorates at a geometric speed until the attacked value reaches the maximum or minimum allowable limit as char-

acterized by the stability region of the subsystem. Geometric attacks can be written as follows:

$$\bar{x}_j(t_i) = x_j(t_i) + \beta \times (1+\alpha)^{i-i_0}, \quad \forall \ i \in [i_0, i_0 + L_a] \tag{12}$$

where β and α define the initial magnitude and the speed of the geometric attack, respectively. A small constant $\beta \in \mathbf{R}$ is added to the true measured output $x_j(t_{i_0})$ at the start of the attack such that $x_j(t_{i_0}) + \beta$ is well below the alarm threshold. At each subsequent time step after t_{i_0}, β is multiplied by a factor $(1+\alpha)$, $\alpha \in (0, 1)$, until $\bar{x}_j$ reaches the maximum allowable attack value bounded by Ω_{ρ_j}. $\bar{x}_j$ is increasing or decreasing geometrically depending on the sign of the parameter β. Attackers will choose the two parameters α and β based on Ω_{ρ_j} and the attack duration L_a.

4.2. *Robustness of decentralized LMPC against cyber-attacks*

Both decentralized and distributed LMPC systems are designed to alleviate the computational complexity and effort of a centralized control problem regulating multiple subsystems. Considering the inherent structure and operating requirement of both systems, the decentralized control system exhibits greater robustness against potential cyber-attacks. Firstly, there must exist inter-controller communication in a distributed control system to exchange information on the control actions calculated by each distributed local controller. With additional communication channels between controllers built for this purpose, it creates a greater exposure surface that is vulnerable to cyber-attacks targeting communication channels. Controllers in the decentralized control system do not share any information, and therefore, eliminating this layer of potential threat. Secondly, every local controller in a distributed control system has knowledge on how the overall process dynamically evolves, and receives full-state measurements of the entire process at every sampling period as required. This means that, if any one of the many sensors of the entire system is falsified, it would result in erroneous calculations in all local controllers, and the error would propagate as the incorrect control actions calculated by the controllers are exchanged. On the other hand, decentralized controllers only have knowledge of the process dynamics of the local subsystem, and are designed to only regulate the states of the respective subsystem. Depending on how the overall process is partitioned, local controllers in the decentralized control system may not need information on feedback measurements of the process states of other subsystems. Therefore, in the case that only one or a few subsystems are attacked, the decentralized controllers regulating those un-attacked subsystems will not be affected at all, and are still able to maintain local subsystem stability. This will be demonstrated in the simulation studies in Section 6.

5. Detection of cyber-attacks

Since physical processes may be prone to structural or parameter changes, and sophisticated stealthy attacks may have knowledge on the process behavior and the underlying model, model-based detection algorithms may be less effective than data-based detection methods (Cárdenas et al., 2011). Most model-based detection methods – such as the Cumulative Sum method – rely on the availability of an accurate model that describes the dynamics of the process, and they determine the presence of an anomaly or a cyber-attack in the system by examining the discrepancy between a model-based predictive estimate and the actual process output. There are some disadvantages to these model-based methods. One disadvantage is that during real-time operation, the process model is subject to structural and parameter changes, and therefore the model used to construct the detector also needs to be adjusted to reflect the same changes. Another disadvantage is that, the intelligent cyber-attacks may have information on the setup of the control system including the design of the model-based detector, and therefore, could adopt attack tactics that counteract the detection scheme. For example, a Cumulative Sum method would not be able to detect surge attacks or any other forms of attacks that have prior knowledge on the detection threshold imposed on the cumulative error in the state measurement. Therefore, these potential shortcomings provide motivation for developing data-based detection methods which are independent from the physical process model and are resilient to process changes and stealthy attacks designed based on process behavior. Amongst data-based methods, machine learning algorithms provide many advanced and flexible capabilities of classification and regression; more specifically, by using different categories of training data and constructing different training algorithms, the machine-learning-based detector may be designed and optimized to perform a variety of different detection tasks. The interested readers may also refer to Wu et al. (2018) for a numerical analysis on the detection accuracy of a machine-learning-based detection method and a model-based statistics detection method. Neural networks (NN), as one of the advanced machine-learning techniques, have proven to be successful in a variety of applications, solving both supervised and unsupervised classification problems. One advantage of NN over other classification methods such as k-nearest-neighbors, random forest, and support vector machine, is that a large number of nonlinear activation functions, tuning parameters, and alternative training algorithms can be used to optimize the overall model accuracy (Tu, 1996).

In our study, we use a feed-forward neural network (FNN) for supervised classification. In supervised classification, the NN will be trained using data with labels corresponding to each target class, and its task will be to classify new data samples into the respective known classes that the new data identifies the most similarities with. A conventional FNN consists of an input layer, an output layer, and a customizable number of hidden layers in between. Each layer undergoes a series of nonlinear functions, which are activation functions of the weight sum of inputs (or neurons in the previous layer) with a bias term, and provides the values for the neurons in the current layer. The neurons in the first hidden layer are derived from the inputs, and the outputs are calculated from the neurons in the last hidden layer.

As the accuracy of the FNN model depends heavily on the types and quality of the training data, selecting the input features for the FNN model that effectively and concisely capture the evolution of the process states is critical. Considering the control objective of the system, the states of the nonlinear subsystems of Eq. (1) are driven towards the operating steady-state under the decentralized or the distributed LMPC subject to Lyapunov-based constraints. The Lyapunov function of the subsystem, $V_j(\bar{x}_j)$, which is a function of the measured process states $\bar{x}_j$ of the respective subsystem j only, captures the dynamic features of all states. Thus,

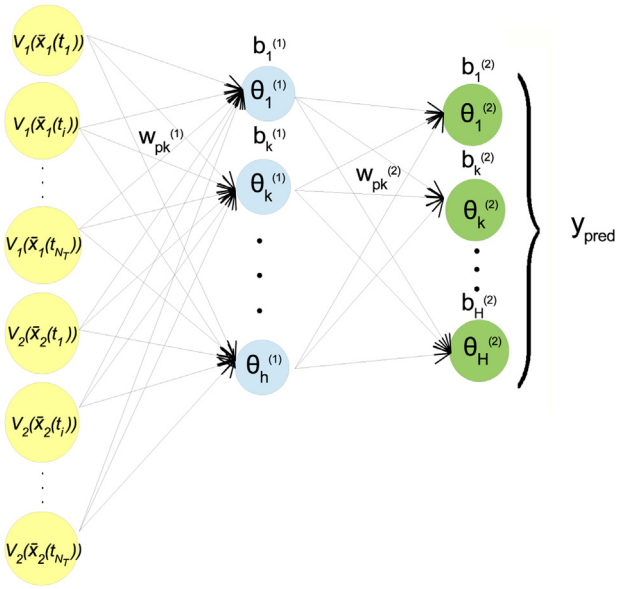

Fig. 1 – Feed-forward neural network structure with 1 hidden layer with inputs being the vector of Lyapunov functions of two subsystems $V_1(\bar{x}_1(t_i))$ and $V_2(\bar{x}_2(t_i))$ with a detection window $i = 1, \ldots, N_T$, and output being the probability of each class label for the examined trajectory of length N_T indicating the status and/or location of cyber-attack.

$V_j(\bar{x}_j)$ is an effective one-dimensional parameter used as the input variables for the attack detection problem. More specifically, since multiple subsystems are involved, we record the Lyapunov function of all subsystems using the state measurements recorded along the time-varying trajectory, i.e., $V_1(\bar{x}_1(t_i))$, $V_2(\bar{x}_2(t_i))$, $i = 1, \ldots, N_T$, and concatenate V_1 and V_2 as a single one-dimensional vector of dimension $1 \times (2N_T)$. The resulting vector contains information on the evolution of the process states of the two subsystems independently, and will be used as the input vector into the feed-forward NN detector model to determine whether abnormalities exist in any of the sensor measurements in the latest time window of $\Delta \times N_T$, where Δ is the sampling period of the LMPC system. Since the information of the state measurements of the two subsystems are recorded and presented in the detector input vector independently, depending on the structure of the NN detector model and the classes of the training data, the FNN detector can be trained to either detect the presence of an attack anywhere in the overall system, or identify the location of an attack (i.e., which subsystem the problematic sensors are located).

The structure of a basic one-hidden-layer FNN model for cyber-attack detection is shown in Fig. 1. On the input layer, each neuron represents the Lyapunov function $V_j(\bar{x}_j(t_i))$ at time instant t_i, for $j = 1, 2$, $i = 1, \ldots, N_T$, and resulting in a total of $p = 1, \ldots, 2N_T$ neurons. The output layer provides an output vector the predicted class label, where the number of neurons corresponds to the number of possible classes. The mathematical formulation of a one-hidden-layer FNN model is shown as follows:

$$\theta_k^{(1)} = g_1\left(\sum_{p=1}^{2N_T} w_{pk}^{(1)} V_j(\bar{x}_j(t_i)) + b_k^{(1)}\right) \tag{13a}$$

$$\theta_k^{(2)} = g_2\left(\sum_{p=1}^{h} w_{pk}^{(2)} \theta_p^{(1)} + b_k^{(2)}\right) \tag{13b}$$

$$y_{pred} = [\theta_1^{(2)}, \theta_2^{(2)}, \ldots, \theta_H^{(2)}]^T \tag{13c}$$

with $\theta_k^{(1)}$ representing neurons in the hidden layer, where $k = 1, \ldots, h$ is the number of neurons in hidden layer. $\theta_k^{(2)}$ represents neurons in the output layer, where $k = 1, \ldots, H$, and H is the number of class labels. In this work, we use one hidden layer for this FNN detector model. Using the similar formulations as Eq. (13), multiple hidden layers can also be constructed. The input layer for each sample consists of variables $V_1(\bar{x}_1(t_i))$ and $V_2(\bar{x}_2(t_i))$ for $i = 1, \ldots, N_T$. The weights connecting neurons p and k in consecutive layers (from layer $l-1$ to layer l) are $w_{pk}^{(l)}$, and the bias term on the kth neuron in the lth layer is $b_k^{(l)}$. Each layer calculates an output using the information received from the previous layer, the optimized weights, biases, which are all passed into a nonlinear activation function g_l (some examples include hyperbolic tangent transfer function $g(z) = \frac{2}{1+e^{-2z}} - 1$, sigmoid function $g(z) = \frac{1}{1+e^{-z}}$, and softmax function $g(z_k) = \frac{e^{z_k}}{\sum_{i=1}^{H} e^{z_i}}$ where H is the number of class labels). The output layer consists of a vector, y_{pred}, which gives the predicted probabilities of each class label, where the neuron with the highest probability indicates the final predicted class label for the examined sample. Depending on the classification problem that the neural network is trained to solve, the predicted class label can provide information on either the presence of a cyber-attack, where the cyber-attack occurs, or the type of cyber-attack.

To obtain an optimal set of weights and biases in Eq. (13), the solver *Adam* is used to minimize a binary cross-entropy loss function, which has the following form:

$$S(w) = \left(N_s \cdot \sum_{q=1}^{N_s} y_{actual,q} \cdot \ln(y_{pred,q})\right)^{-1} \tag{14}$$

where $q = 1, \ldots, N_s$ represents the number of samples in the training dataset, y_{actual} is the vector of target class labels of each sample, and y_{pred} is the vector of the predicted probabilities associated with each class label.

To build an FNN detector model, training data samples are collected, which consist of closed-loop time-varying trajectories of the specific input vector variable suited for the detector model, generated from simulations under different attack scenarios. Sensor cyber-attacks with varying duration L_a are introduced at random times t_{i_0} on specific sensor(s) during the simulation period $N_T \times \Delta$, which is the same length as the detection window. We consider a system where some sensors are attacked and some remain intact. If no attacks occur anywhere in the system within $N_T \times \Delta$, then the measurement signals are labeled and should be classified as "no attack". In order to improve the detection accuracy, various closed-loop state evolutions within the stability region Ω_ρ need to be accounted for; therefore, training data is collected for a broad range of initial conditions within the stability region. Full state measurements $\bar{x}(t)$ are recorded along the time-varying trajectory for $t \in [t_0, t_{N_T}]$, and the Lyapunov functions of both subsystems are computed. Each training sample reflects a different set of initial conditions, and within each class labels, equal number of samples are collected to ensure training accu-

racy. Training and testing accuracies are the ratios between the number of correctly classified samples and total number of samples in the training and testing sets, respectively.

5.1. *Online detection*

When the FNN detector is implemented online with the process controlled by the centralized, the decentralized, or the distributed LMPC system, it uses a moving horizon detection window of a fixed length N_T, which is the same length as the time-varying trajectories of the training data. The detector is activated every time full-state measurements become available; it receives the latest sequences of the process state measurements of all subsystems with a fixed length N_T, and computes the Lyapunov function V_j of each subsystem j respectively. The values of V_j along the time-varying trajectory of length N_T will then be concatenated into a single one-dimensional vector to be used as the input vector for the FNN detector. If the FNN detector is trained to differentiate between "not attacked" and "attacked", then at every sampling period, it makes a decision on whether the sensor measurements over the latest time period of $\Delta \times N_T$ have been tampered. In addition, if the FNN detector is trained with multiple classes where each class represents one particular subsystem being attacked, then by examining the concatenated input vector containing all V_j, $j = 1, \ldots, N_{sys}$ along the detection window N_T, the detector will determine the particular subsystem where the tampered sensors are located. When the occurrence of a sensor cyber-attack has been confirmed, the sensor measurements can no longer be trusted, and all sensors will be switched for secure back-up sensors to completely mitigate the attack.

Remark 2. The availability of back-up sensors is always necessary in case of instrument failure, or in this case, sensor cyber-attacks. These redundant sensors are not connected to the online system until a necessary scenario arises, and therefore will remain secure to any cyber-attacks that target control system in real time. Here, we propose one strategy for mitigating the impact of the cyber-attack by physically switching out the problematic sensors; other mitigation measures have also been proposed in our previous work in Wu et al. (2020) where we use a recurrent neural network model to reconstruct tampered state measurements and restore system stability by using these state observers.

6. Application to a two-CSTR-in-series process

In this study, we simulate a chemical process consisting of two well-mixed, non-isothermal continuous stirred tank reactors (CSTRs) that operate in series. An irreversible second-order exothermic reaction takes place in each reactor, where the feed reactant A is transformed into product B. Reactant material A is fed into each of the two reactors $j = 1, 2$, with inlet concentrations C_{Aj0}, inlet temperatures T_{j0} and the reactor feed volumetric flow rates F_{j0}. On each CSTR, a heating jacket is installed to supply and remove heat at a rate Q_j, $j = 1, 2$. Considering the material and energy balances of the overall process,

Table 1 – Parameter values of the two CSTRs in series.

$T_{10} = 300\,K$	$T_{20} = 300\,K$
$F_{10} = 5\,m^3/h$	$F_{20} = 5\,m^3/h$
$V_{L_1} = 1\,m^3$	$V_{L_2} = 1\,m^3$
$T_{1_s} = 401.9\,K$	$T_{2_s} = 401.9\,K$
$C_{A1_s} = 1.954\,kmol/m^3$	$C_{A2_s} = 1.954\,kmol/m^3$
$C_{A10_s} = 4\,kmol/m^3$	$C_{A20_s} = 4\,kmol/m^3$
$Q_{1_s} = 0.0\,kJ/h$	$Q_{2_s} = 0.0\,kJ/h$
$k_0 = 8.46 \times 10^6\,m^3/kmol\,h$	$\Delta H = -1.15 \times 10^4\,kJ/kmol$
$C_p = 0.231\,kJ/kg\,K$	$R = 8.314\,kJ/kmol\,K$
$\rho_L = 1000\,kg/m^3$	$E = 5 \times 10^4\,kJ/kmol$

the dynamic models of this two-CSTR-in-series process can be represented as follows:

$$\frac{dC_{A1}}{dt} = \frac{F_{10}}{V_{L_1}}(C_{A10} - C_{A1}) - k_0 e^{\frac{-E}{RT_1}} C_{A1}^2 \tag{15a}$$

$$\frac{dT_1}{dt} = \frac{F_{10}}{V_{L_1}}(T_{10} - T_1) + \frac{-\Delta H}{\rho_L C_p} k_0 e^{\frac{-E}{RT_1}} C_{A1}^2 + \frac{Q_1}{\rho_L C_p V_{L_1}} \tag{15b}$$

$$\frac{dC_{A2}}{dt} = \frac{F_{20}}{V_{L_2}} C_{A20} + \frac{F_{10}}{V_{L_2}} C_{A1} - \frac{F_{10} + F_{20}}{V_{L_2}} C_{A2} - k_0 e^{\frac{-E}{RT_2}} C_{A2}^2 \tag{15c}$$

$$\frac{dT_2}{dt} = \frac{F_{20}}{V_{L_2}} T_{20} + \frac{F_{10}}{V_{L_2}} T_1 - \frac{F_{10} + F_{20}}{V_{L_2}} T_2 + \frac{-\Delta H}{\rho_L C_p} k_0 e^{\frac{-E}{RT_2}} C_{A2}^2 + \frac{Q_2}{\rho_L C_p V_{L_2}} \tag{15d}$$

where C_{Aj}, V_{L_j}, T_j and Q_j, $j = 1, 2$ are the concentration of reactant A, the volume of the reacting liquid, the temperature, and the heat input rate in the first and the second reactor, respectively. The reacting liquid has a constant density of ρ_L and a constant heat capacity of C_p for both reactors. E, R, ΔH, and k_0 represent the activation energy, ideal gas constant, enthalpy of the reaction, and pre-exponential constant, respectively. Table 1 lists all parameter values of this process.

For both CSTRs, the manipulated inputs are the inlet concentration of species A and the heat input rate supplied by the heating jacket, which are represented by the deviation variables $\Delta C_{Aj0} = C_{Aj0} - C_{Aj0_s}$, $\Delta Q_j = Q_j - Q_{j_s}$, $j = 1, 2$, respectively. The manipulated inputs are bounded as follows: $|\Delta C_{Aj0}| \leq 3.5\,kmol/m^3$ and $|\Delta Q_j| \leq 5 \times 10^5\,kJ/h$, $j = 1, 2$. The states of the closed-loop system are the concentration of species A and the temperature in the first and the second reactor, which are also represented by their deviation variables such that the equilibrium point of the system is at the origin of the state-space. Therefore, the vector of closed-loop states is $x = [C_{A1} - C_{A1_s} \quad T_1 - T_{1_s} \quad C_{A2} - C_{A2_s} \quad T_2 - T_{2_s}]^T$, where C_{A1_s}, C_{A2_s}, T_{1_s} and T_{2_s} are the steady-state values of concentration of A and temperature in each of the two tanks, respectively.

We analyze and compare three different control architectures in this example. In a centralized framework, the centralized controller receives feedback measurements of all states x, and computes the manipulated inputs for the entire system, $u = [\Delta C_{A10} \ \Delta Q_1 \ \Delta C_{A20} \ \Delta Q_2]^T$. The objective function in the centralized LMPC optimization problem is $L(x, u) = x^T Q x + u^T R u$, where $Q = diag[2 \times 10^3 \ 1 \ 2 \times 10^3 \ 1]$, $R = diag[8 \times 10^{-13} \ 0.001 \ 8 \times 10^{-13} \ 0.001]$. In a decentralized scheme, two LMPCs are designed for the two subsystems respectively, and they are solved in separate processors independently without any inter-controller communications. Due to the structure and the interactions between the two CSTRs, the dynamic behavior of the second CSTR is impacted by that of the first CSTR, but

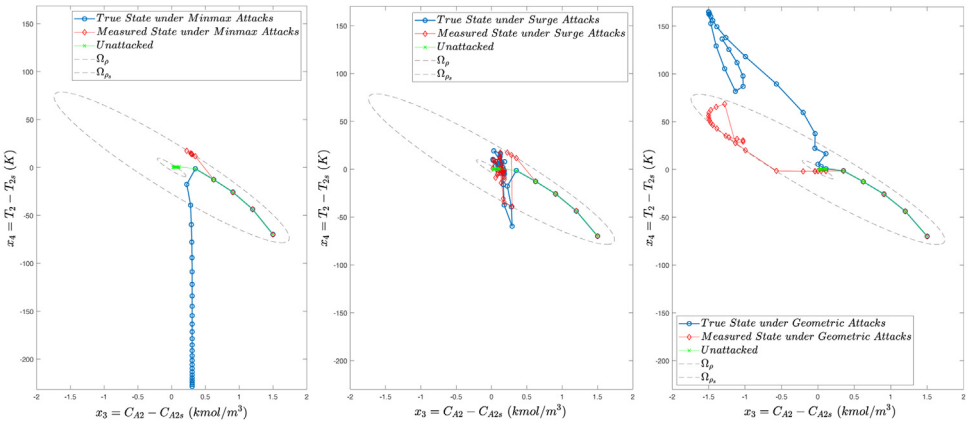

Fig. 2 – State-space closed-loop trajectories of the true, the measured, and the un-attacked process states of the second CSTR under the decentralized LMPC system when the temperature sensor T_2 is attacked by min–max, surge, and geometric attacks, respectively.

not vice versa. Thus, LMPC 1 receives only local state feedback measurements of the first CSTR as other state measurements in the system are not needed to compute the manipulated control actions for the first CSTR. To predict the process states of the second CSTR, LMPC 2 needs feedback information on all states, and therefore receives feedback measurements of both CSTRs to calculate the control actions for the second CSTR. The objective function in the decentralized LMPC j optimization problem has the form $L_j(x_j, u_j) = x_j^T Q_j x_j + u_j^T R_j u_j$, where $Q_j = diag[2 \times 10^3 \ \ 1]$, $R_j = diag[8 \times 10^{-13} \ \ 0.001]$, for $j = 1$, 2, and u_j, $j = 1$, 2 denote the manipulated input vectors of each subsystem j. In a distributed system, both LMPCs are designed to predict the state evolution of the entire process, and then calculate the respective control actions for each CSTR accordingly. Therefore, feedback measurements of both CSTRs are received by both LMPCs in the distributed system. The objective function in each distributed LMPC is $L(x, u_1, u_2) = x^T Q x + u_1^T R_1 u_1 + u_2^T R_2 u_2$, where $Q = diag[2 \times 10^3 \ \ 1 \ \ 2 \times 10^3 \ \ 1]$, $R_1 = R_2 = diag[8 \times 10^{-13} \ \ 0.001]$, and $u_1 = [\Delta C_{A10} \ \ \Delta Q_1]^T$, $u_2 = [\Delta C_{A20} \ \ \Delta Q_2]^T$. We assume that the feedback measurements of all states are available at the same synchronous intervals $\Delta = 10^{-2}$ h. The control Lyapunov function for each decentralized LMPC j is $V_j(x_j) = x_j^T P_j x_j$, for $j = 1$, 2, and the control Lyapunov function for the centralized and the distributed LMPCs is the sum of the control Lyapunov functions for the two CSTRs, i.e., $V(x) = V_1(x_1) + V_2(x_2) = x_1^T P_1 x_1 + x_2^T P_2 x_2$, with the following positive definite P_j matrices:

$$P_1 = P_2 = \begin{bmatrix} 1060 & 22 \\ 22 & 0.52 \end{bmatrix} \tag{16}$$

The closed-loop stability region Ω_{ρ_j} for each subsystem $j = 1$, 2, defined by the largest level sets of the control Lyapunov functions for subsystem 1 and 2 respectively, are $\rho_1 = \rho_2 = 392$. The ultimate bounded regions $\Omega_{\rho_{s_j}}$ are chosen to be $\rho_{s_j} = 7$, for $j = 1$, 2; the positive constants ρ_j and ρ_{s_j} are determined via extensive closed-loop simulations with $u_j \in U_j$.

To numerically simulate the dynamic model of Eq. (15), we use the explicit Euler method with an integration time step of $h_c = 10^{-4}$ h. The nonlinear optimization problems of the centralized, the decentralized, and the distributed LMPCs are solved with the same sampling period of $\Delta = 10^{-2}$ h using PyIpopt, which is a Python module of the IPOPT software package.

6.1. *Closed-loop performance without detection*

To illustrate the impact of the three types of sensor cyber-attacks considered for this system, we compare the closed-loop performance of the decentralized LMPC system when attacked and when not attacked. Fig. 2 shows in state-space plot the closed-loop evolution of the process states of the second CSTR (i.e., $C_{A2} - C_{A2s}$ and $T_2 - T_{2s}$) under the decentralized LMPC system when un-attacked and when one type of sensor attack has been injected during operation. The three types of attacks – min–max, geometric, and surge attacks – are respectively added on the temperature sensor of the second CSTR at time $t = 0.04$ h and lasted until the end of the simulation period of 0.3 h. The tampered sensor measurements, as well as the true process states are shown in the same plots. We can see that the falsified sensor measurements remain within the stability region boundaries, communicating to the controller that the solution to the optimization problem of the system is still feasible, causing the controller to calculate incorrect control actions with respect to the true process state. As a result, the true process states of the system are eventually driven outside of the stability region without triggering any alarms in the absence of an adequate detection mechanism.

While sensors of one subsystem may be compromised, its impact on other subsystems depends on the structure and the resulting actions of the control system. Figs. 3–5 show the true state profiles of the two-CSTR process of Eq. (15) operated under the centralized, the decentralized, and the distributed LMPC systems when under min–max, surge, and geometric sensor attacks, respectively. All types of attacks are added on the temperature sensor T_2 of the second CSTR starting at $t = 0.05$ h. The measured states from attacked sensors are not shown in these figures because depending on the location of the true process state, the attacked sensor measurement will adjust and show different patterns as well. Since the problematic sensor is in the second CSTR subsystem, the true states of the second CSTR (x_3 and x_4) under the centralized, decentralized, and distributed LMPC systems all exhibit destabilized behavior and are no longer driven to the operating steady-state as a result of the cyber-attack. The true states x_1 and x_2 of the first CSTR subsystem, however, may or may not be impacted by the sensor attacks in the second CSTR subsystem depending on the closed-loop control system. The decentralized LMPC system demonstrates robustness against sensor cyber-attacks on the temperature sensor of the second CSTR,

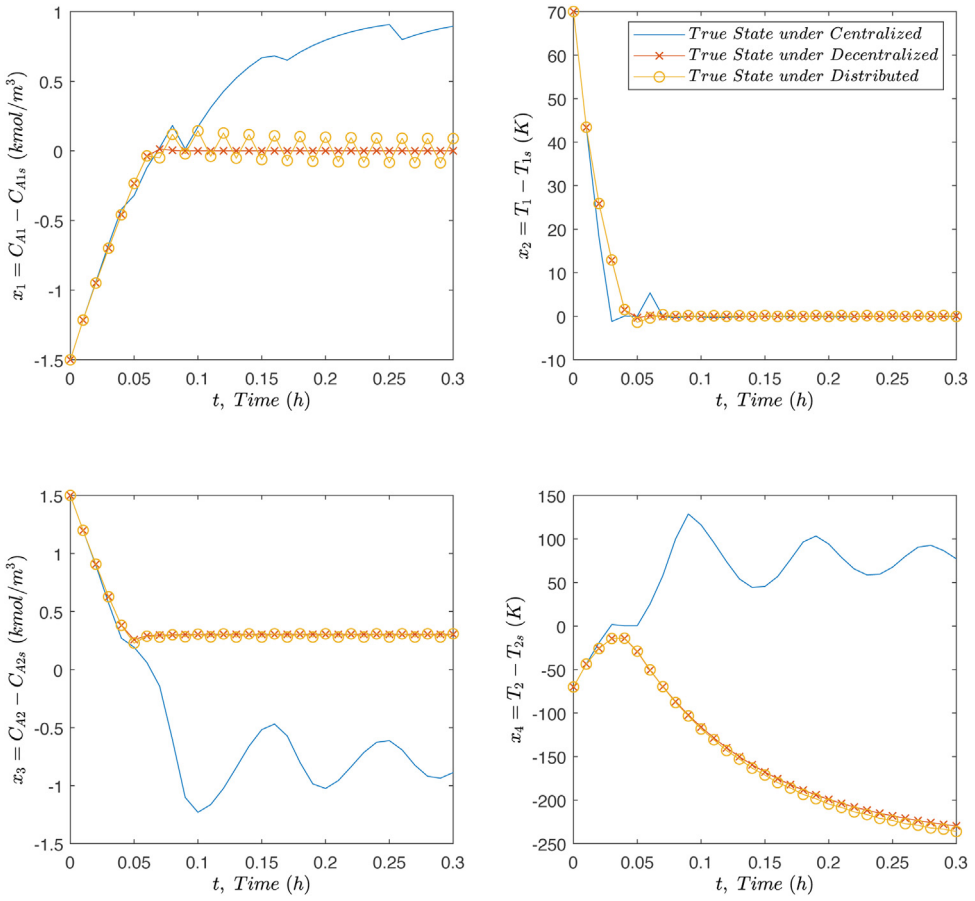

Fig. 3 – Closed-loop trajectories of true states the two-CSTR process operated under centralized, decentralized, and distributed LMPC systems when min–max attacks are added on the temperature sensor T_2 of the second CSTR at t = 0.05 h.

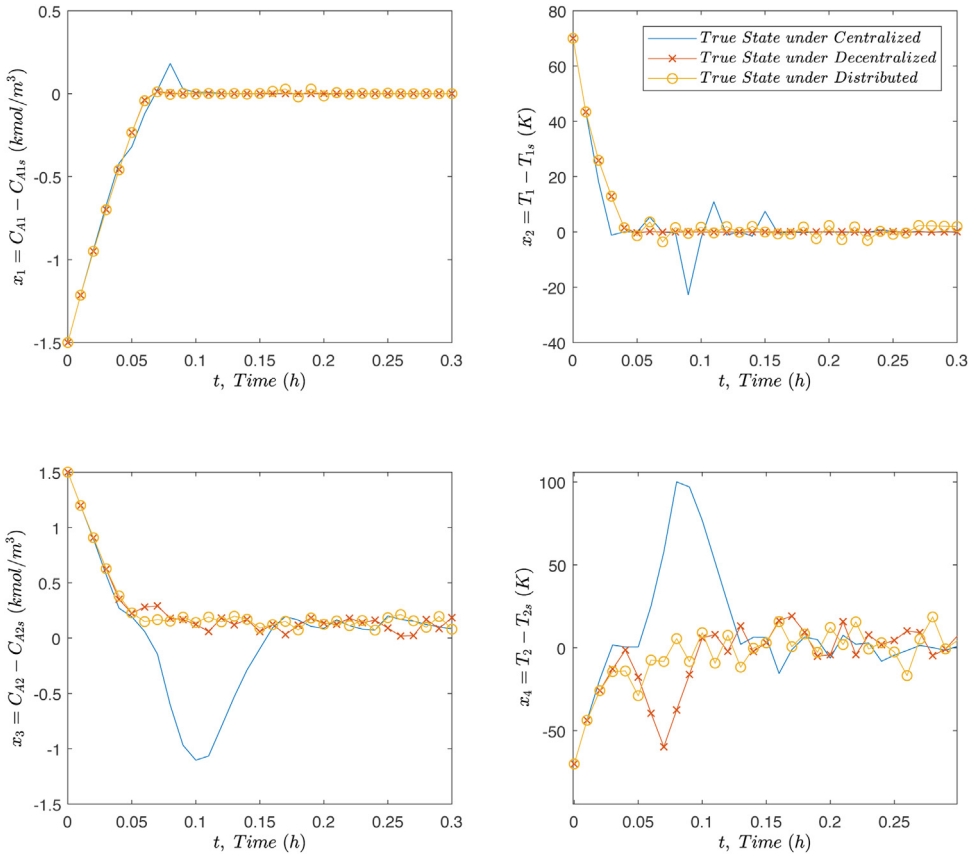

Fig. 4 – Closed-loop trajectories of true states the two-CSTR process operated under centralized, decentralized, and distributed LMPC systems when surge attacks are added on the temperature sensor T_2 of the second CSTR at t = 0.05 h.

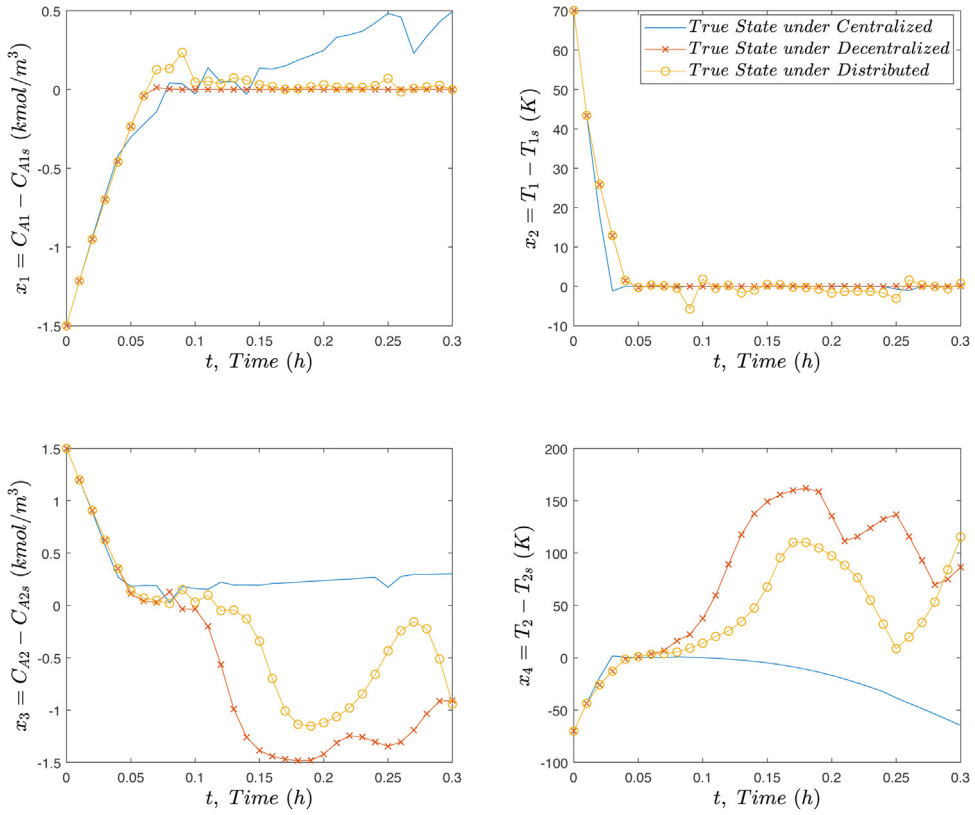

Fig. 5 – Closed-loop trajectories of true states the two-CSTR process operated under centralized, decentralized, and distributed LMPC systems when geometric attacks are added on the temperature sensor T_2 of the second CSTR at $t = 0.05$ h.

Table 2 – Sum of squared percentage error of the first and the second CSTR controlled by decentralized and distributed LMPC systems when under no attacks and when the temperature sensor T_2 is attacked by min–max, surge, geometric cyber-attacks.

	Decentralized SSE-1	Decentralized SSE-2	Distributed SSE-1	Distributed SSE-2
Not attacked	1.4560	1.3815	1.4578	1.3956
Min–max	1.4560	6.9806	1.5081	7.1882
Surge	1.4560	1.5631	1.4586	1.5540
Geometric	1.4560	11.0521	1.4878	4.7149

and ensures the stability of the intact subsystem (i.e., x_1 and x_2 which belong to the first CSTR), which has not experienced cyber-attacks. However, when the process is operated under the centralized and the distributed LMPC systems, varying degrees of oscillations and deviations are seen in the true state profiles of the first CSTR. Moreover, the true state profiles under the centralized LMPC demonstrate significantly worse performance for both CSTR subsystems when under min–max and surge attacks. Although the true state trajectories under the distributed LMPC system may seem overlapped with that under the decentralized LMPC system, the closed-loop performance under the distributed LMPC system is inferior than the decentralized LMPC system when subject to sensor attacks on a particular subsystem. To further illustrate the robustness of the decentralized LMPC system compared to the distributed LMPC system, the sum of squared percentage error of the two CSTRs in the form of $SSE_1 = \int_0^{t_{N_T}} ((\frac{C_{A1}-C_{A1s}}{C_{A1s}})^2 + (\frac{T_1-T_{1s}}{T_{1s}})^2)dt$ and $SSE_2 = \int_0^{t_{N_T}} ((\frac{C_{A2}-C_{A2s}}{C_{A2s}})^2 + (\frac{T_2-T_{2s}}{T_{2s}})^2)dt$ over the simulation period of $t_{N_T} = 0.3$ h are calculated for the decentralized and distributed LMPC systems when the temperature sensor T_2 is under the three attack types and when under no attacks. The SSE results are shown in Table 2. We can see that the SSE of the decentralized LMPC 1 stays constant regardless of the presence of an attack; when either min–max, surge, or geometric attacks are injected into the second CSTR, while the decentralized LMPC 2 does yield higher SSE values as a result of the sensor attack, the decentralized LMPC 1 yields SSE values identical to that of the "not attacked" scenario. However, the SSE values of the distributed LMPC 1 when under the three types of attacks are inconsistent and higher than that under no attacks.

The decentralized LMPC system outperforms the centralized and the distributed LMPC systems because the objective function used in each decentralized LMPC only depends on local state feedback measurements of its own subsystem, and is not impacted by state measurements of other subsystems. Therefore, if a particular subsystem is not attacked, the decentralized LMPC for that subsystem will ensure and sustain its stability. On the other hand, due to the centralized LMPC and the distributed LMPC system's requirements of using full-state measurements, the destabilizing impact of sensor cyber-attacks on one particular subsystem will propagate to other subsystems as well. In both centralized LMPC and distributed LMPC structures, the objective function in the LMPC optimization problem accounts for variations in the feedback measurements of all states in order to bring all states to the operating steady-state.

Table 3 – Detection accuracies of NN detectors in response to min–max, geometric, and surge attacks.

	Detector 1 (2-class) (Attacked vs. not attacked)	Detector 2 (3-class) (T_1 attacked vs. T_2 attacked vs. not attacked)
Min–max on T_1	99.93%	99.87%
Min–max on T_2	99.94%	99.88%
Surge on T_1	99.29%	98.57%
Surge on T_2	98.22%	100.00%
Geometric on T_1	96.88%	97.73%
Geometric on T_2	99.47%	98.93%
Not attacked	100.00%	100.00%

Therefore, if one sensor of the overall process is attacked and reflects false information, then the control actions calculated by the centralized LMPC or the distributed LMPC system will deviate from the supposedly optimal value, and cause destabilizing behavior in the true process states of all subsystems.

6.2. FNN detector modeling

Training data are collected under closed-loop operation for a simulation period of $t_{N_T} = 0.3$ h with $N_T = 30$ and $\Delta = 0.01$ h. Cyber-attacks are added at random times during the simulation period and last until the end of the period. The state measurements $\bar{x}_j, j = 1, 2$ of both CSTRs at every sampling period are recorded, then the Lyapunov function of each subsystem $V_j(\bar{x}_j), j = 1, 2$ is calculated along the time-varying trajectory of length $N_T = 30$ to be concatenated and used as the input vector for the FNN model. Such state measurements were collected for un-attacked scenarios as well as scenarios in which either the temperature sensor of the first CSTR T_1, or the temperature sensor of the second CSTR T_2, has been targeted by min–max attacks. Within each scenario, the training data samples are obtained by simulating the closed-loop process under the decentralized LMPC system starting from different set of initial conditions in the stability region. For training data collection, we only simulate min–max sensor attacks; after obtaining a model with adequate classification accuracy, we test this detector model against surge and geometric sensor attacks to examine the accuracy of the detector against unknown types of sensor cyber-attacks. Depending on

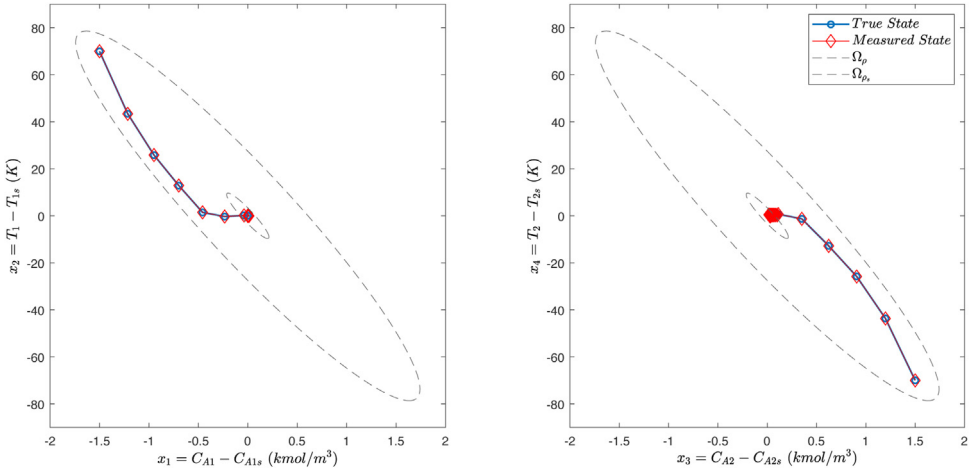

Fig. 6 – Closed-loop trajectories of true states the two-CSTR process operated under the decentralized LMPC system when min–max attacks are added on the temperature sensor T_2 of the second CSTR at $t = 0.30$ h, and detected by the 2-class FNN detector at $t = 0.30$ h.

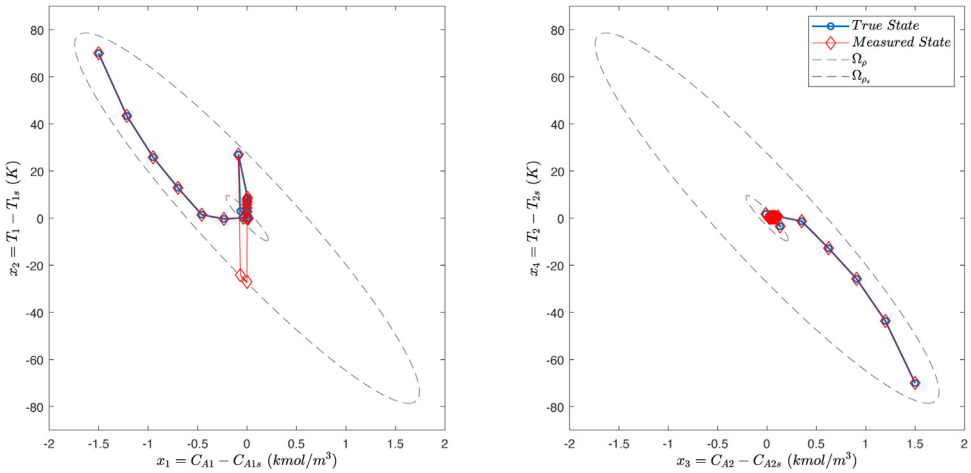

Fig. 7 – Closed-loop trajectories of true states the two-CSTR process operated under the decentralized LMPC system when surge attacks are added on the temperature sensor T_1 of the first CSTR at $t = 0.30$ h and detected by the 2-class FNN detector at $t = 0.32$ h, after which all sensors are switched to their secure back-up sensors and the true process states are driven back to the ultimate bounded region Ω_{ρ_s} around the operating steady-state.

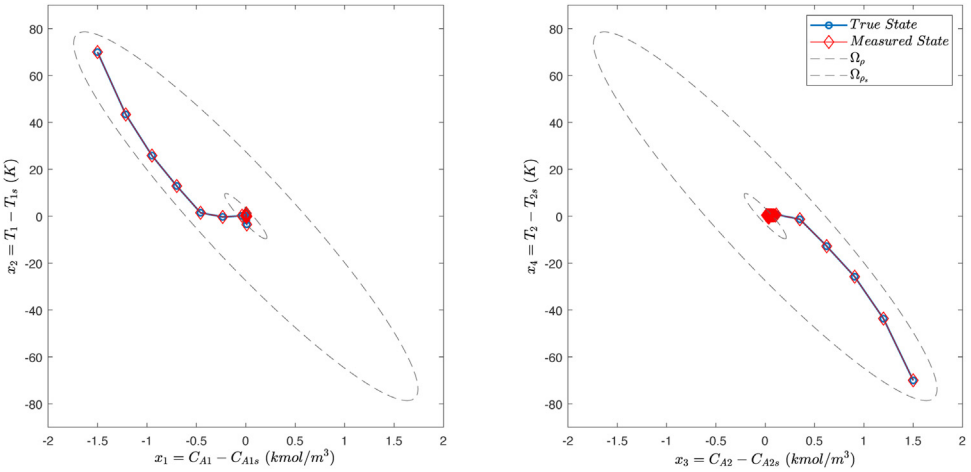

Fig. 8 – Closed-loop trajectories of true states the two-CSTR process operated under the decentralized LMPC system when geometric attacks are added on the temperature sensor T_1 of the first CSTR at $t = 0.30$ h and detected by the 2-class FNN detector at $t = 0.35$ h, after which all sensors are switched to their secure back-up sensors and the true process states are maintained within the ultimate bounded region Ω_{ρ_s} around the operating steady-state.

whether the detector is designed to identify the presence of a sensor cyber-attack where 2 classes are involved (attacked vs. not attacked), or to identify which subsystem the sensor cyber-attack has occurred where 3 classes are involved in this example (CSTR-1 attacked vs. CSTR-2 attacked vs. not attacked), the collected training data are labeled differently. Both the 2-class detector and the 3-class detector are FNN models consisting of one hidden layer with 10 neurons, which uses a *tanh* activation function in the form of $g_1(z) = \frac{2}{1+e^{-2z}} - 1$. Then, on the output layer, a *sigmoid* activation function in the form of $g_2(z) = \frac{1}{1+e^{-z}}$ is used to provide an output value between 0 and 1, which represents the predicted probability of the class labels. The number of layers and the number of neurons in each layer is determined via a grid search method. There are some standard nonlinear activation functions used in feedforward neural networks and available in the *Keras* library; in our simulation, we chose to use *tanh* on the hidden layer, and *sigmoid* on the output layer, as this combination produced the best prediction outcome and model accuracy. A total of 1500 samples are collected for each class label (i.e., attack scenario), where 70% is used for training and 30% is used for testing. The optimizer function used when minimizing the cost function of the network is *Adam* and 50 epochs are carried out during training for both the 2-class and the 3-class FNN detector models. The testing accuracies against the three different attack types (min–max, surge, geometric) on either the temperature sensor of the first CSTR (i.e., $\bar{x}_2$) or the temperature sensor of the second CSTR (i.e., $\bar{x}_4$) are shown in Table 3. Regardless of where the attack happens, the 2-class FNN detector is able to identify the presence of a cyber sensor attack accurately. Both detectors are able to achieve a testing accuracy of above 96% against all attack scenarios. Moreover, the classification accuracy for the "not attacked" scenario is 100% for both detectors, showing that the occurrence of false alarms is very low.

6.3. *Closed-loop operation with FNN detector*

After obtaining the FNN detector models with an adequate classification accuracy, we apply the detector online. At each sampling instant after the state measurements are collected from the sensors and before these measurements are communicated to the controllers, they are sent to the detector. A moving horizon detection window of size $N_T \times \Delta$ is implemented on the detector, where the detector examines the Lyapunov functions of the latest measured state trajectories of length N_T for both subsystems $j = 1, 2$ as the input vector for the FNN model (i.e., $V_j(\bar{x}_j(t_k))$ for $j = 1, 2$ and $k = 1, \ldots, N_T$). Therefore, as the new state measurements are received by the detector, the Lyapunov functions for these latest measurements are calculated and added to the input vector, and the detection window moves forward one sampling step Δ. If the prediction provided by the detector indicates that an attack has occurred within the detection window $N_T \times \Delta$, then all sensors in the process will be deemed un-trustworthy and should be switched to their back-up sensors. In the case that a 3-class detector is used and is able to identify which subsystem is experiencing sensor cyber-attacks, then a possible mitigation response is to only replace the sensors of the targeted subsystem. However, knowing that cyber-attacks has entered the system and leaving the overall network vulnerable, it is still good practice to examine all the sensors and act accordingly. We simulate closed-loop operation of the two-CSTR process under the more robust decentralized LMPC system integrated with the 2-class FNN detector. The total simulation period is 0.6 h with a sampling period of $\Delta = 0.01$ h, where either the temperature sensor for T_1 or T_2 starts experiencing one of the three types of cyber-attacks at $t = 0.3$ h. The measured and the true states of both CSTRs in state-space under different attack scenarios are shown in Figs. 6–8. After the detector indicates the occurrence of a cyber-attack, all sensors of the system are deemed unreliable and are switched to redundant back-up sensors to ensure the security of measurement data. The detector experiences a slight time delay when detecting the presence of certain sensor attacks. More specifically, a detection time delay of 2 sampling periods are observed when min–max and surge attacks are added on the sensor for T_1, and a detection time delay of 5 sampling periods for geometric attacks on T_1. When attacks target T_2, the detection time delays for min–max, surge, and geometric attacks are 0, 0, and 4 sampling periods, respectively. This means that as soon as the sensor for T_2 is attacked by min–max or surge attacks, and the attacked measurements are sent to the detector, the detector flags it immediately and the sensors are switched to their secure back-up sensors before the controller receives these

incorrect measurements. Despite the time delays in some cases, all true process states are maintained inside the stability region and eventually driven back to the terminal set around the operating steady-state within 6 sampling periods (i.e., 0.06 h).

7. Conclusions

As cyber-attacks pose critical risks to large-scale industrial plants, the effects of some standard types of sensor cyber-attacks on the operation of nonlinear processes under centralized, decentralized and distributed model predictive control systems were analyzed. Decentralized control systems have been demonstrated to be more robust than distributed and centralized control systems against sensor cyber-attacks. Using sensor data, a feed-forward neural network model was trained with adequate accuracy to detect the presence of cyber-attacks and was implemented online with a moving horizon detection window. Simulation studies were carried out on a nonlinear chemical process example to demonstrate the effectiveness of the combined architecture of the decentralized control system and the machine-learning-based detector in ensuring the closed-loop stability and the cyber-security of the overall closed-loop process.

Declaration of Competing Interest

The authors report no declarations of interest.

References

Bakule, L., 2008. Decentralized control: an overview. Annu. Rev. Control 32, 87–98.

Bemporad, A., Heemels, M., Johansson, M., 2010. Networked Control Systems, Vol. 406. Springer.

Cárdenas, A., Amin, S., Lin, Z., Huang, Y., Huang, C., Sastry, S., 2011. Attacks against process control systems: risk assessment, detection, and response. In: Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, Honkong, China, pp. 355–366.

Case, D.U., 2016. Analysis of the cyber attack on the Ukrainian power grid. Electricity Information Sharing and Analysis Center, pp. 388.

Chen, S., Wu, Z., Christofides, P.D., 2020a. Cyber-attack detection and resilient operation of nonlinear processes under economic model predictive control. Comput. Chem. Eng. 136, 106806.

Chen, S., Wu, Z., Christofides, P.D., 2020b. A cyber-secure control-detector architecture for nonlinear processes. AIChE J. 66, e16907.

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

Chen, S., Wu, Z., Rincon, D., Christofides, P.D., 2020d. Machine learning-based distributed model predictive control of nonlinear processes. AIChE J. 66, e17013.

Christofides, P.D., Scattolini, R., de la Pena, D.M., Liu, J., 2013. Distributed model predictive control: a tutorial review and future research directions. Comput. Chem. Eng. 51, 21–41.

Farnaaz, N., Jabbar, M.A., 2016. Random forest modeling for network intrusion detection system. Proc. Comput. Sci. 89, 213–217.

Farwell, J.P., Rohozinski, R., 2011. Stuxnet and the future of cyber war. Survival 53, 23–40.

Kim, J., Kim, J., Thu, H.L.T., Kim, H., 2016. Long short term memory recurrent neural network classifier for intrusion detection. In: Proceedings of the International Conference on Platform Technology and Service, Jeju, Korea, pp. 1–5.

Liu, J., Mu noz de la Pe na, D., Christofides, P.D., 2009. Distributed model predictive control of nonlinear process systems. AIChE J. 55, 1171–1184.

Liu, J., Wei, L., Xie, X., Yue, D., 2018. Distributed event-triggered state estimators design for sensor networked systems with deception attacks. IET Control Theory Appl. 13, 2783–2791.

Mohanty, S., Pradhan, A., Routray, A., 2007. A cumulative sum-based fault detector for power system relaying application. IEEE Trans. Power Deliv. 23, 79–86.

Omar, S., Ngadi, A., Jebur, H., 2013. Machine learning techniques for anomaly detection: an overview. Int. J. Comput. Appl. 79, 33–41.

Pang, Z., Liu, G., Zhou, D., Hou, F., Sun, D., 2016. Two-channel false data injection attacks against output tracking control of networked systems. IEEE Trans. Ind. Electron. 63, 3242–3251.

Shon, T., Moon, J., 2007. A hybrid machine learning approach to network anomaly detection. Inform. Sci. 177, 3799–3821.

Singh, J., Nene, M.J., 2013. A survey on machine learning techniques for intrusion detection systems. Int. J. Adv. Res. Comput. Commun. Eng. 2, 4349–4355.

Stouffer, K., Falco, J., Scarfone, K., 2011. Guide to Industrial Control Systems (ICS) Security. NIST Special Publication, pp. 800.

Teixeira, A., Amin, S., Sandberg, H., Johansson, K.H., Sastry, S.S., 2010. Cyber security analysis of state estimators in electric power systems. In: Proceedings of the 49th IEEE Conference on Decision and Control (CDC), Atlanta, Georgia, USA, pp. 5991–5998.

Teixeira, A., Pérez, D., Sandberg, H., Johansson, K., 2012. Attack models and scenarios for networked control systems. In: Proceedings of the 1st International Conference on High Confidence Networked Systems, Beijing, China, pp. 55–64.

Tu, J., 1996. Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. J. Clin. Epidemiol. 49, 1225–1231.

Vinayakumar, R., Soman, K.P., Poornachandran, P., 2017. Applying convolutional neural network for network intrusion detection. In: Proceedings of the International Conference on Advances in Computing, Communications and Informatics, Udupi, India, pp. 1222–1228.

Wu, Z., Albalawi, F., Zhang, J., Zhang, Z., Durand, H., Christofides, P.D., 2018. Detecting and handling cyber-attacks in model predictive control of chemical processes. Mathematics 6, 173.

Wu, Z., Chen, S., Rincon, D., Christofides, P.D., 2020. Post cyber-attack state reconstruction for nonlinear processes using machine learning. Chem. Eng. Res. Des. 159, 248–261.

Zhang, H., Han, Z., Li, C., 2003. Support vector machine based nonlinear model predictive control. Syst. Eng. Electron. 25, 330–334.