Title
Enhancing the Performance and Security of Networked Control Systems Using Identification-Integrated Model-Based Control

Author
Zedan, Amr

Publication Date
2021

Peer reviewed|Thesis/dissertation

Enhancing the Performance and Security of Networked Control Systems Using Identification-Integrated Model-Based Control

By

Amr Zedan

Dissertation

Submitted in partial satisfaction of the requirements for the degree of

Doctor of Philosophy

in

Chemical Engineering

in the

Office of Graduate Studies

of the

University of California

Davis

Approved:

Nael H. El-Farra, Chair

Ahmet N. Palazoğlu

A. Nazli Gündeş

Committee in Charge

2021

To my family and friends.

Contents

List of Figures

List of Tables

Abstract

Enhancing the Performance and Security of Networked Control Systems Using Identification-Integrated Model-Based Control

Modern industrial plants have become increasingly dependent on networked control system architectures in which dedicated sensor-controller, controller-actuator, and controller-controller links are replaced by real-time shared (wired or wireless) digital communication networks that operate over specialized industrial networks and protocols. While networked control systems offer a multitude of economic and operational benefits, the increased reliance on shared communication networks comes with a host of fundamental challenges that need to be addressed. For example, challenges such as network resource constraints, data losses, communication delays and real-time scheduling constraints are tied to the inherent limitations on the transmission and processing capabilities of the communication medium, and if left unaddressed can cause operational instabilities or closed-loop performance deterioration. These challenges have motivated a significant and growing body of research work on the analysis and design of networked control systems. An approach that has been proposed to address a number of these challenges is the use of model-based control, however, a central problem that has yet to be addressed is the robustness of the control system to plant-model mismatches that could arise due to things like fouling in heat-exchanger systems or deactivation of catalysts in catalytic reactors.

Motivated by these considerations, this dissertation aims to develop an identification-integrated model-based control framework that enhances the performance and security of networked control systems, and address framework implementation issues such as communication resource constraints, lack of full state measurements, distributed systems, communication strategies, nonlinearities, and measurement errors. Finally, the implementation and effectiveness of the developed framework through simulated chemical process examples.

Acknowledgments

I would like to express my utmost gratitude to my advisor, Professor Nael H. El-Farra for his continuous guidance, support and encouragement. Throughout my doctoral program, he has served as an excellent mentor from whom I learnt so much about scientific research, critical thinking, professional rigor and scientific curiosity. He sets a great role model as an excellent researcher, instructor and supervisor.

I am grateful for Professor Ahmet N. Palazoğlu and Professor A. Nazli Gündeş for dedicating their time and effort to serve as members of my qualifying exam and dissertation committees. I would like to also thank and acknowledge Professors Robert Powell, Adam Moulé and Greg Miller for their constant support, mentoring and guidance throughout my program.

I would like to thank everyone in my research group: Da, James, Eric and Gustavo, it was my pleasure and honor to work alongside such brilliant minds, thank you for all your support.

Finally, I would like to thank my parents, Mohamed and Madiha, for their immeasurable sacrifices and unwavering love and support, I am eternally grateful to you. To all my siblings, Assmaa, Abdulrahman, Rehab, Salma, Youssuf, and Ahmad, you have filled my life with happiness, thank you for all of your love and support.

Chapter 1

Introduction

Process systems engineering, or controls engineering, is a discipline concerned with the planning, designing, operating, and controlling of any kind of unit operation [1]. Traditionally, the control of such operations takes place through dedicated feedback links as depicted in Figure 1.1. Through these dedicated links, the controller receives data from the sensor at the output of the process, computes a control action, then sends an appropriate signal to the actuator at the inlet of the process to try and achieve the desired setpoint. While this traditional control architecture has proven to be successful, it has a number of limitations. To understand these limitations, we must first look at a plant-wide application of this control architecture. Figure 1.2 shows an example of a plant that utilizes a traditional feedback control architecture. In this plant there are three processes, a plant supervisory control unit from which operators make control decisions and establish setpoints for each controller in order to meet plant production specifications, and a plant monitoring unit in which plant parameters are monitored for safety and performance tracking. In the traditional control architecture, dedicated links from the plant supervisory control unit to each controller, as well as from each sensor to the plant monitoring unit, have to be established. While this example only shows three unit operations, in real applications there can be hundreds of units that are spaced hundreds of meters apart and installing cables in such a set-up can be very costly. Additionally, due to its wired nature, this architecture is not very modular; that is, it is not flexible to the addition of extra units. Moreover, plants that utilize this traditional control ar-

chitecture lack process level integration; the local sensors and controllers in each unit do not have access to data from any other units, which limits the ability for these units to respond quickly to disturbances and faults. Finally, this architecture does not lend itself to real-time data monitoring, a paradigm in which plant parameters can be accessed in real-time by plant operators as well as management to enable the possibility for tight integration between plant production and market demand when it comes to high level business decisions [2].

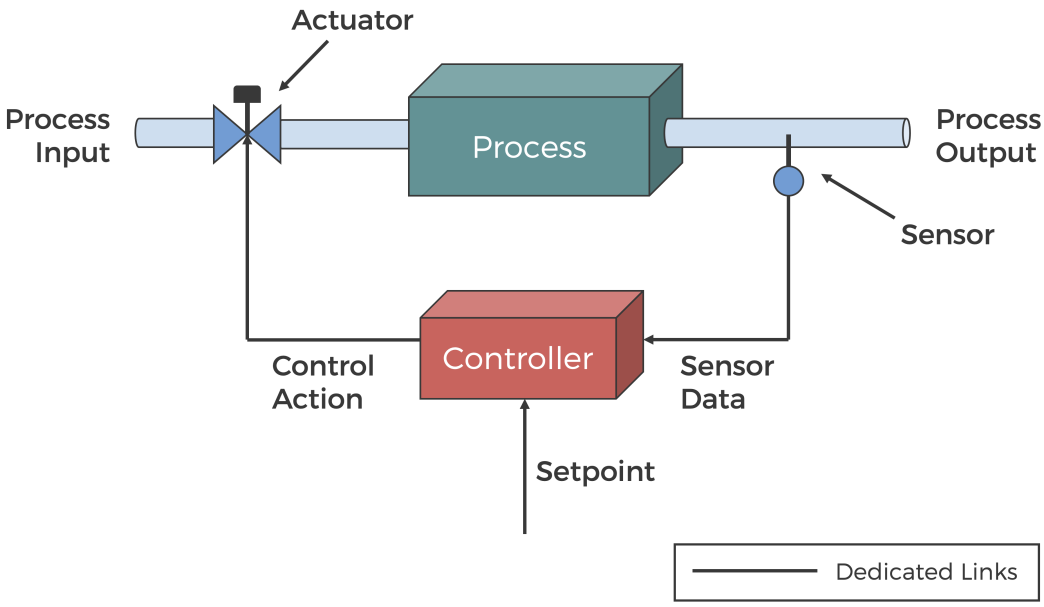

Figure 1.1. Traditional feedback control architecture for a single unit

To address these limitations, modern industrial plants have become increasingly dependant on networked control system architectures in which dedicated sensor-controller, controller-actuator, and controller-controller links are replaced by real-time shared (wired or wireless) digital communication networks that operate over specialized industrial networks and protocols such as Ethernet, HART, WirelessHART, ISA100, Profibus, Modbus and DeviceNet. On the unit operation level, the controllers, actuators, and sensors all exchange information through a local shared network as shown in Figure 1.3. Through this local shared network, processes can exchange information more readily, thus, enabling tight process integration. Additionally, a plant-wide shared communication network, across which information can be passed between the local networks and the plant

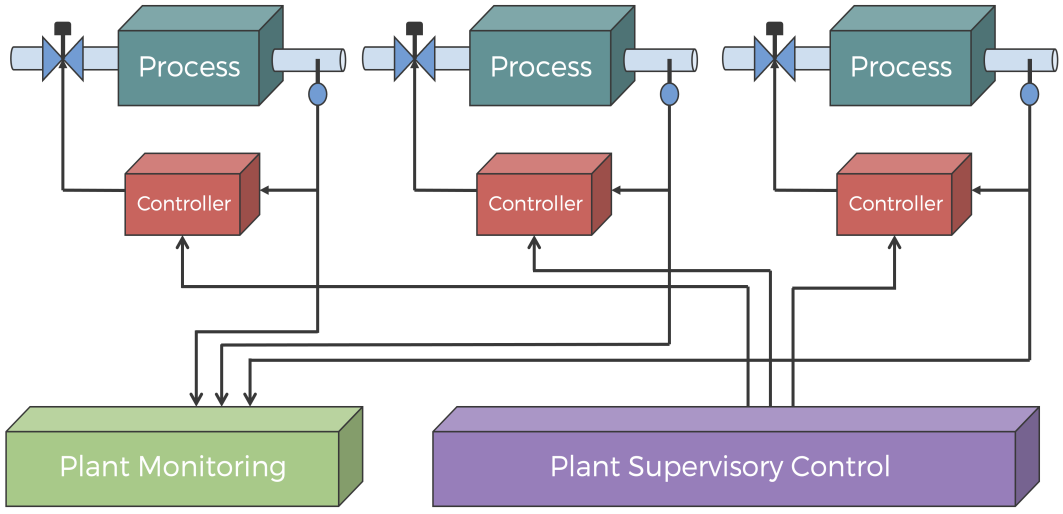

Figure 1.2. Example of a plant that utilizes a traditional feedback control architecture

supervisory control and monitoring unit, is established. An example of this architecture can be seen in Figure 1.4. Leveraging these shared digital communication protocols and networks can significantly improve the operational flexibility and fault-tolerance capabilities of an industrial control system, as well as reduce the installation, reconfiguration, and maintenance times and costs by reducing the amount of field wiring required for these control systems [3].

While networked control systems offer a multitude of economic and operational benefits, the increased reliance on shared communication networks comes with a host of fundamental challenges that need to be addressed. For example, challenges such as network resource constraints, data losses, communication delays and real-time scheduling constraints are tied to the inherent limitations on the transmission and processing capabilities of the communication medium, and if left unaddressed can cause operational instabilities or closed-loop performance deterioration. These challenges have motivated a significant and growing body of research work on the analysis and design of networked control systems (see, for example, [4–14], for some surveys of results and references in this area).

An approach that has been previously considered to address these challenges is the use of model-based control (see, for example, [15–22]). The idea of model-based control

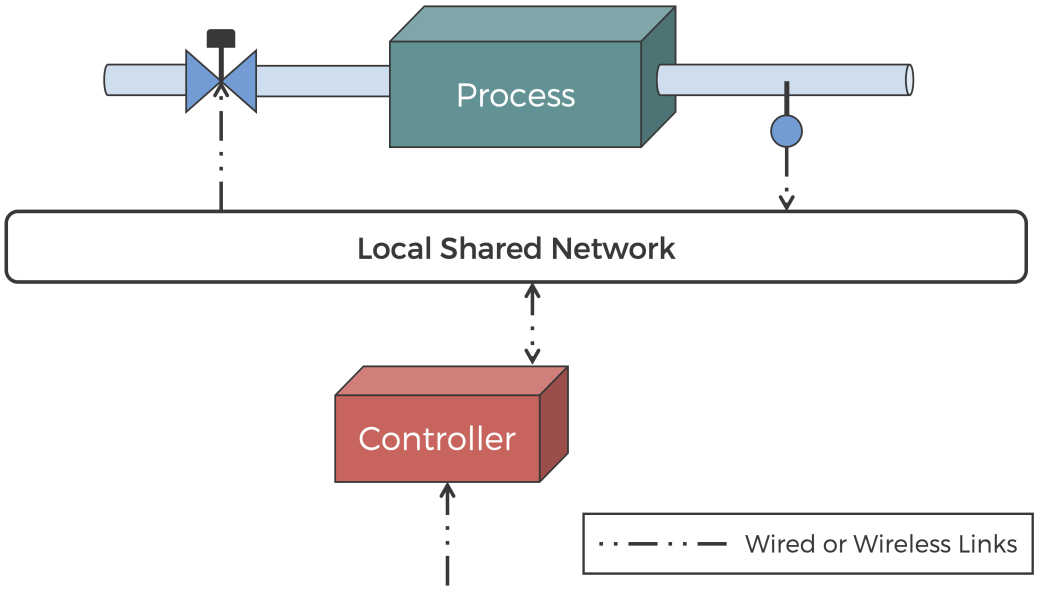

Figure 1.3. Networked feedback control architecture for a single unit

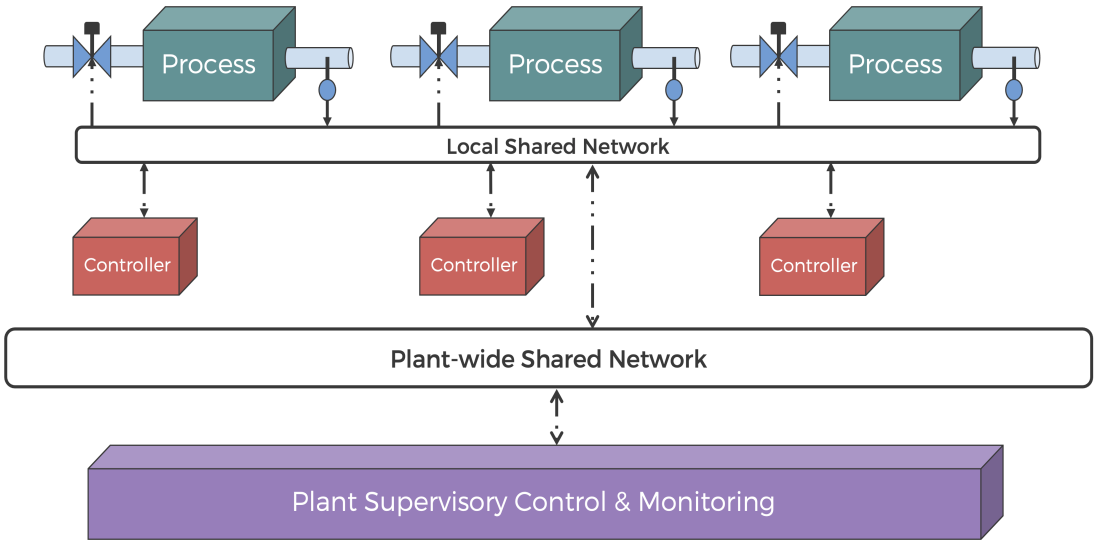

Figure 1.4. Example of a plant that utilizes a networked feedback control architecture

is to embed a predictive model within the process' local controller to generate estimates of the process states. By generating these estimates, the controller can compensate for lapses in communication by using these estimates to compute control actions, and when communication can be established, the available state measurements are used at discrete times. An additional advantage of utilizing model-based control is the ability to design a control architecture in which periodic or event-based communication can be established in order to maximize the achievable savings in network resource utilization. This is especially important in systems where network bandwidth is limited or processes that rely on batteries in their operation. Figure 1.5 depicts this strategy for a single unit operation.

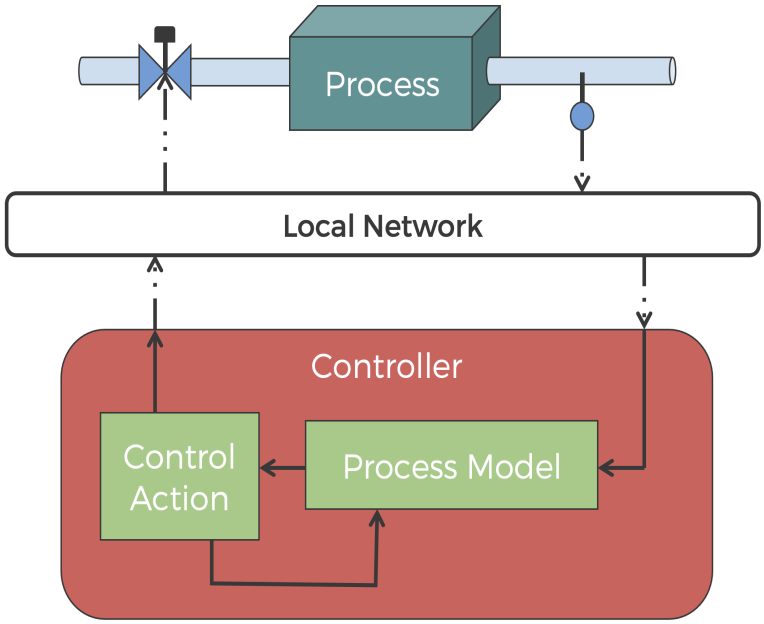

Figure 1.5. Model-based control concept

While a model-based control strategy generally helps reduce the control system's reliance on the communication medium, and therefore reduces its vulnerability to sensor failures and communication outages, its applications thus far have relied on the use of a single model with fixed parameters. The use of a fixed parameter model, however, is not robust. For example, consider the case when process parameters change over time, due to, for example, fouling in heat-exchanger systems or deactivation of catalysts in catalytic

reactors. Due to these variations, the model no longer captures the accurate state of the process, and plant-model mismatch occurs. If left unaddressed, this mismatch may limit the achievable savings in network resource utilization or cause performance degradation and even instability.

Another challenge that networked control systems face, especially in the context of wireless networks, is the issue of network cybersecurity. The deployment of sensor and actuator networks on a large-scale, together with the open nature of wireless networks, make networked control systems more vulnerable to cyberattacks. Cyberattacks take many forms, including deception and denial-of-service attacks (e.g., [23]), replay attacks (e.g., [24, 25]), covert attacks (e.g., [26]) and false data injection attacks (e.g., [27, 28]). These attacks generally aim to alter process inputs and cause them to deviate from their normal operating values. If left unchecked, these cybersecurity risks can potentially lead to high recovery costs, injury, death or physical damage (e.g., [29–34]), and are therefore a critical problem to address.

A number of approaches have been proposed in the literature for dealing with cybersecurity risks. Traditionally, these approaches are based on computer science, information technology, computer hardware, or networking solutions (e.g., see [28, 35–37]). More recently, several efforts within the process control community have been made to address the problem of cybersecurity from a control system perspective. Some of these efforts focused on the development of methods for the detection of cyberattacks. Examples of contributions in this direction include Bayesian detection techniques as binary hypothesis tests with prior probability (e.g., [38, 39]), weighted least square detection approaches (e.g., [40, 41]), χ^2-detector based on Kalman filters (e.g., [42, 43]), quasi-fault detection and isolation methods (e.g., [44]), Gaussian mixture clustering techniques to classify incoming measurements (e.g., [45]) and dynamic watermarking techniques wherein actuators inject private excitations into the system to reveal malicious signal tampering (e.g., [46]). Other studies have proposed techniques for mitigating the effects of cyberattacks. Examples of contributions in this direction include event-triggered control strategies under denial-of-service attacks (e.g., [47]), the design of attack-resilient state estimators (e.g., [48]) and

economic model predictive control mitigation strategies (e.g., [49, 50]).

An approach to mitigate the risk of cyberattacks in networked control systems is to minimize the control system's reliance on the communication network as much as possible. This idea has been pursued in earlier studies in the context of the design of resource-aware networked control systems (e.g., [15, 16, 21, 22, 51, 52]), and is realized through the design of a model-based networked control algorithm that enforces closed-loop stability with minimal sensor-controller communication. The control strategy involves suspending sensor-controller communication for some time during which the control action is computed based on the model predictions, and then updating the model states using the sensor measurements transmitted when sensor-controller communication is restored. In doing this, a minimum communication rate that guarantees closed-loop stability can be established. While this approach helps reduce the vulnerability of the control system to network-induced cyberattacks, it is a passive approach in the sense that it does not provide explicit robustness guarantees against cyberattacks, and does not include mechanisms by which cyberattacks can be actively detected or mitigated.

Motivated by these considerations, the overall objectives of this dissertation are to:

- Develop an identification-integrated model-based control framework that enhances the performance and security of networked control systems.
- Address framework implementation issues such as communication resource constraints, lack of full state measurements, distributed systems, communication strategies, nonlinearities, and measurement errors.
- Demonstrate the implementation and effectiveness of the developed framework through simulated chemical process examples.

To achieve these objectives, in Chapter 2 we introduce the idea of integrating model identification and model-based control in order to maintain closed-loop stability and reduce network resource utilization, while enhancing controller performance and accounting for plant drift and plant-model mismatch. Initially, a model-based controller with a well-characterized model state update rate is designed and implemented. An error detection

scheme with a time-varying alarm threshold is devised to track the state evolution and trigger model parameter updates. When the instability threshold is breached, the communication rate is temporarily increased to avoid instability and, in the meantime, the data collected during this time period are used to identify a new model online based on subspace identification techniques. The networked closed-loop stability region associated with the new model is characterized and used to identify a suitable model state update rate that can restore the communication frequency to its original level.

The approach in Chapter 2 is based on the assumption that all process states were available as measurements, which were used for both controller implementation and model parameter estimation. In many practical applications, measurements of the full-state are not available, and only a limited number of measured outputs are accessible. This problem typically arises due to technological constraints on the sensing techniques, which translate into restrictions on the ability to measure certain physical variables in real-time such as concentrations of reactive intermediates. The lack of full-state measurements imposes limitations on the implementation of full-state model-based feedback control and the data-based identification of model parameters, which need to be addressed. To advance the proposed framework, in Chapter 3 we present an approach for augmenting time-triggered model-based output feedback control with event-triggered online parameter re-identification for process systems subject to limited output measurements, process parameter variations and sensor-controller communication constraints.

At this point, we have only considered implementation strategies for systems modeled by ordinary differential equations (ODEs), however, many applications are modeled by highly-dissipative partial differential equations (PDEs). In Chapter 4 we present a framework for the integration of model-based control and model identification for spatially-distributed process systems modeled by highly-dissipative PDEs, subject to sensor-controller communication constraints and process parameter variations. The framework aims to enhance the stability and performance properties of the networked closed-loop system in the presence of process parameter variations and external disturbances, while simultaneously reducing the rate of sensor-controller information transfer required.

Initially, a networked feedback controller is designed on the basis of an approximate finite-dimensional model that captures the dominant dynamics of the infinite-dimensional system. The maximum allowable model state update rate is explicitly characterized in terms of the model parameters and the control actuator locations, and this characterization is used to devise a time-triggered model state update policy that guarantees closed-loop stability. An error monitoring scheme with a time-varying instability alarm threshold is then developed to track the state evolution and trigger model re-identification and model parameter updates in the event of process parametric drift. When the alarm threshold is breached, a safe-parking protocol is initiated by temporarily increasing the sensor-controller communication rate to counter the destabilizing influence of parametric drift. In the meantime, the input and state data collected during the safe-parking period are used to identify, on-line, a new finite-dimensional model based on subspace identification techniques. The networked closed-loop stability region associated with the newly-identified reduced-order model is then characterized and used to identify a suitable model state update rate that can restore the control performance and network utilization to their pre-drift levels.

While the approach in Chapter 4 is successful at improving the achievable level of communication savings by keeping the plant-model mismatch to a minimum, the implementation of this approach requires that a minimum fixed communication rate be established in order to achieve closed-loop stability, and model re-identification is triggered only in response to destabilizing parametric drifts. The use of a fixed communication rate strategy is not always optimal with respect to the achievable savings in network resource utilization. Event-triggered control strategies, on the other hand, can lead to more substantial reductions in network utilization and provide the process with the flexibility needed to adapt to changes in the operating environment. In Chapter 5 we present an event-based approach for the integration of model-based control and parameter identification in networked distributed processes subject to sensor-controller communication constraints and process parametric drift. The approach aims to improve the performance and communication constraint-handling capabilities of the networked closed-loop system,

and simultaneously address practical implementation issues arising from the uncertainty in process parameter values. Following this work, in Chapter 6 we address the limitations imposed by the lack of full-state measurements on the implementation of the integrated control and identification event-based approach developed in Chapter 5 in the context of event-based networked control of distributed processes. In Chapter 6 We focus on systems described by highly-dissipative PDEs subject to parametric drift and a limited number of measured outputs, and address the problem on the basis of a suitable reduced-order model that captures the slow dynamics of the infinite-dimensional system.

To address the implementation issues of nonlinear systems control, in Chapter 7 we present a framework for augmenting model-based feedback control with error-triggered parameter re-identification in spatially-distributed systems described by nonlinear parabolic PDEs subject to sensor-controller communication constraints and process parametric variations. The framework aims to maintain closed-loop stability in the presence of varying levels of plant-model mismatch during periods of parametric drift, while simultaneously keeping the rate of sensor-controller communication to a minimum and accounting explicitly for the presence of nonlinearities. In Chapter 8 we advance the algorithm further to address control under measurement errors by developing a model-based framework for the design of finite-dimensional sampled-data feedback controllers for spatially-distributed systems described by nonlinear parabolic PDEs subject to discretely-sampled measurements, bounded measurement errors, and bounded model uncertainty.

While measurement errors can occur naturally in sensor systems, they can also occur due to deliberate cyberattacks. To explore this, in Chapter 9 we present an integrated approach for the active detection, identification and mitigation of cyberattacks in a class of networked control systems subject to false data injection cyberattacks. The approach brings together tools from supervised machine learning, which are used for attack detection and identification, and model-based networked controller stabilization techniques, which are used for attack mitigation. Initially, a model-based networked control architecture in which the sensor and the controller communicate over a resource-limited communication medium is designed, and the networked closed-loop stability region is explicitly

characterized in terms of the measurement error resulting from the falsified measurement cyberattack, as well as the communication rate and the controller design parameters. This characterization is obtained by modeling the cyberattack in the closed-loop system formulation thus making it possible to identify the range of feasible operating conditions that guarantee robust stability under the attack. This characterization reveals the key parameters that can be used to actively mitigate the effects of these attacks when they arise. The implementation of these mitigation measures requires knowledge of the existence of the cyberattack as well as an estimate of its magnitude. To that end, we utilize machine learning methods (e.g., [53], [54], [55], [56]) to build a neural network (NN) based detection system. To detect both the existence and the magnitude of the cyberattack, the NN is trained using data obtained from simulating the system under normal operation and the system under different cyberattack magnitudes. While NN models are generally utilized as classification tools, training the NN with different cyberattack magnitudes allows for the classification of both the existence of a cyberattack and the approximate magnitude of this attack.

Finally, proofs of proposed theorems are provided in the appendix.

Chapter 2

Integrating Parameter Identification and Model-Based Control

In this chapter, we present a framework for integrating parameter identification and model-based control of networked control systems in order to maintain closed-loop stability and reduce network resource utilization, while enhancing controller performance and accounting for plant drift and plant-model mismatch. Initially, a model-based controller with a well-characterized model state update rate is designed and implemented. An error detection scheme with a time-varying alarm threshold is devised to track the state evolution and trigger model parameter updates. When the instability threshold is breached, the communication rate is temporarily increased to avoid instability and, in the meantime, the data collected during this time period are used to identify new model parameters online based on subspace identification techniques. The networked closed-loop stability region associated with the new model is characterized and used to identify a suitable model state update rate that can restore the communication frequency to its original level. Finally, the results are illustrated using a chemical process example.

The rest of the chapter is organized as follows: following some preliminaries in Section 2.1, the model-based controller is designed in Section 2.2 and a stability analysis is performed for the closed-loop system in Section 2.2.2. The integration of parameter identification and model-based control framework is then discussed in Section 2.3. Finally, the implementation of the proposed framework is illustrated in Section 2.4 using a chemical

process example. The results of this chapter were first published in [51].

2.1 Preliminaries

We consider a plant comprised of a network of interconnected subsystems with the following state-space representation:

$$\dot{x}_i = A_i x_i + B_i u_i + \sum_{j=1, j\neq i}^{n} A_{ij} x_j \tag{2.1}$$

where x_i is the state vector, u_i is the manipulated input vector, A_i and B_i are the state and input matrices, respectively, all of the i-th subsystem, A_{ij} is the interconnection matrix which captures the coupling between the i-th and j-th subsystems, and n is the total number of subsystems. The various subsystems have local control systems that exchange state measurements over a resource-constrained communication medium. The control objective is to stabilize the overall system at the desired steady-state, while simultaneously reducing unnecessary utilization of the communication medium. To simplify the presentation of the main results, we will focus on the state feedback control problem where the states of all the units are assumed to available as measurements.

2.2 Model-Based Control with Time-Triggered Communication

2.2.1 Local controller synthesis

To address the control objective, we consider a model-based control strategy similar to the one considered in [15]. The main idea is to embed within the local control system of each unit a set of dynamic models which are used to provide estimates of the states of the neighbouring units in the event when communication between the plant subsystems is suspended. When communication is re-established, the model states are updated using the plant states, at discrete time instances, based on the sensor measured values. The

resulting control and update laws are given by:

$$
\begin{aligned}
u_i(t) &= K_i x_i(t) + \sum_{j=1, j\neq i}^{n} K_{ji}\widehat{x}_j^i(t),\ i \in \{1, 2, \cdots, n\} \\
\dot{\widehat{x}}_j^i(t) &= \widehat{A}_j\widehat{x}_j^i(t) + \widehat{B}_j\widehat{u}_j(t) + \widehat{A}_{ji}x_i(t) + \sum_{l=1, l\neq i, l\neq j}^{n} K_{jl}\widehat{x}_l^i(t),\ t \in (t_k, t_{k+1}) \\
\widehat{u}_j^i(t) &= K_l\widehat{x}_j^i(t) + K_{ji}x_i(t) + \sum_{l=1, l\neq i, l\neq j}^{n} A_{jl}\widehat{x}_l^i(t),\ t \in (t_k, t_{k+1}) \\
\dot{\widehat{x}}_j^i(t_k) &= x_j(t_k),\ k \in \{0, 1, 2, \cdots\}
\end{aligned}
\tag{2.2}
$$

where K_i is the local feedback gain, K_{ij} is the controller gain that compensates for the interactions, $\widehat{x}_j^i$ is the state of the model capturing the dynamics of the j-th unit embedded in the i-th unit, $\widehat{A}_j$, $\widehat{A}_{ji}$, $\widehat{A}_{jl}$ and $\widehat{B}_j$ are constant matrices associated with the model of the j-th unit, and t_k is the update time. Note that, for simplicity, the states of all models are assumed to be updated at the same time.

2.2.2 Closed-loop stability analysis

An important measure of the extent of network utilization is the update period, h, which is defined as the difference between two successive update times, i.e., $h = t_{k+1} - t_k$. To achieve closed-loop stability with minimal communication, it is important to characterize the maximum allowable update period for the control system which dictates the minimum communication frequency required for stability. This characterization can be obtained through a closed-loop stability analysis. Specifically, it can be shown (see [15] for the details) that the response of the closed-loop system of (2.1) - (2.2) in terms of the update period is given by:

$$
\xi(t) = e^{\Lambda(t-t_k)}(I_s e^{\Lambda h} I_s)^k \xi_0 = e^{\Lambda(t-t_k)} M^k \xi_0 \tag{2.3}
$$

for $k \in \{0, 1, 2, \cdots\}$, where $\xi(t) := [s^T(t) \quad e^T(t)]^T$ is an augmented vector of the plant states and model estimation errors between the model and plant states, Λ is the augmented closed-loop matrix which depends on the plant and model matrices as well as the controller gains, I_s is an augmented identity matrix, and $\xi_0 = [x^T(t_0) \quad 0]^T$ is the initial condition. By analysing this response, it can be verified that a necessary and sufficient condition for closed-loop stability is to have all eigenvalues of the stability test matrix M lie strictly

inside the unit circle. This is represented by $\lambda_{max}(M)[A_i, A_{ij}, B_i, \widehat{A}_i, \widehat{A}_{ij}, \widehat{B}_i, K_i, K_{ij}, h] < 1$ where λ_{max} is the maximum eigenvalue magnitude of M. This condition can be used to explicitly characterize the maximum allowable update period in terms of the plant-model mismatch and controller gains.

2.3 Integrating Model Identification and Model-Based Control

In this section, an overview of the proposed methodology is presented first, and is then followed by a discussion of the key implementation issues.

2.3.1 Overview of the proposed methodology

Figure 2.1 summarizes the proposed approach for integrating model identification and model-based control. Initially, the control system is operated at a suitably chosen update period that minimizes communication while ensuring closed-loop stability. The evolution of the closed-loop state is continuously monitored and checked against an instability alarm threshold to determine when the model parameters need to be updated. In the event of a threshold breach, a "safe-parking" protocol is activated to momentarily stabilize the closed-loop system while additional data is collected and a new model is identified. The new model is then checked for stability, and the corresponding communication rate is determined. Finally, the model parameters are updated and the new communication rate is implemented. The algorithm is repeated every time the instability alarm threshold is breached.

2.3.2 Triggering criterion

To determine when a new model needs to be identified, a process monitoring scheme is implemented whereby the closed-loop states are continuously checked to detect potential instabilities caused by the drift in plant parameters. An instability alarm can be designed on the basis of the nominal closed-loop response in (2.3). Specifically, it can be verified that, in the absence of process parameter variations, the following time-varying bound is

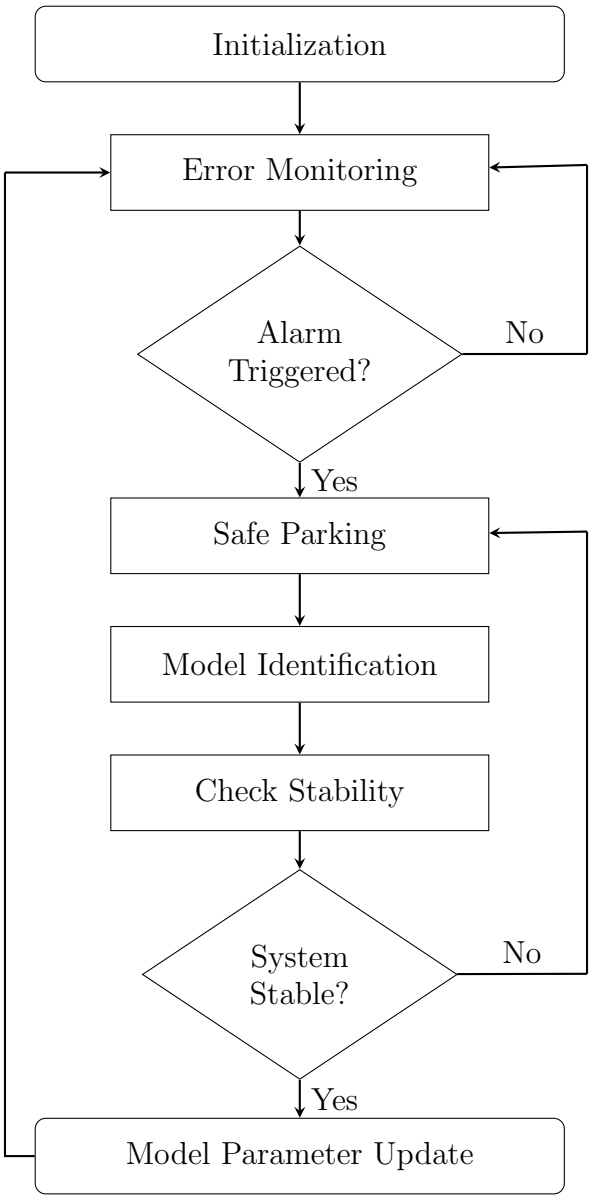

Figure 2.1. Flowchart of the algorithm for integrating model identification and model-based control

satisfied:

$$\|x(t)\| \leq \alpha e^{-\beta(t-t_0)}\|x(t_0)\| \tag{2.4}$$

where $\alpha > 1$ and $\beta > 0$ are tunable constants that depend on the choice of the controller design parameters. Using this bound as a time-varying alarm threshold, instability can be detected once the norm of the state violates the threshold. This acts as a trigger for subsequent steps in the process, culminating in the update of the model parameters. Note that, unlike the model state updates which are time-triggered periodically, the model parameter updates are event-triggered and thus occur less frequently.

2.3.3 Safe parking and model identification

Once the instability threshold is breached, the plant is switched temporarily to a safe-parking mode of operation where the model state update period is reduced for a certain period of time. The temporary increase in the communication rate serves to maintain closed-loop stability and to collect sufficient data for the model identification step. Model identification is then triggered and new model parameters are obtained. While the choice of the identification method depends on the particular application, subspace techniques are used in this study to identify the new model matrices based on the closed-loop input and state data to keep plant-model mismatch to a minimum. Note that in the case of time-varying uncertainty where the mismatch continues to deteriorate over time, the safe-parking step may need to be repeated to ensure closed-loop stability.

2.3.4 Stability check

Once a new model is identified, the closed-loop stability properties are analysed based on the eigenvalues of the new stability test matrix M to determine the range of stabilizing update periods associated with the new model. If either no stabilizing update period exists, or the maximum allowable update period is less than the safe-parking update period, the plant is kept in the safe-parking mode until a stabilizing model with a lower update frequency requirement can be found (possibly using a different identification method); otherwise, the model parameters are updated and the corresponding update period is implemented.

2.4 Application to a chemical process example

We consider a networked plant comprised of two interconnected non-isothermal continuous stirred tank reactors with a material recycle stream. The control objectives are to stabilize the system at the open-loop unstable steady-state using the heat input rates as the manipulated variables, to minimize the required communication between the two control systems and to account for process parameter variations.

To simulate plant parametric drift, we introduce uncertainty in the plant parameters. We define $\delta = (\Delta H^m - \Delta H)/\Delta H$, where ΔH^m is the nominal values of the heat of reaction used in the plant models. The operating conditions were chosen such that the model states are updated every 4.8 minutes and that there is initially no mismatch between the model and plant states; that is, $h = 0.080h$ and $\delta_1 = 0$. After 1 hour of closed-loop stable operation (see Figures 2.2 - 2.3), an uncertainty of $\delta_1 = -0.178$ is introduced in the plant causing a mismatch between the plant and model parameters. It can be seen in Figures 2.2 - 2.3 that, starting at approximately the 4^{th} hour of operation, the reactor temperature starts to diverge from the operating steady-state in an oscillatory fashion until the instability alarm threshold is breached and the safe-parking mode is triggered to regain stability.

The safe-parking update period is determined from the stability plot in Figure 2.4 (see dashed line) which shows that the update period can be reduced to $h = 0.073h$ to stabilize the system. It is noted that it is sufficient to increase the communication frequency to achieve stability; however, the objective here is to maintain stability with low communication frequencies, and model re-identification allows us to continue achieving this balance even after the process drifts. After data has been collected during the safe-parking period, the plant model is re-identified at time $t = 10h$. At that point, the stability region of the new model is characterized based on the new stability matrix M and the update period is increased to $h = 0.084h$ to switch the plant out of the safe-parking mode; and, as seen in Figure 2.2, maintain stability for the rest of the simulation.

The effect of uncertainty and model re-identification on closed-loop stability can be seen in Figure 2.4 which shows the stabilizing ranges for the update period before the

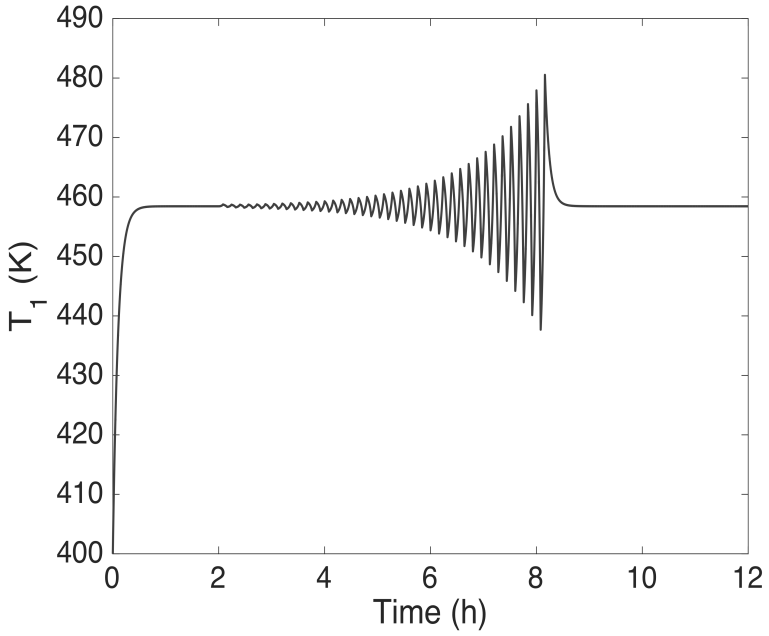

Figure 2.2. Closed-loop temperature profile for the first reactor

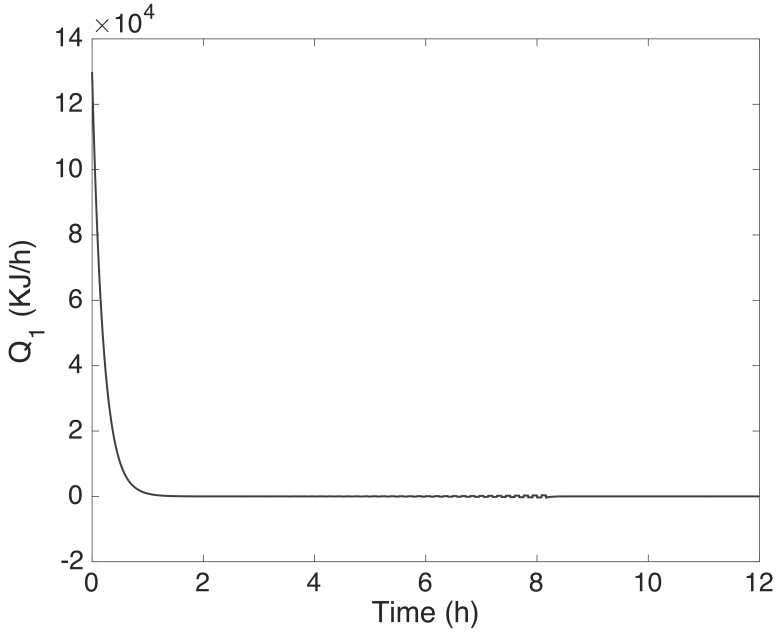

Figure 2.3. Heat input rate profile for the first reactor

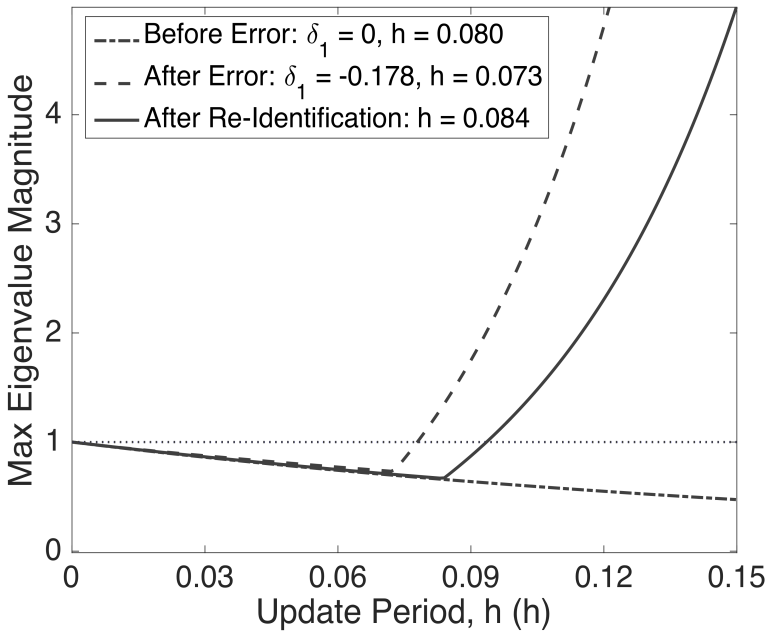

Figure 2.4. $\lambda_{max}(M)$ as a function of the update period under normal operation (dashed-dotted), under plant-model mismatch (dashed) and for the identified model (solid)

uncertainty was introduced, after the error was introduced and the safe-parking mode was invoked, and after the model re-identification took place. Eigenvalues below the critical unit line correspond to stable update periods. It can be seen that when the error is introduced, the stability region shrinks; however, after model re-identification takes place, the stability region increases, thus yielding a larger feasible range of update periods. This translates into stabilization with a lower communication frequency after model uncertainty is introduced.

2.5 Conclusions

A framework for the integration of time-triggered model-based control and event- triggered model identification for networked process systems subject to process parameter variations and communication constraints was presented in this chapter. It was shown that by minimizing the plant-model mismatch, model identification techniques can help enhance closed-loop stability with reduced communication requirements. The next chapter will focus on generalizing this approach to address the output feedback control problem and also explore the effect of the particular choice of the model-identification method on the

achievable savings in communication resource utilization.

Chapter 3

Output Feedback Model-Based Networked Control of Process Systems with Parameter Re-Identification

In Chapter 2, the developed framework was based on the assumption that all process states were available as measurements, which were used for both controller implementation and model parameter estimation. In many practical applications, measurements of the full-state are not available, and only a limited number of measured outputs are accessible. This problem typically arises due to technological constraints on the sensing techniques, which translate into restrictions on the ability to measure certain physical variables in real-time such as concentrations of reactive intermediates. The lack of full-state measurements imposes limitations on the implementation of full-state model-based feedback control and the data-based identification of model parameters, which need to be addressed.

In this chapter, we present an approach for augmenting time-triggered model-based output feedback control with event-triggered online parameter re-identification for process systems subject to limited output measurements, process parameter variations and sensor-controller communication constraints. The rest of the chapter is organized as follows: following some preliminaries and an overview of the control problem in Section 3.1, a model-based output feedback controller is designed and analyzed in Section 3.2. The

controller generates the control action using the predictions of a locally embedded model during periods of communication suspension, and the model state is periodically updated when communication is resumed. Owing to the lack of full-state measurements, a state observer is embedded within the sensor to generate state estimates using the measured outputs which are transmitted over the network to update the model states. An explicit characterization of the minimum allowable communication rate in terms of the various process, controller and observer design parameters is obtained.

The proposed methodology for augmenting model-based control with online parameter re-identification tools is then presented in Section 3.3, where an error monitoring scheme with a time-varying alarm threshold is developed on the basis of the nominal (drift-free) closed-loop stability properties. When the alarm threshold is breached during periods of parametric drift, a safe-parking mode of operation is triggered by adjusting the communication rate and/or the controller/observer design parameters to counteract the destabilizing tendencies of the increased plant-model mismatch. The process data collected during the safe-parking period are used to identify new model parameters, and the resulting networked closed-loop stability region is assessed to determine the appropriate post-drift communication rate and/or controller design parameters. At this point, the process exits the safe-parking regime and the model parameters are updated. Finally, the implementation of the developed methodology is illustrated in Section 3.4 using a chemical process example. The results of this chapter were first published in [57].

3.1 Preliminaries

In this work, we consider the class of dynamic linear processes described by the following state-space representation:

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{3.1}$$

$$y(t) = Cx(t) \tag{3.2}$$

where $x \in \mathbb{R}^n$ is the state vector, $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are the state and input matrices, respectively, $u \in \mathbb{R}^m$ is the vector of manipulated inputs, $y \in \mathbb{R}^q$ is the vector of measured outputs, and $C \in \mathbb{R}^{q\times n}$ is the output matrix.

We consider a networked control architecture in which the measurement sensors associated with the process of (3.1) communicate with the feedback controller at discrete times over a shared, resource-constrained communication medium. The communication medium is shared by other processes and units, and it is therefore desirable to optimize the rate at which the controlled process requires sensor feedback over the network. The control objective is thus to stabilize the closed-loop system at the desired steady-state while simultaneously reducing the required sensor-controller communication rate so as to reduce network resource utilization by the controlled process (which helps free up network resources for other units or processes sharing the network). This objective is to be achieved subject to process parametric variations and limited output measurements.

3.2 Model-Based Output Feedback Control

3.2.1 Controller synthesis

To achieve the control objective, a model-based control strategy wherein a model of the process is embedded within the controller is utilized. The purpose of the model is to provide the controller with estimates of the system states when communication is suspended, and the control actions are generated based on these estimates. When communication is re-established, the model states are updated over the network, at discrete instances. The embedded model takes the following form:

$$\dot{\widehat{x}}(t) = \widehat{A}\widehat{x}(t) + \widehat{B}u(t) \tag{3.3}$$

$$\widehat{y}(t) = \widehat{C}\widehat{x}(t) \tag{3.4}$$

where $\widehat{x} \in \mathbb{R}^n$ is the vector of model states, $\widehat{A} \in \mathbb{R}^{n\times n}$ and $\widehat{B} \in \mathbb{R}^{n\times m}$ are the model state and input matrices, respectively, $\widehat{y}$ is the vector of model outputs, and $\widehat{C}$ is the model output matrix. Note that, due to the plant-model mismatch, in general we have $A \neq \widehat{A}$, $B \neq \widehat{B}$, and $C \neq \widehat{C}$.

The model of (3.3) is used to design a stabilizing feedback controller of the form:

$$u(t) = K\widehat{x}(t) \tag{3.5}$$

where K is the feedback gain, which is chosen to place the poles of the closed-loop model

in the left-half of the complex plane. While the values of $\widehat{x}$ are continuously available to the controller, it is important to periodically update $\widehat{x}$ to reflect the state of the actual system dynamics and correct for any inaccuracies caused by the plant-model mismatch.

The implementation of this model state update is complicated when state measurements are unavailable. To address this issue, a state observer is embedded on the sensor side of the network to provide estimates of the process states using the available output measurements. To this end, we consider a dynamic observer of the following form:

$$\dot{\bar{x}}(t) = \widehat{A}\bar{x}(t) + \widehat{B}u(t) + L\widehat{C}(x(t) - \bar{x}(t)), \tag{3.6}$$

where $\bar{x} \in \mathbb{R}^n$ is the observer state vector and $L \in \mathbb{R}^{n\times q}$ is the observer gain. The observer states are transmitted over the network to update the model state at discrete times as follows:

$$\widehat{x}(t_k) = \bar{x}(t_k), \quad k \in \{0, 1, 2, \cdots\} \tag{3.7}$$

where t_k is the update time instance, and $h = t_{k+1} - t_k$ is the update period.

3.2.2 Analysis of networked closed-loop stability

To characterize the maximum allowable update period for the networked control system, we formulate an augmented closed-loop system that captures the collective dynamics of the process, the model and the observer. To this end, we define the augmented state vector $\xi(t) := [x^T(t) \;\; \bar{x}^T(t) \;\; e^T(t)]^T \in \mathbb{R}^{3n}$, where $e(t) = \bar{x}(t) - \widehat{x}(t) \in \mathbb{R}^n$ is the estimation error, which is defined as the discrepancy between the model and observer states. The dynamics of the augmented closed-loop system are given by:

$$\dot{\xi}(t) = \Lambda\xi(t), \quad t \in [t_k, t_{k+1}) \tag{3.8}$$

where $\Lambda \in \mathbb{R}^{3n\times 3n}$ is the augmented closed-loop matrix given by:

$$\Lambda = \begin{bmatrix} A & BK & -BK \\ LC & \widehat{A} - L\widehat{C} + \widehat{B}K + L\widetilde{D}K & -BK + L\widetilde{D}K \\ LC & -L\widehat{C} + L\widetilde{D}K & \widehat{A} - L\widetilde{D}K \end{bmatrix} \tag{3.9}$$

$$\widetilde{D} = D - \widehat{D} \tag{3.10}$$

It can be shown (e.g., see [15], [17]) that the response of the closed-loop system in (3.8)-(3.10) in terms of the update period is given by:

$$\xi(t) = e^{\Lambda(t-t_k)} M^k \xi_0,\ t \in [t_k, t_{k+1}),\ k \in \{0, 1, 2, \cdots\} \tag{3.11}$$

where

$$M = I_o e^{\Lambda h} I_o \tag{3.12}$$

is a special matrix whose eigenvalues determine closed-loop stability,

$$I_o = \begin{bmatrix} I & O & O \\ O & I & O \\ O & O & O \end{bmatrix} \tag{3.13}$$

is a matrix that captures the model state update logic under output feedback (the fact that the model state is updated using the observer state, rather than the actual state), $I \in \mathbb{R}^{n \times n}$ is the identity matrix, and $\xi_0 = [x^T(t_0)\ \ \bar{x}^T(t_0)\ \ 0]^T$ is the initial condition.

Based on the response of the closed-loop system in (3.11) it can be verified that a necessary and sufficient condition for closed-loop stability is to have all the eigenvalues of the stability test matrix M lie strictly inside the unit circle. This is represented by the following inequality:

$$\lambda_{max}(M)[A, B, \widehat{A}, \widehat{B}, K, L, h] < 1 \tag{3.14}$$

where $\lambda_{max}(M)$ is the maximum eigenvalue magnitude of M. This condition can be leveraged to explicitly characterize the maximum allowable update period as a function of the plant-model mismatch, the update period, the controller gain and the observer gain.

3.3 Augmenting model-based control with parameter re-identification

3.3.1 Overview of the proposed approach

Figure 3.1 summarizes the proposed approach for augmenting the model-based output feedback control strategy presented in Section 3.2 with model re-identification tools with

the goal of enhancing the stability and performance properties of the networked closed-loop system. Initially, the networked control system is operated at a stabilizing update period that simultaneously reduces sensor-controller communication. The evolution of the model estimation error is continuously monitored and checked against an instability alarm threshold to determine when the model parameters need to be updated. Upon breach of the threshold, a safe-parking protocol is triggered to momentarily stabilize the closed-loop system while additional data is collected and new model parameters are re-identified. The identified parameters are then examined for stability and achievable communication savings, and the corresponding communication rate and controller/observer design parameters are determined. Finally, the model parameters are updated and the new communication rate and controller parameters are implemented. The algorithm is repeated every time the instability alarm threshold is breached.

3.3.2 Error monitoring and instability alarms

To determine when new model parameters need to be identified, a monitoring scheme is implemented whereby the model estimation error is continuously checked to detect potential instabilities caused by the drift in process parameters. An instability alarm can be designed on the basis of the nominal closed-loop response in (3.11)-(3.13). Specifically, it can be verified that, in the absence of process parametric drift, the estimation error obeys the following time-varying bound:

$$\|e(t)\| \leq \alpha e^{-\beta(t-t_0)}\|\xi_0\| \tag{3.15}$$

where $\alpha \geq 1$ and $\beta > 0$ are tunable parameters that depend on the choice of the controller and observer gains. Using this bound as a time-varying alarm threshold, instability can be detected once the norm of the estimation error violates the threshold, i.e., when $\|e(t_b)\| > \alpha e^{-\beta(t_b-t_0)}\|\xi_0\|$, where t_b is the time of the threshold breach. This breach triggers subsequent steps, culminating in the update of the model parameters. To minimize possible detection delays, it is important to choose the controller and observer design parameters judiciously to ensure that the alarm threshold is tight enough. Note that, unlike the update of the model state which occurs periodically at a constant rate, the

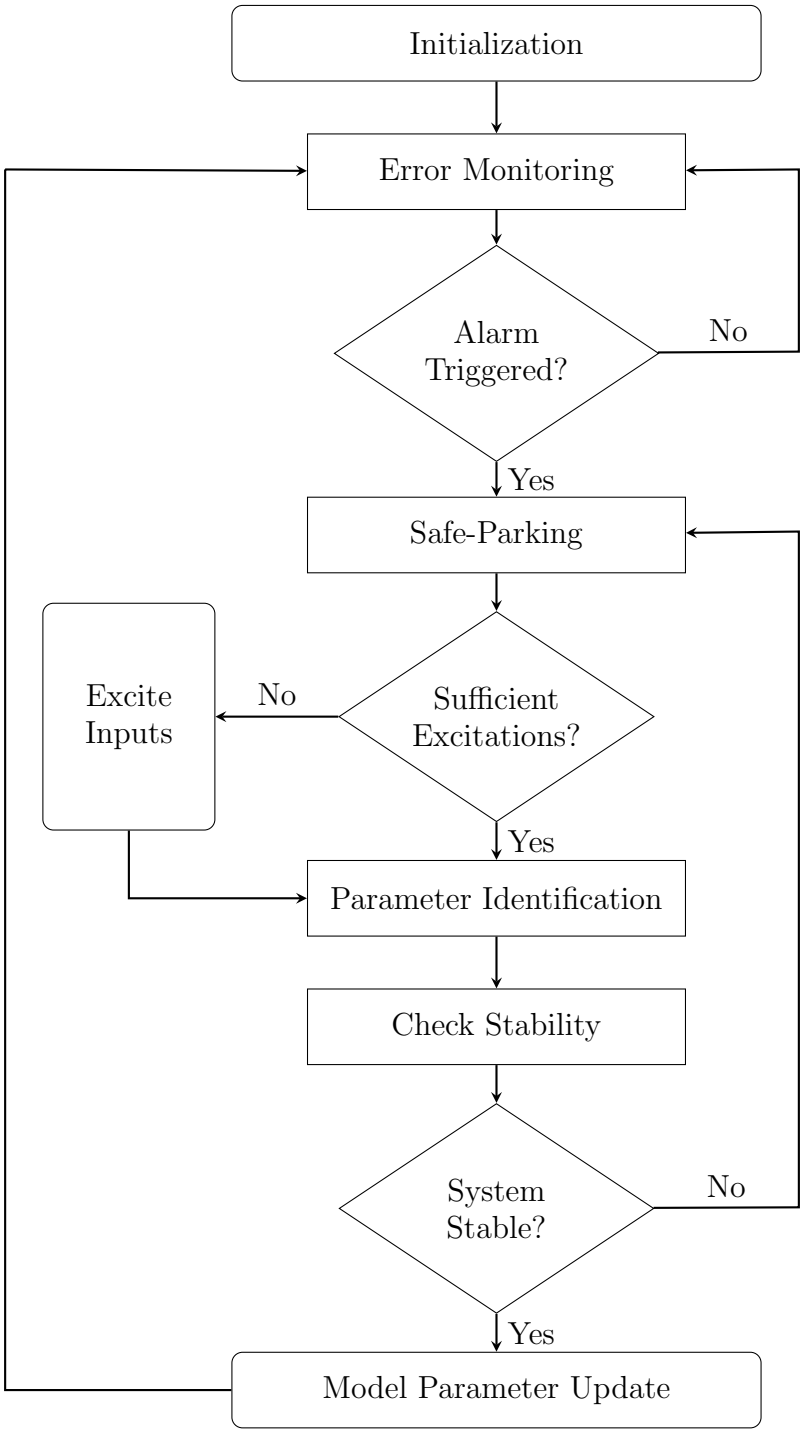

Figure 3.1. A flowchart summarizing the proposed approach for augmenting model-based control with model identification under output feedback control

model parameter updates are event-triggered and thus are generally less frequent.

3.3.3 An error-triggered safe-parking mode of operation

Following the breach of the alarm threshold, the process is switched temporarily to a safe-parking mode of operation in order to maintain stability while collecting sufficient data for parameter re-identification.

One way to achieve closed-loop stability during the safe-parking mode is to temporarily switch the model state update period according to the following switching criterion:

$$h(t) = \begin{cases} h_i \quad , & 0 \leq t < t_b \\ h_{sp} \quad , & t \geq t_b \end{cases} \tag{3.16}$$

where t_b is the time when the alarm threshold is breached and the safe-parking protocol is activated, h_i is the initial update period used prior to t_b, and h_{sp} is the stabilizing (safe-parking) update period chosen such that $\lambda_{max}(h_{sp}) < 1$. This safe-parking method comes at the cost of increasing the communication frequency for the affected process (which may take a way resources from other units or processes sharing the communication medium); nevertheless, this temporary increase in the communication rate serves to maintain closed-loop stability and to collect sufficient data for the model parameter identification step. Inherently, this approach poses a trade-off between requirements of model re-identification (which may favor a longer safe-parking period to collect sufficient data) and the need to reduce communication costs (which favors a shorter safe-parking period).

Remark 3.1. *It should be noted here that the possibility of increasing the communication rate during the safe-parking period requires that a larger share of the bandwidth be made available to the control system of the process affected by the drift. In a shared network setting, this increase comes at the expense of other units or processes that utilize the network, and whose share of the bandwidth (or network access) may be temporarily reduced. While it is important for the plant's supervisory control system to allocate and divert network resources to the units that need it the most (i.e., units on the verge of instability), it is also important to factor in the potential impact of this resource re-allocation during the safe-parking period on the stability and performance of the other units in deciding the*

duration of the safe-parking period.

Another method that can be utilized to maintain closed-loop stability during the safe-parking step is to adjust the observer gain while fixing the sensor-controller communication frequency. This can be achieved by switching the observer gain according to the following criterion:

$$L(t) = \begin{cases} L_i \quad , & 0 \leq t < t_b \\ L_{sp} \quad , & t \geq t_b \end{cases} \tag{3.17}$$

where L_i is the initial observer gain used before the threshold is breached, and L_{sp} is the observer gain during the safe-parking period chosen such that $\lambda_{max}(L_{sp}) < 1$. Using this safe-parking method could help avoid the communication cost increase associated with the former method; however, changing the observer gain could affect the closed-loop system performance and this impact needs to be weighed against the increased communication cost associated with the other approach to decide on a suitable safe-parking strategy. Both safe-parking approaches will be investigated in the context of the simulation study in Section 3.4.

3.3.4 Check for sufficient excitations

The accuracy of the parameters identified in the parameter identification step is governed by quality of the data collected during the safe-parking mode of operation and whether we collected enough data to excite all the frequencies (the slow, intermediate, and fast dynamics). Therefore, it is essential to check that the data collected results in sufficient frequency excitations. If the excitations are sufficient, the model parameters are re-identified, otherwise, the system inputs are perturbed in order to generate sufficient excitations. One way to guarantee sufficient excitations is to perturb the system inputs using simple input signals such as the Pseudorandom Binary Sequence (PRBS) which varies each of the inputs between 2 values only.

Remark 3.2. *This type of perturbation is sufficient for Linear Time Invariant (LTI) systems. The magnitude scales linearly in LTI systems, therefore, the use of a simple PRBS signal to identify the system parameters can be performed easily while still obtaining*

the correct LTI system. This can be done while maintaining the system in the safe-parking mode of operation.

3.3.5 Model parameter re-identification

The choice of the identification method needed for parameter estimation generally depends on the particular application. For illustration purposes, subspace identification techniques (see, for example, [58], [59], [60], [61]) are used in this study to identify the new model parameters based on the closed-loop input and output data to keep the plant-model mismatch to a minimum following parametric drift. This is achieved by solving the following subspace equations using least squares methods:

$$\begin{pmatrix} X_{i+1}^d \\ Y_i \end{pmatrix} = \begin{pmatrix} \bar{A} & \bar{B} \\ \bar{C} & 0 \end{pmatrix} \begin{pmatrix} X_i^d \\ U_i \end{pmatrix} \tag{3.18}$$

where X^d is the deterministic state sequence, Y_i is the output block Hankel matrix, and U_i is the input block Hankel matrix. Note that the subspace identification method is not unique and any other parameter identification method can be utilized here.

3.3.6 A post-identification closed-loop stability check

Once the new model parameters ($\bar{A}$ and $\bar{B}$) are identified, the stability properties of the new closed-loop system need to be analyzed based on the eigenvalues of the new stability test matrix M (per (3.14)) to determine the range of stabilizing update periods and/or observer gains associated with the new model parameters. For the safe-parking approach where the observer gain is fixed and the update period is adjusted, if the new model does not yield a maximum allowable update period that is greater than h_{sp}, then the closed-loop system should remain in the safe-parking mode until a stabilizing model with a lower update frequency is found (possibly by using a different identification method); otherwise, the model parameters are updated and the corresponding update period is implemented:

$$\widehat{A}(t_u) \;=\; \bar{A}, \;\; \widehat{B}(t_u) \;=\; \bar{B}, \;\; h(t) = h_f \;, \quad t \geq t_u \tag{3.19}$$

where t_u is the time that the model parameters are updated, and h_f is the new stabilizing update period. We note here that the process parametric drift is considered to be slow enough such that continuous or frequent updates of the model parameters are not needed.

For the safe-parking approach where the update period is fixed and the observer gain is varied, if the new model does not yield a satisfactory range of feasible observer gains (that can recover some of the potential closed-loop performance losses induced by the safe-parking observer gain), the closed-loop system is kept in the safe-parking mode until a stabilizing model with an observer gain that yields satisfactory closed-loop performance is identified; otherwise, the model parameters are updated and the corresponding observer gain is implemented as follows:

$$\widehat{A}(t_u) \;=\; \bar{A}, \;\; \widehat{B}(t_u) \;\; \bar{B}, \; L(t) = L_f \; , \quad t \geq t_u \tag{3.20}$$

where L_f is the newly identified stabilizing observer gain.

3.4 Simulation Study: Application to a Chemical Process Example

To illustrate the implementation of the methodology described in Sections 3.2 and 3.3, we consider in this section a jacketed, well-mixed, non-isothermal continuous stirred tank reactor in which an irreversible elementary reaction, $A \rightarrow B$, is carried out. The feed stream consists of pure reactant at flow rate F, concentration C_{Ao} and temperature T_o. The reaction is exothermic and a cooling jacket with non-negligible dynamics is used for heat management. Cooling water is added to the jacket at a flow rate F_j and an inlet temperature T_{jo}.

Under standard modeling assumptions, the following process dynamic model can be obtained from material and energy balances:

$$\begin{aligned} \frac{dC_A}{dt} &= \frac{F}{V_r}(C_{Ao} - C_A) - k_o e^{-\frac{E}{RT_r}} C_A \\ \frac{dT_r}{dt} &= \frac{F(T_o - T_r)}{V_r} + \frac{-\Delta H_r k_o}{\rho_m c_{pm}} e^{-\frac{E}{RT_r}} C_A + \frac{UA_r(T_j - T_r)}{\rho_m c_{pm} V_r} \\ \frac{dT_j}{dt} &= \frac{F_j}{V_j}(T_{jo} - T_j) - \frac{UA_r}{\rho_j c_{pj} V_j}(T_j - T_r) \end{aligned} \tag{3.21}$$

where the process parameters are given in Table 3.1. The control objective is to stabilize reactor operation at the open-loop unstable steady-state ($C_{As} = 5.4$ mol/L, $T_{rs} = 328$

Table 3.1. Process parameters and steady-state values

$$
\begin{aligned}
V_r &= 1.00 \quad m^3 \\
V_j &= 0.109 \quad m^3 \\
A_r &= 23.226 \quad m^2 \\
C_{A0} &= 8.0 \quad kmol \cdot m^{-3} \\
U &= 226 \quad kcal \cdot hr^{-1} \cdot m^{-2} \cdot K^{-1} \\
T_0 &= 300.17 \quad K \\
T_{j0} &= 294.4 \quad K \\
R &= 1.987 \quad kcal \cdot kmol^{-1} \cdot K^{-1} \\
\Delta H_r &= -1.67 \times 10^4 \quad kcal \cdot kgmol^{-1} \\
k_0 &= 7.09 \times 10^{10} \quad hr^{-1} \\
E &= 1.67 \times 10^4 \quad kcal \cdot kg^{-1} \\
c_{pm} &= 0.231 \quad kcal \cdot kg^{-1} \cdot K^{-1} \\
c_{pj} &= 0.308 \quad kcal \cdot kg^{-1} \cdot K^{-1} \\
\rho_m &= 809.0 \quad kg \cdot m^{-3} \\
\rho_j &= 1000.0 \quad kg \cdot m^{-3} \\
F &= 1.13 \quad m^3 \cdot hr^{-1} \\
F_j &= 5.38 \quad m^3 \cdot hr^{-1}
\end{aligned}
$$

K, $T_{js} = 320$ K), by manipulating the the flow rate of the cooling water F_j, using measurements of the reactor and jacket temperatures, T and T_j, respectively, in the presence of process parametric drift during operation. The output measurements are assumed to be transmitted to the control system over a bandwidth-limited communication network whose resources are shared among multiple processes and units. From a network resource utilization perspective, the goal is to try to keep the extent to which the process needs to access the network minimal.

The control problem is addressed on the basis of the linearization of the process around the unstable steady-state, which takes the form of (3.1) with the following state, input, and output matrices:

$$A = \begin{bmatrix} -1.6630 & -0.2262 & 0 \\ 47.6263 & -9.0081 & 28.0881 \\ 0 & 156.3528 & -205.7106 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ -235.1890 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Using the results presented in Section 3.2, a model-based output feedback controller is initially designed. The feedback controller gain is chosen to place the poles of the closed-loop model at [-2, -8, -10], while the observer gain is selected to place the closed-loop observer poles at [-50, -75, -100]. To simulate process parametric drift, uncertainty is introduced in the heat of reaction, by defining the following error parameter: $\delta = (\Delta H^m - \Delta H)/\Delta H$, where ΔH^m is the nominal value of the heat of reaction. The process initially operates at steady-state with a 5% mismatch between the model and process parameters. Based on the chosen controller and observer design parameters, the following alarm threshold is used to detect parametric drift: $\|e(t)\| > 3.5e^{-0.009t}\|\xi(t_o)\|$. In the remainder of this section, we explore the implementation of the proposed methodology for integrating model-based control and identification in the context of two different safe-parking approaches and simulation scenarios.

3.4.1 Safe-parking using sensor-controller communication rate

For this case, the operating conditions are chosen initially such that the model states are updated every 17.9 minutes; that is, $h_i = 0.298$ hr. To simulate the effect of a process parametric drift, at the 10-th hour of operation the uncertainty in the heat of reaction is increased such that the plant-model mismatch is now greater. The error parameter thus changes value from $\delta = -0.05$ to $\delta = -0.15$ at $t = 10$ hr.

Figure 3.2 shows the response of the reactor temperature, the cooling jacket inlet flow rate and the norm of the estimation error for the case when safe-parking is performed using the update period. It can be seen that starting at approximately the 21-st hour of operation, the reactor temperature starts to diverge from the operating steady-state in an oscillatory fashion. If left unaddressed, the plant-model mismatch would cause the system to remain unstable.

At approximately the 26-th hour of operation, the norm of the estimation error violates the alarm threshold and the safe-parking protocol is initiated to restore closed-loop stability in preparation for the parameter re-identification phase.

To determine the stabilizing update period for the safe-parking mode, the maximum eigenvalue magnitude of the stability test matrix is calculated for different uncertainty and update period values. Figure 3.3 is a contour plot of $\lambda_{\max}(M)$ as a function of the plant-model mismatch, δ, and the update period, h. The region depicted in white is the closed-loop stability region which represents the points at which the maximum eigenvalue magnitude of the stability test matrix M is less than 1 as established by (3.14). By examining the stability landscape in Figure 3.3 it can be seen that for small h values (more frequent communication) there is a large tolerance for uncertainty between the process and model parameters. However, for larger h values (less frequent communication) the range of tolerable uncertainties decreases and more communication is needed to achieve stability for high values of δ. This trend can be leveraged to safe-park process operation and ensure stability following parametric drifts.

Figure 3.4 shows the maximum eigenvalue magnitude for different update period values at different operating points. It can be seen that before the drift was introduced, the

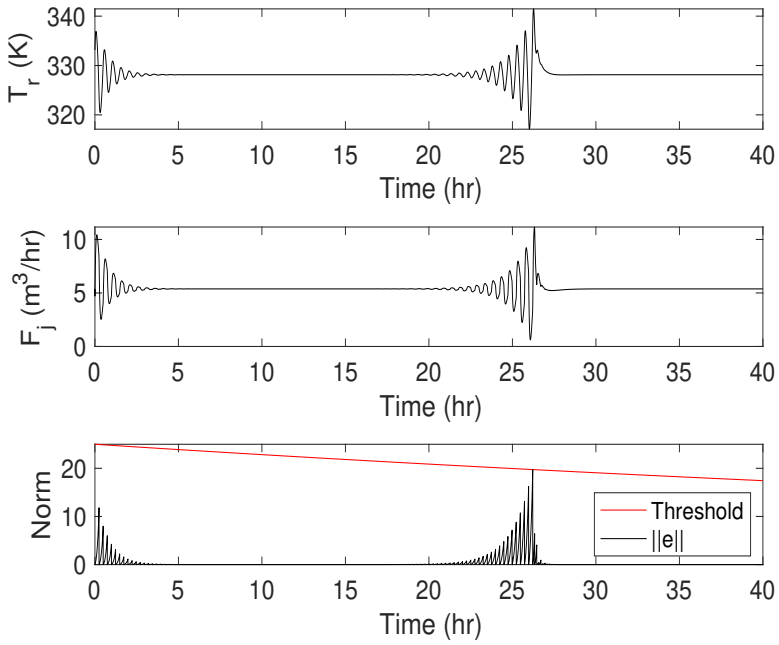

Figure 3.2. Closed-loop reactor temperature profile (top), cooling water inlet flow rate profile (middle) and the norm of the estimation error profile (bottom) when a parametric drift occurs at $t = 10$ hr, the alarm threshold is breached and safe-parking is triggered at $t = 26$ hr, and the model parameters are re-identified and updated at $t = 30$ hr

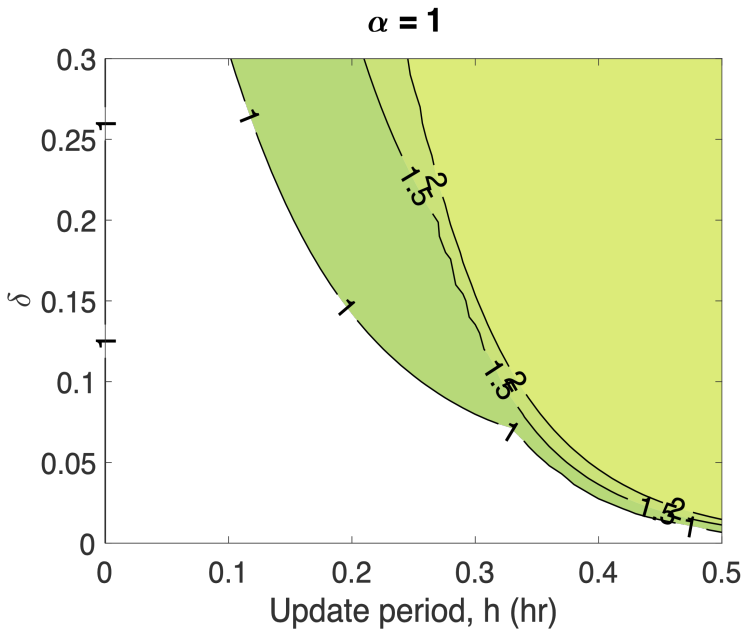

Figure 3.3. Contour plot showing the closed-loop stability region (white) where $\lambda_{max}(M) < 1$ for different δ and h values

maximum stabilizing update period value is $h_i = 0.298$ hr. During the safe-parking period, however, the maximum stabilizing update period is reduced to $h_{sp} = 0.210$ hr. The system is operated at the new stabilizing update period value while closed-loop data is collected for the parameter identification step. Once a sufficient amount of data is collected the model parameters are re-identified and updated at the 30-th hour of operation. It can be seen in Figure 3.4 that the maximum stabilizing update period after identification takes place is now $h_f = 0.315$ hr. Since this new update period value is greater than the update period value during the safe-parking step, the system is switched out of the safe-parking mode and the model state updates now occur at the new update rate. Figure 3.2 shows that the system continues to operate stably after the model parameters are re-identified and updated and the communication rate decreased.

3.4.2 Safe-parking using the observer pole placement

As mentioned earlier in Section 3.3.3, reducing the update period to maintain closed-loop stability during the safe-parking period can lead to an increase in network utilization that may be undesirable. An alternative safe-parking approach is to keep the update rate

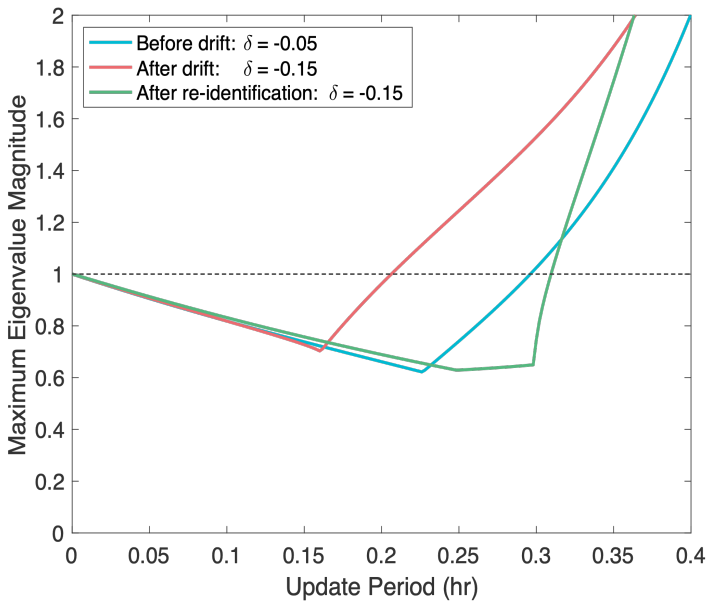

Figure 3.4. λ_{max}(M) as a function of the update period under normal operation (blue), under increased plant-model mismatch (red) and after parameter re-identification (green)

fixed, and instead adjust the observer gain to temporarily achieve closed-loop stability. To capture the effect that adjusting the observer gain has on closed-loop stability, we will vary the placement of the closed-loop observer poles. This is achieved by introducing a pole placement parameter, α, to serve as a multiplier to the poles, as follows: [-50α, -75α, -100α]. The baseline value of $\alpha = 1$ corresponds to the initial placement of the poles based on the initial choice of the observer gain. As α is increased, the poles are shifted farther to the left in the complex plane.

Figure 3.5 shows the closed-loop stability region (in white) as a function of α and h, after the process drift takes place (with $\delta = -0.15$). It can be seen that for small h values (more frequent communication) the range of stabilizing α values is greater than the stabilizing range for large h values (less frequent communication). For large enough update periods, it appears that no choice of α is stabilizing. Conversely, for α values less than 1, there is no update period value that stabilizes the closed-loop system; however, for values of $\alpha > 1$ the range of stabilizing update periods continues to increase slightly until an asymptote is reached around $\alpha = 9$.

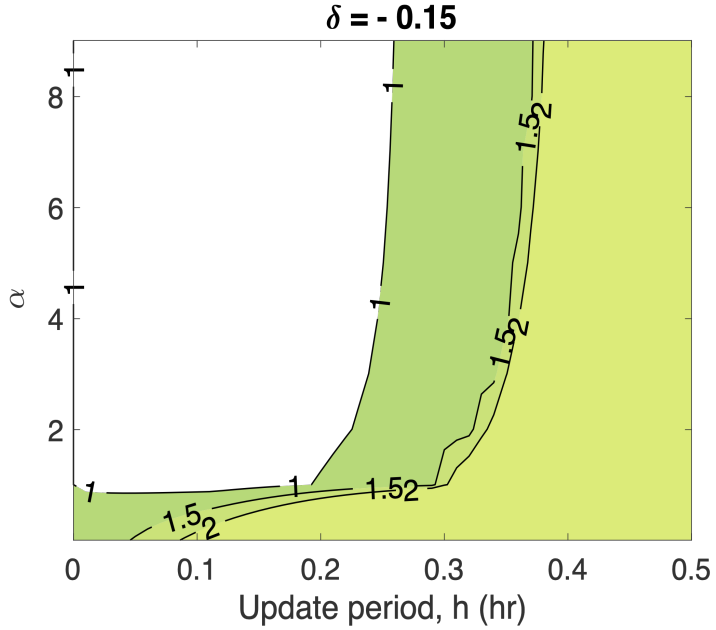

Figure 3.5. Contour plot showing the closed-loop stability region (white) where $\lambda_{max}(M) < 1$ for different α and h values, and with $\delta = -0.15$

Figure 3.6 shows the response of the reactor temperature, the cooling jacket inlet flow

rate and the norm of the estimation error for the case when safe-parking is performed using the observer pole placement α. Similar to the previous safe-parking case, a drift is introduced at the 10-th hour of operation and the error parameter is set to $\delta = -0.15$ at that time. It can be seen from Figure 3.7 that the maximum stabilizing update period decreases from 0.298 hr to 0.210 hr immediately after the drift. Starting at approximately the 20-th hour of operation, the reactor temperature starts to diverge away from the operating steady-state in an oscillatory fashion until the instability alarm threshold is breached and the safe-parking protocol is triggered a few hours later.

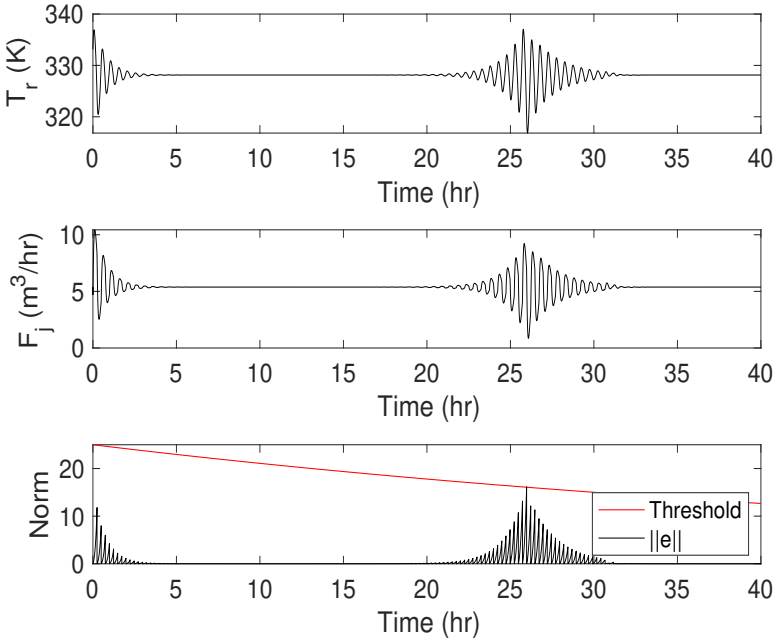

Figure 3.6. Closed-loop reactor temperature profile (top), cooling water inlet flow rate profile (middle) and the norm of the estimation error profile (bottom) when a parametric drift occurs at $t = 10$ hr, the alarm threshold is breached and safe-parking is triggered at $t = 26$ hr, and the model parameters are re-identified and updated at $t = 30$ hr

Unlike the previous case where closed-loop stability was regained by decreasing the update period following the parametric drift, in this case increasing α enables us to recover stability while maintaining the original operating update period that was used before the drift occurred. This can be seen by the solid red line in Figure 3.7 where α is increased to a value of 2.8.

This approach allows us to maintain closed-loop stability without the incurred cost

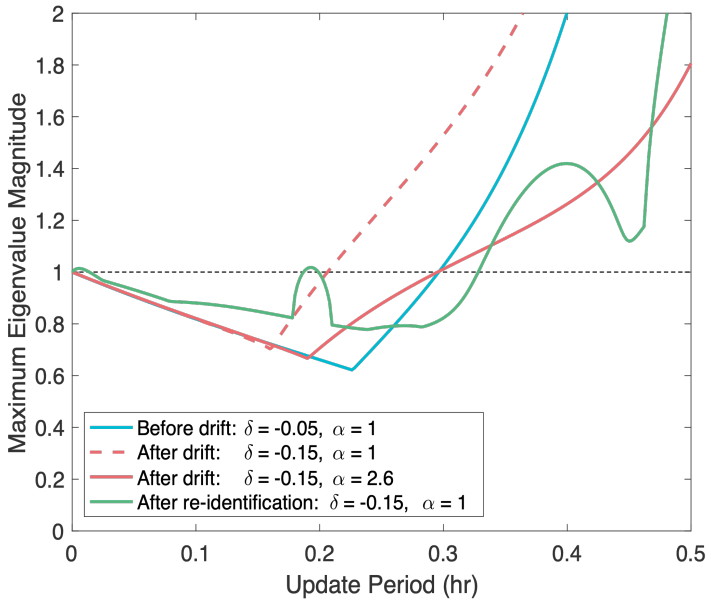

Figure 3.7. λ_{max}(M) as a function of the update period under normal operation (blue), under increased plant-model mismatch with $\alpha = 1$ (dashed red), under increased plant-model mismatch with increased α (solid red) and after parameter re-identification with $\alpha = 1$ (green)

of increased communication; however, the trade-off here is that the performance of the closed-loop response is impacted due to the use of the observer gain as the stabilizing method during the safe-parking mode of operation. It can be seen in Figure 3.6 that the response is significantly slower at reaching the steady-state compared to the response time in Figure 3.2. Depending on the system being controlled, this impact needs to be weighed against the increased communication cost associated with the alternate safe-parking approach to decide on a suitable safe-parking strategy.

When a sufficient amount of data has been collected, the model parameters are re-identified and updated after a closed-loop stability check is performed. Specifically, it can be seen from the green curve in Figure 3.6 that, at the operating update period, closed-loop stability is maintained using the newly identified model parameters and the old $\alpha = 1$ value before the drift took place. Based on this result, the process is taken out of the safe-parking mode by restoring α restored to its pre-drift value to recover some of the closed-loop performance losses, and the new model parameters are implemented.

3.5 Conclusions

This chapter developed a methodological framework for enhancing closed-loop performance and network resource utilization achieved using model-based networked control subject to limited output measurements, process parameter variations and sensor-controller communication constraints. The main idea was to design a model-based output feedback controller with well-characterized closed-loop stability properties and to augment its implementation using error-triggered online parameter re-identification that helps maintain acceptable levels of network utilization and closed-loop performance in the event of process parametric drift. The proposed methodology was demonstrated using a chemical process example, and it was shown that parameter re-identification can help enhance the closed-loop stability for process systems with limited output measurements while reducing network resource utilization. Future work will focus on investigating the effect of measurement noise on parameter identification and the benefits of explicitly modeling the output noise to enhance the accuracy of the identified parameters and further minimize the plant-model mismatch.

Chapter 4

Model-Based Networked Control of Spatially-Distributed Processes with Parameter Re-identification

At this point, we have only considered implementation strategies for systems modeled by ordinary differential equations (ODEs), however, many applications are modeled by highly-dissipative PDEs. The problem of stabilizing spatial profiles in spatially-distributed processes using feedback control has been the subject of significant research work in the distributed parameter systems area (e.g., [62], [63], [64]). This problem is significant because the dynamic behavior of many industrial processes, such as fluid flows and transport-reaction processes, is characterized by spatiotemporal variations and uncertainties, which are captured by Partial Differential Equation (PDE) models. An important class of PDEs to which finite-dimensional controller design techniques can be applied is the class of highly-dissipative PDEs whose dominant dynamics are low-dimensional (e.g., [65], [66], [67], [68], [69]). The majority of existing research work on control of highly-dissipative PDE systems is based on the classical paradigm of feedback control in which sensor-controller communication is assumed to take place through dedicated channels, ignoring real-time implementation issues such as sensor-controller communication resource constraints which are becoming increasingly important with the increased integration of communication networks in feedback control systems (e.g., [7], [8]).

An established strategy for dealing with the communication resource constraints problem is through the use of model-based control which aims to achieve closed-loop stabilization with minimal feedback from the sensor to the controller over the network (e.g., [19], [17]). While this strategy is appealing in that it can potentially reduce the reliance of the feedback control system on the communication medium, and can thus free up the network for other tasks, the extent of communication savings achieved is highly influenced by the quality of the model used and the extent of plant-model mismatch.

To address this implementation gap, in this chapter, we present a framework for the integration of model-based control and model identification for spatially-distributed process systems modeled by highly-dissipative PDEs, subject to sensor-controller communication constraints and process parameter variations. The framework aims to enhance the stability and performance properties of the networked closed-loop system in the presence of process parameter variations and external disturbances, while simultaneously reducing the rate of sensor-controller information transfer required. Initially, a networked feedback controller is designed on the basis of an approximate finite-dimensional model that captures the dominant dynamics of the infinite-dimensional system. The maximum allowable model state update rate is explicitly characterized in terms of the model parameters and the control actuator locations, and this characterization is used to devise a time-triggered model state update policy that guarantees closed-loop stability.

An error monitoring scheme with a time-varying instability alarm threshold is then developed to track the state evolution and trigger model re-identification and model parameter updates in the event of process parametric drift. When the alarm threshold is breached, a safe-parking protocol is initiated by temporarily increasing the sensor-controller communication rate to counter the destabilizing influence of parametric drift. In the mean-time, the input and state data collected during the safe-parking period are used to identify, on-line, a new finite-dimensional model based on subspace identification techniques. The networked closed-loop stability region associated with the newly-identified reduced-order model is then characterized and used to identify a suitable model state update rate that can restore the control performance and network utilization to

their pre-drift levels. The development and implementation of the proposed framework are demonstrated using a representative diffusion-reaction process example. The results of this chapter were first published in [52].

4.1 Preliminaries

In this work we focus on spatially-distributed processes modeled by highly-dissipative PDEs with low-order dynamics. A common example of this class of systems are parabolic PDEs described by:

$$\frac{\partial \bar{x}(z,t)}{\partial t} = \alpha \frac{\partial^2 \bar{x}(z,t)}{\partial z^2} + \beta \bar{x}(z,t) + \omega \sum_{i=1}^{m} b_i(z) u_i(t) \tag{4.1}$$

subject to the following initial and boundary conditions:

$$\begin{aligned} \bar{x}(z,0) &= \bar{x}_0(z) \\ \bar{x}(0,t) &= \bar{x}(\pi,t) = 0 \end{aligned} \tag{4.2}$$

where $\bar{x}(z,t) \in \mathbb{R}$ is the state variable, $z \in [0,\pi]$ is the spatial coordinate, $t \in [0,\infty)$ is the time, u_i is the i-th manipulated input, m is the number of manipulated inputs, and $b_i(z)$ is the actuator distribution function. The parameters $\alpha > 0$, β, and ω are constants and $x_0(z)$ is a smooth function of z.

Using standard techniques from infinite-dimensional system theory (e.g., see [70]), the PDE in (4.1) can be formulated as an infinite-dimensional system of the form:

$$\dot{x}(t) = \mathcal{A}x(t) + \mathcal{B}u(t) \tag{4.3}$$

with an initial condition $x(0) = x_0$, where $x(t) = \bar{x}(z,t)$, for $t > 0$ and $z \in [0,\pi]$, is the state function defined on the Hilbert space $\mathcal{H} = L_2(0,\pi)$, $\mathcal{A}$ is the spatial differential operator whose spectrum is characterized by a large separation between a finite set of slow (possibly unstable) eigenvalues and an infinite set of fast stable eigenvalues (e.g., see [63]), $\mathcal{B}$ is the input operator that describes the spatial placement of the control actuators, and $u = [u_1 \cdots u_m]^T \in \mathbb{R}^m$ is the vector of manipulated inputs. Taking advantage of the low-order dynamics of this class of systems, standard modal decomposition can be applied

to the system in (4.3) to obtain the following interconnected systems:

$$\begin{aligned} \dot{x}_s(t) &= \mathcal{A}_s x_s(t) + \mathcal{B}_s u(t) \\ \dot{x}_f(t) &= \mathcal{A}_f x_f(t) + \mathcal{B}_f u(t) \end{aligned} \tag{4.4}$$

with initial conditions $x_s(0) = \mathcal{P}_s x_0$ and $x_f(0) = \mathcal{P}_f x_0$, where $x_s = \mathcal{P}_s x$ is the state of the finite-dimensional system that describes the evolution of the slow eigenmodes, $x_f = \mathcal{P}_f x$ is the state of an infinite-dimensional system that describes the evolution of the fast stable eigenmodes, $\mathcal{A}_s = \mathcal{P}_s\mathcal{A}$, $\mathcal{A}_f = \mathcal{P}_f\mathcal{A}$, $\mathcal{B}_s = \mathcal{P}_s\mathcal{B}$, $\mathcal{B}_f = \mathcal{P}_f\mathcal{B}$, and $\mathcal{P}_s$ and $\mathcal{P}_f$ are the orthogonal projection operators for the slow and fast eigenmodes, respectively.

Applying model reduction techniques [63] we can obtain a finite-dimensional ODE system that describes the time evolution of the amplitudes of the dominant eigenmodes. The ODE system takes the following form:

$$\dot{a}_s = A_s a_s + B_s(z_a)u \tag{4.5}$$

where $a_s = [a_1 \cdots a_m]^T \in \mathbb{R}^m$, a_i is the amplitude of the i-th eigenmode, A_s is an $m \times m$ diagonal matrix containing the first m slow eigenvalues of $\mathcal{A}$, and B_s is an $m \times m$ input matrix whose elements are parameterized by the control actuator locations, z_a.

Remark 4.1. *When state-feedback control is used (i.e., u is chosen as a function of the slow states only) the closed-loop slow subsystem in 4.4 becomes decoupled from the fast subsystem. In this case, stabilization of the closed-loop slow subsystem is sufficient to guarantee stability of the infinite-dimensional system since the fast subsystem is stable and driven by a bounded and converging input. This has important implications for model identification (see Remark 4.4).*

4.2 Model-Based State-Feedback Controller Design

4.2.1 Controller synthesis

To address the control objective, we consider a typical model-based networked control configuration with periodic sensor-controller communication. The idea behind model-based control is to embed within the control system a dynamic model which is used to

provide estimates of the states in the event when communication between the controller and the sensor is suspended. When communication is re-established, the model states are updated using the plant states, at discrete time instances, based on the sensor-measured values. The resulting model-based state-feedback control law is given by:

$$\begin{aligned} u(t) &= K\widehat{a}_s(t) \ , \quad t \in [t_k, t_{k+1}) \\ \dot{\widehat{a}}_s(t) &= \widehat{A}_s\widehat{a}_s(t) + \widehat{B}_s u(t) \ , \quad t \in [t_k, t_{k+1}) \\ \widehat{a}_s(t_k) &= a_s(t_k) \ , \quad k \in \{0, 1, 2, \cdots\} \\ h &\triangleq t_{k+1} - t_k \end{aligned} \tag{4.6}$$

where K is the feedback gain which is chosen to stabilize the closed-loop model, $\widehat{a}_s$ is the state of the model and $\widehat{A}_s$ and $\widehat{B}_s$ are constant matrices that approximate the plant matrices A_s and B_s, respectively, t_k is the update time, and h is the update period which is assumed to be constant.

4.2.2 Closed-loop stability analysis

In order to achieve closed-loop stability with minimal communication, it is important to characterize the maximum allowable update period for the networked control system. The maximum allowable update period dictates the minimum communication frequency required for stability. This characterization is obtained by performing a closed-loop stability analysis as shown in [19]. By performing the stability analysis it can be shown that the response of the closed-loop system in terms of the update period is given by:

$$\xi(t) = e^{\Lambda(t-t_k)}(I_s e^{\Lambda h} I_s)^k \xi_0 = e^{\Lambda(t-t_k)} M^k \xi_0 \tag{4.7}$$

where $t \in [t_k, t_{k+1})$, $\xi(t) := [a_s^T(t) \ \ e^T(t)]^T$ is an augmented vector of the plant states and model estimation errors, defined as the difference between the model and plant states, $e(t) = a_s(t) - \widehat{a}_s(t)$, I_s is an augmented identity matrix, $\xi_0 = [a_s^T(t_0) \ \ 0]^T$ is the initial condition, $M = (I_s e^{\Lambda h} I_s)$ is the stability test matrix, and Λ is the augmented closed-loop matrix which depends on the plant and model matrices as well as the controller gain and is given by:

$$\Lambda = \begin{bmatrix} A_s + B_s(z_a)K & -B_s(z_a)K \\ \widetilde{A}_s + \widetilde{B}_s(z_a)K & \widehat{A}_s - \widetilde{B}_s(z_a)K \end{bmatrix} \tag{4.8}$$

where $\widetilde{A}_s = A - \widehat{A}_s$ and $\widetilde{B}_s(z_a) = B(z_a) - \widehat{B}_s(z_a)$.

By analyzing the closed-loop response in (4.7)-4.8, it can be verified that a necessary and sufficient condition for closed-loop stability is to have all eigenvalues of the stability test matrix M lie strictly inside the unit circle. This can be described by the following maximum eigenvalue magnitude function:

$$\lambda_{max}(M)[A_s, B_s(z_a), \widehat{A}_s, \widehat{B}_s(z_a), K, h] < 1 \tag{4.9}$$

where λ_{max} is the maximum eigenvalue magnitude of M.

Remark 4.2. *The condition in* (4.9) *can explicitly characterize the maximum allowable update period, or the minimum allowable communication frequency, necessary to maintain closed-loop stability in terms of the plant-model mismatch and the controller design parameters (the feedback gain and the location of the actuator). Alternatively, for a fixed communication rate, this condition can be used to identify the feasible range of actuator locations. Both characterizations are important because they provide the basis for any recourse action (in terms of adjusting the update period and/or the control actuator location) that may be needed to prevent drift-induced instability (see Section 4.3.3).*

4.3 Integrating Model Identification and Model-Based Control

In this section, we present a methodology for the integration of model identification into the model-based networked control strategy discussed in Section 4.2. We first present an overview of the proposed integration methodology followed by a discussion of the key implementation steps. Figure 4.1 summarizes the proposed approach.

4.3.1 Overview of the proposed methodology

Initially, the networked control system is operated at a chosen update period that minimizes sensor-controller communication while simultaneously ensuring closed-loop stability. As the system continuous to operate, the evolution of the closed-loop state is continuously monitored in order to detect plant-model mismatch. When a significant plant-model mismatch is detected, it becomes necessary to update the model parameters. This is triggered

by continuously checking the closed-loop state against an instability alarm threshold. In the event of a threshold breach, a safe-parking protocol is implemented to momentarily stabilize the closed-loop system by modifying the update period. In the mean time, additional data is collected during the safe-parking period and serves as the basis for the identification of the new model parameters. Once the new model parameters are identified, the new model is then checked for stability and the corresponding stability region is determined. Based on this characterization, the decision to move the system out of the safe-parking regime and update the model parameters is made. The algorithm is then repeated every time the instability alarm threshold is breached.

4.3.2 Model re-identification triggering criterion

In order to detect potential instabilities caused by the drift in process parameters and determine when new model parameters need to be identified, a monitoring scheme is implemented whereby the closed-loop states are continuously checked against some alarm threshold. To that end, we design an instability alarm based on the nominal closed-loop response in (4.7). It can be verified that, in the absence of process parameter variations, there exist positive constants $\delta > 1$ and $\theta > 0$ such that the following time-varying bound is satisfied:

$$\|a_s(t)\| \leq \delta \, e^{-\theta(t-t_0)} \, \|a_s(t_0)\| \tag{4.10}$$

This bound follows from the exponential closed-loop stability of the origin of the networked closed-loop system in (4.7). This bound serves as a time-varying alarm threshold and utilizing it enables us to detect instability once the norm of the state violates the threshold. This step acts as a trigger for the subsequent steps in the methodology culminating in the update of the model parameters. It is important to note that unlike the model state updates which are time-triggered periodically, the model parameter updates are event-triggered and thus occur less frequently.

Remark 4.3. *The threshold defined in* (4.10) *serves as an upper bound on the time evolution of the state of the system. The magnitude of this upper bound depends on the actuator location which could lead to larger upper bounds or smaller upper bounds*

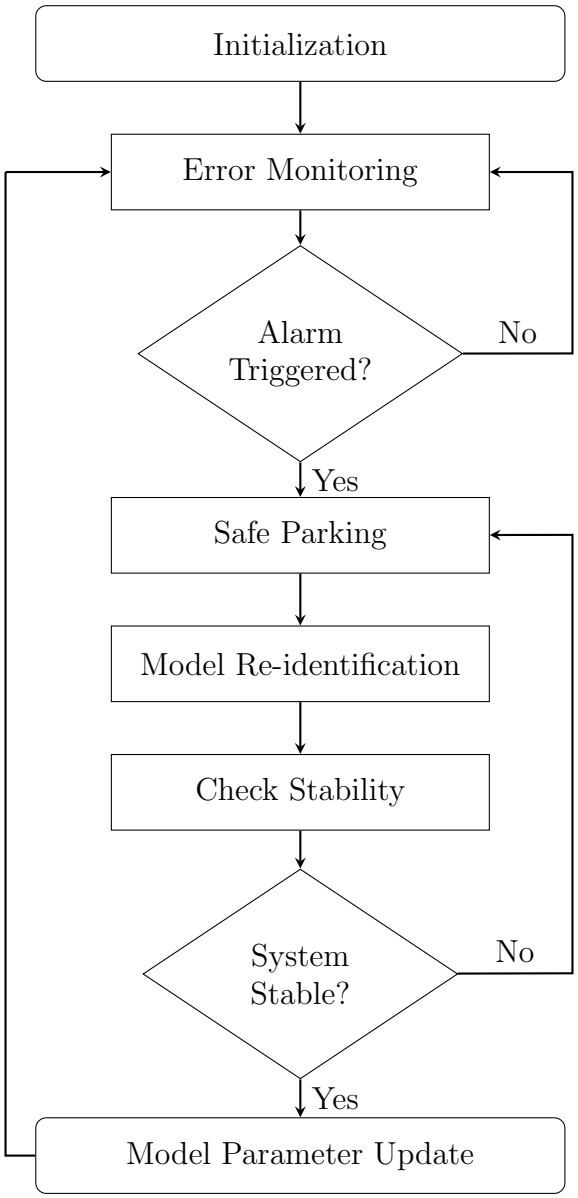

Figure 4.1. A flowchart of the integrated control and identification methodology for networked control systems with periodic communication

depending on the choice of actuator location. This range of error thresholds could lead to cases when the triggering criterion takes longer to detect the drift, thus causing momentary instability. To minimize detection delays, it is therefore important to choose the control actuator location such that the alarm threshold is sufficiently tight.

4.3.3 Safe-parking

When the instability threshold is breached, that is when $\|a_s(t_b)\| > \delta\ e^{-\theta(t_b - t_0)}\ \|a_s(t_0)\|$, where t_b is the time when the threshold is breached, it is imperative to first stabilize the process before proceeding with the model parameter re-identification. In this safe-parking protocol, the plant is switched temporarily to a safe mode of operation. One way to achieve closed-loop stability during the safe-parking step is to temporarily switch the model state update period according to the following switching criterion:

$$h(t) = \begin{cases} h_i \quad , & 0 \leq t < t_b \\ h_{sp} \quad , & t \geq t_b \end{cases} \tag{4.11}$$

where t_b is the time when the instability threshold is breached and the safe-parking protocol is initiated, h_i is the initial update period before t_b, and h_{sp} is the stabilizing (safe-parking) update period chosen such that $\lambda_{max}(h_{sp}) < 1$. This safe-parking method comes at the cost of increased communication frequency; nevertheless, this temporary increase in the communication rate serves to maintain closed-loop stability and to collect sufficient data for the model parameter identification step.

4.3.4 Model parameter re-identification

After the closed-loop system is temporarily stabilized and when a sufficient amount of data has been collected, model re-identification is triggered and new model parameters are obtained, $\bar{A}_s$ and $\bar{B}_s$. The choice of the model re-identification method depends on the particular application. In this work, subspace identification techniques for discrete linear time-invariant systems are used to identify the new model parameters based on the input and state data, in order to minimize plant-model mismatch. Utilizing the subspace methodology and algorithm (e.g., see [58], [59]), new model parameters are obtained by solving the following set of equations using linear least squares algorithms:

$$\begin{pmatrix} X_{i+1}^d \\ Y_i \end{pmatrix} = \begin{pmatrix} \bar{A}_s & \bar{B}_s \\ I & 0 \end{pmatrix} \begin{pmatrix} X_i^d \\ U_i \end{pmatrix} \tag{4.12}$$

where Y_i and U_i depend on the input and state data and are defined as the output and input block Hankel matrices, respectively, and X^d is the deterministic state sequences.

Remark 4.4. *In this work, we consider the state-feedback control problem where full-state measurements of the slow eignemodes are available. As discussed in Remark 4.1, this results in decoupling the evolution of the closed-loop slow eigenmodes from the fast eigenmodes. This decoupling is important for model identification since the exclusion of the fast eigenmodes will not affect model parameter identification because the control and identification problems are addressed on the basis of the slow subsystem only. In the case of output feedback control, however, where only output measurements are available, the dynamics of the slow and fast subsystems will be coupled due to the dependence of the output on both the slow and fast states. In this case, using the output measurements to re-identify the slow subsystem (i.e., reduced order model) introduces errors due to the contribution of the fast states. These errors, however, become negligible if the separation between the slow and fast eigenvalues of the differential operator is sufficiently large. This argument can be justified using singular perturbation techniques [63].*

4.3.5 Closed-loop stability check

Once the new model parameters, $\bar{A}_s$ and $\bar{B}_s$, are identified, it is important to verify that the networked closed-loop stability condition for the newly-obtained model is satisfied. This can be achieved by analyzing the stability of the new model parameters based on the eigenvalues of the test matrix M. Using (4.9), the range of stabilizing update periods associated with the new model can be obtained. If the new model does not yield a maximum allowable update period that is greater than h_{sp}, then the closed-loop system remains in the safe-parking mode until a stabilizing model with a lower update frequency is found (possibly by using a different identification method); otherwise, the model parameters are updated and the corresponding update period is implemented:

$$\widehat{A}_s(t_u) \;=\; \bar{A}, \;\; \widehat{B}_s(t_u) \;=\; \bar{B}, \;\; h(t) = h_f \;, \quad t \geq t_u \tag{4.13}$$

where t_u is the time when the model parameters are updated, and h_f is the newly identified stabilizing update period. It is important to note here that even though the process parameters are assumed to change over time, this drift is generally considered to be slow and does not require continuous or frequent updates of the model parameters.

Remark 4.5. *The need for the post-identification closed-loop stability check performed her stems from the fact that the model re-identification method used does not take the networked closed-loop stability explicitly into account. Based purely on the input and state data available during the safe-parking period, the parameter re-identification method yields parameter values that minimize some measure of the discrepancy between the actual states and inputs on the one hand, and the predicted states and inputs, on the other, with no a priori guarantees regarding networked closed-loop stability. If such guarantees are desired, one may incorporate the networked closed-loop stability condition of* (4.9) *as an additional constraint on the optimization problem formulation for parameter re-identification. While this approach may increases the complexity of the parameter identification algorithm, it eliminates the need for a stability check since the identified model automatically satisfies the closed-loop stability requirement.*

4.4 Simulation Study: Application to a Diffusion-Reaction Process Example

To illustrate the developed methodology, we consider a diffusion-reaction process with the following space-time dynamics:

$$\frac{\partial \bar{x}(z,t)}{\partial t} = \frac{\partial^2 \bar{x}(z,t)}{\partial z^2} + [\beta_T \gamma e^{-\gamma} - \beta_U]\bar{x}(z,t) + \beta_U b(z)u(t) \tag{4.14}$$

subject to the following Dirichlet boundary conditions:

$$\bar{x}(0,t) = \bar{x}(\pi,t) = 0$$

where $\bar{x}(z,t)$ is the state, u is the control input, and $b(z)$ is the control actuator distribution function. It can be shown that the operating steady-state $\bar{x}(z,t) = 0$ is unstable for the following choice of process parameter values: $\beta_T = 80$, $\gamma = 4$, and $\beta_U = 2.0$. The control objective is to stabilize the state profile at the open-loop unstable steady-state with reduced sensor-controller communication. To achieve this objective, we consider the use of a single point control actuator (with finite support) and assume that a sufficiently large number of point measurement sensors that provide accurate state measurements are available. To simulate process parametric drift, we consider parametric uncertainty in

the heat of reaction coefficient, β_U. By solving the eigenvalue problem for the differential operator of the PDE of (4.14), it can be determined that only the first eigenvalue is unstable, and, therefore, we consider the first unstable eigenvalue to be the dominant one and apply the Galerkin's method to obtain the following ODE:

$$\dot{a}_1 = \lambda a_1 + 2\sqrt{\frac{2}{\pi}}\sin(z_a)u \tag{4.15}$$

which describes the evolution of the amplitude of the first eigenmode. Using this ODE, we consider a feedback controller design method based on pole-placement ideas, where the feedback controller gain is independent of the control actuator location.

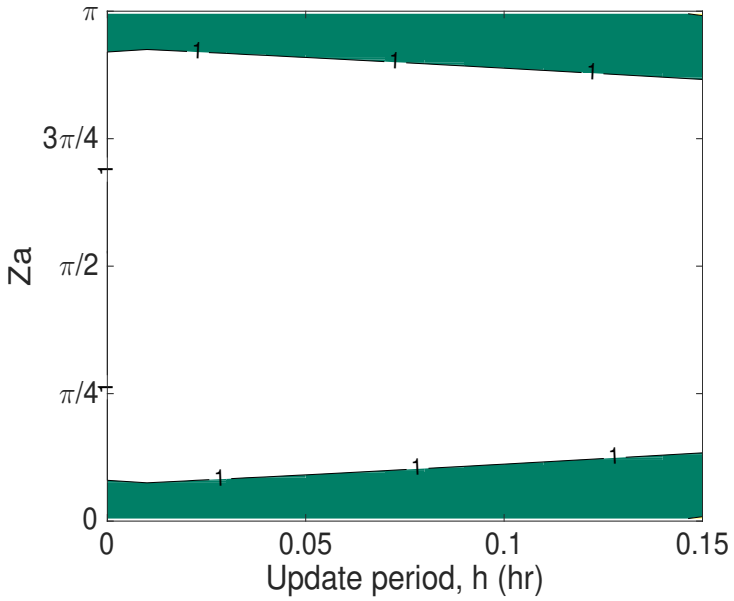

Figure 4.2. Contour plot of the stability region as a function of the update period and actuator location using the pole placement design for nominal operation

For this design method, the actuator location was initially set to $z_a = \pi/2$ and the controller gain was chosen as $K = -8.06$ in order to place the pole of the closed-loop model at -10. Doing so ensures that the closed-loop system is stable for all actuator locations (excluding the boundary points where the system is uncontrollable). It is important to note that for this design method the feedback gain is independent of actuator location and, as a result, the closed-loop response is sensitive to the actuator location (where the middle location yields the fastest response). Using the aforementioned settings, we

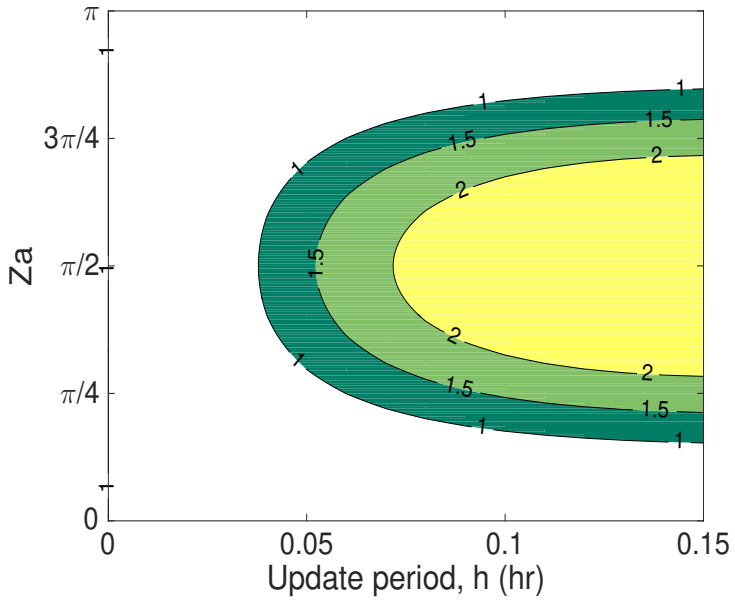

Figure 4.3. Contour plot of the stability region as a function of the update period and actuator location using the pole placement design for operation under drift

are able to characterize the networked closed-loop stability region in terms of the model update period and the control actuator location as shown in Figures 4.2 - 4.3. These figure shows the contour plots of the closed-loop stability region during normal operation (Figure 4.2) and after a drift takes place (Figure 4.3). The region contained inside the unit contour line (the uncolored region) represents the stable region of operation where $\lambda_{max}(M) < 1$. It can be seen that before the drift occurs, the operating ranges of update periods and actuator locations are quite broad; however, after the drift takes place, the range of possible operating update periods shrinks symmetrically around the middle actuator location $za = \pi/2$. This suggests that placing the actuator closer to the boundaries helps reduce the communication cost, since the system can be operated at a longer update period without loss of closed-loop stability. This trend is a result of the fact that the closed-loop response is faster when the actuator is located at $z_a = \pi/2$ and the response is more sensitive to model uncertainties.

Initially, the closed-loop system is stabilized using an update period of $h = 0.15$. When the parametric drift takes place, however, this operating point is no longer stable and safe-parking must take place in order to stabilize the closed-loop system and re-

identify the model parameters. Safe-parking in this case can be performed in one of two ways: one can either safe-park the system by decreasing the update period (i.e., increasing the communication frequency) or by moving the control actuator towards the boundaries of the feasible actuator locations. In what follows, we focus only on the first safe-parking approach where the system is stabilized during the drift by decreasing the update period and increasing sensor-controller communication. The temporary increase in communication serves the dual purpose of maintaining closed-loop stability and collection of frequent and sufficient data for the parameter re-identification step. Figures 4.4 - 4.8 show the time-evolution of the amplitude of the dominant eigenmode, the manipulated input profile, the maximum eigenvalue magnitude as a function of the update period for different operating conditions, the operating update period as a function of time, and the closed-loop state profile, respectively.

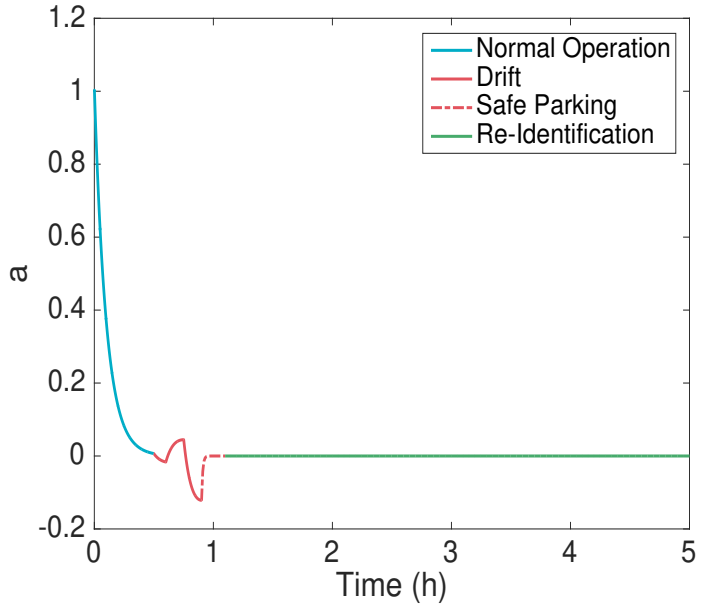

Figure 4.4. Dominant eigenmode profile for the closed-loop system at a fixed actuator location $z_a = \pi/2$ and a pole-placement based gain design

The blue line in Figure 4.4 represents the eigenmode profile when the process is operating normally. A drift is introduced after roughly 30 minutes of operation, and the closed-loop state starts to diverge from the steady-state as represented by the red line. When the norm of the state violates a pre-determined instability threshold, safe-parking

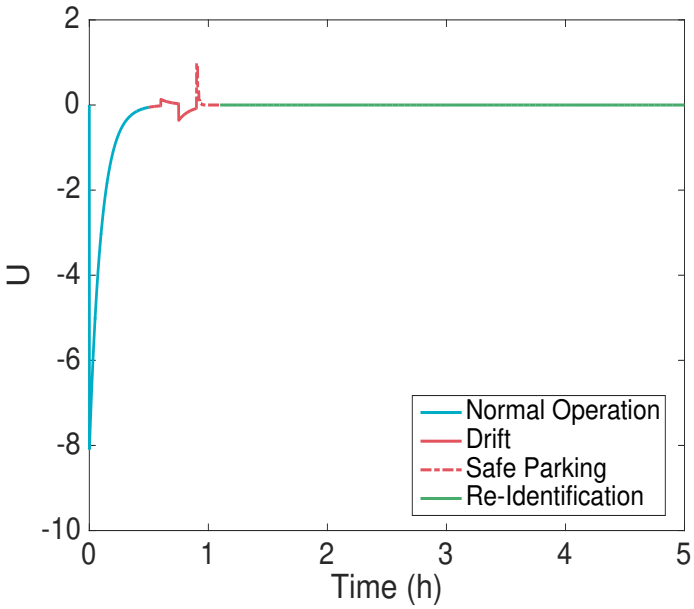

Figure 4.5. Manipulated input profile for the closed-loop system at a fixed actuator location $z_a = \pi/2$ and a pole-placement based gain design

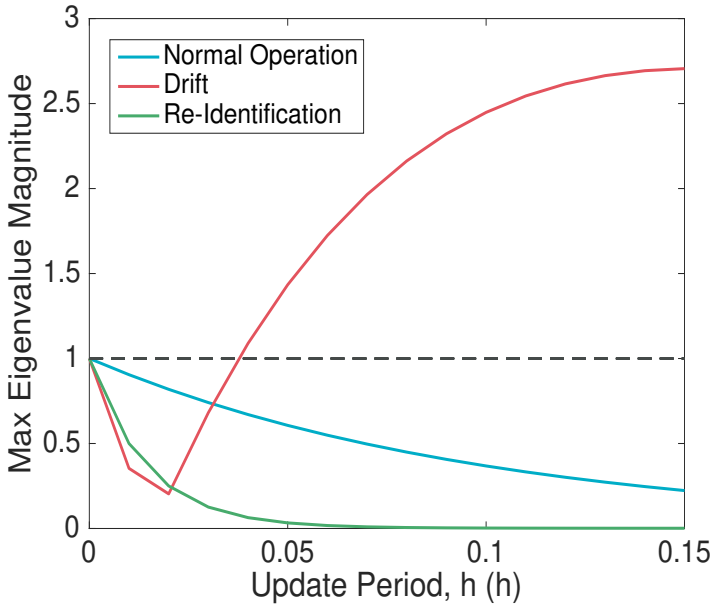

Figure 4.6. Maximum eigenvalue magnitude with respect to the update period for different operating conditions

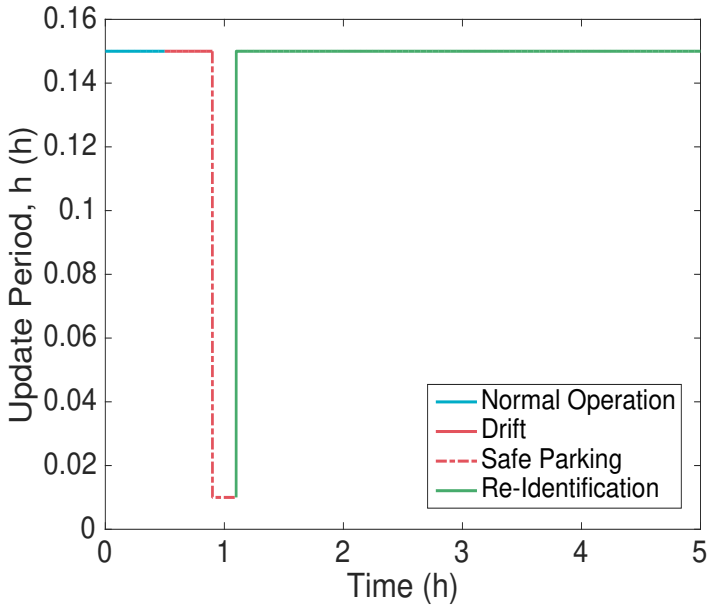

Figure 4.7. Operating update period with respect to time

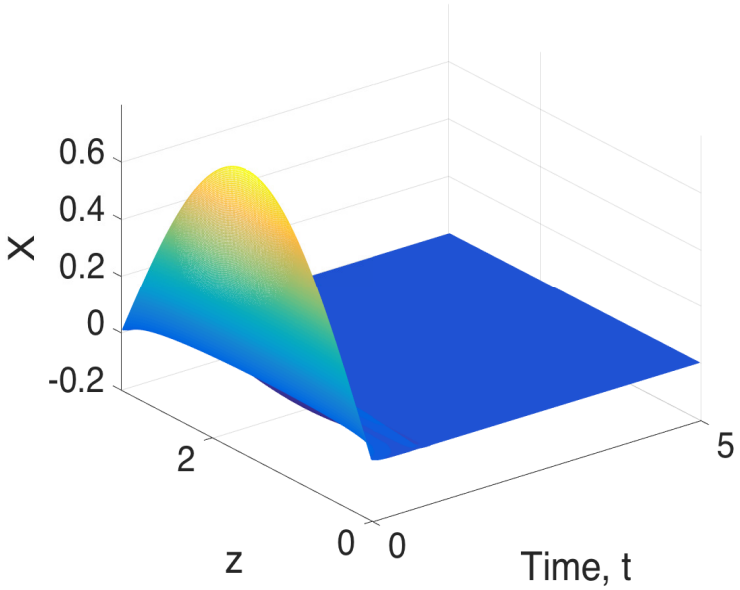

Figure 4.8. Closed-loop state profile

is initiated and the closed-loop system is stabilized. This is represented by the dashed red line. After a sufficient amount of data is collected, parameter re-identification is initiated and the model parameters are updated upon verification of closed-stability. This is represented by the green line.

To visualize the maximum allowable update period, in Figure 4.6 the maximum eigenvalue magnitude of the test matrix M is plotted for the different operating conditions. Values below the unit line represent the stable eigenvalue magnitudes. It can be seen that initially during normal operation the system can be operated up to an update period of $h_{\max} = 0.15$ and maintain closed-loop stability. However, when a drift is introduced the maximum allowable update period drops to $h_{\max} = 0.01$. This means that we need to achieve a higher communication frequency to maintain closed-loop stability which is realized during the safe-parking period. After the model parameters are re-identified we are able to reduce the communication frequency back to its original value while maintaining stability. Figure 4.7 captures the change in update period values for the different operating conditions over the operating time course of the simulated scenario (normal operation, safe-parking and post-identification regimes).

4.5 Conclusions

In this chapter, we presented a methodology for the integration of time-triggered model-based networked control and event-triggered model parameter re-identification for spatially-distributed processes modeled by highly-dissipative PDEs subject to sensor-controller communication constraints and process parametric drift. A monitoring scheme was devised to detect increased plant-model mismatch due to parameter drift using a time-varying instability alarm threshold. A breach of the threshold triggered a safe-parking protocol that aimed to maintain closed-loop stability while allowing the collection of additional input and state data that were used to re-identify the model parameters and minimize the plant-model mismatch. A stability check was then performed for the newly-identified model to determine if the model parameters could be updated and pre-drift operating conditions could be restored. The results were illustrated using a diffusion-

reaction process example.

Chapter 5

Model-Based Event-Triggered Networked Control of Spatially-Distributed Processes with Parameter Re-identification

While the approach in Chapter 4 is successful at improving the achievable level of communication savings by keeping the plant-model mismatch to a minimum, the implementation of this approach requires that a minimum fixed communication rate be established in order to achieve closed-loop stability, and model re-identification is triggered only in response to destabilizing parametric drifts. The use of a fixed communication rate strategy is not always optimal with respect to the achievable savings in network resource utilization. Event-triggered control strategies, on the other hand, can lead to more substantial reductions in network utilization and provide the process with the flexibility needed to adapt to changes in the operating environment.

Motivated by these considerations, in this chapter we present an event-based approach for the integration of model-based control and parameter identification in networked distributed processes subject to sensor-controller communication constraints and process parametric drift. The approach aims to improve the performance and communication constraint-handling capabilities of the networked closed-loop system, and simultaneously address practical implementation issues arising from the uncertainty in process parame-

ter values. The rest of the chapter is organized as follows: following some preliminaries and an overview of the control problem in Section 5.1, a model-based event-triggered controller is designed and analyzed in Section 5.2. The proposed methodology for augmenting model-based event-triggered control with parameter re-identification is then presented in Section 5.3 and the implementation of the developed methodology is illustrated in Section 5.4 using a chemical process example. The results of this chapter were first published in [71].

5.1 Preliminaries

Our focus in this work is on processes that are spatially-distributed and can be modeled by highly-dissipative PDEs with low-order dominant dynamics. A representative example of these systems are those described by parabolic PDEs:

$$\frac{\partial \bar{x}(z,t)}{\partial t} = \bar{\alpha}\frac{\partial^2 \bar{x}(z,t)}{\partial z^2} + \bar{\beta}\bar{x}(z,t) + \bar{\omega}\sum_{i=1}^{m} b_i(z)u_i(t) \tag{5.1}$$

subject to the following boundary and initial conditions:

$$\bar{x}(0,t) = \bar{x}(\pi,t) = 0;\ \bar{x}(z,0) = \bar{x}_0(z) \tag{5.2}$$

where $t \geq 0$ is the time, $0 \leq z \leq \pi$ is the spatial domain, $\bar{x}(z,t) \in \mathbb{R}$ is the state variable, $u_i \in \mathbb{R}$ is the i-th manipulated input, $b_i(z)$ is the spatial distribution function associated with the i-th control actuator and $m > 0$ is the number of manipulated inputs. The process parameters $\bar{\alpha} > 0$, $\bar{\beta}$ and $\bar{\omega}$ are constants and $\bar{x}_0(z)$ is a smooth function of its argument.

An infinite-dimensional state-space formulation of the PDE in (5.1)-(5.2) can be obtained as follows:

$$\dot{x}(t) = \mathcal{A}x(t) + \mathcal{B}u(t),\ x(0) = x_0 \tag{5.3}$$

where $x(t) \in \mathcal{H} = L_2(0,\pi)$ is the state function defined on the Hilbert space of square integrable functions, $\mathcal{A}$ is the spatial differential operator, $\mathcal{B}$ is the input operator which describes the spatial placement of the control actuators and $u \in \mathbb{R}^m$ is the vector of manipulated inputs.

For parabolic PDEs, a key structural characteristic of the spectrum of $\mathcal{A}$ is the large separation that exists between a finite set of slow (possibly unstable) eigenvalues and an infinite set of fast stable eigenvalues (e.g., see [63]), which enables the application of standard modal decomposition techniques to transform the system of (5.3) to the following interconnected subsystems:

$$\begin{aligned} \dot{x}_s(t) &= \mathcal{A}_s x_s(t) + \mathcal{B}_s u(t),\ x_s(0) = \mathcal{P}_s x_0 \\ \dot{x}_f(t) &= \mathcal{A}_f x_f(t) + \mathcal{B}_f u(t),\ x_f(0) = \mathcal{P}_f x_0 \end{aligned} \tag{5.4}$$

where $x_s = \mathcal{P}_s x$ is the state of a finite-dimensional slow subsystem that describes the evolution of the slow eigenmodes, $x_f = \mathcal{P}_f x$ is the state of an infinite-dimensional system that describes the evolution of the fast stable eigenmodes, $\mathcal{A}_s = \mathcal{P}_s\mathcal{A}$, $\mathcal{A}_f = \mathcal{P}_f\mathcal{A}$, $\mathcal{B}_s = \mathcal{P}_s\mathcal{B}$, $\mathcal{B}_f = \mathcal{P}_f\mathcal{B}$; $\mathcal{P}_s$ and $\mathcal{P}_f$ are the orthogonal projection operators for the slow and fast eigenmodes, respectively.

An alternative representation of the above subsystems – and one which is more convenient for practical controller synthesis and analysis – is in terms of the time evolution of the amplitudes of the slow and fast eigenmodes. To this end, let a_i denote the amplitude of the i-th eigenmode (which can be obtained by taking the inner product of the state, $\bar{x}(z,t)$, with the i-th eigenfunction of the differential operator). We define $a_s = [a_1 \cdots a_m]^T \in \mathbb{R}^m$ as the slow state, which contains the amplitudes of the first m slow eigenmodes, and define a_f as the fast state, which contains the amplitudes of the remaining fast stable eigenmodes. The dynamics of the slow and fast subsystems can be described by:

$$\begin{aligned} \dot{a}_s &= A_s a_s + B_s(z_a)u \\ \dot{a}_f &= A_f a_f + B_f(z_a)u \end{aligned} \tag{5.5}$$

where A_s is an $m \times m$ diagonal matrix containing the first m slow eigenvalues of $\mathcal{A}$, B_s is an $m \times m$ input matrix whose elements are parameterized by the control actuator locations, z_a. A_f and B_f are "infinite-dimensional" state and input matrices associated with the fast subsystem, where A_f contains the stable fast eigenvalues of $\mathcal{A}$. Owing to its finite-dimensional nature, only the slow subsystem of (5.5) is used for controller design as explained in the next section.

5.2 Model-Based Control with Event-Triggered Sensor-Controller Communication

In this section, we use the finite-dimensional slow subsystem of (5.5) to design a model-based networked controller that utilizes event-triggered sensor-controller communication and characterize its closed-loop stability properties under full-state measurements. The model-based control strategy involves embedding a finite-dimensional dynamic model of the slow subsystem within the controller and using the model to generate estimates of the slow states which are used to generate the control action for as long as the communication between the controller and the sensor is suspended. When communication is re-established, the model states are updated based on the available sensor-measured values.

We consider a model-based controller of the following form:

$$\begin{aligned} u(t) &= -K\widehat{a}_s(t) \\ \dot{\widehat{a}}_s(t) &= \widehat{A}_s\widehat{a}_s(t) + \widehat{B}_s u(t) \end{aligned} \tag{5.6}$$

where $\widehat{a}_s$ is the state of the model estimating the slow state a_s, $\widehat{A}_s$ and $\widehat{B}_s$ are constant matrices that approximate the matrices of the slow subsystem A_s and B_s, respectively, and satisfy the following bounds:

$$\|\delta_A\| := \|A_s - \widehat{A}_s\| \leq \Delta_A,\ \|\delta_B\| := \|B_s - \widehat{B}_s\| \leq \Delta_B \tag{5.7}$$

where $\Delta_A > 0$ and $\Delta_B > 0$ are bounds on the plant-model mismatch; and K is the feedback gain which is chosen to exponentially stabilize the origin of the closed-loop model. This choice guarantees the existence of a positive-definite symmetric matrix $P = P^T > 0$ that satisfies the following Lyapunov equation:

$$-Q = (\widehat{A}_s - \widehat{B}_s K)^T P + P(\widehat{A}_s - \widehat{B}_s K) \tag{5.8}$$

for some positive-definite symmetric matrix Q, which in turn implies that the time-derivative of the Lyapunov function $V = \widehat{a}_s^T P \widehat{a}_s$ along the trajectories of the closed-loop model satisfies a bound of the form $\dot{V}(\widehat{a}_s) \leq -\alpha \|\widehat{a}_s\|^2$, where $\alpha = \lambda_{min}(Q) > 0$ is the minimum eigenvalue of Q.

To assess the stabilizing capabilities of the controller of (5.6) when implemented on the slow subsystem of (5.5), we consider the Lyapunov function $V(a_s) = a_s^T P a_s$, where P satisfies (5.8), and evaluate the time-derivative of V along the trajectories of the closed-loop slow subsystem as follows:

$$\begin{aligned} \dot{V}(a_s) &= \dot{a}_s^T P a_s + a_s^T P \dot{a}_s \\ &= (A_s a_s - B_s K \widehat{a}_s)^T P a_s + a_s^T P (A_s a_s - B_s K \widehat{a}_s) \end{aligned} \tag{5.9}$$

Introducing the model estimation error, $e = a_s - \widehat{a}_s$, and exploiting (5.8) together with the uncertainty bounds $\|\delta_A\| \leq \Delta_A$ and $\|\delta_B\| \leq \Delta_B$, an upper bound on the time-derivative of V can be obtained as follows:

$$\begin{aligned} \dot{V}(a_s) &\leq -\alpha \|a_s\|^2 + 2\Delta_A \|P\| \|a_s\|^2 \\ &+ 2\|P B_s K\| \|a_s\| \|e\| \\ &= -\frac{\alpha}{2}\|a_s\|^2 - \bar{\gamma}\|a_s\|(\epsilon\|a_s\| - \|e\|) \end{aligned} \tag{5.10}$$

where $\bar{\gamma} = 2(\|P\widehat{B}_s K\| + \Delta_B \|P\| \|K\|) > 0$ and the parameter ϵ is given by:

$$\epsilon = \frac{\alpha - 4\Delta_A \|P\| - 4\Delta_B \|P\| \|K\|}{4(\|P\widehat{B}K\| + \Delta_B \|P\| \|K\|)} \tag{5.11}$$

From (5.10) it can be seen that if the norm of the model estimation error $\|e(t)\|$ satisfies the following bound:

$$\|e(t)\| \leq \epsilon \|a_s(t)\| \ \ \forall t \geq 0, \tag{5.12}$$

where $\alpha > 4\Delta_A \|P\| + 4\Delta_B \|P\| \|K\|$, the time-derivative of V is guaranteed to satisfy $\dot{V}(a_s(t)) \leq (-\alpha/2)\|a_s(t)\|^2$ for all $t \geq 0$, which ensures exponential closed-loop stability. The right-hand side in (5.12) can be viewed as a time-varying alarm threshold that the model estimation error needs to remain below to ensure closed-loop stability. This condition can therefore be used to devise an event-triggered control strategy whereby the controller uses the model state (i.e., sensor-controller communication is suspended) as long as (5.12) is satisfied, and when (5.12) is violated the model state gets updated (i.e., sensor-controller communication is permitted) to reset the model estimation error to zero

and ensure the negative-definiteness of $\dot{V}$. This event-based update logic can be formally described as follows:

$$\|e(t_k)\| > \epsilon \|a_s(t_k)\| \Longrightarrow \widehat{a}_s(t_k) = a_s(t_k)\ , k \in \{0, 1, 2, \cdots\} \tag{5.13}$$

where t_k is the k-th update time.

Remark 5.1. *Note that the implementation of this strategy requires monitoring the evolution of the model estimation error over time to determine if (and when) it breaches the time-varying alarm threshold. Under full-state feedback (where measurements of the slow states are assumed to be available), the event-triggered control strategy can be implemented directly since the availability of the slow states as measurements enables the determination of both the estimation error and the communication-triggering threshold.*

Remark 5.2. *The frequency of model state updates (which is an indicator of the extent of sensor-controller communication) is dictated by how often the time-varying alarm threshold in* (5.12) *is breached. This is determined in part by the size of the threshold coefficient, ϵ, where a large value of ϵ makes it less likely for the threshold to be breached (leading to smaller communication frequency) whereas a small value of ϵ makes it more likely that a breach may take place (leading to higher communication frequency). In this sense, the threshold coefficient may be used as an (indirect) measure of the extent of network utilization. The advantage of using the threshold coefficient as such is that, as shown in* (5.11)*, it is explicitly parameterized by the model and controller design parameters and can therefore be computed and optimized a priori (prior to controller implementation).*

Remark 5.3. *The expression in* (5.11) *provides an explicit characterization of the dependence of the communication-triggering threshold coefficient, ϵ, on the size of the plant-model mismatch, which is reflected in the uncertainty bounds Δ_A and Δ_B. As these bounds get smaller, ϵ gets larger and less frequent model updates are triggered. However, it is crucial to note that it is possible for ϵ to become negative if the uncertainty bounds are too large. In such cases, the event-triggered communication strategy becomes infeasible, leading to continuous communication with no a priori guarantees of closed-loop stability. To mitigate this effect, we require that α be large enough such that $\alpha >$*

$4\Delta_A\|P\| + 4\Delta_B\|P\|\|K\|$. *In the presence of large uncertainty, this requirement can be fulfilled by increasing* α *appropriately. As a consequence of doing so, the controller will become more aggressive in order to compensate for the large model uncertainty. This requirement may also be viewed as an implicit constraint on the size of the uncertainty that the system can tolerate, in the sense that* ϵ *will be positive if the uncertainty is small enough. While* α *can be used to try to mitigate the effect of the uncertainty, ultimately there is a limit on how large the uncertainty bounds can be without jeopardizing closed-loop stability.*

Remark 5.4. *The event-triggered control strategy described in this section guarantees exponential stability not only for the closed-loop slow subsystem of* (5.5)*, but also for the infinite-dimensional system of* (5.5)*. Specifically, under the model state update law of* (5.13)*, the slow state is guaranteed to converge exponentially to zero owing to the fact that* $\dot{V}(a_s) \leq (-\alpha/2)\|a_s\|^2$*. Furthermore, the fact that the fast operator,* A_f*, in* (5.5) *is itself stable, together with the fact that the model state* $\widehat{a}_s$ *(which drives the fast subsystem through the control input) is exponentially stable by controller design, implies that the fast state is also exponentially stable.*

5.3 Integrating Event-Based Parameter Identification and Model-Based Control

In this section, we present a methodology for integrating the event-triggered model-based networked control approach discussed in Section 5.2 with an event-based parameter re-identification scheme to mitigate the impact of process parametric drift on both closed-loop stability and the extent of network utilization. We first present an overview of the proposed integration methodology followed by a discussion of the key implementation steps. Figure 5.1 summarizes the proposed approach.

5.3.1 Overview of the proposed methodology

Initially, the networked control system is operated at a suitably chosen communication-triggering threshold coefficient, ϵ, that minimizes communication while ensuring closed-loop stability. Under full-state feedback the evolution of the closed-loop slow state is

continuously monitored and the model estimation error is checked against the threshold in (5.12) to determine when the model states need to be updated. This event-triggered communication strategy is highlighted in green in Figure 5.1.

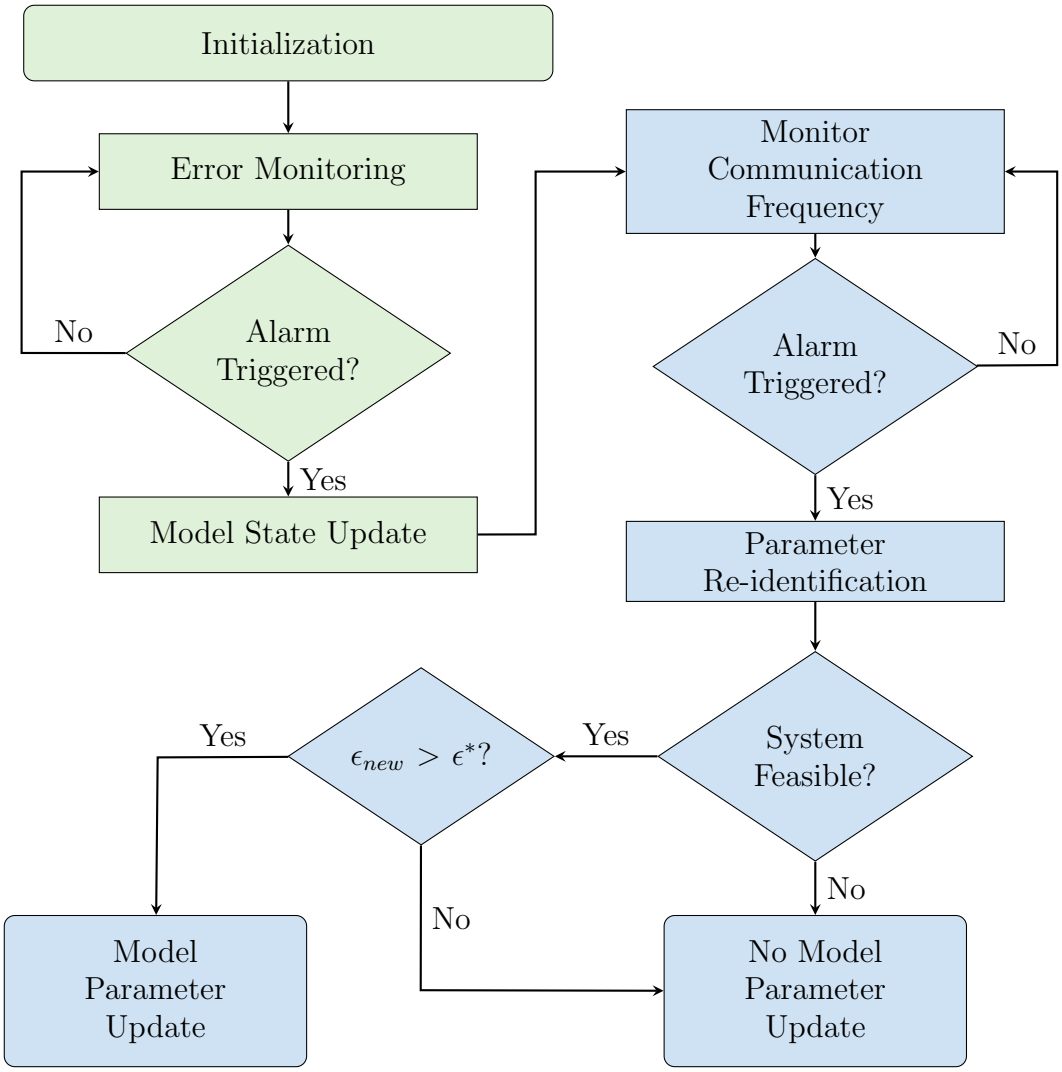

Figure 5.1. Algorithm of integrating parameter identification and event-triggered model-based control

To detect process parametric drift and determine when model parameters need to be re-identified, the frequency of sensor-controller communication over the network is continuously monitored. When a sustained increase in the communication frequency is observed, parameter identification is triggered and new model parameters are identified. A new communication-triggering threshold coefficient value, ϵ, is calculated based on the

newly identified model parameters and is checked for feasibility. A desirable ϵ value is one that results in a lower communication frequency. When this is achieved, the model parameters are updated. This event-triggered parameter identification strategy is highlighted in blue in Figure 5.1.

5.3.2 Communication frequency monitoring

When model state updates occur, the sensor-controller communication frequency is continuously monitored in order to detect possible process parametric drift. As the process parameters drift, the plant-model mismatch increases and the event-triggered control strategy compensates for the increased mismatch by increasing the sensor-controller communication to stabilize the closed-loop system (note that closed-loop stability is guaranteed as long as the drift does not exceed the uncertainty bounds Δ_A and Δ_B). To detect this increase in the communication frequency, we introduce the following indicator function:

$$\delta_i = \begin{cases} 1\ , & (t_i - t_{i-1}) < (t_{i-1} - t_{i-2}) \\ 0\ , & \text{otherwise} \end{cases} \tag{5.14}$$

where t_i is the i-th update time. This function takes a value of 1 whenever the time interval between the most recent consecutive update instances, $(t_i - t_{i-1})$, is smaller than interval between the previous two update instances, $(t_{i-1} - t_{i-2})$. While this decrease in the update period can be taken as a sign of potential increase in communication, this increase by itself is not sufficient to indicate the need for model re-identification because under normal event-triggered control the update period is expected to be non-uniform to begin with as there can be short-lived (transient) increases and decreases in the communication frequency. What the monitoring scheme, therefore, should be designed to look for is a sustained increase in the communication frequency. To this end, we consider comparing the sizes of consecutive update intervals over some horizon, N, and add up the values of δ_i over that horizon. The resulting sum at any given time indicates the number of instances that the update period decreased over the horizon. The need for parameter re-identification is then triggered if the sum exceeds a certain pre-defined threshold, i.e., $\sum_{i=k}^{k-N} \delta_i > \gamma'$.

Remark 5.5. *The choice of the horizon length, N, and the update frequency threshold magnitude, γ', in* (5.14) *influence the accuracy of the parametric drift detection algorithm. A longer horizon or a larger threshold value could make the algorithm indifferent to communication frequency changes, whereas a short horizon or a smaller threshold value could make the algorithm very sensitive to minor frequency changes which could lead to unnecessary parameter identification events. The choice of these parameters requires some experience and knowledge of the system being controlled, and can also serve as tuning parameters to strike the desired balance between the describable level of control performance and the frequency of parameter re-identification events.*

Remark 5.6. *It should be noted that unlike the algorithm proposed in [52] for the case of periodic sensor-controller communication, no safe-parking of the process following the parametric drift is needed in the current approach. In [52], the drift was detected when the state became unstable. Since the communication period was held constant, safe-parking was utilized to temporarily stabilize the system immediately following the drift by either increasing the communication frequency or by adjusting the actuator location. Realization of the safe-parking step requires a search for the stabilizing update periods and/or the feasible actuator locations. By contrast, the use of event-triggered control in the current work eliminates the need for safe-parking as the sensor-controller communication frequency is automatically adjusted by the control system in response to the parametric drift.*

5.3.3 Model parameter re-identification

Once the parametric drift is detected as described in the previous subsection, new model parameters need to be identified. The choice of the model re-identification method depends on the particular application. In this study, subspace identification techniques for finite-dimensional discrete linear time-invariant systems are used to identify new model parameters, $\bar{A}_s$ and $\bar{B}_s$, based on the input and state data (under full-state feedback). The aim of parameter re-identification is to minimize the plant-model mismatch. For most identification methods, the quality of the model fit is based on how well the newly identified parameters are consistent with the data, and the quality of the results are dependent on the choice of the re-identification technique. For example, grey box estimation

tools take advantage of prior knowledge of some system parameters which enables it to identify the drifted parameters more accurately. On the other hand, black box estimation tools do not require prior knowledge of system parameters, but could result in fits that are less accurate.

5.3.4 Feasibility check and model parameter updates

Once new model parameters are identified using the subspace identification method as discussed in Section 5.3.3, a decision has to be made in terms of whether to actually update the model parameters or not. A key consideration in making this decision is whether the newly identified parameters achieve a desirable post-drift level of sensor-controller communication. Note that standard parameter identification methods do not in general account for sensor-controller communication considerations, and the newly identified parameters are typically based solely on how well they are represented by the data. This, however, does not guarantee that the desired communication requirements will be met. Recall also that the communication-triggering threshold ϵ can be used as a measure of the communication frequency (see Remark 5.2), where a large value usually indicates less frequent updates. In light of this, the new model parameters need to be analyzed to determine what the new ϵ value will be before a decision on whether the new parameters should be updated or not is made. Note that it is possible for the newly calculated ϵ value to be less than the original value obtained prior to the drift, resulting in more frequent communication. In this case, the decision to update the model parameters hinges on the whether the decrease in ϵ is acceptable or not. For example, if we denote by ϵ^* the minimum value of the communication-triggering threshold coefficient that is deemed appropriate for the desired communication level, the model parameters can be updated as long as the newly obtained ϵ is greater than ϵ^*, i.e.,

$$\epsilon_{\text{new}}(t_u) \geq \epsilon^* \Longrightarrow \widehat{A}_s(t_u) \;=\; \bar{A}, \;\; \widehat{B}_s(t_u) \;=\; \bar{B}, \tag{5.15}$$

where t_u is the time at which the model parameter updating takes place. At a minimum, the new ϵ has to be greater than zero to ensure that the event-triggered update strategy continues to be feasible.

5.4 Simulation Example

In this section, we demonstrate the application of the proposed methodology using a simulated diffusion-reaction process with the following dynamics:

$$\frac{\partial \bar{x}(z,t)}{\partial t} = \frac{\partial^2 \bar{x}(z,t)}{\partial z^2} + [\beta_T \hat{\gamma} e^{-\hat{\gamma}} - \beta_U]\bar{x}(z,t) + \beta_U \sum_{i=1}^{2} b_i(z) u_i(t) \tag{5.16}$$

subject to the initial and boundary conditions in (5.2). For this system with $u_1 = u_2 = 0$, the spatially-uniform steady-state, $\bar{x}(z,t) = 0$, can be shown to be unstable for the following choice of process parameter values: $\beta_T = 120$, $\hat{\gamma} = 4$ and $\beta_U = 2.0$. The control objective is to stabilize the state profile at the open-loop unstable steady-state using two point-control actuators (with finite-support) with minimal sensor-controller communication, subject to process parametric variations. The variations in process parameters are simulated by a drift in the heat of reaction, β_U.

Based on the solution to the eigenvalue problem associated with the differential operator of above diffusion-reaction PDE, it can be shown that only the first two eigenvalues are unstable. Therefore, we consider the first two unstable eigenvalues to be the dominant ones and apply modal decomposition to obtain the following slow subsystem:

$$\dot{a}_s = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} a_s + \omega \begin{bmatrix} \phi_1(z_{a1}) & \phi_1(z_{a2}) \\ \phi_2(z_{a1}) & \phi_2(z_{a2}) \end{bmatrix} u$$

where $\lambda_i = \bar{\beta} - i^2$ is the i-th eigenvalue, $\omega = 2$, $\phi_i(z) = \sqrt{2/\pi}\ \sin(iz)$ is the i-th eigenfunction, $z_{a1} = \pi/4$ is the location of the first actuator and $z_{a2} = \pi/2$ is the location of the second actuator. We initially consider a finite-dimensional model with

$$\widehat{A}_s = \begin{bmatrix} 5.792 & 0 \\ 0 & 2.792 \end{bmatrix}, \quad \widehat{B}_s = \begin{bmatrix} 1.128 & 1.596 \\ 1.596 & 0 \end{bmatrix}$$

and set the controller gain at:

$$K = \begin{bmatrix} 0 & -14.3 \\ -9.9 & 10.1 \end{bmatrix}$$

in order to place the poles of the closed-loop model at $[-10\ -20]$. With these settings the communication-triggering threshold coefficient was determined to be $\epsilon = 0.1248$. In the remainder of this section, we explore the implementation of the proposed approach under parametric drift. In this scenario, the control and identification approaches are based on the finite-dimensional model, but are applied to a sufficiently high-order Galerkin discretization of the PDE.

Parametric drift is introduced in the heat of reaction coefficient, β_U, at time $t = 24$ min. Figures 5.2 - 5.8 show the results of the simulation. After the drift occurs, the plant-model mismatch increases, and the event-triggered communication strategy compensates for this by increasing the sensor-controller communication in order to stabilize the system. This can be seen in the continuous red band of update instances in Figure 5.4. Figure 5.6 shows how the resulting sustained increase in communication is detected by the frequency monitoring algorithm.

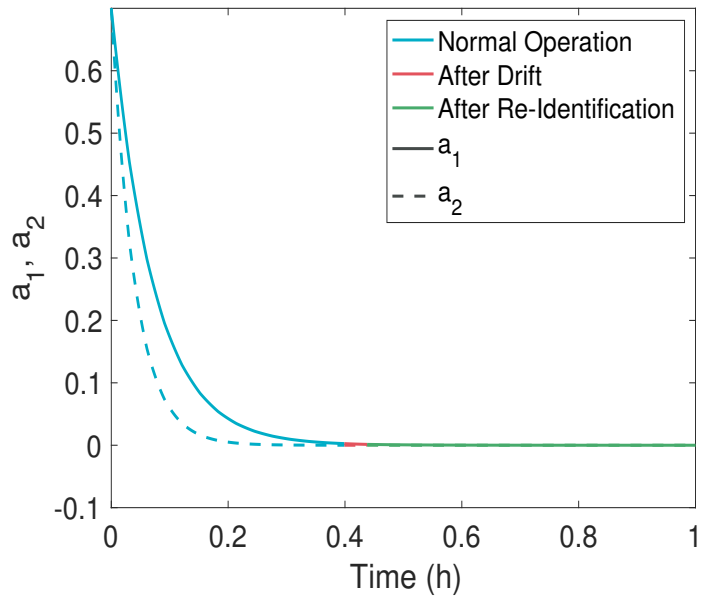

Figure 5.2. Dominant eigenmode profile for the closed-loop system under the integrated event-triggered feedback control and event-based parametric re-identification approach

A horizon value of $N = 50$ and a frequency threshold value of $\gamma' = 40$ are utilized in the communication frequency monitoring scheme. In this figure, the cumulative sum of the indicator function values over the chosen horizon is plotted at each time instance, and it

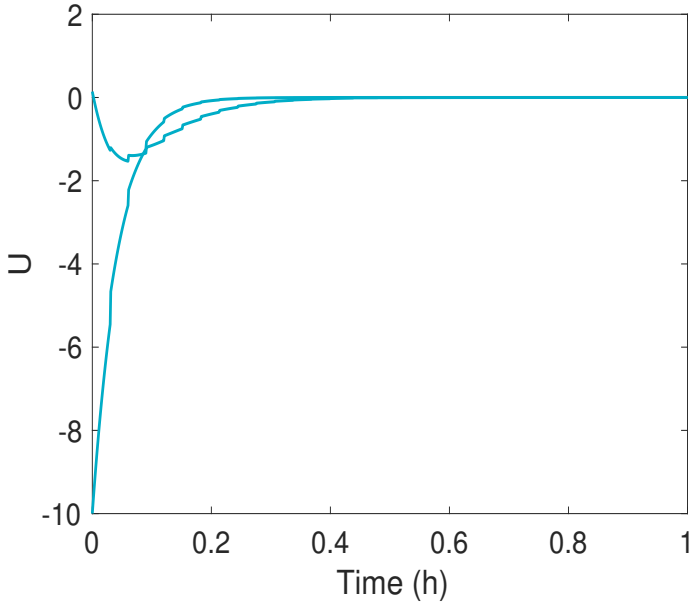

Figure 5.3. Manipulated input profile for the closed-loop system under the integrated event-triggered feedback control and event-based parametric re-identification approach

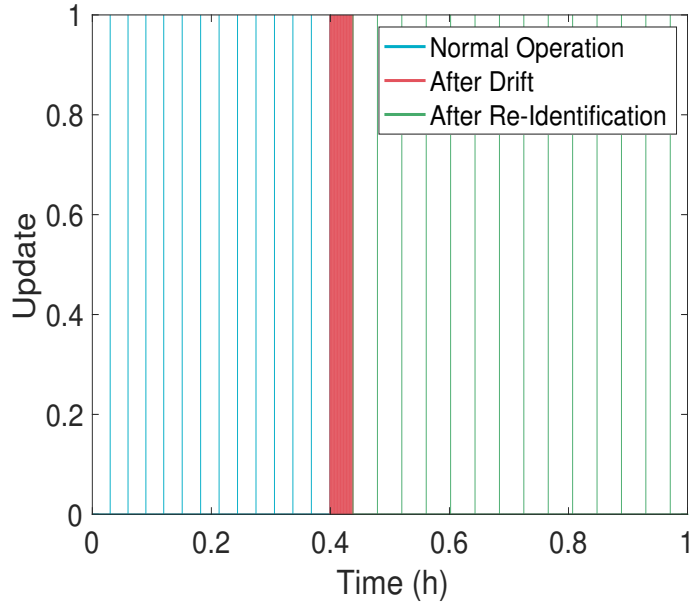

Figure 5.4. Update instances vs. time

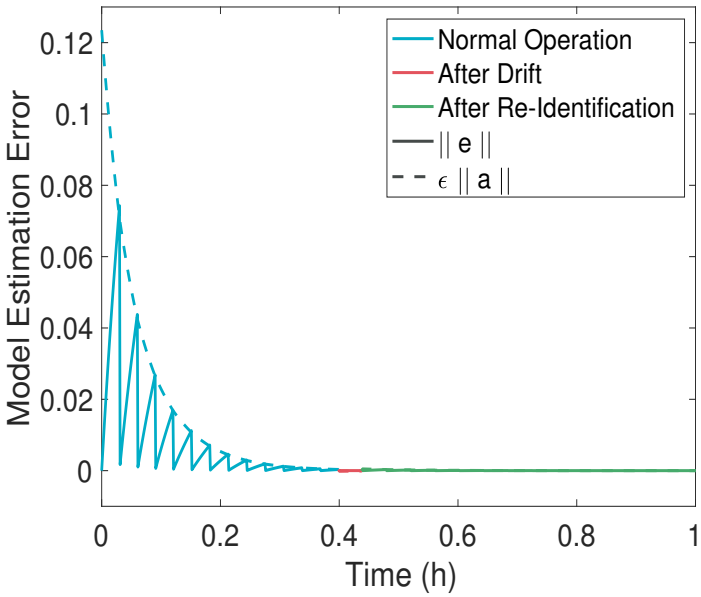

Figure 5.5. Model estimation error relative to the time-varying threshold

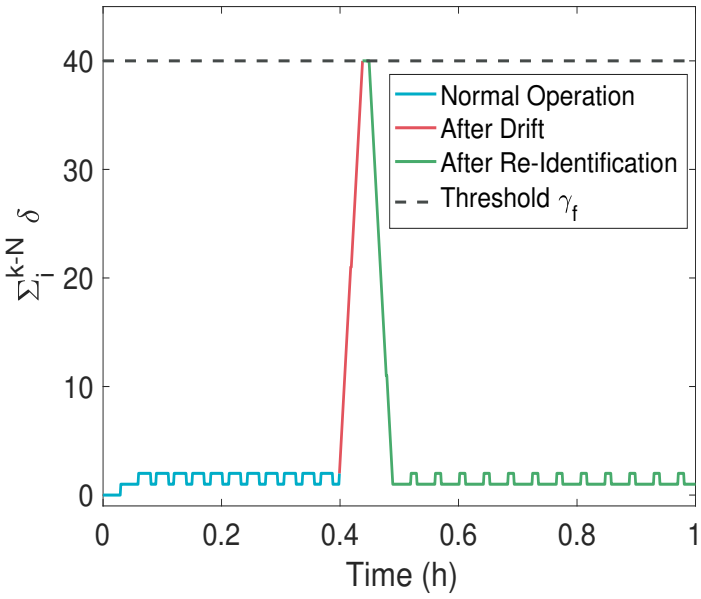

Figure 5.6. The accumulated sum of the indicator function values over the horizon at each time instance

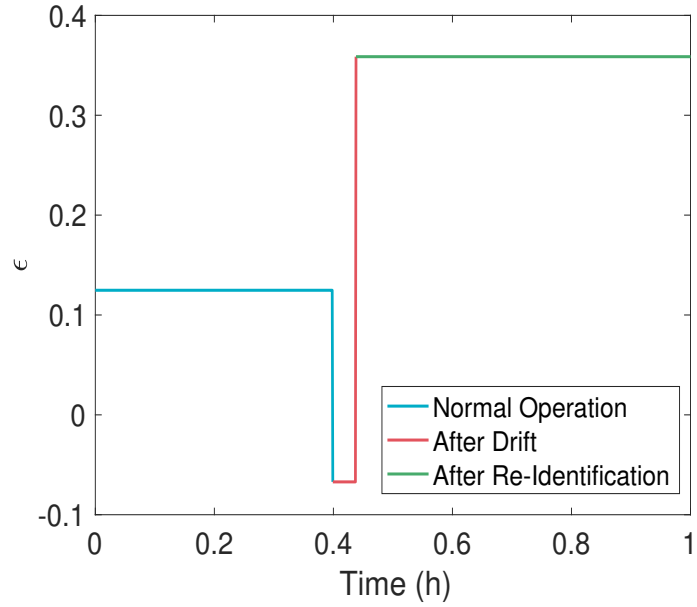

Figure 5.7. Model estimation error threshold value

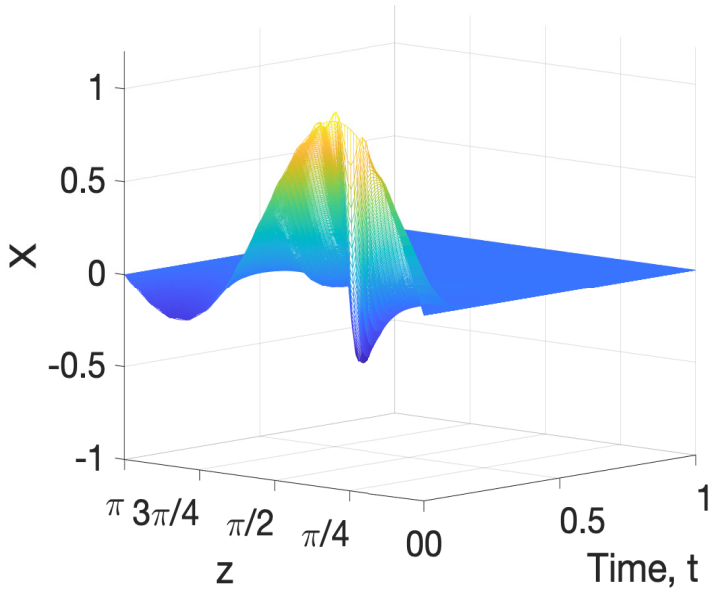

Figure 5.8. Closed-loop state profile

can be seen that at $t = 26$ min the sum breaches the specified threshold, indicating that a drift has occurred and triggering parameter re-identification. Using subspace identification techniques, the following model system and input matrices are identified:

$$\bar{A} = \begin{bmatrix} 0.238 & 0 \\ 0 & -1.000 \end{bmatrix}, \quad \bar{B} = \begin{bmatrix} 1.128 & 1.596 \\ 1.596 & 0 \end{bmatrix}$$

Based on the new model, a new communication-triggering threshold coefficient value is computed and found to be $\epsilon_{new} = 0.3512$. Since this value exceeds the pre-drift threshold coefficient value, the model parameters are updated. This is represented by the green color in Figures 5.2 - 5.8. It can be seen from Figure 5.4 that by identifying new model parameters and minimizing the model estimation error along with increasing the model estimation error threshold coefficient, the overall post-update communication rate is decreased compared to the nominal (pre-drift) operating conditions (blue region) and the operating conditions during drift (red region).

5.5 Conclusions

In this chapter, we presented a methodology for the integration of time-triggered model-based networked control and event-triggered model parameter re-identification for spatially-distributed processes modeled by highly-dissipative PDEs subject to sensor-controller communication constraints and process parametric drift. A monitoring scheme was devised to detect increased plant-model mismatch due to parameter drift using a time-varying instability alarm threshold. A breach of the threshold triggered a safe-parking protocol that aimed to maintain closed-loop stability while allowing the collection of additional input and state data that were used to re-identify the model parameters and minimize the plant-model mismatch. A stability check was then performed for the newly-identified model to determine if the model parameters could be updated and pre-drift operating conditions could be restored. The development and implementation of the proposed framework were demonstrated using a representative diffusion-reaction process example. A key point of investigation in the simulation example has been the tradeoff between the achievable closed-loop performance and the extent of network utilization re-

alized by different safe-parking approaches. It was shown that safe-parking by adjusting the update period yields better performance but comes at the expense of increased communication cost, where as safe-parking via actuator re-location avoids the communication cost increase but could cause performance deterioration.

Chapter 6

Output Feedback Event-Triggered Networked Control of Spatially-Distributed Processes with Parameter Re-Identification

The objective of this chapter is to address the limitations imposed by the lack of full-state measurements on the implementation of the integrated control and identification approach developed earlier in the context of event-based networked control of distributed processes. In practical applications, full-state measurements are not always available, especially in the case of spatially-distributed systems where it might be difficult to measure the entire spatial profile of a given state variable (as this requires a large number of measurement sensors), or to measure certain physical variables in real-time due to physical or technological constraints (e.g., species concentrations). In this chapter, we focus on systems described by highly-dissipative PDEs subject to parametric drift and a limited number of measured outputs, and address the problem on the basis of a suitable reduced-order model that captures the slow dynamics of the infinite-dimensional system. The rest of the chapter is organized as follows: Section 6.1 introduces some preliminaries that define the scope of the current study. The event-based model-based control strategy is presented in Section 6.2, where the state feedback design is initially reviewed and then augmented with a state estimation scheme to generate estimates of the slow states

using the available output measurements. Taking the state estimation error into account, the state feedback-based communication-triggering criterion is modified to allow the implementation of the event-triggered communication strategy based on the available state estimates. The methodological framework for the integration of event-triggered parameter re-identification and updates into the developed event-triggered output feedback control strategy is the described in Section 6.3. Finally, an illustrative simulation case study is presented in Section 6.4. The results of this chapter were first published in [72].

6.1 Preliminaries

In this study we consider a class of spatially-distributed processes described by highly-dissipative PDEs that lend themselves to finite-dimensional control techniques owing to their low-order dominant dynamics. An example are systems modeled by parabolic PDEs with distributed control actuators and measurement sensors as follows:

$$\begin{aligned} \frac{\partial \bar{x}(z,t)}{\partial t} &= \bar{\alpha}\frac{\partial^2 \bar{x}(z,t)}{\partial z^2} + \bar{\beta}\bar{x}(z,t) + \bar{\omega}\sum_{i=1}^{m} b_i(z)u_i(t) \\ y_i(t) &= \int_0^{\pi} q_i(z)\bar{x}(z,t)dz, \quad i \in \{1,2,\cdots,l\} \end{aligned} \tag{6.1}$$

subject to the following boundary and initial conditions:

$$\bar{x}(0,t) = \bar{x}(\pi,t) = 0;\ \bar{x}(z,0) = \bar{x}_0(z) \tag{6.2}$$

where $t \geq 0$ is the time, $0 \leq z \leq \pi$ is the spatial domain, $\bar{x}(z,t) \in \mathbb{R}$ is the state variable, $u_i \in \mathbb{R}$ is the i-th manipulated input, $b_i(z)$ is the spatial distribution function associated with the i-th control actuator, and $m > 0$ is the number of manipulated inputs. The constants $\bar{\alpha} > 0$, $\bar{\beta}$, and $\bar{\omega}$ are process parameters, $y_i(t) \in \mathbb{R}$ is the i-th measured output, $q_i(z)$ is the spatial distribution function associated with the i-th measurement sensor, $l > 0$ is the number of measurement sensors, and $\bar{x}_0(\cdot)$ is a smooth function of its argument.

An infinite-dimensional state-space formulation of the PDE in (6.1)-(6.2) can be obtained as follows:

$$\begin{aligned} \dot{x}(t) &= \mathcal{A}x(t) + \mathcal{B}u(t),\ x(0) = x_0 \\ y(t) &= \mathcal{Q}x(t) \end{aligned} \tag{6.3}$$

where $x(t) \in \mathcal{H} = L_2(0, \pi)$ is the state function defined on the Hilbert space of square integrable functions, $\mathcal{A}$ is the spatial differential operator, $\mathcal{B}$ is the input operator which dictates the spatial placement of the control actuators, $u \in \mathbb{R}^m$ is the vector of manipulated inputs, $y \in \mathbb{R}^l$ is the vector of measured outputs, and $\mathcal{Q}$ is the output (or measurement) operator which dictates the spatial placement of the measurement sensors.

An integral structural characteristic of highly-dissipative PDEs, such as the one described above, is the existence of a large separation in the spectrum of $\mathcal{A}$ between a finite set of slow (possibly unstable) eigenvalues and an infinite set of fast stable eigenvalues (e.g., see [63, 65–67, 69]). This characteristic enables the application of standard modal decomposition techniques to transform the system of (6.3) to the following interconnected subsystems:

$$\begin{aligned} \dot{x}_s(t) &= \mathcal{A}_s x_s(t) + \mathcal{B}_s u(t),\ x_s(0) = \mathcal{P}_s x_0 \\ \dot{x}_f(t) &= \mathcal{A}_f x_f(t) + \mathcal{B}_f u(t),\ x_f(0) = \mathcal{P}_f x_0 \end{aligned} \tag{6.4}$$

$$y(t) = \mathcal{Q}_s x_s(t) + \mathcal{Q}_f x_f(t) \tag{6.5}$$

where $x_s = \mathcal{P}_s x$ is the state of a finite-dimensional slow subsystem that describes the evolution of the slow eigenmodes, $x_f = \mathcal{P}_f x$ is the state of an infinite-dimensional system that describes the evolution of the fast stable eigenmodes, $\mathcal{A}_s = \mathcal{P}_s \mathcal{A}$, $\mathcal{A}_f = \mathcal{P}_f \mathcal{A}$, $\mathcal{B}_s = \mathcal{P}_s \mathcal{B}$, $\mathcal{B}_f = \mathcal{P}_f \mathcal{B}$; $\mathcal{P}_s$ and $\mathcal{P}_f$ are the orthogonal projection operators for the slow and fast eigenmodes, respectively; and $\mathcal{Q}_s$ and $\mathcal{Q}_f$ are the measurement operators associated with the slow and fast subsystems, respectively.

The above subsystems can be represented in a more convenient form, in terms of the time evolution of the amplitudes of the slow and fast eigenmodes, for the purpose of practical controller synthesis and analysis. To that end, we define the vector of slow eigenmodes as $a_s = [a_1 \cdots a_m]^T \in \mathbb{R}^m$ (where a_i denotes the amplitude of the i-th eigenmode and can be obtained by taking the inner product of the state, $\bar{x}(z, t)$, with the i-th eigenfunction of the differential operator), and define a_f as the vector of the remaining fast eigenmodes. The dynamics of the slow and fast subsystems can now be described by:

$$\dot{a}_s = A_s a_s + B_s(z_a) u,\ y_s = Q_s(z_s) a_s \tag{6.6}$$

$$\dot{a}_f = A_f a_f + B_f(z_a) u,\ y_f = Q_f(z_s) a_f \tag{6.7}$$

$$y = y_s + y_f \tag{6.8}$$

where A_s is an $m \times m$ diagonal matrix containing the first m slow eigenvalues of $\mathcal{A}$, B_s is an $m \times m$ input matrix whose elements are parameterized by the spatial locations of the control actuators, z_a, and Q_s is an $l \times m$ output matrix with its elements parameterized by the spatial locations of the measurement sensors; A_f, B_f and Q_f are the "infinite-dimensional" state, input and output operators associated with the fast subsystem, where A_f contains the stable fast eigenvalues of $\mathcal{A}$. The outputs of the slow and fast subsystems are denoted by y_s and y_f, respectively. As will be discusses in the next section, only the slow subsystem of (6.6) is used for controller design owing to its finite-dimensional nature.

Remark 6.1. *It should be noted that while the example of a single parabolic PDE subject to Dririchlet boundary conditions is used for motivation and illustration purposes, the results of this work are not limited to this class of systems. The results apply to parabolic PDEs with other types of boundary conditions (including Neumann and mixed-type), as well as systems of parabolic and other highly-dissipative PDEs for which the spectral separation property discussed above holds and the decomposition of* (6.4) *can be written.*

6.2 Design and Implementation of Event-Triggered Model-Based Feedback Control

In this section we discuss the design and implementation of a model-based networked control strategy that utilizes event-triggered sensor-controller communication and characterize the closed-loop stability properties under both full and incomplete state measurements, all based on the finite-dimensional slow subsystem of (6.6). The model-based control strategy requires that a finite-dimensional dynamic model of the slow subsystem be embedded within the controller. Utilizing this model, estimates of the slow states used to generate the control action when the communication between the controller and the sensor is suspended are generated, and when communication is re-established, the model states are updated using the available sensor-measured values. In the remainder of this section, we first recall the state feedback result for this type of control and then extend these results to the case when only output measurements are available.

6.2.1 Implementation using full-state measurements

Following [71], a model-based controller of the following form is considered:

$$\begin{aligned} u(t) &= -K\widehat{a}_s(t) \\ \dot{\widehat{a}}_s(t) &= \widehat{A}_s\widehat{a}_s(t) + \widehat{B}_s(z_a)u(t) \end{aligned} \tag{6.9}$$

where $\widehat{a}_s$ is the state of the model estimating the slow state a_s; $\widehat{A}_s$ and $\widehat{B}_s$ are constant matrices that approximate the matrices of the slow subsystem A_s and B_s, respectively, and satisfy the following bounds:

$$\|\delta_A\| := \|A_s - \widehat{A}_s\| \leq \Delta_A,\ \|\delta_B\| := \|B_s - \widehat{B}_s\| \leq \Delta_B$$

where $\Delta_A > 0$ and $\Delta_B > 0$ are bounds on the parametric model uncertainty; and K is the feedback gain which is chosen to exponentially stabilize the origin of the closed-loop model. This choice guarantees the existence of a positive-definite symmetric matrix $P = P^T > 0$ that satisfies the following Lyapunov equation:

$$-Q = (\widehat{A}_s - \widehat{B}_sK)^TP + P(\widehat{A}_s - \widehat{B}_sK) \tag{6.10}$$

for some positive-definite symmetric matrix Q. This, in turn, implies that the time-derivative of the Lyapunov function $V = \widehat{a}_s^TP\widehat{a}_s$ along the trajectories of the closed-loop model satisfies $\dot{V}(\widehat{a}_s) \leq -\alpha\|\widehat{a}_s\|^2$, where $\alpha = \lambda_{min}(Q) > 0$ is the minimum eigenvalue of Q.

To map the stabilizing capabilities of the controller of (6.9) when implemented on the slow subsystem of (6.6), the Lyapunov function candidate $V(a_s) = a_s^TPa_s$, where P satisfies (6.10), is considered and its time-derivative along the trajectories of the closed-loop slow subsystem is evaluated.

By doing this, it can be shown that if the norm of the model estimation error $\|e(t)\|$, where $e(t) = a_s(t) - \widehat{a}_s(t)$, satisfies the following bound:

$$\|e(t)\| \leq \epsilon\|a_s(t)\|,\ \forall t \geq 0, \tag{6.11}$$

where

$$\epsilon = \frac{\alpha - 4\Delta_A\|P\| - 4\Delta_B\|P\|\|K\|}{4(\|P\widehat{B}K\| + \Delta_B\|P\|\|K\|)}, \tag{6.12}$$

and $\alpha > 4\Delta_A\|P\| + 4\Delta_B\|P\|\|K\|$, the time-derivative of V is guaranteed to satisfy $\dot{V}(a_s(t)) \leq (-\alpha/2)\|a_s(t)\|^2$ for all $t \geq 0$, which ensures exponential closed-loop stability for both the slow subsystem and the infinite-dimensional system (the latter is due to the fact the the fast subsystem itself is stable and driven by a bounded and converging input).

The right-hand side of the inequality in (6.11) represents a state-dependent threshold that the model estimation error needs to respect to ensure closed-loop stability. Using this condition we can devise an event-triggered control strategy where the controller uses the model state during times when communication is suspended as long as (6.11) is satisfied. When (6.11) is violated the model state gets updated through the network to reset the model estimation error to zero and ensure the negative-definiteness of $\dot{V}$. This event-based update logic can be formally described as follows. Let $t_k > 0$ be a time instant such that the estimation error breaches the threshold:

$$\|e(t_k)\| > \epsilon\|a_s(t_k)\| \tag{6.13}$$

Then resetting the model state according to the following update law:

$$\widehat{a}_s(t_k) = a_s(t_k) \tag{6.14}$$

resets the estimation error to zero and ensures exponential closed-loop stability (for both the finite-dimensional slow subsystem and the infinite-dimensional system).

6.2.2 Implementation using limited state measurements

When only a finite number of measured outputs are available, the event-triggered control strategy discussed in the previous subsection cannot be implemented directly due to the unavailability of the slow state measurements, and must therefore be modified. One way to address this problem is to generate estimates of the slow states from the available outputs and use those estimates to redesign and implement the model state update logic. To this end, we consider as an illustration the case when the measurement operator is invertible (or pseudo-invertible in the case of a non-square system). This requirement can typically be achieved by the appropriate choice of the number and location of the sensors. An estimate of the slow state can be generated as follows:

$$\tilde{a}_s(t) \;=\; Q_s^{-1}(z_s)y(t) \tag{6.15}$$

where $\tilde{a}_s$ is an estimate of a_s. Since the output reflects the contributions of both the slow and fast states, the above estimation approach introduces an error that needs to be accounted for in the design and implementation of the sensor-controller communication logic. To illustrate how this error can be accounted for, let us first consider a "certainty-equivalence" approach whereby the slow state in the state feedback design, a_s, is simply replaced by the estimate, $\tilde{a}_s$. Recall that closed-loop stability under full-state feedback is guaranteed as long as (6.11) is satisfied. What is needed here is an alternative condition, expressed in terms of $\tilde{a}_s$, whose satisfaction suffices to guarantees that (6.11) is satisfied. Let us define the new error variable $\tilde{e} = \tilde{a}_s - \widehat{a}_s$, and analyze the consequences of requiring:

$$\|\tilde{e}(t)\| \;\leq\; \epsilon\|\tilde{a}_s(t)\| \tag{6.16}$$

Using (6.15), and the definition of $\tilde{e}$, it can be verified that:

$$\begin{aligned} \|\tilde{e}(t)\| &= \|Q_s^{-1}y(t) - \widehat{a}_s(t)\| \\ &\geq \|a_s(t) - \widehat{a}_s(t)\| - \|Q_s^{-1}Q_f a_f(t)\| \\ &= \|e(t)\| - \|Q_s^{-1}Q_f a_f(t)\| \end{aligned} \tag{6.17}$$

Therefore, if (6.16) holds, we will have:

$$\|e(t)\| \;\leq\; \epsilon\|a_s(t)\| + (\epsilon+1)\|Q_s^{-1}Q_f a_f(t)\| \tag{6.18}$$

Comparing this result with the condition in (6.11), it is clear that satisfaction of (6.16) does not ensure satisfaction of (6.11) due to the additional term on the right-hand side of the inequality in (6.16), which is related to the error introduced in the estimation of the slow state. To ensure that (6.11) is satisfied, one idea is to tighten the upper bound in (6.16) by subtracting the additional term containing the contribution of the fast states. Specifically it can be shown that if the following condition is satisfied:

$$\|\tilde{e}(t)\| \;\leq\; \epsilon\|\tilde{a}_s(t)\| - (\epsilon+1)\|Q_s^{-1}Q_f a_f(t)\| \tag{6.19}$$

then (6.11) is also satisfied and closed-loop stability is guaranteed (provided that the right-hand side in (6.19) is positive; see Remark 6.2 below). To eliminate the need for the

fast states in evaluating the new time-varying alarm threshold in (6.19), we can exploit the stability properties of the fast subsystem which, together with the boundedness of the input and measurement operators, implies the existence of positive real constants, γ_1, γ_2, and β, such that

$$\|Q_s^{-1}Q_f a_f(t)\| \leq \gamma_1 e^{-\beta t} + \gamma_2 \int_0^t e^{-\beta(t-\tau)} u(\tau) d\tau := \varphi(t) \tag{6.20}$$

where the control action, $u(\tau)$, is computed based on the model state from (6.9) and $\varphi(\cdot)$ is a monotonically decreasing function of time. Notice that the evaluation of the above bound requires storing the control input profile (or the model state profile) over time. Recall also that the closed-loop model is exponentially stable by design and, therefore, the control input term can be upper-bounded by an exponentially-decreasing function of time that can be integrated explicitly leading to a more explicit time-varying upper bound that eliminates the need for the integral in the above expression (albeit at the expense of a possibly more conservative bound).

Using the bound in (6.20), it can now be shown that the following chain of inequalities holds:

$$\begin{aligned} &\|\tilde{e}(t)\| \leq \epsilon \|\tilde{a}_s(t)\| - (\epsilon+1)\varphi(t) \\ \implies &\|\tilde{e}(t)\| \ \leq \ \epsilon \|\tilde{a}_s(t)\| - (\epsilon+1)\|Q_s^{-1}Q_f a_f(t)\| \\ \implies &\|e(t)\| \leq \epsilon \|a_s(t)\| \ \implies \dot{V}(a_s) < 0 \end{aligned} \tag{6.21}$$

The above analysis can be used to devise a modified event-triggered control strategy to be implemented under incomplete state measurements. The idea is to monitor the evolution of the error variable, $\tilde{e}$, over time, and trigger an update of the model state using the available estimate, $\tilde{a}_s$, whenever the modified threshold is breached. This can be formalized as follows. Let t_k be a time instance such that the modified threshold is breached as follows:

$$\|\tilde{e}(t_k)\| > \epsilon \|\tilde{a}_s(t_k)\| - (\epsilon+1)\varphi(t_k) \tag{6.22}$$

Then the model state update law given by

$$\widehat{a}_s(t_k) \ = \ \tilde{a}_s(t_k) \tag{6.23}$$

resets the error variable to zero and ensures that (6.19) and (6.21) are satisfied, thus ensuring stability.

Remark 6.2. *It should be noted that for the implementation of this modified update strategy to be feasible, the estimation error (arising due to the contribution of the fast states to the output) needs to be small enough such that the new update-triggering threshold is positive. This implies that we need to have* $\|\tilde{a}_s(t)\| > (1+\epsilon^{-1})\varphi(t)$*, which in turn requires that the fast states decay sufficiently fast to ensure that* $\varphi(t)$ *is sufficiently small. A singular perturbation framework can be used here to establish that this fast convergence is achievable provided that the separation between the fast and slow eigenvalues is sufficiently large (i.e, the finite-dimensional slow subsystem is chosen to be of sufficiently high order), at least after an arbitrarily small boundary layer time interval has elapsed. It is also worth noting that owing to the dependence of the estimation error on the measurement operator (which in turn depends on the locations of the measurement sensors), a judicious choice of the sensor locations can also help maintain small estimation errors.*

Remark 6.3. *A possible alternative approach to address the implementation of the state feedback event-triggered control strategy under limited state measurements is to use a dynamic slow state observer (in lieu of the static estimator in* (6.15)*) to generate estimates of the slow states. While this approach does not require the invertibility of the measurement operator, it is challenged by the need to impose additional assumptions on the convergence properties of the observer estimation error to ensure implementation feasibility and closed-loop stability (see [73] for a discussion on this issue).*

6.3 Augmenting Model-Based Control with Event-Based Parameter Re-Identification

To address the potential impact of process parametric variations on both the stability of the closed-loop system and the extent of sensor-controller communication required, we describe in this section a methodology for augmenting the event-triggered model-based networked control strategy presented in the previous section with an event-based parameter re-identification scheme that aims keep the plant-model mismatch.

Initially, an appropriate communication-triggering threshold coefficient, ϵ, that ensures closed-loop stability with reduced sensor-controller communication is chosen (based on (6.12)) and the networked control system is operated at that value. The time evolution of the estimated closed-loop slow states is then continuously monitored and the estimation error variable, $\tilde{e}(t)$, is computed and compared to the threshold on the right-hand side in (6.22) to determine if a threshold breach has taken place, at which time the model states need to be updated.

As the drift in process parameter values acts to exacerbate the size of the plant-model mismatch over time (i.e., causing the error $\tilde{e}(t)$ to become larger), the alarm threshold in (6.22) is more easily violated, thus jeopardizing closed-loop stability. While the event-triggered control strategy has the ability to compensate for this by increasing the sensor-controller communication rate (provided the parametric drift does not exceed the prescribed uncertainty bounds), this solution may not be as efficient in terms of network resource utilization in the sense that it may lead to a premature increase in the communication rate.

A potentially more efficient solution is to have the model parameters re-identified and updated if necessary. To assess the need for parameter re-identification at any give time, a monitoring scheme is utilized wherein the sensor-controller communication pattern over the network is continuously monitored. The idea is to detect possible sustained increases in the communication rate and use them as an indication of parametric drift. To this end, an indicator function of the following form is utilized:

$$\delta_i = \begin{cases} 1\ , & (t_i - t_{i-1}) < (t_{i-1} - t_{i-2}) \\ 0\ , & \text{otherwise} \end{cases} \tag{6.24}$$

where t_i refers to the i-th update time. Whenever the time interval between the most recent consecutive update instances, $(t_i - t_{i-1})$, is shorter than the interval between the previous two update instances, $(t_{i-1} - t_{i-2})$ (see Figure 6.1 for an illustration), the indicator function records a value of 1. Otherwise, the function takes a value of zero. By summing up the values recorded over some horizon, one can identify a sustained increase in the communication frequency if the accumulated sum over the horizon exceeds a certain pre-

defined threshold; i.e., if the following holds

$$\sum_{i=k}^{k-N} \delta_i > \gamma' \tag{6.25}$$

where $N > 0$ is the horizon and $\gamma' > 0$ is the threshold. Both N and γ' can be viewed as design parameters which can be used to tune the sensitivity of the monitoring scheme and how frequently the model parameters are re-identified. When a sustained increase in

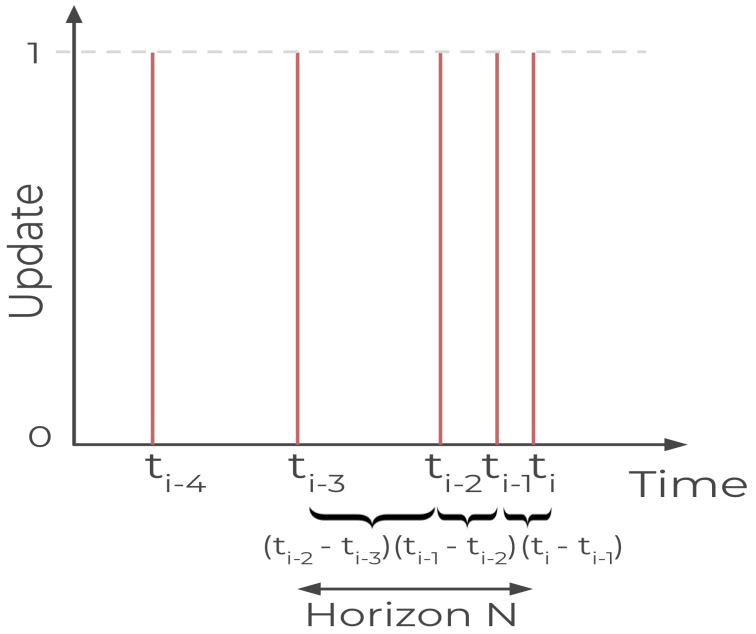

Figure 6.1. Illustration of communication frequency monitoring by tracking update instances over a moving horizon

the communication frequency is observed, parameter re-identification is triggered and new model parameters are identified using the available input and output data (e.g., using subspace identification techniques which are appropriate for linear systems). Once new model parameters are identified, a new communication-triggering threshold coefficient value, ϵ, is calculated and compared to the pre-drift value used originally to decide whether an update of the model parameters should be triggered. Specifically, if the newly-obtained ϵ is larger, then the model parameters should be updated since a larger threshold coefficient generally will lead to reduced communication occurrences.

6.4 Simulation Study

In this section, we use a simulated diffusion-reaction process to illustrate the application of the methodology presented in the previous two sections. The process dynamics and output measurements are given by:

$$\frac{\partial \bar{x}(z,t)}{\partial t} = \frac{\partial^2 \bar{x}(z,t)}{\partial z^2} + [\beta_T \gamma e^{-\gamma} - \beta_U]\bar{x}(z,t) + \beta_U \sum_{i=1}^{2} b_i(z) u_i(t)$$

$$y_i(t) = \int_0^{\pi} q_i(z)\bar{x}(z,t)dz, \; i \in \{1,2\}$$

subject to the initial and boundary conditions in (6.2). For a choice of process parameter values such that $\beta_T = 120$, $\gamma = 4$, and $\beta_U = 2.0$, the spatially-uniform open-loop steady-state, $\bar{x}(z,t) = 0$, is unstable. The control objective is to stabilize the state profile at the open-loop unstable steady-state using two point-control actuators (with finite-support) and two point-measurement sensors (with finite support) with minimal sensor-controller communication, in the presence of process parametric variations which are simulated by a drift in the heat of reaction parameter, β_U.

By solving the eigenvalue problem for the differential operator, it can be shown that, for the given process parameters, only the first two eigenvalues are unstable, and as a result we take the first two unstable eigenvalues as the dominant ones and apply modal decomposition to obtain the following second-order slow subsystem:

$$\dot{a}_s = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} a_s + \omega \begin{bmatrix} \phi_1(z_{a1}) & \phi_1(z_{a2}) \\ \phi_2(z_{a1}) & \phi_2(z_{a2}) \end{bmatrix} u$$

$$y_s = \begin{bmatrix} \phi_1(z_{s1}) & \phi_2(z_{s1}) \\ \phi_1(z_{s2}) & \phi_2(z_{s2}) \end{bmatrix} a_s$$

where $\lambda_i = \bar{\beta} - i^2$ is the i-th eigenvalue, $\omega = 2$, $\phi_i(z) = \sqrt{2/\pi}\ \sin(iz)$ is the i-th eigenfunction, and $z_{a1} = \pi/4$ and $z_{a2} = \pi/2$ are the locations of the first and second control actuators, respectively, and $z_{s1} = \pi/3$ and $z_{s2} = 2\pi/3$ are the locations of the first and second measurement sensors. For controller implementation, we initially consider a

finite-dimensional nominal model of the slow subsystem with

$$\widehat{A}_s = \begin{bmatrix} 5.792 & 0 \\ 0 & 2.792 \end{bmatrix}, \quad \widehat{B}_s = \begin{bmatrix} 1.128 & 1.596 \\ 1.596 & 0 \end{bmatrix}$$

Based on this model, a choice of the controller gain as:

$$K = \begin{bmatrix} 0 & -14.3 \\ -9.9 & 10.1 \end{bmatrix}$$

is made to place the poles of the closed-loop model at $[-10\ -20]$. Based on these settings, the communication-triggering threshold coefficient was determined to be $\epsilon = 0.125$. In the remainder of this section, we explore the implementation of the proposed approach both under normal operating conditions and when parametric drift is introduced. In all scenarios, the controller design and parameter re-identification are based on the finite-dimensional model, but are applied to a sufficiently high-order Galerkin discretization of the PDE.

6.4.1 Nominal operation under event-triggered control

Figures 6.2 - 6.6 show the time evolution of the two dominant eigenmodes, the manipulated input profiles, the model state update instances, the evolution of the model estimation error relative to the communication-triggering threshold, and the closed-loop state profile under the event-triggered control strategy when no parametric drift is introduced. It can be seen that the event-triggered control strategy is able to successfully stabilize the system at the desired steady state with frequent communication as can be seen from Figure 6.4.

6.4.2 Operation subject to parametric drift

In this case, a drift in the heat of reaction coefficient, β_U, is introduced into the simulation starting at time $t = 24$ min. The simulation results for this case are given in Figures 6.7 - 6.13. The dense red band of update instances in Figure 6.9 represents the immediate response of the event-triggered communication strategy aimed at maintaining closed-loop stability following the drift as the plant-model mismatch increases. The detection of this sustained increase in sensor-controller communication by the monitoring scheme is shown

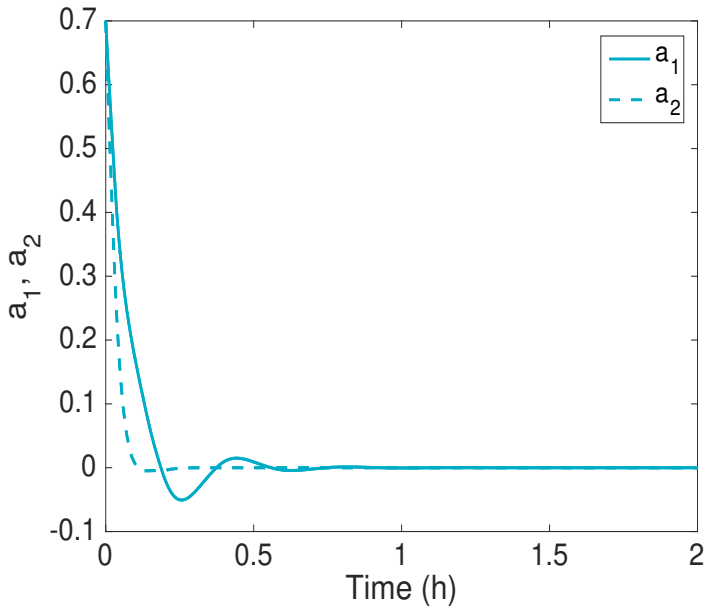

Figure 6.2. Dominant eigenmode profile for the closed-loop system under event-triggered output feedback control and no parametric drift

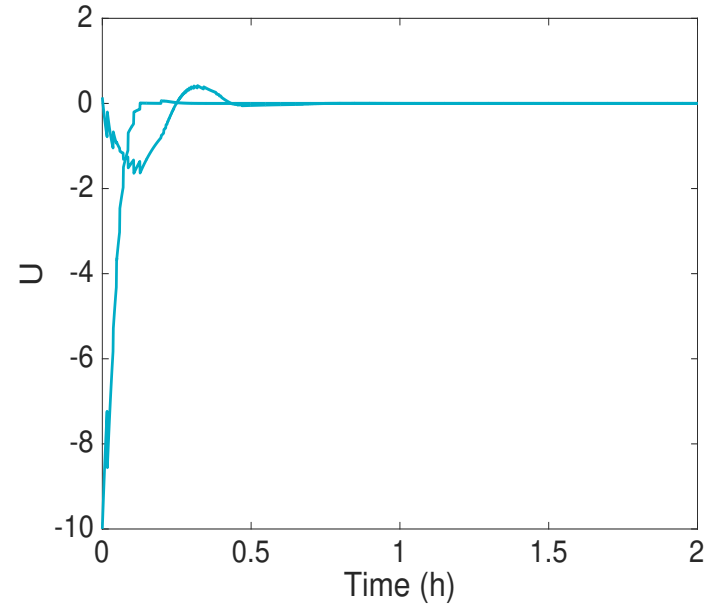

Figure 6.3. Manipulated input profile for the closed-loop system under event-triggered output feedback control and no parametric drift

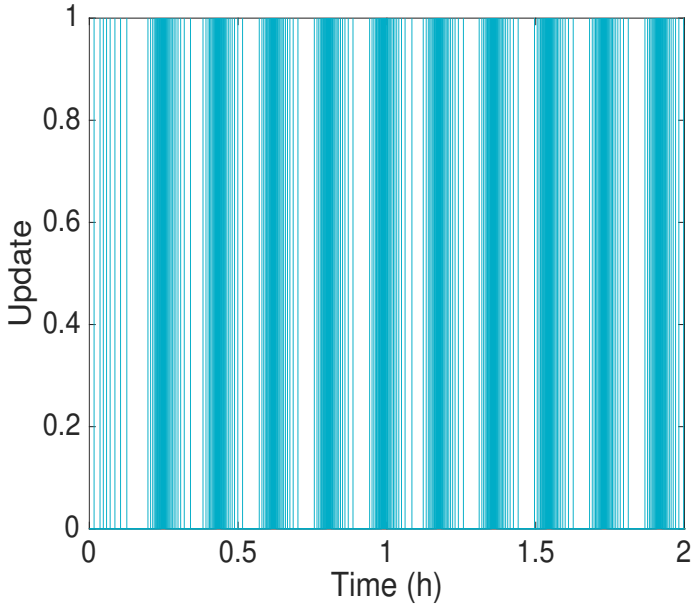

Figure 6.4. Update instances vs. time

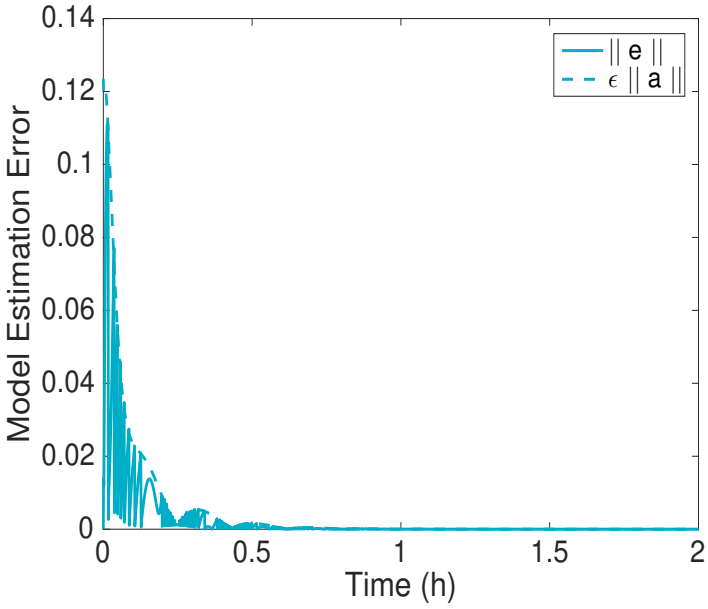

Figure 6.5. Model estimation error evolution relative to the alarm threshold

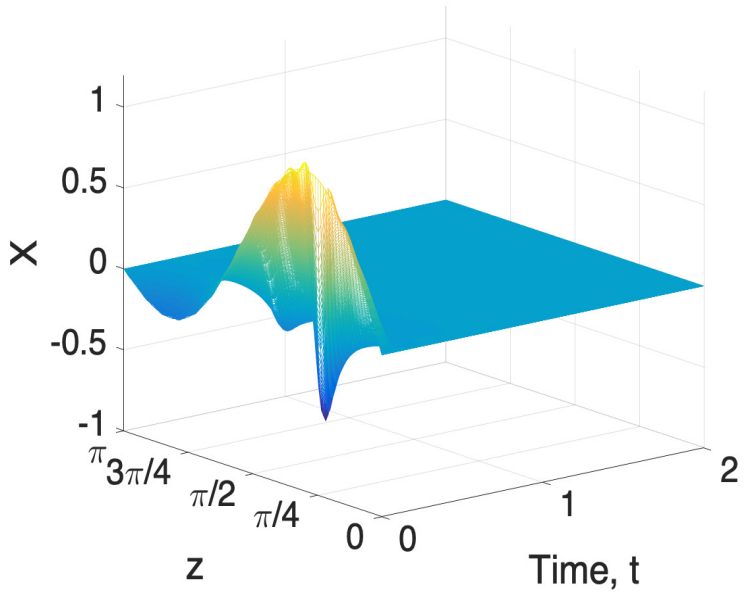

Figure 6.6. Closed-loop state profile

in Figure 6.11, where the time evolution of the cumulative sum of the recorded indicator function values over the horizon ($N = 50$) is plotted, and can be seen to exceed the specified threshold ($\gamma' = 40$) at $t = 26$ min, thus indicating that a drift has occurred and triggering parameter re-identification. It should be noted here that, without monitoring the communication frequency and taking action to re-identify the model parameters, the post-drift continuous red band in Figure 6.9 would continue for all future times.

Following the detection of a sustained increase in communication, the input and output data (available from the infinite-dimensional system) are used, together with subspace identification techniques, to identify the following model matrices:

$$\bar{A} = \begin{bmatrix} 0.238 & 0 \\ 0 & -1.000 \end{bmatrix}, \bar{B} = \begin{bmatrix} 1.128 & 1.596 \\ 1.596 & 0 \end{bmatrix} \tag{6.26}$$

Based on the newly-identified model parameters, the new communication-triggering threshold coefficient value is found to be $\epsilon_{new} = 0.351$, which exceeds the pre-drift value of $\epsilon = 0.125$. As a result, the model parameters are updated according to (6.26). The post-update phase of operation is depicted by the green color in Figures 6.7 - 6.13. Figure 6.9 shows that, as a result of applying the integrated control and identification approach,

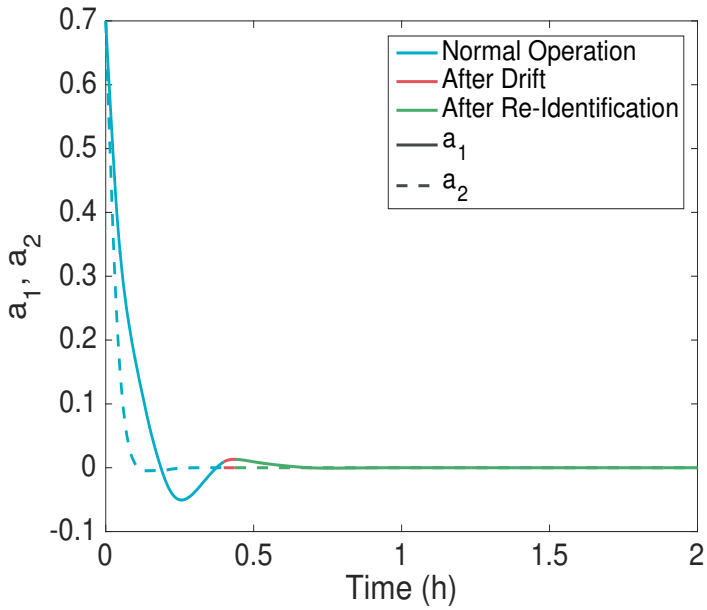

Figure 6.7. Dominant eigenmode profile for the closed-loop system under event-triggered output feedback control and parametric drift

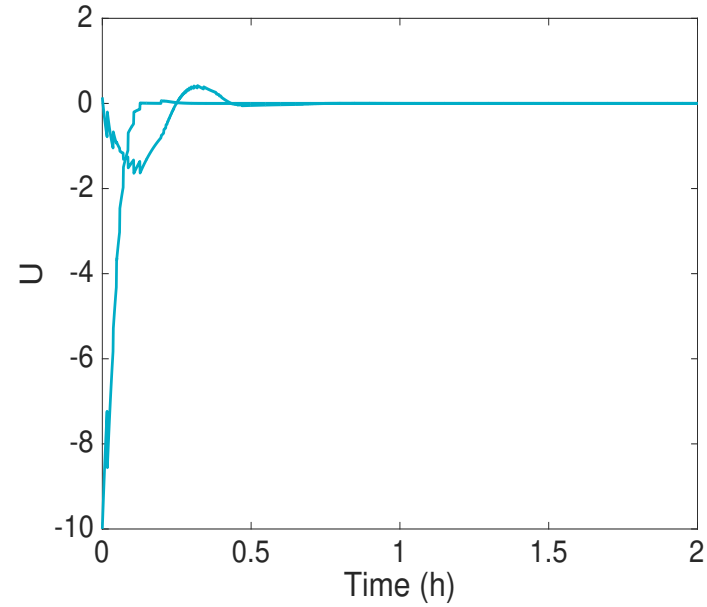

Figure 6.8. Manipulated input profile for the closed-loop system under event-triggered output feedback control and parametric drift

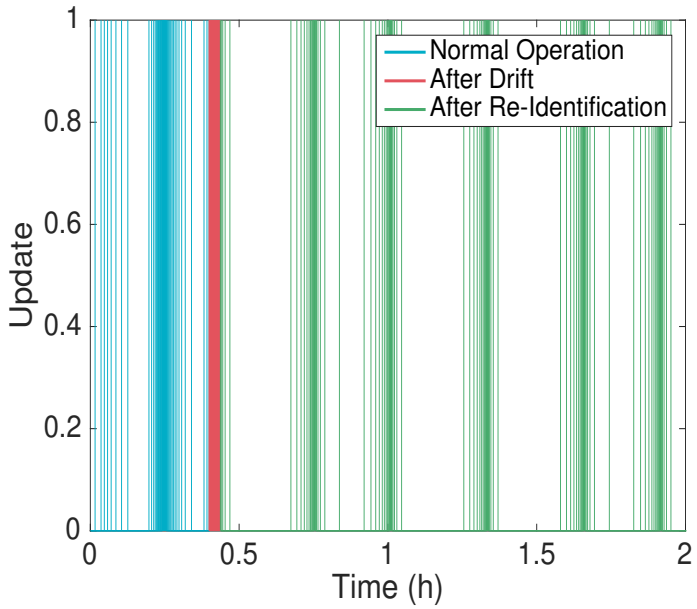

Figure 6.9. Update instances vs. time

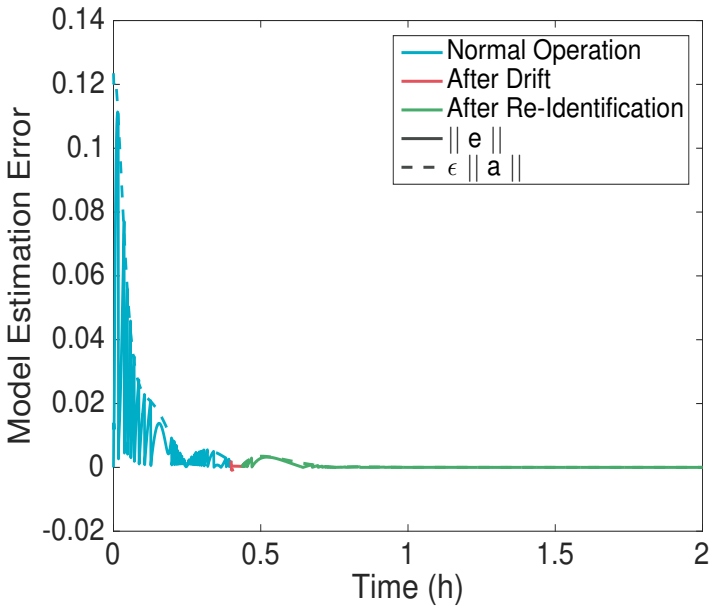

Figure 6.10. Model estimation error evolution relative to the alarm threshold

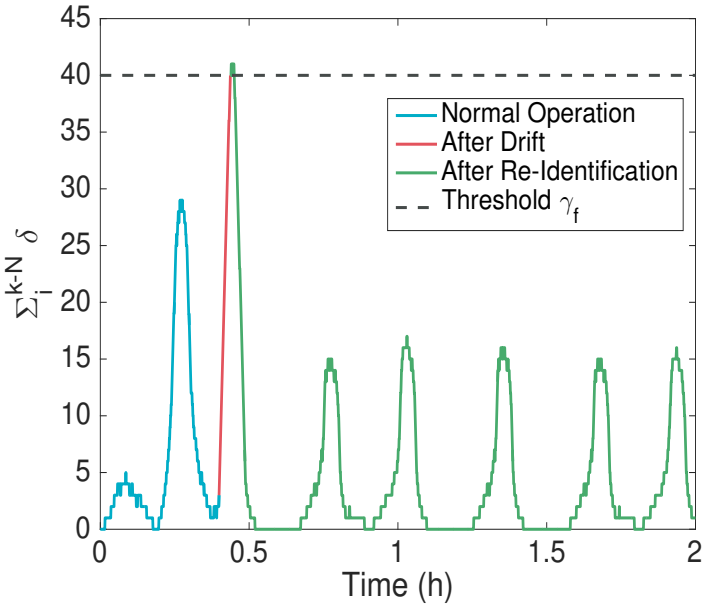

Figure 6.11. The accumulated sum of the indicator function values over the horizon at each time instance

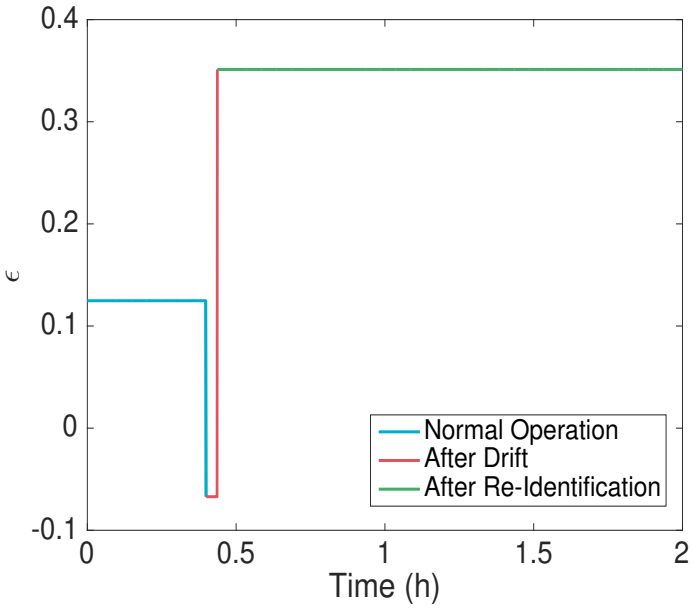

Figure 6.12. Communication-triggering threshold coefficient during the pre-drift, drift, and post-drift phases

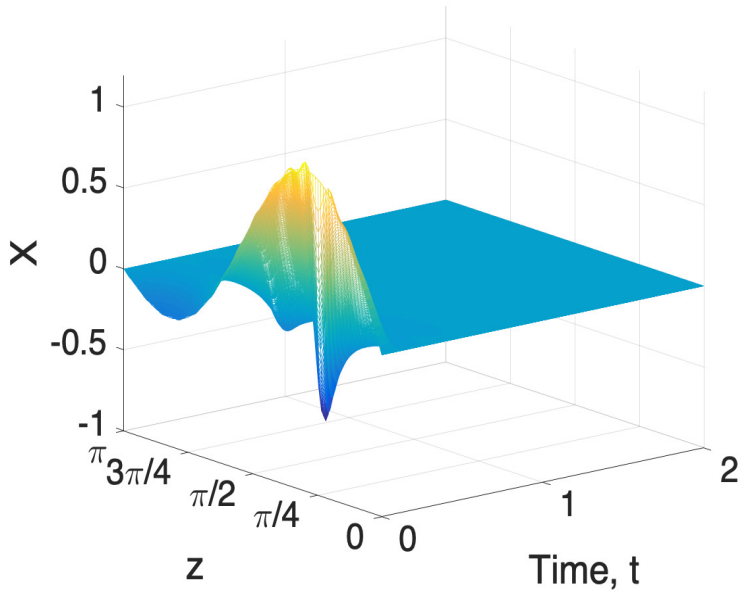

Figure 6.13. Closed-loop state profile

the post-update communication rate is reduced compared to that in the pre-drift phase (blue region) and in the initial drift phase (red region).

6.5 Conclusions

In this chapter, we presented a methodology for the integration of time-triggered model-based networked control and event-triggered model parameter re-identification for spatially-distributed processes modeled by highly-dissipative PDEs subject to sensor-controller communication constraints and process parametric drift. A monitoring scheme was devised to detect increased plant-model mismatch due to parameter drift using a time-varying instability alarm threshold. A breach of the threshold triggered a safe-parking protocol that aimed to maintain closed-loop stability while allowing the collection of additional input and state data that were used to re-identify the model parameters and minimize the plant-model mismatch. A stability check was then performed for the newly-identified model to determine if the model parameters could be updated and pre-drift operating conditions could be restored. The development and implementation of the proposed framework were demonstrated using a representative diffusion-reaction process example. A key point of investigation in the simulation example has been the tradeoff

between the achievable closed-loop performance and the extent of network utilization realized by different safe-parking approaches. It was shown that safe-parking by adjusting the update period yields better performance but comes at the expense of increased communication cost, where as safe-parking via actuator re-location avoids the communication cost increase but could cause performance deterioration.

Chapter 7

Nonlinear Networked Control of Parabolic PDE Systems with Parametric Drift and Re-Identification

To address the implementation issues of nonlinear systems control, in this chapter we present a framework for augmenting model-based feedback control with error-triggered parameter re-identification in spatially-distributed systems described by nonlinear parabolic PDEs subject to sensor-controller communication constraints and process parametric variations. The framework aims to maintain closed-loop stability in the presence of varying levels of plant-model mismatch during periods of parametric drift, while simultaneously keeping the rate of sensor-controller communication to a minimum and accounting explicitly for the presence of nonlinearities.

The rest of the chapter is organized as follows: following some preliminaries in Section 7.1, a stabilizing nonlinear state feedback controller is designed in Section 7.2 based on an approximate finite-dimensional model of the infinite-dimensional system, and an explicit characterization of the closed-loop stability region is obtained in terms of the communication rate, the parametric uncertainty bounds, and the controller design parameters. An error monitoring scheme with a time-varying instability alarm threshold is then introduced in Section 7.3 to determine on-line if and when the model parameters need to

be updated. A breach of the instability threshold triggers a safe-parking step in which the sensor-controller communication rate is adjusted to mitigate the destabilizing impact of increased plant-model mismatch. The measurements collected during the safe-parking mode are used to obtain new estimates of the process parameters using nonlinear grey-box parameter estimation techniques. An explicit characterization of the closed-loop stability region associated with the new model parameters is obtained to determine the appropriate post-drift sensor-controller communication rate that should be used when the model parameters are updated. Finally, the results are illustrated using a simulation case study in Section 7.4.

7.1 Preliminaries

In this work we consider spatially-distributed systems modeled by nonlinear parabolic PDEs of the following form:

$$\frac{\partial \bar{x}}{\partial t} = \alpha \frac{\partial^2 \bar{x}}{\partial z^2} + \beta \bar{x} + \bar{f}(\bar{x}, p) + \omega \sum_{i=1}^{m} b_i(z) u_i(t) \tag{7.1}$$

$$\bar{x}(0,t) = \bar{x}(\pi,t) = 0, \ \bar{x}(z,0) = \bar{x}_0(z) \tag{7.2}$$

where $\bar{x}(z,t) \in \mathbb{R}$ represents the state variable, $z \in [0,\pi]$ and $t \in [0,\infty)$ are the spatial and temporal coordinates, respectively, $\bar{f}(\cdot)$ is a smooth nonlinear function that satisfies $\bar{f}(0,p) = 0$, p is a vector of constant parameters associated with the nonlinear part of the dynamics, u_i is the i-th manipulated input, $b_i(\cdot)$ is the i-th actuator distribution function, m is the number of control actuators, the parameters $\alpha > 0$, β, and ω are constants, and $\bar{x}_0(z)$ is a smooth function of z.

To facilitate controller synthesis and subsequent analysis, the PDE of (7.1)-(7.2) is initially formulated as an infinite-dimensional system of the following form (e.g., see [63]):

$$\dot{x}(t) = \mathcal{A}x(t) + \mathcal{B}u(t) + f(x(t),p), \ \ x(0) = x_0, \tag{7.3}$$

where $x(t) = \bar{x}(z,t)$, for $t > 0$ and $z \in [0,\pi]$, is the state function defined on the Hilbert space $\mathcal{H} = L_2(0,\pi)$, $\mathcal{A}$ is the spatial differential operator whose spectrum can be partitioned into a finite set of slow (possibly unstable) eigenvalues and an infinite

complement of fast stable eigenvalues, $\mathcal{B}$ is the input operator that describes the spatial placement of the control actuators, $u = [u_1 \cdots u_m]^T \in \mathbb{R}^m$ is the manipulated input vector, $f(\cdot)$ is a smooth nonlinear function, and x_0 is the initial condition. Leveraging the low-order dynamics of this class of systems, standard modal decomposition can be applied to transform the system in (7.3) into the following interconnected subsystems:

$$\begin{aligned} \dot{x}_s(t) &= \mathcal{A}_s x_s(t) + \mathcal{B}_s u(t) + f_s(x_s, x_f, p_s), \quad x_s(0) = \mathcal{P}_s x_0 \\ \dot{x}_f(t) &= \mathcal{A}_f x_f(t) + \mathcal{B}_f u(t) + f_f(x_s, x_f, p_f), \quad x_f(0) = \mathcal{P}_f x_0 \end{aligned} \tag{7.4}$$

where $x_s = \mathcal{P}_s x$ is the state of a finite-dimensional slow subsystem that describes the evolution of the slow eigenmodes, $x_f = \mathcal{P}_f x$ is the state of an infinite-dimensional fast subsystem that describes the evolution of the fast stable eigenmodes, $\mathcal{A}_s = \mathcal{P}_s \mathcal{A}$, $\mathcal{A}_f = \mathcal{P}_f \mathcal{A}$, $\mathcal{B}_s = \mathcal{P}_s \mathcal{B}$, $\mathcal{B}_f = \mathcal{P}_f \mathcal{B}$, $f_s(\cdot) = \mathcal{P}_s f(\cdot)$ and $f_f(\cdot) = \mathcal{P}_f f(\cdot)$ are nonlinear functions associated with the fast and slow subsystems, respectively, and $\mathcal{P}_s$ and $\mathcal{P}_f$ are the orthogonal projection operators for the slow and fast eigenmodes, respectively. The vectors p_s and p_f are the parameters associated with the nonlinear terms in the slow and fast subsystems, respectively.

Applying model reduction techniques to the system of (7.4), a finite-dimensional system that describes the time evolution of the amplitudes of the dominant (slow) eigenmodes can be obtained as follows:

$$\dot{a}_s = A_s a_s + B_s(z_a) u + f_s(a_s, p_s) \tag{7.5}$$

where $a_s = [a_1 \cdots a_m]^T \in \mathbb{R}^m$, a_i is the amplitude of the i-th eigenmode, A_s is an $m \times m$ diagonal matrix that contains the first m slow eigenvalues of $\mathcal{A}$, and B_s is an $m \times m$ input matrix whose elements are parameterized by the spatial placement of the control actuator locations, z_a, and $f_s(\cdot)$ is a sufficiently smooth function that satisfies $f_s(0, p_s) = 0$.

The system of (7.5) will be used in the remainder of the chapter as the basis for the integrated controller design and parameter identification methodology. This is justified by the fact that, for sufficiently large separation between the slow and fast eigenvalues, closed-loop stabilization of the finite-dimensional system of (7.5) is sufficient to stabilize the infinite-dimensional closed-loop system (see [63] for a justification using singular perturbation arguments).

7.2 Model-Based Networked Control

7.2.1 Problem formulation

Referring to the system of (7.5), the overarching control objective is to asymptotically stabilize the origin of the closed-loop system using minimal sensor-controller communication so as to conserve the limited resources of the communication channel. To meet this objective, we consider a model-based networked control configuration with periodic sensor-controller communication. The general idea is to initially design a model-based controller that stabilizes the closed-loop model, implement the model-based control action on the plant, and periodically update the model state using the actual state measurement at discrete times to curb the potentially destabilizing influence of the plant-model mismatch. The main task is to characterize the maximum allowable update period that can be used without loss of closed-loop stability. To help illustrate the key ideas, we focus on the state feedback control problem.

7.2.2 Model-based controller synthesis and implementation

To aid in controller synthesis, we assume that an approximate model of the system of (7.5) is available and takes the following form:

$$\dot{\widehat{a}}_s = \widehat{A}_s \widehat{a}_s + \widehat{B}_s(z_a)u + \widehat{f}_s(\widehat{a}_s, \widehat{p}_s) \tag{7.6}$$

where $\widehat{a}_s$ is the state of the model and $\widehat{A}_s$, $\widehat{B}_s$ and $\widehat{f}_s(\cdot)$ are models that approximate A_s, B_s and $f_s(\cdot)$, respectively. It is important to note that $\widehat{A}_s$, $\widehat{B}_s$ and $\widehat{f}_s(\cdot)$ are in general not equal to A_s, B_s and $f_s(\cdot)$ to allow for possible plant-model mismatch. The plant-model mismatch is defined as:

$$\begin{gathered} \widetilde{A}_s = A_s - \widehat{A}_s,\ \widetilde{B}_s = B_s - \widehat{B}_s \\ \theta(\cdot, \delta_p) = f_s(\cdot, p_s) - \widehat{f}_s(\cdot, \widehat{p}_s) \end{gathered} \tag{7.7}$$

where $\delta_p = p_s - \widehat{p}_s$ and the functions $\widehat{f}_s(\cdot)$ and $\theta(\cdot)$ are assumed to be sufficiently smooth (with $\widehat{f}_s(0, \widehat{p}_s) = 0$ and $\theta(0, \delta_p) = 0$) and therefore satisfy the following growth bounds:

$$\begin{gathered} \left\| \widehat{f}_s(x, \widehat{p}_s) - \widehat{f}_s(y, \widehat{p}_s) \right\| \leq L_f(\widehat{p}_s)\|x - y\| \\ \|\theta(x, \delta_p) - \theta(y, \delta_p)\| \leq L_\theta(\delta_p)\|x - y\| \end{gathered} \tag{7.8}$$

for all x, y in some compact neighborhood $\Omega \subset \mathbb{R}^m$ containing the origin in its interior, where $L_f(\widehat{p}_s)$ and $L_\theta(\delta_p)$, are positive constants.

Based on model of (7.6), the next step is to design a nonlinear state feedback control law of the general form:

$$u(t) = k(\widehat{a}_s(t)) \tag{7.9}$$

such that the origin of the closed-loop model of (7.6)-(7.9) satisfies a bound of the form:

$$\|\widehat{a}_s(t)\| \leq \alpha \, \|\widehat{a}_s(t_0)\| e^{-\beta(t-t_0)}, \; \forall \, t \geq t_0 \tag{7.10}$$

where t_0 is the initial time, $\alpha \geq 1$, $\beta > 0$, for all $\widehat{a}_s(t_0) \in \Omega$. The controller function, $k(\cdot)$ is a sufficiently smooth function and satisfies the following growth bound:

$$\|k(x) - k(y)\| \leq L_k \|x - y\| \tag{7.11}$$

for all x, y in Ω, where L_k is a positive constant.

We note that the above stability requirement is imposed to ensure that during times of sensor-controller communication suspension (when the controller uses the model state), the input to the plant is stable. However, the requirement of exponential stability is not necessary and can be relaxed and replaced by either asymptotic or bounded stability. The advantage of exponential stability, however, is that it facilitates the formulation of an explicit stability condition for the networked closed-loop system which is presented in the next subsection.

The implementation of the model-based controller can now be described as follows:

$$\begin{aligned} u(t) &= k(\widehat{a}_s(t)), \; t \in [t_k, t_{k+1}) \\ \dot{\widehat{a}}_s(t) &= \widehat{A}_s \widehat{a}_s(t) + \widehat{B}_s(z_a) u(t) + \widehat{f}_s(\widehat{a}_s(t), p_s) \\ \widehat{a}_s(t_k) &= a_s(t_k), \; k \in \{0, 1, 2, \cdots\} \end{aligned} \tag{7.12}$$

where t_k is the k-th update time, and $h = t_{k+1} - t_k$ is defined as the update period, which is the time period between two consecutive updates. The update period is the measure that we use to quantify the extent of network resource utilization, and is therefore an important parameter to be optimized in the resource-constrained stabilization problem.

7.2.3 Networked closed-loop stability characterization

To achieve the goal of closed-loop stabilization with minimal communication, it is important to first characterize the range of feasible update periods that can be used for the networked control system without loss of stability. This characterization can be obtained by analyzing the dynamics of the model estimation error, $e = a_s - \widehat{a}_s$, and exploiting the available growth bounds in (7.8) and (7.11). Specifically, it can be shown that the origin of the closed-loop system of (7.5) and (7.12) is asymptotically stable if the update period is chosen to satisfy the following condition:

$$F_1(h) := 1 - \alpha\Big(e^{-\beta h} + \big(e^{L_e h} - e^{-\beta h}\big)\frac{L_w}{\beta + L_e}\Big) > 0 \tag{7.13}$$

where $L_e = \left\|\widehat{A}_s\right\| + \left\|\widetilde{A}_s\right\| + L_\theta(\delta_p) + L_f(\widehat{p}_s)$ and $L_w = \left\|\widetilde{A}_s\right\| + \left\|\widetilde{B}_s\right\| L_k + L_\theta(\delta_p)$. The proof of this result is conceptually similar to the one given in [74]. This result implies that there exists a class $\mathcal{KL}$ function, $\beta_s(\cdot,\cdot)$, such that the norm of the closed-loop state satisfies a bound of the following form:

$$\|a_s(t)\| \leq \beta_s(\|a_s(t_0)\|, t - t_0),\ \forall t \geq t_0 \tag{7.14}$$

The condition in (7.13) captures the influence of plant-model mismatch on the range of allowable update periods, and how the controller design parameters can be used to modulate this influence. It can be seen, for example, that as the uncertainty bound L_w (which combines the uncertainties in A_s, B_s and $f_s(\cdot)$) increases, the range of stabilizing update periods shrinks. Conversely, as the uncertainty bound shrinks, the range of stabilizing update periods increases. In the limit as L_w tends to zero and we approach the case of a perfect model, the update period attains its maximum feasible range, which for the case when $\alpha = 1$, includes any update period no matter how large. Note also that the parameter β, which quantifies the response speed of the closed-loop model state, also influences the stabilizing range of update periods, where larger values of β help expand the feasible range, for a given uncertainty bound.

7.3 Augmenting Model-Based Control Using Parameter Re-Identification

As discussed earlier, process parametric drift over time can increase the level of plant-model mismatch and thus may force the networked control system to operate at reduced update periods to maintain closed-loop stability. To alleviate the resulting increased demand on network resources, it becomes necessary to enhance the predictive capabilities of the model by updating its parameters. In this section, a framework for augmenting the model-based networked control strategy discussed in Section 7.2 with an event-triggered parameter re-identification scheme is presented. An overview of the proposed framework is first presented and then followed by a more detailed discussion of the various implementation steps.

7.3.1 Overview of proposed framework

Figure 7.1 summarizes the proposed approach. As shown in this figure, and following initialization of the networked control system at some suitable update period, the impact of parametric drift is continuously assessed by monitoring the evolution of the closed-loop state relative to a pre-specified instability alarm threshold. When the threshold is breached at some time, operation is switched temporarily to a safe-parking mode where the update period is adjusted to avert instability. During this mode, nonlinear parameter estimation tools are employed to identify the drifted plant parameters based on the available closed-loop state and input data. The estimated parameters are then examined for stability, and a characterization of the new closed-loop stability region is obtained to decide whether to move the system out of the safe-parking mode and update the model parameters. Following parameter updates, process monitoring continues and the aforementioned steps are repeated every time the instability alarm threshold is breached in the future.

7.3.2 Initialization step

In this step, a suitable update period is chosen to operate the networked control system in a way that ensures minimal network resource utilization without compromising closed-

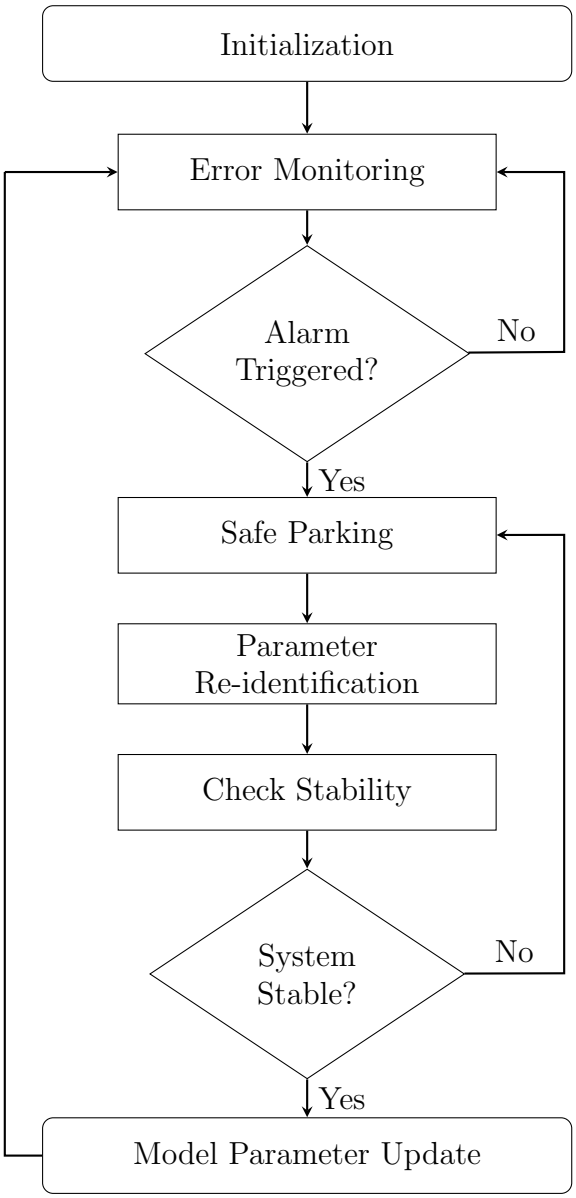

Figure 7.1. Summary of the proposed framework for integrated model-based control and parameter identification

loop stability. The initial update period is denoted by h_0, where $F_1(h_0) > 0$.

7.3.3 Closed-loop system monitoring step

To assess the impact of parametric drift on closed-loop stability and determine if new model parameters need to be identified at any time, a closed-loop monitoring scheme is implemented. The scheme involves continuously tracking the evolution of the closed-

loop states relative to a certain alarm threshold. The threshold is designed to allow the detection of potential instabilities that could be caused by the drift. The instability alarm can be designed on the basis of the nominal closed-loop stability properties highlighted in Section 7.2. Specifically, the closed-loop stability bound in (7.14) is a natural candidate for use as a time-varying alarm threshold, and utilizing it enables the detection of instability once the given bound is violated. This is captured by the following criterion:

$$\|a_s(t_b)\| > \beta_s(\|a_s(t_0)\|, t_b - t_0) \tag{7.15}$$

where t_b is the time that the alarm threshold is exceeded. The threshold breach at t_b triggers a chain of events which include safe-parking and parameter re-identification. The use of a time-varying threshold is advantageous in that it helps reduce the lag that generally exists between the onset of the drift and the triggering of the alarm.

7.3.4 Safe-parking and data collection step

Following the instability threshold breach described by (7.15), the closed-loop system needs to be stabilized prior to proceeding with the parameter re-identification step. In this safe-parking step, the model state update period is reduced to maintain closed-loop stability. Recall from (7.13) that an increase in the plant-model mismatch reduces the maximum allowable update period. The operating update period is adjusted according to the following switching criterion:

$$h(t) = \begin{cases} h_0 \quad , & t_0 \leq t < t_b \\ h_{sp} \quad , & t \geq t_b \end{cases} \tag{7.16}$$

where h_{sp} is the stabilizing (safe-parking) update period chosen such that $F_1(h_{sp}) > 0$. This update period is typically based on the worst-case plant-model mismatch that is expected to be caused by the drift, and may therefore be conservative and costly from a communication standpoint. This highlights the fact that the safe-parking action generally comes at the cost of increased communication frequency, but it is nonetheless necessary to maintain closed-loop stability and to collect sufficient data for the model parameter identification step.

7.3.5 Model parameter re-identification

Following the stabilization of the drifted closed-loop system, when a sufficient amount of data has been collected, parameter re-identification is initiated and new parameter estimates are obtained. The choice of the parameter re-identification method depends on the particular system structure. In this work, nonlinear grey-box parameter estimation techniques are used to identify the new model parameters based on the input and state data. In this class of estimation methods, the structure of the model is assumed to be known, but the parameters are unknown and are estimated using uniformly sampled time data. The nonlinear model is generally represented in this approach using the following form:

$$\begin{aligned} \dot{x}(t) &= G(t, x(t), u(t), p_1, p_2, \ldots, p_N) \\ y(t) &= H(t, x(t), u(t), p_1, p_2, \ldots, p_N) + e(t) \\ x_0 &= x(t_b) \end{aligned} \tag{7.17}$$

where $G(\cdot)$ and $H(\cdot)$ are the known nonlinear functions, $x(t)$ and $u(t)$ are the state and input data (which are collected during the safe-parking mode), $p_1, \ldots, p_N$ are the unknown parameters that need to be estimated following the drift, $y(t)$ is the output data, $e(t)$ is possible noise which maybe present in the output measurements, and x_0 is the initial state data available at the breach time.

The general idea is to determine the values of the unknown parameters, $p_1, \ldots, p_N$, that minimize the mismatch between the predicted state values obtained from (7.17) and the actual state measurements. For the state feedback problem under consideration, we have $y(t) = x(t)$. In cases where the model structure is unknown, black-box nonlinear model identification techniques can be used instead.

It should be noted that the use of nonlinear model identification tools is motivated by the presence of strong nonlinearities and the need to represent the system operation over a range of operating points. In cases of weak nonlinearities, however, it maybe acceptable (and also less computationally demanding) to fit several linear models, where each model is accurate at specific operating conditions. Such linear models can also serve as initial

models for nonlinear estimation that can further improve the fit.

7.3.6 Post-identification analysis of closed-loop stability

Whereas parameter estimation methods generally aim to determine the parameter values that minimize the discrepancy between the predicted and actual state values, an additional key requirement that needs to be imposed on the parameter estimation in the context of the resource-constrained stabilization problem is to find parameter values that reduce the communication load as well. This consideration is not typically taken into account in the optimization-based estimation problem formulation. For this reason, it is important, once the new parameters are identified, to verify that the networked closed-loop stability condition for the newly-obtained model is satisfied. This can be achieved by analyzing the stability of the closed-loop system with the new model parameters. Specifically, using (7.13), the range of stabilizing update periods associated with the new model can be obtained. If the resulting maximum allowable update period is not greater than h_{sp}, then the closed-loop system remains safe-parked until a stabilizing model with a lower update frequency is found; otherwise, the model parameters are updated and the corresponding update period is implemented as follows:

$$\begin{gathered} \widehat{A}_s(t_u) = \bar{A}_s(t_u),\ \widehat{B}_s(t_u) = \bar{B}_s(t_u),\ \widehat{p}_s(t_u) = \bar{p}_s(t_u) \\ h(t) = h_f\ , \quad t \geq t_u \end{gathered} \tag{7.18}$$

where $\bar{A}_s$, $\bar{B}_s$ and $\bar{p}_s$ are the newly identified parameters, t_u is the time when the model parameters are updated, and h_f is the newly determined stabilizing update period which satisfies $F_1(h_f) > 0$.

It is important to note here that in the course of parametric drift, the process parameters are generally assumed to change slowly over time. This gradual change would trigger future alarm threshold breaches and subsequent safe-parking, parameter re-identification and parameter update events. Handling situations involving time-varying parametric drift therefore requires that the steps of the proposed algorithm be implemented repeatedly. This point will be investigated further in the simulation case study.

Following the model parameter update, closed-loop state monitoring needs to continue

to account for possible future parametric drifts. To this end, a new alarm threshold based on the new model parameters needs to be designed and implemented.

7.4 Simulation Example

For illustration purposes, we consider in this section a nonlinear diffusion-reaction process with the following space-time dynamics:

$$\frac{\partial \bar{x}}{\partial t} = \frac{\partial^2 \bar{x}}{\partial x^2} - \beta_U \bar{x} + \beta_T [e^{-\gamma/(1+\bar{x})} - e^{-\gamma}] + \beta_U b(z) u(t)$$

subject to the following initial and boundary conditions:

$$\bar{x}(0,t) = \bar{x}(\pi,t) = 0,\ \bar{x}(z,0) = \bar{x}_0(z)$$

where $\bar{x}(z,t)$ represents the dimensionless temperature; the manipulated input, u, represents the dimensionless temperature of the coolant; $\beta_T = 50$ is the dimensionless heat of reaction, $\gamma = 2$ is the dimensionless activation energy, $\beta_U = 4$ is the dimensionless heat transfer coefficient and $b(z)$ is the control actuator distribution function. It can be shown that for the chosen process parameter values (β_T, γ, and β_U), the zero steady-state solution of the open-loop system is unstable. The control objective is to stabilize the temperature at this unstable steady-state by manipulating u with reduced sensor-controller communication.

To achieve the control objective, we consider the use of a single point control actuator (with finite support) and assume that a sufficiently large number of point measurement sensors that provide accurate state measurements are available. Parametric uncertainty in the heat of reaction coefficient, β_T, will be considered when simulating process parametric drift.

By solving the eigenvalue problem associated with (7.4) and taking the first eigenvalue as the dominant one, we can obtain the following ODE using Galerkin's method:

$$\dot{a}_1 = \lambda_1 a_1 + g(z_a) u + f(a_1) \tag{7.19}$$

where $\bar{x}(z,t) = \sum_{i=1}^{\infty} a_i(t)\phi_i(z)$, $\phi_i(z) = \sqrt{\frac{2}{\pi}}\sin(iz)$, $\lambda_1 = -1-\beta_u$, $g(z_a) = \beta_U \langle \phi_1(z), b(z) \rangle$, $f(a_1) = \beta_T \langle \phi_1(z), h(a_1) \rangle$, $h(a_1) = e^{-\gamma/(1+a_1\phi(z))} - e^{-\gamma}$ and $\langle \cdot, \cdot \rangle$ is the inner product defined on $\mathcal{H} = L_2(0,\pi)$.

An approximate model of (7.19) for controller synthesis is constructed as follows:

$$\dot{\widehat{a}}_1 = \widehat{\lambda}_1 \widehat{a}_1 + \widehat{g}(z_a)u + \widehat{f}(\widehat{a}_1) \tag{7.20}$$

where, for simplicity, $\widehat{\lambda}_1 = \lambda_1$, $\widehat{g}(z_a) = 1.6$, $\widehat{f}(a_1) = (\beta_{Tnom} + \delta_T)\langle \phi_1(z), h(\widehat{a}_1)\rangle$, with $\beta_{Tnom} = 50$. The model-based controller is of the feedback linearizing type and takes the form:

$$u = \widehat{g}^{-1}(z_a)[\lambda_c - \widehat{f}(\widehat{a}_1)] \tag{7.21}$$

where the actuator location is set to $z_a = \pi/2$ and λ_c is a controller parameter that is chosen to place the eigenvalue of the closed-loop model at the desired value. In the following subsections, we will investigate the stability region of the closed-loop system and show the results of the implementation of the proposed integration methodology when time-varying parametric drift takes place.

7.4.1 Characterization of the closed-loop stability region

To understand how the closed-loop stability region as defined by the stability condition in (7.13) is affected by the choice of the controller design parameters (which is represented by β in (7.10)) and the parametric uncertainty (δ_T), a sensitivity analysis is performed on each of those variables independently. Figure 7.2 shows the effect of varying β while holding δ_T constant (note that for the closed-loop model of (7.20)-(7.21), $\beta = -\widehat{\lambda}_1 - \lambda_c$). Values of $F_1(h)$ above the zero line correspond to the range of feasible stabilizing update period values. It can be observed generally that as β increases in value (i.e., as the controller becomes more aggressive), higher update period values can be utilized to stabilize the networked closed-loop system. Figure 7.3 shows the effect of varying δ_T while holding β constant. The trend here is reversed from the previous case; as δ_T increases in value, the range of stabilizing update period values becomes smaller (notice that theoretically at $\delta_T = 0$, any update period value h no matter how large can be used to stabilize the system since this is the limiting case of a perfect model). This result is expected since a larger plant-model mismatch would result in control actions that are based on model estimates of the state that are far from the actual states. For the controller to stabilize the closed-loop system under these conditions, knowledge of the actual system states is

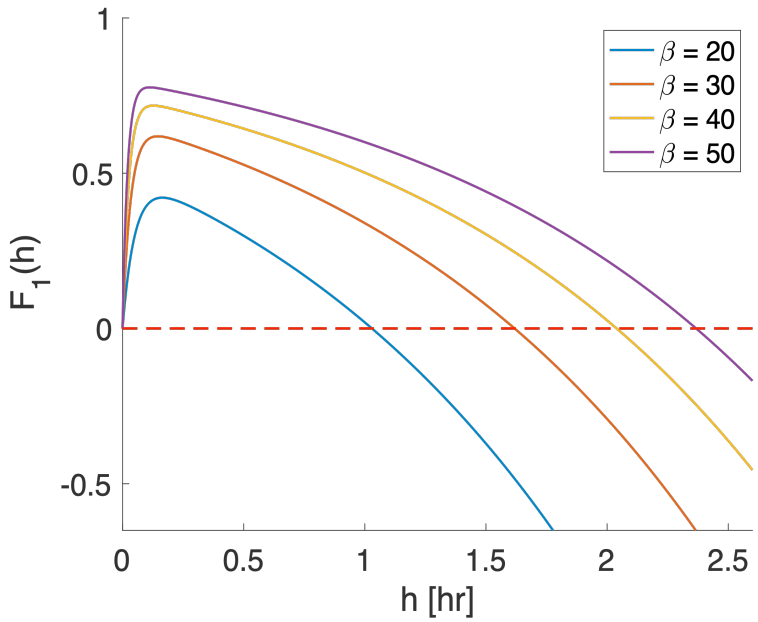

Figure 7.2. F_1 as a function of the update period for different β values when $\delta_T = 5$ and $\alpha = 1$. Larger β values correspond to a larger feasible range of stabilizing update period values

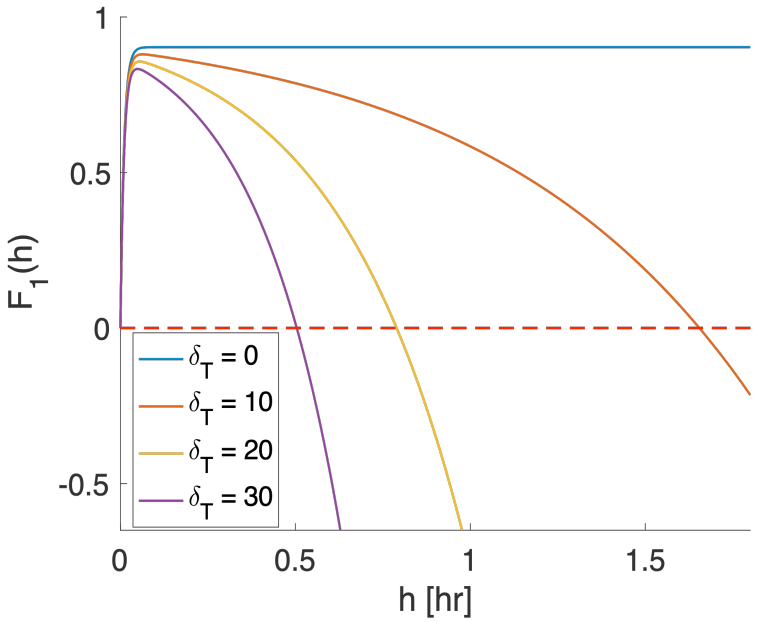

Figure 7.3. F_1 as a function of the update period for different δ_T values when $\beta = 100$ and $\alpha = 1$. Larger δ_T values correspond to a smaller feasible range of stabilizing update period values

required, thus more frequent feedback through communication between the controller and the sensor needs to be established.

7.4.2 Simulation results

Initially, an update period value of $h = 1.2$ hr is chosen to operate the control system, and as can be seen from Figure 7.5 (see the initial blue portion of the profile) it successfully stabilizes the closed-loop system. Meanwhile, a time-varying drift in the heat of reaction parameter is assumed to take place according to the following logistic function form:

$$\beta_T = a + \frac{130 - a}{1 + \exp\left(\frac{c-t}{b}\right)} \tag{7.22}$$

where $a = 50$, $c = 3000$, $b = 500$, and t is time. Figure 7.4 captures the behavior of the time-varying drift in β_T (blue profile) as well the value of the estimated model parameter $\widehat{\beta}_T$ over time (red profile) as parameter re-identification takes place at discrete times. As the parametric drift continues to increase, the plant-model mismatch continues

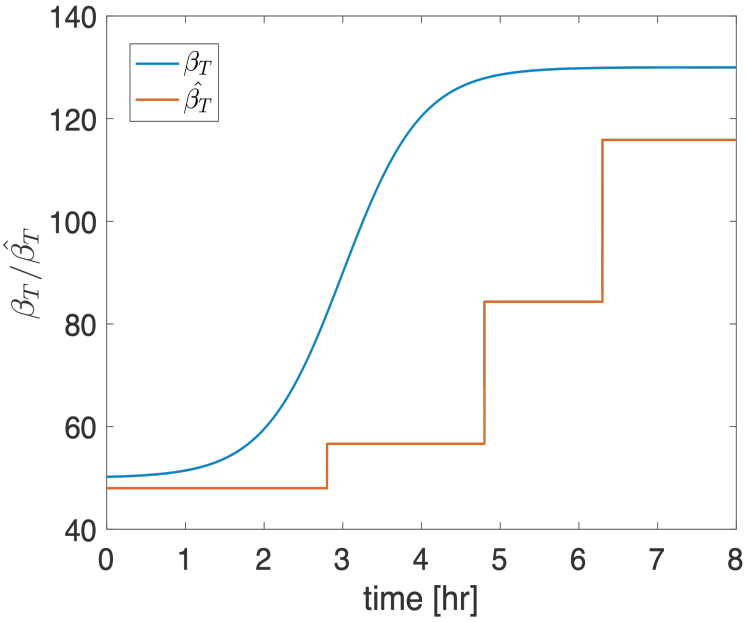

Figure 7.4. Time-varying drift in the parameter β_T, and the identified model parameter value $\widehat{\beta}_T$ obtained at discrete times

to grow and the initial update period value no longer stabilizes the closed-loop system. Specifically, it can be seen from the plot of the amplitude of the dominant eigenmode in Figure 7.5 that at $t = 2.4$ hr the closed-loop state starts to diverge noticeably away from

the steady-state. At this point, the first safe-parking step is initiated to restore stability by reducing the update period to $h_{sp} = 0.1$ hr (this is represented by the red portion of the profile; see also Figure 7.6). Once enough data is collected, $\widehat{\beta}_T$ is re-identified and the corresponding maximum allowable update period is determined ($h = 0.55$ hr), checked for stability and implemented together with the parameter update at $t = 2.8$ hr. At this time, the system is switched out of the safe-parking step, and as can be seen from the green portion of the state profile in Figure 7.5, the closed-loop system is stabilized.

As the drift continues to grow, however, the mismatch between the drifting parameter value and the previously estimated one grows (see Figure 7.4), and as a result a second alarm is triggered now at $t = 4.5$ hr and a second safe-parking step is initiated. During this second safe-parking period, the closed-loop system is re-stabilized (by returning the update period to $h = h_{sp}$), a new value for $\widehat{\beta}_T$ is identified from the available data, and a new update period is computed and verified for stability ($h = 0.3$ hr). The system exists the safe-parking mode when the model parameter is updated and the new stabilizing update period is implemented at $t = 4.8$ hr (see Figure 7.6).

This process repeats one more time as the drift continues to grow, leading to the third and final alarm breach at $t = 5.5$ hr followed by the third safe-parking period which lasts until $t = 6.5$ hr. By the end of this period, the actual parameter value has settled, and a new (improved) estimate of the model parameter is obtained. Based on the newly-identified parameter value, a new stabilizing update period is computed and implemented for all future times ($h = 0.7$ hr).

Figure 7.6 shows the operating update period over the time of the simulation for the normal operation step, the safe-parking steps, and the post-parameter-update steps. It can be seen that, overall, as the difference between the plant and model parameter increases, the update period value decreases (communication frequency increases) to maintain stability. This mismatch, however, is minimized after the third parameter re-identification event as can be seen in Figure 7.4 where the actual parameter settles at its final value. Due to the diminished mismatch at this point, the achievable stabilizing update period after the third re-identification event is greater than update period values

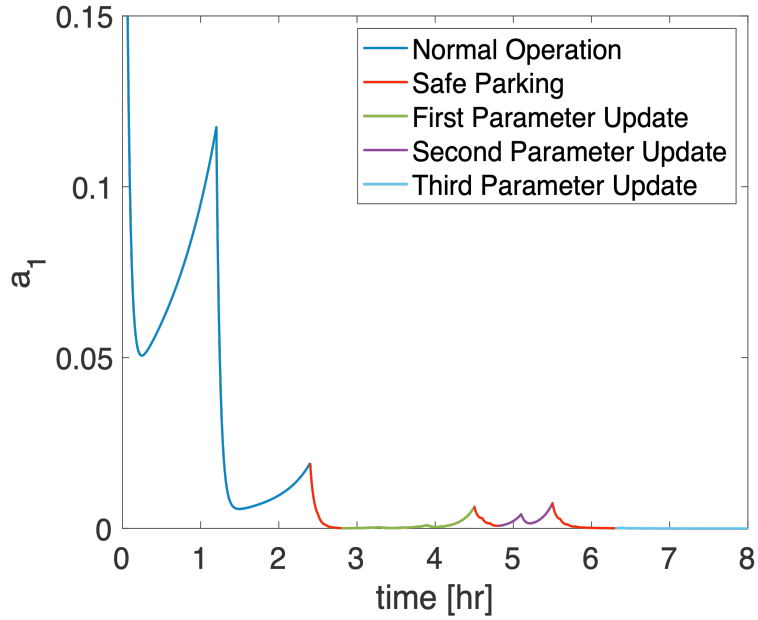

Figure 7.5. Closed-loop evolution of the amplitude of the dominant eigenmode under time-varying parametric drift, with $\beta = 100$ and $\alpha = 1$

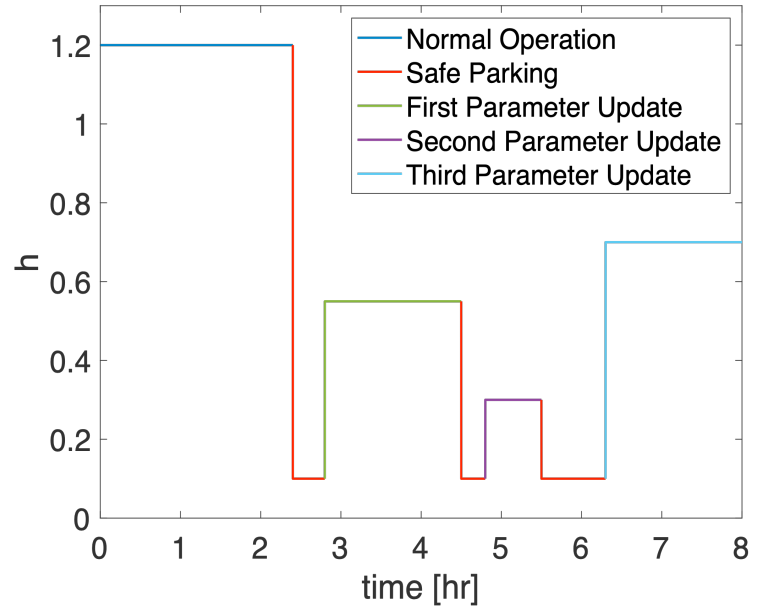

Figure 7.6. Operating update period as a function of time during normal operation, safe-parking and after each parameter update

obtained in the previous two parameter identification events.

This is further confirmed by analyzing the closed-loop stability region prior to the drift and during the different drift conditions (this is depicted in Figure 7.7). It is important to

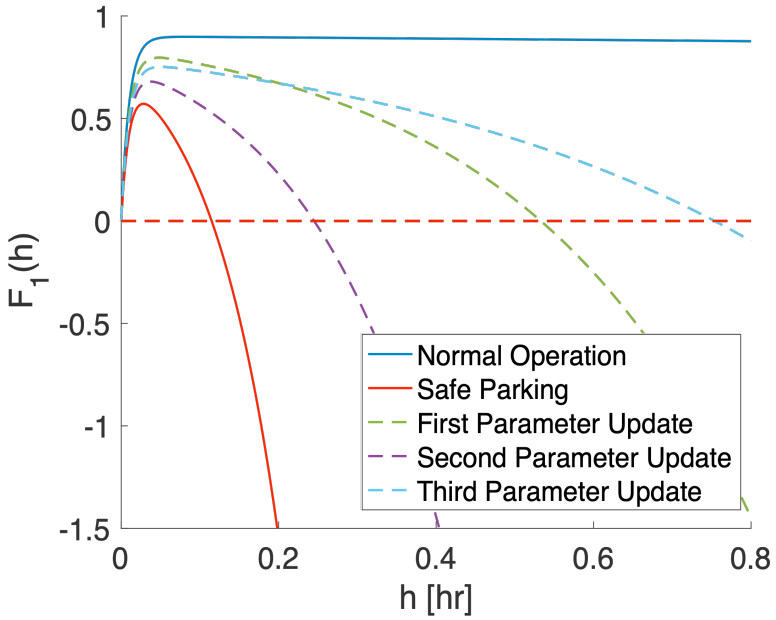

Figure 7.7. F_1 as a function of the update period for normal operation, safe-parking and for each re-identification step

note here that the update period value used during the safe-parking steps is the same and determined from the maximum allowable update period obtained under the worst-case uncertainty bound caused by the drift. This safe-parking update period value guarantees that stability is achieved between the parameter re-identification events.

7.5 Conclusions

In this chapter, a framework for augmenting model-based feedback control with error-triggered parameter re-identification in spatially-distributed systems described by nonlinear parabolic PDEs subject to sensor-controller communication constraints and process parametric variations was presented. The framework aims to maintain closed-loop stability in the presence of varying levels of plant-model mismatch during periods of parametric drift, while simultaneously keeping the rate of sensor-controller communication to a minimum and accounting explicitly for the presence of nonlinearities. We were able to show through simulations that the developed framework was successful at stabilizing the process subject to time varying drifts using parameter re-identification.

Chapter 8

Nonlinear Sampled-Data Control of Parabolic PDE Systems with Measurement Errors, Parametric Drift, and Re-Identification

An issue that is not considered in earlier chapters is the presence of measurement errors and its impact on the stability of the closed-loop system. Such errors can arise from sensor faults or measurement noise, and in some cases can be the direct result of deliberate measurement falsification attempts such as the case in cyber-security attacks which will be discussed in more detail in Chapter 9. Explicit account of the nonlinear process dynamics as well as the measurement errors must therefore be taken not only in the controller synthesis but also in subsequent closed-loop stability characterization. The current chapter aims to bridge this gap.

Motivated by these considerations, we present in this chapter a model-based framework for the design of finite-dimensional sampled-data feedback controllers for spatially-distributed systems described by nonlinear parabolic PDEs subject to discretely-sampled measurements, bounded measurement errors, and bounded model uncertainty. The rest of the chapter is organized as follows: following some preliminaries in Section 8.1 that define the class of systems considered, standard modal decomposition and model reduction techniques are applied to obtain an approximate finite-dimensional system that captures

the dominant dynamics of the infinite-dimensional system. The control problem is then formulated in Section 8.2 and a finite-dimensional nonlinear model-based state feedback controller that stabilizes the closed-loop model is then introduced. The controller uses the model predictions to compute the control action between sampling times, and its state is updated using the available state measurements at the sampling times. The sampled-data closed-loop system, subject to the measurement errors, is analyzed in Section 8.3 and a sufficient condition for closed-loop stability is derived. The analysis leads to an explicit characterization of the closed-loop stability region in terms of the sampling period, the measurement error bound, the model uncertainty bounds, and the controller design parameters. Finally, the results are illustrated using a simulated diffusion-reaction process in Section 8.4.

8.1 Preliminaries

In this work we consider spatially-distributed systems modeled by nonlinear parabolic PDEs of the following form:

$$\frac{\partial \bar{x}}{\partial t} = \alpha \frac{\partial^2 \bar{x}}{\partial z^2} + \beta \bar{x} + \bar{f}(\bar{x}) + \omega \sum_{i=1}^{m} b_i(z) u_i(t) \tag{8.1}$$

subject to the following boundary and initial conditions:

$$\bar{x}(0,t) = \bar{x}(\pi,t) = 0, \ \bar{x}(z,0) = \bar{x}_0(z) \tag{8.2}$$

where $\bar{x}(z,t) \in \mathbb{R}$ represents the state variable; $z \in [0,\pi]$ and $t \in [0,\infty)$ are the spatial and temporal coordinates, respectively; $\bar{f}(\cdot)$ is a smooth nonlinear function that satisfies $\bar{f}(0) = 0$; u_i is the i-th manipulated input; $b_i(\cdot)$ is the i-th actuator distribution function, m is the number of control actuators; the parameters $\alpha > 0$, β, and ω are constant process parameters, and $\bar{x}_0(\cdot)$ is a smooth function of its argument.

To facilitate controller synthesis and subsequent analysis, the PDE of (8.1)-(8.2) is initially formulated as an infinite-dimensional system of the following form (e.g., see [63]):

$$\dot{x}(t) = \mathcal{A}x(t) + \mathcal{B}u(t) + f(x(t)), \ \ x(0) = x_0, \tag{8.3}$$

where $x(t) = \bar{x}(z,t)$, for $t > 0$ and $z \in [0,\pi]$, is the state function defined on the Hilbert space $\mathcal{H} = L_2(0,\pi)$, $\mathcal{A}$ is the spatial differential operator whose spectrum can

be partitioned into a finite set of slow (possibly unstable) eigenvalues and an infinite complement of fast stable eigenvalues, $\mathcal{B}$ is the input operator that describes the spatial placement of the control actuators, $u = [u_1 \cdots u_m]^T \in \mathbb{R}^m$ is the manipulated input vector, $f(\cdot)$ is a smooth nonlinear function, and x_0 is the initial condition. Exploiting the low-order dynamics of this class of systems, standard modal decomposition can be applied to transform the system in (8.3) into the following interconnected subsystems:

$$\begin{aligned} \dot{x}_s(t) &= \mathcal{A}_s x_s(t) + \mathcal{B}_s u(t) + f_s(x_s, x_f), \quad x_s(0) = \mathcal{P}_s x_0 \\ \dot{x}_f(t) &= \mathcal{A}_f x_f(t) + \mathcal{B}_f u(t) + f_f(x_s, x_f), \quad x_f(0) = \mathcal{P}_f x_0 \end{aligned} \tag{8.4}$$

where $x_s = \mathcal{P}_s x$ is the state of a finite-dimensional slow subsystem that describes the evolution of the slow eigenmodes, $x_f = \mathcal{P}_f x$ is the state of an infinite-dimensional fast subsystem that describes the evolution of the fast stable eigenmodes, $\mathcal{A}_s = \mathcal{P}_s \mathcal{A}$, $\mathcal{A}_f = \mathcal{P}_f \mathcal{A}$, $\mathcal{B}_s = \mathcal{P}_s \mathcal{B}$, $\mathcal{B}_f = \mathcal{P}_f \mathcal{B}$, $f_s(\cdot) = \mathcal{P}_s f(\cdot)$ and $f_f(\cdot) = \mathcal{P}_f f(\cdot)$ are nonlinear functions associated with the fast and slow subsystems, respectively, and $\mathcal{P}_s$ and $\mathcal{P}_f$ are the orthogonal projection operators for the slow and fast eigenmodes, respectively.

Applying model reduction techniques to the system of (8.4), a finite-dimensional system that describes the time evolution of the amplitudes of the dominant (slow) eigenmodes can be obtained as follows:

$$\dot{a}_s = A_s a_s + B_s(z_a) u + f_s(a_s) \tag{8.5}$$

where $a_s = [a_1 \cdots a_m]^T \in \mathbb{R}^m$, a_i is the amplitude of the i-th eigenmode, A_s is an $m \times m$ diagonal matrix that contains the first m slow eigenvalues of $\mathcal{A}$, and B_s is an $m \times m$ input matrix whose elements are parameterized by the spatial placement of the control actuator locations, z_a, and $f_s(\cdot)$ is a sufficiently smooth function that satisfies $f_s(0) = 0$. The system of (8.5) will be used in the remainder of the chapter as the basis for the sampled-data controller design and analysis. This is justified by the fact that, for sufficiently large separation between the slow and fast eigenvalues, closed-loop stabilization of the finite-dimensional system of (8.5) is sufficient to guarantee stabilization of the infinite-dimensional closed-loop system (see [63] for a justification using singular perturbation arguments).

8.2 Model-Based Sampled-Data Feedback Controller Design

8.2.1 Control problem formulation

Referring to the system of (8.5), the control objective is to stabilize the origin of the closed-loop system using discretely-sampled measurements in the presence of measurement errors. To this end, we consider the case where all states are sampled periodically at a constant sampling rate, Δ. To capture the effect of measurement errors, we define the vector of measured states as follows:

$$y_m(t) = \Xi_M a_s(t) \tag{8.6}$$

where $y_m(t) \in \mathbb{R}^m$, $\Xi_M := \text{diag}\{\sigma_i\}$ is a diagonal matrix that multiplies the state vector and thus distorts its values. Each element in this matrix, σ_i, scales the value of the i-th state by a certain factor whose size reflects status and severity of the error in each state measurement. A value of $\sigma_i = 1$, for example, indicates that the measured value of the i-th state is error-free, while any other value, i.e., $\sigma_i \neq 1$, reflects the presence of some error. The size of the scaling factor, and how much it deviates from 1, determines the severity of the measurement error.

To achieve the control objective, we consider a model-based sampled-data control structure in which an inter-sample model predictor is used to compensate for the unavailability of measurements between sampling times. The basic idea is to initially design a model-based controller that stabilizes the closed-loop model, implement the model-based control action on the plant, and periodically update the model state using the sampled state measurement at discrete times to curb the potentially destabilizing influence of the plant-model mismatch. The main task is to characterize the maximum allowable sampling period that can be used without loss of closed-loop stability in terms of the measurement error and the controller design parameters. To help illustrate the key ideas, we focus in the following development on the state feedback control problem. Extensions to the output feedback problem are possible and the subject of other research work.

8.2.2 Controller synthesis and implementation

To compute the control action between sampling times, we assume that an approximate model of the system of (8.5) is available and takes the following form:

$$\dot{\widehat{a}}_s = \widehat{A}_s \widehat{a}_s + \widehat{B}_s(z_a)u + \widehat{f}_s(\widehat{a}_s) \tag{8.7}$$

where $\widehat{a}_s$ is the state of the model, and $\widehat{A}_s$, $\widehat{B}_s$ and $\widehat{f}_s(\cdot)$ are models that approximate A_s, B_s and $f_s(\cdot)$, respectively. It is important to note that $\widehat{A}_s$, $\widehat{B}_s$ and $\widehat{f}_s(\cdot)$ are in general not equal to A_s, B_s and $f_s(\cdot)$ to allow for possible plant-model mismatch. The plant-model mismatch is defined by:

$$\begin{gathered} \widetilde{A}_s = A_s - \widehat{A}_s,\ \widetilde{B}_s = B_s - \widehat{B}_s \\ \theta(\cdot) = f_s(\cdot) - \widehat{f}_s(\cdot) \end{gathered} \tag{8.8}$$

where the functions $\widehat{f}_s(\cdot)$ and $\theta(\cdot)$ are assumed to be sufficiently smooth and therefore satisfy the following growth bounds:

$$\begin{gathered} \left\| \widehat{f}_s(x) - \widehat{f}_s(y) \right\| \leq L_f \|x - y\| \\ \|\theta(x) - \theta(y)\| \leq L_\theta \|x - y\| \end{gathered} \tag{8.9}$$

for all x, y in some compact neighborhood, $\Omega \subset \mathbb{R}^m$, containing the origin in its interior, where L_f and L_θ are positive constants.

Based on the model of (8.7), the next step is to design a nonlinear state feedback control law that exponentially stabilizes the closed-loop model. To avoid limiting the analysis to a particular class of nonlinear control laws, we assume that a controller with the desired closed-loop properties is available as stated in the following Assumption.

Assumption 8.1. *There exists a smooth nonlinear feedback control law of the general form:*

$$u = k(\widehat{a}_s) \tag{8.10}$$

such that the origin of the closed-loop model of (8.7)-(8.10) *satisfies a bound of the form:*

$$\|\widehat{a}_s(t)\| \leq \alpha \, \|\widehat{a}_s(t_0)\| e^{-\beta(t-t_0)},\ \forall\, t \geq t_0 \tag{8.11}$$

where t_0 is the initial time, $\alpha \geq 1$, $\beta > 0$, for all $\widehat{a}_s(t_0) \in \Omega := \{\widehat{a}_s \in \mathbb{R}^m : \|\widehat{a}_s\| \leq M\}$, for some compact neighborhood $\Omega \subset \mathbb{R}^m$, containing the origin in its interior.

The controller function, $k(\cdot)$ is assumed to be sufficiently smooth and satisfies a growth bound of the form:

$$\|k(x) - k(y)\| \leq L_k \|x - y\| \tag{8.12}$$

for all x, y in Ω, where L_k is a positive constant.

Remark 8.1. *We note that the above stability requirement is imposed to ensure that between sampling times (when the controller uses the model state), the input to the plant is stable. However, the requirement of exponential stability is not necessary and can be relaxed and replaced by either asymptotic or bounded stability. The advantage of exponential stability, however, is that it facilitates the formulation of an explicit stability criterion for the sampled-data closed-loop system which is presented in the next section.*

The implementation of the model-based controller can now be described as follows:

$$\begin{aligned} u(t) &= k(\widehat{a}_s(t)),\ t \in [t_k, t_{k+1}) \\ \dot{\widehat{a}}_s(t) &= \widehat{A}_s \widehat{a}_s(t) + \widehat{B}_s(z_a) u(t) + \widehat{f}_s(\widehat{a}_s(t)) \\ \widehat{a}_s(t_k) &= y_m(t_k) = \Xi_M a_s(t_k),\ k \in \{0, 1, 2, \cdots\} \end{aligned} \tag{8.13}$$

where t_k is the k-th sampling time at which the model state is updated, and $\Delta = t_{k+1} - t_k$ is the sampling period, which is the time interval between two consecutive sampling times. Note that the model state is updated using the measured state which generally contains errors. The impact of such errors on closed-loop stability is analyzed in the next section.

8.3 Closed-loop stability analysis

In this section, we characterize the stability properties of the sampled-data closed-loop system in terms of the interplay between the various system and control parameters, including the sampling period, the plant-model mismatch, the measurement errors, and the controller parameters. We first formulate the sampled-data closed-loop system and then present and discuss the stability criteria.

8.3.1 Formulation of the sampled-data closed-loop system

To formulate the sampled-data closed-loop system, we first define the model estimation error, $e = a_s - \widehat{a}_s$, which represents the difference between the system state and the

model state. Substituting the model-based control law of (8.13) into (8.5), and using (8.8), the dynamics of the closed-loop system can be captured by the following combined discrete-continuous system:

$$\begin{aligned} \dot{a}_s &= A_s a_s + B_s(z_a)k(\widehat{a}_s) + f_s(a_s) \\ \dot{e} &= A_s e + \widetilde{A}_s \widehat{a}_s + \widetilde{B}_s(z_a)k(\widehat{a}_s) + \widehat{f}_s(e + a_s) \\ &- \widehat{f}_s(\widehat{a}_s) + \theta(a_s) \\ e(t_k) &= (I - \Xi_M)a_s(t_k),\ k \in \{0, 1, 2, \cdots\},\ \Delta = t_{k+1} - t_k \end{aligned} \tag{8.14}$$

where the process state evolves continuously in time, while the model estimation error is reset at each sampling instance when the model state is updated. Note that in the absence of any measurement errors, i.e., when $\Xi_M = I$, the model estimation error is reset to zero.

8.3.2 Characterization of closed-loop stability properties

The following theorem provides a sufficient condition for the stability of the sampled-data closed-loop system in terms of the sampling period, the model uncertainty, the size of the measurement error, and the controller design parameters. To simplify the statement of the theorem, we introduce the following definitions:

$$L_F = \|A_s\| + L_f + L_\theta \tag{8.15}$$

$$L_w = \|\widetilde{A}_s\| + \|\widetilde{B}_s\|L_k + L_\theta,\ L_0 = \theta(0) \tag{8.16}$$

$$\begin{aligned} F_1(s) &:= 1 - \alpha\|\Xi_M\| \left(e^{-\beta s} + (e^{L_F s} - e^{-\beta s})\tfrac{L_w}{\beta + L_F}\right) \\ &- \|I - \Xi_M\|e^{L_F s} \end{aligned} \tag{8.17}$$

$$\psi_1 = \max_{s\in[0,\Delta]} (1 - F_1(s)) \tag{8.18}$$

$$F_2(s) = \frac{L_0}{L_F}(e^{L_F s} - 1) \tag{8.19}$$

$$\psi_2 = \max_{s\in[0,\Delta]} F_2(s) \tag{8.20}$$

$$M' = \frac{M - \psi_2}{\psi_1} \tag{8.21}$$

Theorem 8.1. *Consider the closed-loop system of* (8.5)*, subject to the control and update laws of* (8.13) *for which Assumption 1 holds, with* $\|a_s(t_0)\| \leq M'$*. Then, if* Δ *is chosen such that*

$$F_1(\Delta) > 0 \tag{8.22}$$

and

$$\psi_1 r(\Delta) + \psi_2 \leq M', \tag{8.23}$$

where $r(\Delta) = F_2(\Delta)/F_1(\Delta)$*, the states of the sampled-data closed-loop system are bounded and satisfy:*

$$\left\|a_s(t_{k+1}^-)\right\| < \|a_s(t_k)\| \text{ for all } \|a_s(t_k)\| > r(\Delta) \tag{8.24}$$

for $k \in \{0, 1, 2, \cdots\}$*. Furthermore,* $\lim_{t\to\infty} \|a_s(t)\| \leq \psi_1 r(\Delta) + \psi_2$*.*

Remark 8.2. *The result of Theorem 8.1 establishes that if the sampling period is chosen such that* (8.22) *and* (8.23) *are satisfied, the norm of the sampled closed-loop state is guaranteed to decrease at successive sampling times as long as it is outside some terminal neighborhood of the origin (the size of which is fixed by the choice of the sampling period). This implies that the sampled closed-loop state is guaranteed to converge in finite-time to the terminal set where it remains confined for all future times. Note that the continuous closed-loop state is also guaranteed to converge in finite time to a terminal set around the origin; albeit one that is larger than that for the sampled state owing to the fact that between consecutive sampling times, the closed-loop state can grow a certain amount (see* (A.7) *in Appendix A). This growth, however, is limited since the sampling period is finite.*

Remark 8.3. *The restriction on the set of initial conditions such that* $\|a_s(t_0)\| \leq M'$*, where* M' *is given in* (8.21)*, is imposed to ensure that the closed-loop state remains confined for all times within the set* Ω*, which in turn ensures that the growth bounds in* (8.9) *and* (8.12) *remain valid for all times. This restriction takes account of the maximal growth of the closed-loop state between sampling times. Note that for the problem to be well-posed,* M' *must be positive, and this imposes a restriction on the choice of the sampling period. Also the requirement in* (8.23) *is needed to ensure that the terminal set to*

which the closed-loop state ultimately converges is a subset of the set of initial conditions. This requirement represents another constraint on the choice of the sampling period.

Remark 8.4. *Notice that the stability test function, $F_1(\cdot)$, defined in* (8.17)*, is dependent on several important parameters that influence closed-loop stability, including the size of the plant-model mismatch, the size of the measurement error, the choice of the sampling period and the choice of the control parameters. The stability condition of* (8.22) *represents a constraint that ties all these parameters together, and can therefore be used to assess the interplay between these parameters in shaping the stability region of the sampled-data closed-loop system. For example, for a fixed inter-sample model predictor, a fixed controller, and a fixed measurement error,* (8.22) *can be used to estimate the range of stabilizing sampling periods. Alternatively, for a fixed choice of the sampling period,* (8.22) *can be used to estimate the range of measurement errors that can be tolerated by the closed-loop system for a given model-based controller.*

Remark 8.5. *The condition in* (8.22) *captures the influence of measurement errors on the range of allowable sampling periods. It can be seen, for example, that as the size of the errors increase (i.e., as either $\|\Xi_M\|$ or $\|I - \Xi_M\|$ becomes larger), the range of sampling periods that satisfy* (8.22) *shrinks. In the limit as the bound on the measurement error $\|\Xi_M\|$ shrinks and approaches 1 (or as Ξ_M approaches I), the range of stabilizing sampling periods increases until it attains its maximum feasible range, which is ultimately limited at that point by the size of the plant-model mismatch. Note that in the absence of any measurement errors ($\Xi_M = I$) and in the absence of any plant-model mismatch ($L_w = 0$), and for the case when $\alpha = 1$, any sampling period, no matter how large, satisfies* (8.22)*. Note also that the parameter β, which quantifies the response speed of the closed-loop model state, also influences the stabilizing range of sampling periods, where larger values of β could help expand the feasible range; however, this influence is limited by the size of the measurement error and the plant-model mismatch.*

Remark 8.6. *Another important implication of the result of Theorem 8.1 is that it suggests ways by which the influence of measurement errors on closed-loop stability could be*

mitigated. By inspection of (8.22)*, for example, it can be seen that for large errors, which may destabilize the closed-loop system, reducing the sampling period and/or increasing the control parameter* β *(i.e., making the controller more aggressive) tend to have a restorative stabilizing effect. These measures, however, are ultimately limited by the the size of the errors as well as the plant-model mismatch.*

Remark 8.7. *The result of Theorem 8.1 is developed for the general case where the plant-model mismatch is assumed to be non-vanishing in the sense that the plant and the model do not have the same equilibrium point. In the special case where the uncertainty is vanishing (i.e.,* $\theta(0) = 0$*), the terminal set collapses to the origin. This follows from that fact that* $F_2(\Delta) = 0$ *when* $L_0 = 0$*. In this case, and in lieu of ultimate boundedness, the stability condition of* (8.22) *guarantees asymptotic convergence of the closed-loop state to the origin.*

Remark 8.8. *Since the focus of this work is on the analysis and characterization of the impact of measurement errors on the stability of the sampled-data closed-loop system, it is assumed that a mechanism for the detection and estimation of measurement errors is already in place. While beyond the scope of this work, such mechanism is important since knowledge of the error bound is required for the characterization of the closed-loop stability region. Generally speaking, data-based fault detection and estimation methods could be used for this purpose and are the subject of other research work.*

8.4 Simulation Example

As an illustrative example, we consider in this section a nonlinear diffusion-reaction process described by:

$$\frac{\partial \bar{x}}{\partial t} = \frac{\partial^2 \bar{x}}{\partial x^2} - \beta_U \bar{x} + \beta_T [e^{-\gamma/(1+\bar{x})} - e^{-\gamma}] + \beta_U b(z) u(t) \tag{8.25}$$

subject to the following boundary and initial conditions:

$$\bar{x}(0,t) = \bar{x}(\pi,t) = 0,\ \bar{x}(z,0) = \bar{x}_0(z) \tag{8.26}$$

where $\bar{x}(z,t)$ represents the dimensionless temperature; the manipulated input, u, represents the dimensionless temperature of the coolant; β_T is the dimensionless heat of

reaction, $\gamma = 2$ is the dimensionless activation energy, $\beta_U = 4$ is the dimensionless heat transfer coefficient and $b(z)$ is the control actuator distribution function. It can be verified that for the given process parameter values, the zero steady-state solution of the open-loop system is unstable. The control objective is to stabilize the process temperature at this unstable, spatially-uniform steady-state using a single point-control actuator (with finite support) and a sufficiently large number of point-measurement sensors that provide discretely-sampled temperature measurements.

By solving the eigenvalue problem associated with (8.25)-(8.26) and taking the first eigenvalue as the dominant one, we can obtain the following ODE using Galerkin's method:

$$\dot{a}_1 = \lambda_1 a_1 + g(z_a)u + f(a_1) \tag{8.27}$$

where $\bar{x}(z,t) = \sum_{i=1}^{\infty} a_i(t)\phi_i(z)$, $\phi_i(z) = \sqrt{\frac{2}{\pi}}\sin(iz)$, $\lambda_1 = -1-\beta_u$, $g(z_a) = \beta_U\langle\phi_1(z), b(z)\rangle$, $f(a_1) = \beta_T\langle\phi_1(z), h(a_1)\rangle$, $h(a_1) = e^{-\gamma/(1+a_1\phi(z))} - e^{-\gamma}$ and $\langle\cdot,\cdot\rangle$ is the inner product defined on $\mathcal{H} = L_2(0,\pi)$. An approximate model of (8.27) for controller synthesis is constructed as follows:

$$\dot{\widehat{a}}_1 = \widehat{\lambda}_1\widehat{a}_1 + \widehat{g}(z_a)u + \widehat{f}(\widehat{a}_1) \tag{8.28}$$

where, for simplicity, $\widehat{\lambda}_1 = \lambda_1$, $\widehat{g}(z_a) = 1.6$ and $\widehat{f}(a_1) = (\beta_{Tnom} + \delta_T)\langle\phi_1(z), h(\widehat{a}_1)\rangle$, where $\delta_T = \beta_T - \beta_{Tnom}$ represents the model uncertainty. A model-based feedback linearizing controller is synthesized and takes the form:

$$u = \widehat{g}^{-1}(z_a)[\lambda_c - \widehat{f}(\widehat{a}_1)] \tag{8.29}$$

where the actuator location is set to $z_a = \pi/2$ and λ_c is a controller parameter that is chosen to place the eigenvalue of the closed-loop model at the desired value. In the following subsections, we will characterize the closed-loop stability region and show the influences of the sampling rate, the measurement error, the pole placement and plant-model mismatch on the stability region.

8.4.1 Interplay between sampling rates and measurement errors

The result of Theorem 8.1 suggests that a potential method for stabilizing systems that may become unstable due to measurement errors is to manipulate the sampling period

Δ. In this section, we will investigate the interplay between the sampling rate and the measurement error and show the effect their sizes have on the closed-loop stability region. Figure 8.1 shows the values of the stability test function, $F_1(\Delta)$, as a function of the sampling period Δ for different measurement error values σ. Values above the zero line represent the stable region. Generally, it can be seen that as the measurement error increases, the range of stabilizing sampling periods shrinks; and that, for a given measurement error, closed-loop stability is lost when the sampling period becomes too large. A more detailed representation of these trends is captured by the contour plot in Figure 8.2 where the stability region is the area enclosed by within the zero contour line.

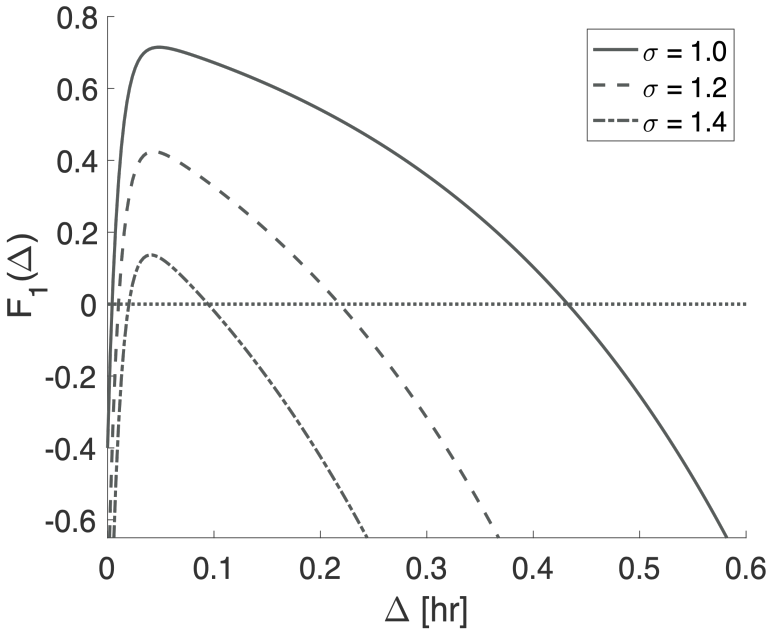

Figure 8.1. A plot of F_1 as a function of the sampling period Δ for different error values with $\delta_T = 25$, $\alpha = 1$ and $\lambda_c = 25$

It can be seen that the closed-loop stability region shrinks as σ deviates from 1 (the error-free case). The region also tends to shrink as the sampling period increases. An important implication of these trends is that if the measurement error drives the system outside the stability region, it may be possible to mitigate the problem and restore stability by reducing the sampling period.

To confirm these trends, a simulation of the closed loop system is shown in Figures 8.3 - 8.8. In both scenarios, the sampling period was fixed at $\Delta = 0.4\ hr$; however, the

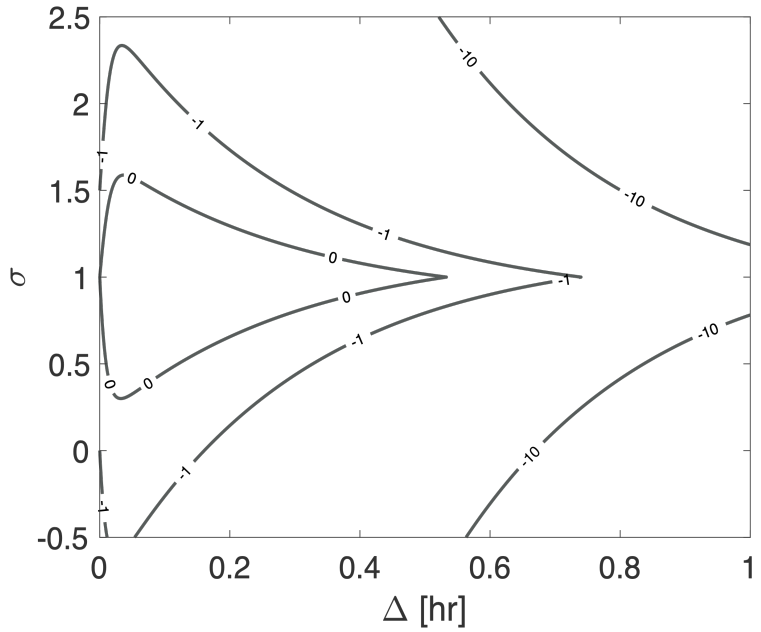

Figure 8.2. A contour plot of F_1 as a function of the measurement error σ and the sampling period Δ with $\delta_T = 25$, $\alpha = 1$ and $\lambda_c = 25$

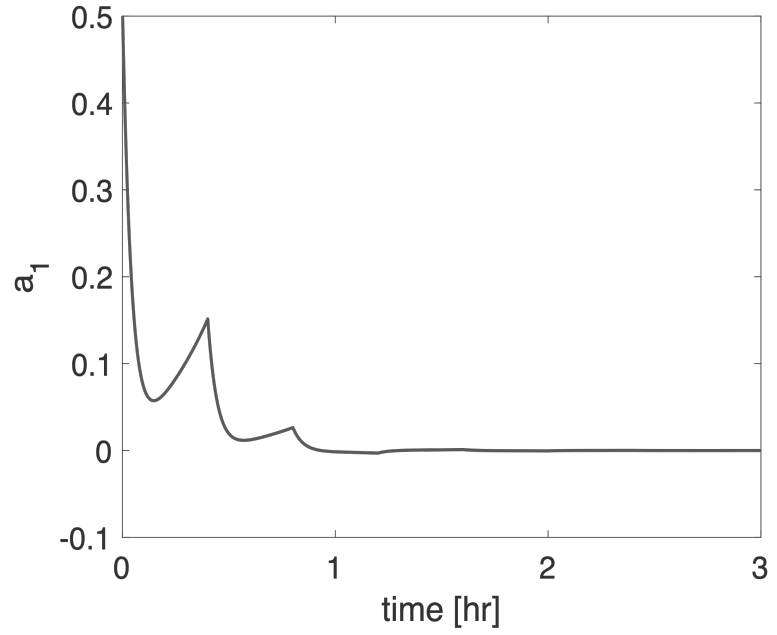

Figure 8.3. Evolution of dominant eigenmode for the closed-loop system for $\Delta = 0.4hr$ $\sigma = 1.2$, $\delta_T = 25$, $\alpha = 1$ and $\lambda_c = 25$

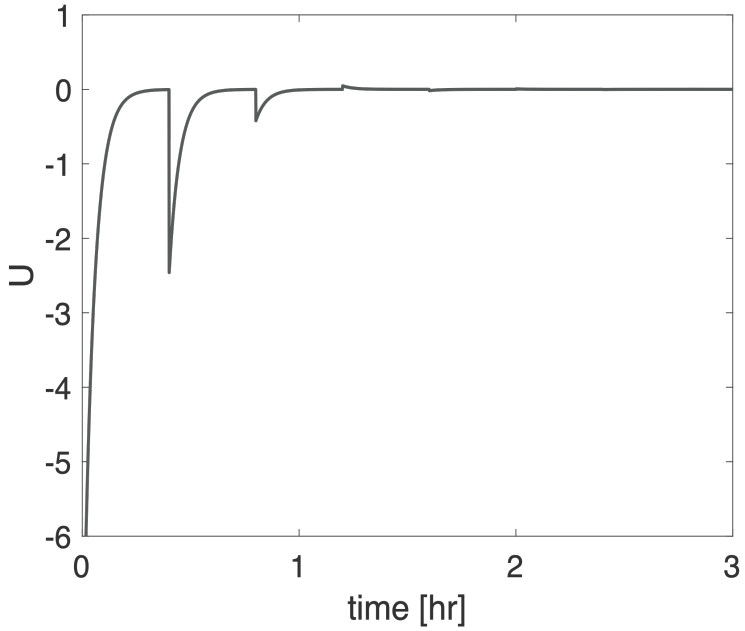

Figure 8.4. Manipulated input profile for the closed-loop system for $\Delta = 0.4hr$ $\sigma = 1.2$, $\delta_T = 25$, $\alpha = 1$ and $\lambda_c = 25$

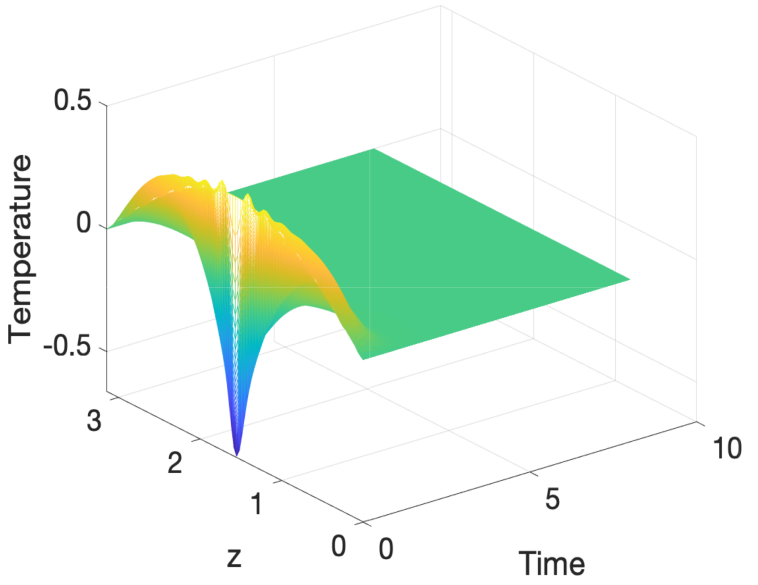

Figure 8.5. Closed-loop state profile for $\Delta = 0.4hr$ $\sigma = 1.2$, $\delta_T = 25$, $\alpha = 1$ and $\lambda_c = 25$

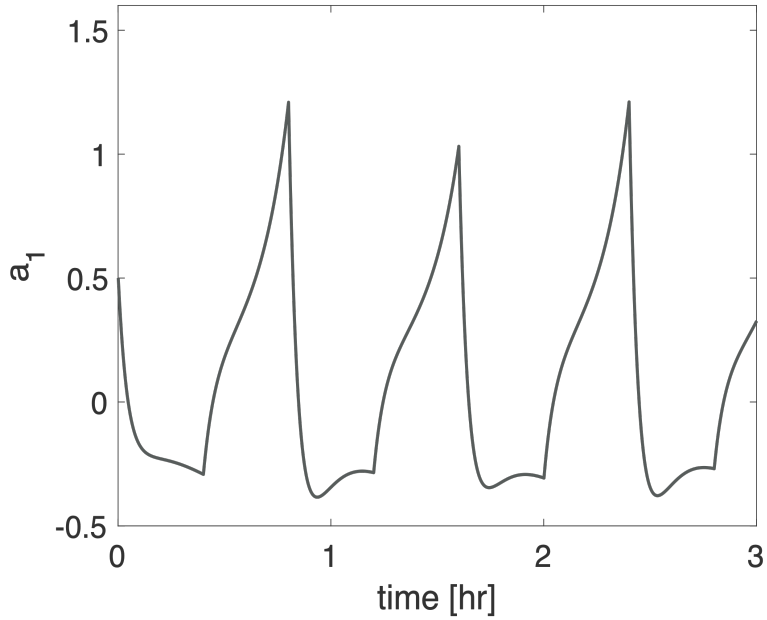

Figure 8.6. Evolution of dominant eigenmode for the closed-loop system for $\Delta = 0.4hr$ $\sigma = 1.7$, $\delta_T = 25$, $\alpha = 1$ and $\lambda_c = 25$

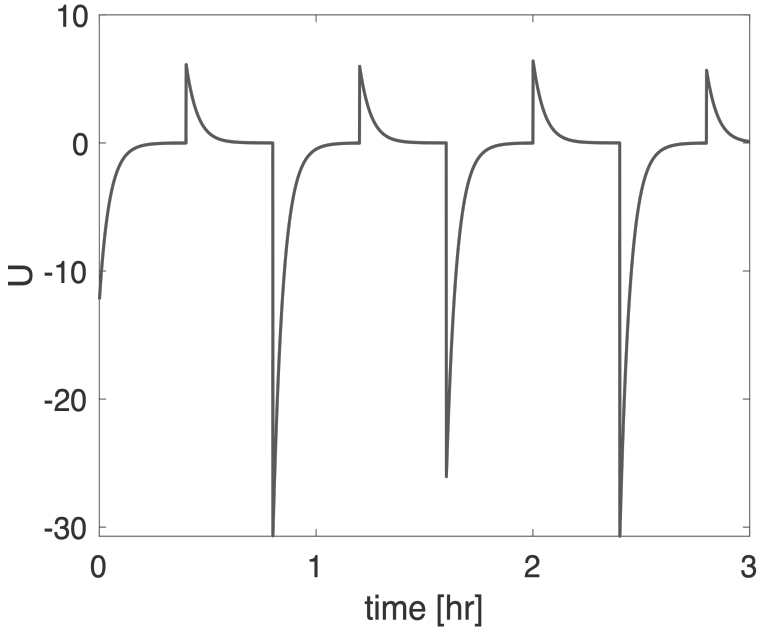

Figure 8.7. Manipulated input profile for the closed-loop system for $\Delta = 0.4hr$ $\sigma = 1.7$, $\delta_T = 25$, $\alpha = 1$ and $\lambda_c = 25$

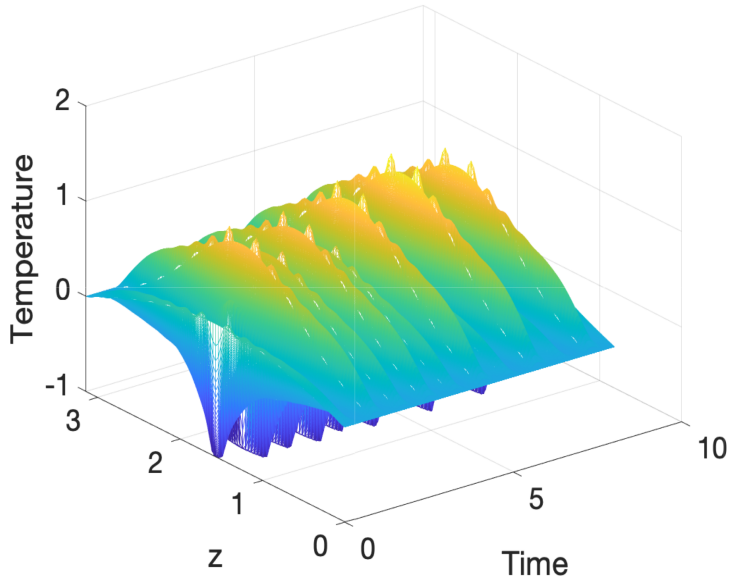

Figure 8.8. Closed-loop state profile for $\Delta = 0.4hr$ $\sigma = 1.7$, $\delta_T = 25$, $\alpha = 1$ and $\lambda_c = 25$

measurement error was varied from $\sigma = 1.2$ (a stable operating point) as shown in Figures 8.3 - 8.5 to $\sigma = 1.7$ (an unstable operating point) as shown in Figures 8.6 - 8.8. It can be seen that the closed-loop system becomes unstable when operating outside the stability region.

8.4.2 Interplay between pole-placement and measurement errors

An alternative approach to stabilizing a system that can potentially become unstable due to measurement errors is through the manipulation of the pole placement parameter λ_c. To understand the the interplay between λ_c and σ, we generated a plot of the stability test function, F_1, as a function of the measurement error, σ, for different λ_c values. This is shown in Figure 8.9. It can observed that the range of tolerable errors (for which $F_1(\sigma) > 0$) expands as λ_c increases (i.e., as the controller becomes more aggressive).

This trend is further confirmed by the contour plot in Figure 8.10 which shows the closed-loop stability region (the area within the zero contour line) as a function of σ and λ_c. This plot shows that as the error size increases (greater departure of σ from 1), the minimum λ_c need for stability increases as well indicating that the closed-loop pole needs

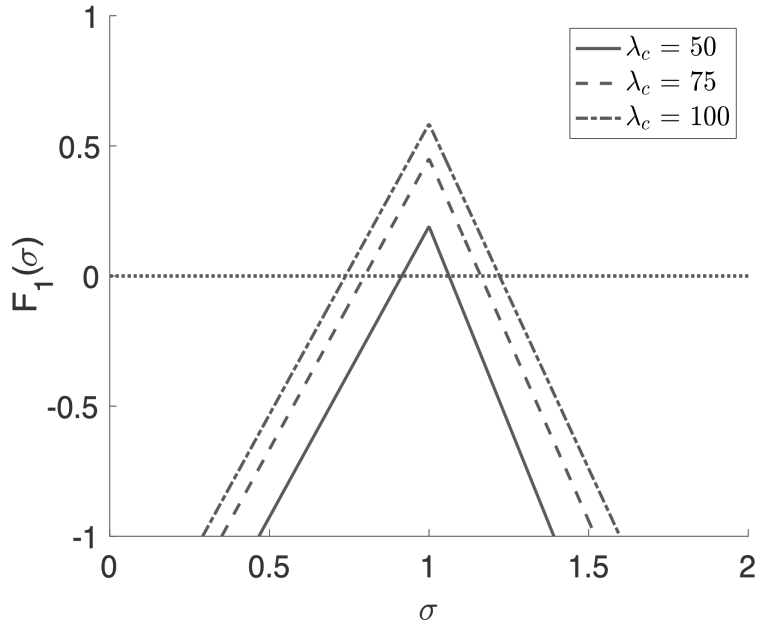

Figure 8.9. A plot of F_1 as a function of the measurement error σ, for different λ_c values, with $\Delta = 0.3$, $\delta_T = 25$ and $\alpha = 1$

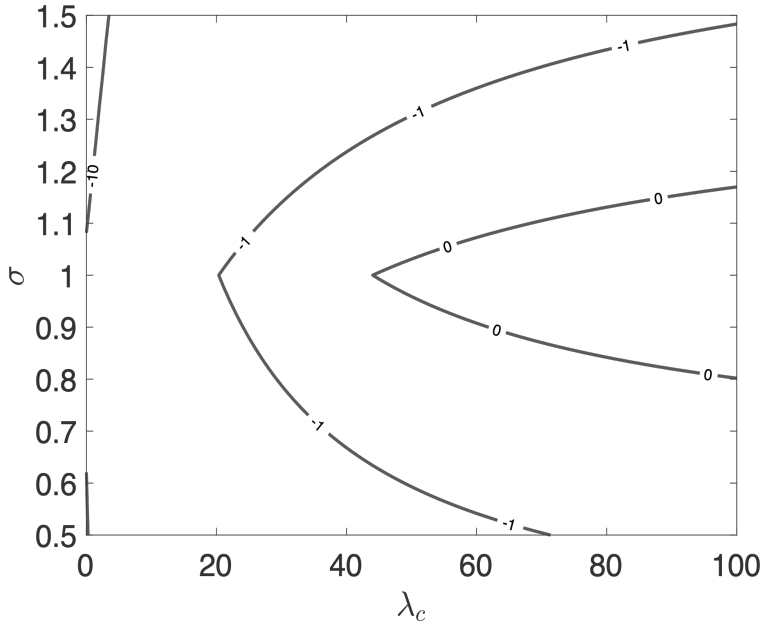

Figure 8.10. A contour plot of F_1 values as a function of the measurement error, σ, and λ_c, with $\Delta = 0.3$, $\delta_T = 25$ and $\alpha = 1$

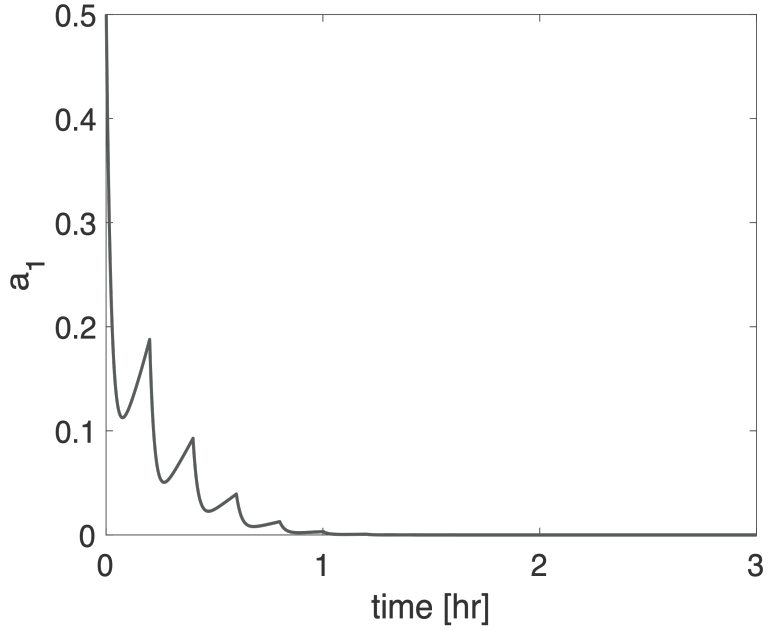

Figure 8.11. Evolution of dominant eigenmode for the closed-loop system for $\Delta = 0.3$, $\lambda_c = 75$, $\sigma = 1.0$, $\delta_T = 25$ and $\alpha = 1$

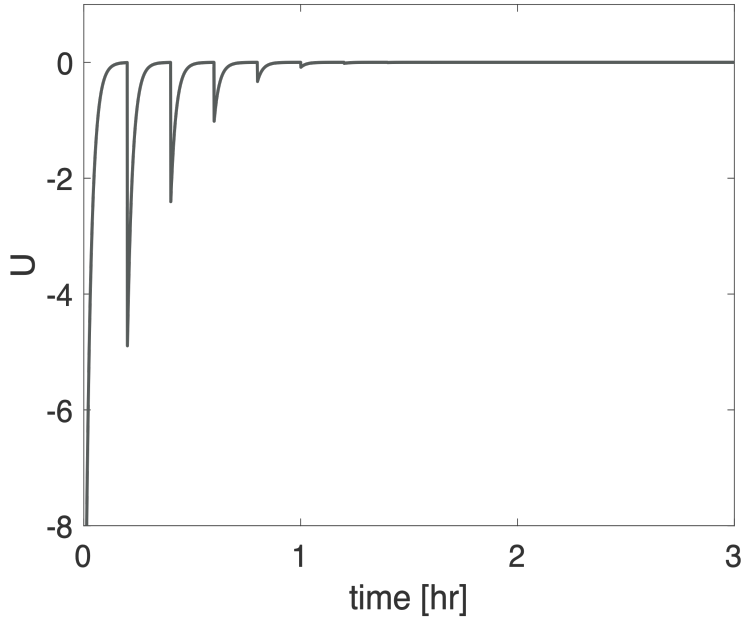

Figure 8.12. Manipulated input profile for the closed-loop system for $\Delta = 0.3$, $\lambda_c = 75$, $\sigma = 1.0$, $\delta_T = 25$ and $\alpha = 1$

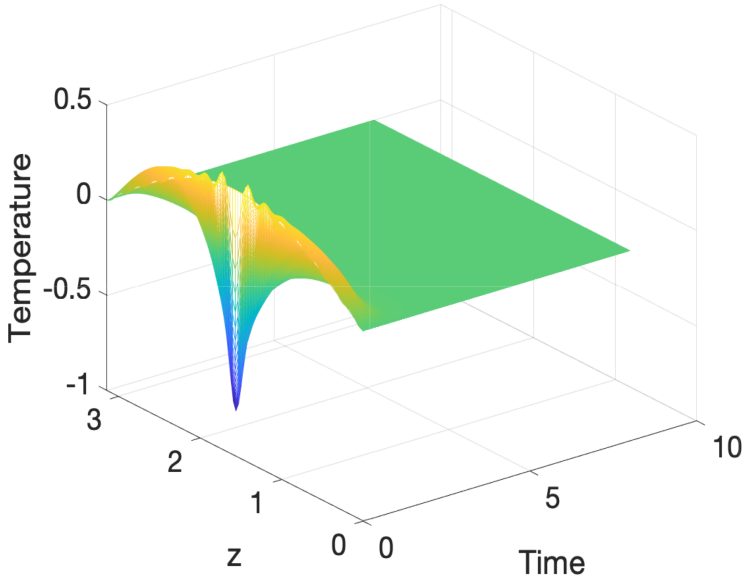

Figure 8.13. Closed-loop state profile for $\Delta = 0.3$, $\lambda_c = 75$, $\sigma = 1.0$, $\delta_T = 25$ and $\alpha = 1$

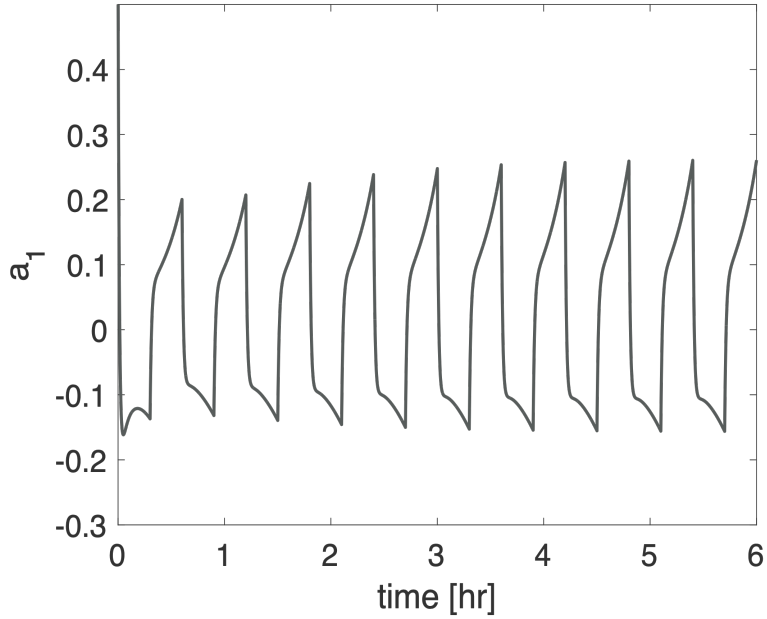

Figure 8.14. Evolution of dominant eigenmode for the closed-loop system for $\Delta = 0.3$, $\lambda_c = 75$, $\sigma = 1.3$, $\delta_T = 25$ and $\alpha = 1$

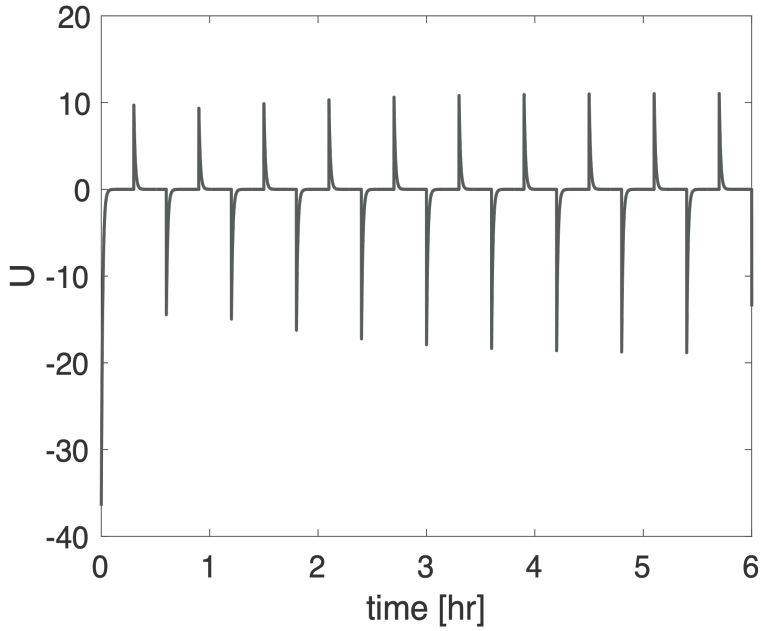

Figure 8.15. Manipulated input profile for the closed-loop system for $\Delta = 0.3$, $\lambda_c = 75$, $\sigma = 1.3$, $\delta_T = 25$ and $\alpha = 1$

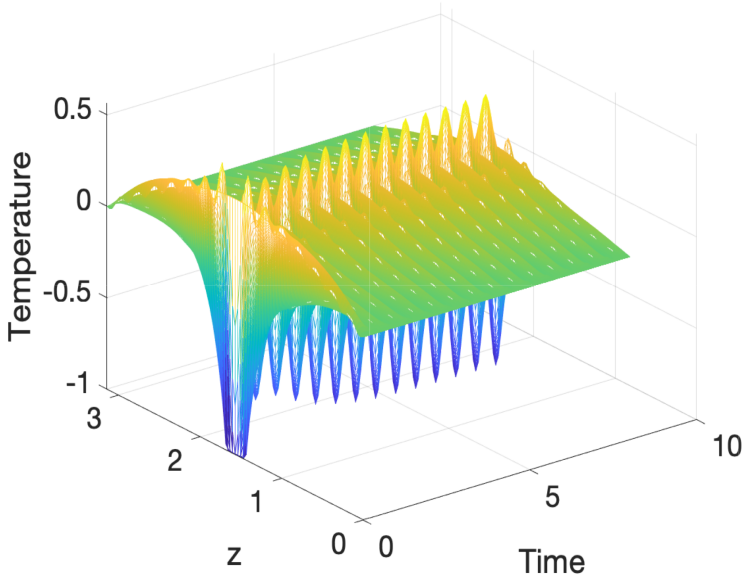

Figure 8.16. Closed-loop state profile for $\Delta = 0.3$, $\lambda_c = 75$, $\sigma = 1.3$, $\delta_T = 25$ and $\alpha = 1$

to be pushed further to the left in the left half of the complex plane. It can also be seen that there is a minimum value for λ_c below which the closed-loop system is unstable even in the absence of measurement errors. An implication of these trends is that if the measurement error is severe enough to drive the closed-loop system outside the stability region, one may be able to mitigate this effect and restore stability by increasing λ_c.

Illustrative simulation results for the cases when σ and λ_c are chosen such that the operating point is within and outside the stable region are shown in Figures 8.11 - 8.16. It can be seen that the closed-loop state profile in both cases follows the expected outcome based on the stability trends depicted in Figures 8.9-8.10.

8.5 Conclusions

In this chapter, a model-based approach for sampled-data feedback control of spatially-distributed processes modeled by nonlinear PDEs with measurement errors and model uncertainty was presented. The interplay between the sampling rate, the measurement errors, the model uncertainty and the controller design parameters in shaping the closed-loop stability region was analyzed and explicitly characterized. The implications of the results for the development of strategies to mitigate the effect of measurement errors were discussed and demonstrated using a simulated diffusion-reaction process example.

Chapter 9

Machine Learning Based Classification and Mitigation of Cyberattacks in Model-Based Networked Process Control Systems

In this chapter, we present in an integrated approach for the active detection, identification and mitigation of cyberattacks in a class of networked control systems subject to false data injection cyberattacks. The approach brings together tools from supervised machine learning, which are used for attack detection and identification, and model-based networked controller stabilization techniques, which are used for attack mitigation. Initially, a model-based networked control architecture in which the sensor and the controller communicate over a resource-limited communication medium is designed, and the networked closed-loop stability region is explicitly characterized in terms of the measurement error resulting from the falsified measurement cyberattack, as well as the communication rate and the controller design parameters. This characterization is obtained by modeling the cyberattack in the closed-loop system formulation thus making it possible to identify the range of feasible operating conditions that guarantee robust stability under the attack. This characterization reveals the key parameters that can be used to actively mitigate the effects of these attacks when they arise. The implementation of these mitigation measures requires knowledge of the existence of the cyberattack as well as an estimate of its

magnitude. To that end, we utilize machine learning methods (e.g., [53], [54], [55], [56]) to build a neural network (NN) based detection system. To detect both the existence and the magnitude of the cyberattack, the NN is trained using data obtained from simulating the system under normal operation and the system under different cyberattack magnitudes. While NN models are generally utilized as classification tools, training the NN with different cyberattack magnitudes allows for the classification of both the existence of a cyberattack and the approximate magnitude of this attack. Finally, the implementation of the integrated cyberattack identification and mitigation strategies is demonstrated using a chemical process example.

9.1 Preliminaries

9.1.1 Process system description

We consider process systems described by continuous-time nonlinear ordinary differential equation systems with the following state-space representation:

$$\dot{x} = f(x) + g(u), \tag{9.1}$$

where $x \in \mathbb{R}^n$ is the vector of process state variables, $u \in \mathbb{R}^m$ is vector of manipulated inputs, and $f(\cdot)$ and $g(\cdot)$ and sufficiently smooth $n \times 1$ nonlinear vector functions given by:

$$\begin{aligned} f(x) &= \widehat{f}(x) + w(x) \\ g(u) &= \widehat{g}(u) + m(u) \end{aligned} \tag{9.2}$$

where $\widehat{f}(\cdot)$, $w(\cdot)$, $\widehat{g}(\cdot)$, and $m(\cdot)$ are sufficiently smooth nonlinear functions that satisfy the following growth bounds for all x and y in some compact (closed and bounded) neighborhood of the origin:

$$\begin{aligned} \| \widehat{f}(x) - \widehat{f}(y) \| &\leq L_f \| x - y \| \\ \| \widehat{g}(x) - \widehat{g}(y) \| &\leq L_g \| x - y \| \\ \| w(x) - w(y) \| &\leq L_w \| x - y \| \\ \| m(x) - m(y) \| &\leq L_m \| x - y \| \end{aligned} \tag{9.3}$$

where L_f, L_g, L_w and L_m are positive real constants. The functions $\widehat{f}(\cdot)$ and $\widehat{g}(\cdot)$ represent the known (or certain) part of the process dynamics, whereas the functions $w(\cdot)$ and $m(\cdot)$ capture the uncertain part (i.e., the plant-model mismatch). Without loss of generality, it is assumed that the origin of, $(x, u) = (0, 0)$, is an equilibrium point of the nominal system, i.e., $\widehat{f}(0) = \widehat{g}(0) = 0$.

The control architecture under consideration is a networked control system in which the embedded controller of a process communicates with the neighboring process' embedded controllers at discrete times over a shared, resource-constrained, communication medium. The objective is to design a model-based networked process control system that has well-characterized robustness margins in the presence of false data injection cyberattacks and to develop active strategies for mitigating the impact of such attacks.

False data injection cyberattacks are ones wherein the attacker deliberately falsifies the sensor measurements sent to the feedback controller, resulting in incorrect signals to the control actuators [28]. These types of attacks can be modeled as multiplicative attacks and take the following form:

$$y(t) = \Phi\, x(t) \tag{9.4}$$

where $y \in \mathbb{R}^n$ is the vector of falsified state measurements and $\Phi = \text{diag}\{\phi_i\} \in \mathbb{R}^{n\times n}$ is a diagonal matrix (denoted as the cyberattack matrix for the rest of the paper) where ϕ_i captures the extent of the cyberattack on the i-th measurement sensor. A value of $\phi_i = 1$ represents a healthy sensor measurement with no cyberattack present, while values of $\phi_i \neq 1$ represent the existence of a cyberattack of magnitude ϕ_i. If left unaddressed, the falsification of process state measurements can have detrimental effects of the stability and performance of the closed-loop system and must therefore be addressed.

9.1.2 Illustrative Example

In this section, we introduce a benchmark example of a process composed of two non-isothermal continuous stirred-tank reactors (CSTRs) in series with a recycle stream as shown in the process flow diagram in Figure 9.1. This example will be used in subsequent sections to demonstrate the implementation of the model-based networked control system design discussed in Section 9.2, and to evaluate the efficacy of the integrated cyberattack

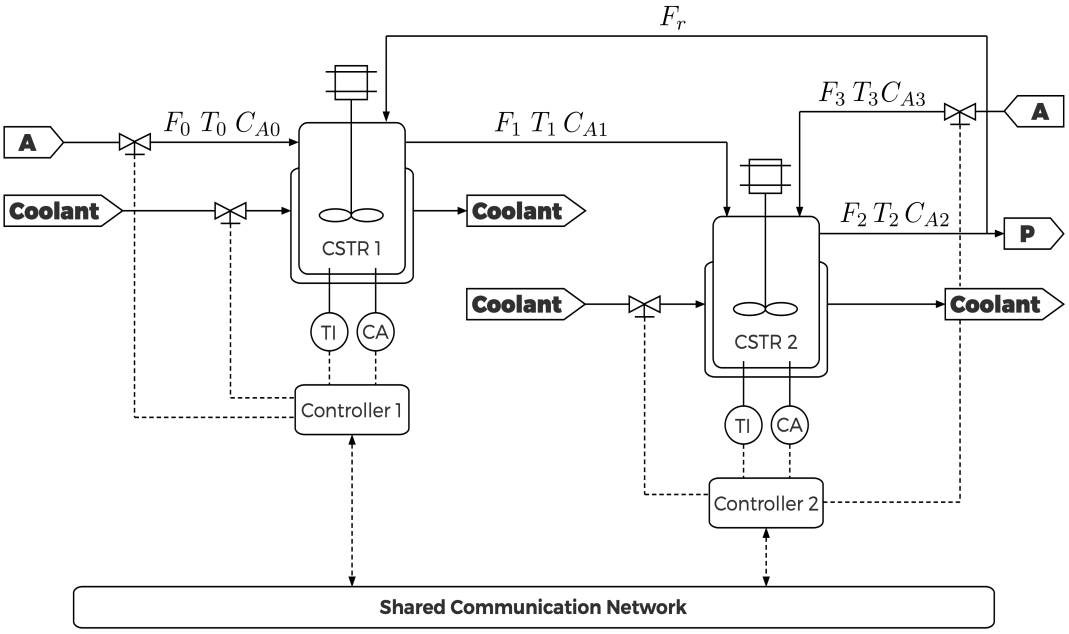

Figure 9.1. Process flow diagram of two interconnected non-isothermal continuous stirred tank reactors

detection and mitigation framework to be proposed. The reactions occurring in this process are three parallel irreversible exothermic reactions that consume reactant A. The temperature is stabilized in each reactor through a heat exchange jacket. A model of this process that is based on first principles can be described by the following differential equations:

$$\begin{aligned}
\dot{T}_1 &= \frac{F_0}{V_1}(T_0 - T_1) + \frac{F_r}{V_1}(T_2 - T_1) + \sum_{i=1}^{3} G_i(T_1)C_{A1} + \frac{Q_1}{\rho c_p V_1} \\
\dot{C}_{A1} &= \frac{F_0}{V_1}(C_{A0} - C_{A1}) + \frac{F_r}{V_1}(C_{A2} - C_{A1}) - \sum_{i=1}^{3} R_i(T_1)C_{A1} \\
\dot{T}_2 &= \frac{F_1}{V_2}(T_1 - T_2) + \frac{F_3}{V_2}(T_3 - T_2) + \sum_{i=1}^{3} G_i(T_2)C_{A2} + \frac{Q_2}{\rho c_p V_2} \\
\dot{C}_{A2} &= \frac{F_1}{V_2}(C_{A1} - C_{A2}) + \frac{F_3}{V_2}(C_{A3} - C_{A2}) - \sum_{i=1}^{3} R_i(T_2)C_{A2}
\end{aligned} \tag{9.5}$$

where $R_i(T_j) = k_{i0}\, exp(\frac{-E_i}{RT_j})$, $G_i(T_j) = \frac{-\Delta H_i}{\rho c_p} R_i(T_j)$, for $j = \{1, 2\}$. T_j, C_{Aj}, F_j, V_j and Q_j represent the temperature, reactant concentration, flow rate, volume, and heat rate for the j-th reactor, respectively. The process parameters ΔH_i, k_i, E_i, $i = \{1, 2, 3\}$, represent the enthalpies, pre-exponential constants and activation energies of the three reactions,

respectively; and c_p and ρ represent the heat capacity and density, respectively. Given the process parameters in Table 9.1, and specifying that at steady-state $Q_1 = Q_2 = 0$, $C_{A0} = C^s_{A0}$, $C_{A3} = C^s_{A3}$ and $r = 0.5$, where r is the recycle rate, it can be shown that the process has three steady-states; two locally asymptotically stable and one unstable at $T^s_1 = 457.9K$, $C^s_{A1} = 1.77kmol/m^3$, $T^s_2 = 415.5K$ and $C^s_{A2} = 1.75kmol/m^3$.

Table 9.1. Process parameters and steady-state values

F_0	=	$4.998 \quad m^3/h$	ΔH_3	=	$-5.4 \times 10^4 \quad kJ/kmol$
F_1	=	$39.996 \quad m^3/h$	k_1	=	$3.0 \times 10^6 \quad hr$
F_3	=	$30.0 \quad m^3/h$	k_2	=	$3.0 \times 10^5 \quad hr$
F_r	=	$34.998 \quad m^3/h$	k_3	=	$3.0 \times 10^5 \quad hr$
V_1	=	$1.0 \quad m^3$	E_1	=	$5.0 \times 10^4 \quad kJ/kmol$
V_2	=	$3.0 \quad m^3$	E_2	=	$7.53 \times 10^4 \quad kJ/kmol$
R	=	$8.314 \quad kJ/kmol\,K$	E_3	=	$7.53 \times 10^4 \quad kJ/kmol$
T_0	=	$300.0 \quad K$	ρ	=	$1000.0 \quad kg/m^3$
T_3	=	$300.0 \quad K$	c_p	=	$0.231 \quad kJ/kg\,K$
C^s_{A0}	=	$4.0 \quad kmol/m^3$	T^s_1	=	$457.9 \quad K$
C^s_{A3}	=	$2.0 \quad kmol/m^3$	C^s_{A1}	=	$1.77 \quad kmol/m^3$
ΔH_1	=	$-5.0 \times 10^4 \quad kJ/kmol$	T^s_2	=	$415.5 \quad K$
ΔH_2	=	$-5.2 \times 10^4 \quad kJ/kmol$	C^s_{A2}	=	$1.75 \quad kmol/m^3$

The control objective is to stabilize the process at the open-loop unstable steady-state by manipulating Q_1, C_{A0}, Q_2 and C_{A3}, while actively detecting and mitigating potential cyberattacks.

To help illustrate the key ideas of the proposed framework, we focus in this case study

on cyberattacks taking place in the temperature sensor in the first reactor, such that:

$$\Phi = \begin{bmatrix} \phi_1 & 0 & 0 & 0 \\ 0 & \phi_2 & 0 & 0 \\ 0 & 0 & \phi_3 & 0 \\ 0 & 0 & 0 & \phi_4 \end{bmatrix} = \begin{bmatrix} \phi_1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9.6}$$

Attacks in the other measurement sensors (i.e., ϕ_2, ϕ_3, ϕ_4), as well as simultaneous attacks in multiple sensors, can also be considered within the proposed framework, but are not pursued in this study.

9.2 Model-Based Networked Controller Design

To achieve the desired control objective, we present in this section a model-based control strategy in which a model of the plant is embedded in the control system, and is utilized to generate estimates of the plant states when communication between the sensors and the controller is suspended. The control action is calculated based on these estimates. When communication is re-established, the model states are updated using the state measurements transmitted over the shared communication medium. This strategy enables the controller to operate independent of the sensor measurements (for certain periods of time) and adds some leverage for enhanced cyber-security.

In designing this model-based control system, the first step is to use a nominal model of the process to synthesize a nonlinear feedback controller that stabilizes the closed-loop model at the origin. Once the controller is synthesized, its closed-loop stability properties are analyzed and explicitly characterized in the presence of both communication constraints and cyberattacks.

Based on the process description in (9.1), we consider the following nominal model which captures the known part of the process dynamics:

$$\dot{\widehat{x}} = \widehat{f}(\widehat{x}) + \widehat{g}(u) \tag{9.7}$$

where $\widehat{x} \in \mathbb{R}^n$ is the vector of model states, and the functions $\widehat{f}(\cdot)$ and $\widehat{g}(\cdot)$ are sufficiently smooth nonlinear functions that satisfy the growth bounds in (9.3) over some compact

neighborhood of the origin. Based on the model in (9.7) a nonlinear feedback control of the following general form:

$$u = k(\widehat{x}) \tag{9.8}$$

can be designed to exponentially stabilize the origin of the closed-loop model of (9.7)-(9.8), where $k(\cdot)$ is a sufficiently smooth nonlinear function that satisfies a growth bound of the form:

$$\| k(x) - k(y) \| \leq L_k \| x - y \| \tag{9.9}$$

for all x, y in Ω, where Ω is a compact neighborhood of the origin, and L_k is a positive constant. In the interest of generalizing the results presented in this work, we will not restrict the choice of the function $k(\cdot)$ to any particular class of nonlinear control laws; instead we will assume that a suitable control law with the desired closed-loop stability properties exists, such that the state of the closed-loop model of (9.7)-(9.8) satisfies an exponentially decreasing bound of the form:

$$\| \widehat{x}(t) \| \leq \alpha \| \widehat{x}(t_0) \| e^{-\beta(t-t_0)} \tag{9.10}$$

for some $\alpha \geq 1$, $\beta > 0$, for all $\widehat{x}(t_0) \in \Omega \subset \mathbb{R}^m$, where Ω is a compact set given by $\Omega := \{\widehat{x} \in \mathbb{R}^m : \| \widehat{x} \| \leq M\}$.

The implementation of the model-based controller of (9.7)-(9.8) in the presence of periodic sensor-controller communication and falsified state measurement attacks can now be described as follows:

$$\begin{aligned} u(t) &= k(\widehat{x}(t)),\ t \in [t_k, t_{k+1}) \\ \dot{\widehat{x}}(t) &= \widehat{f}(\widehat{x}(t)) + \widehat{g}(k(\widehat{x}(t)) \\ \widehat{x}(t_k) &= y(t_k) = \Phi x(t_k),\ k \in \{0, 1, 2, \cdots\} \end{aligned} \tag{9.11}$$

where t_k is the k-th update time at which the model state is updated, and $h = t_{k+1} - t_k$ is the update period, which is the time interval between two consecutive update times (and is assumed to be constant for the purposes of this study). Note that at the update instances, t_k, the model state is updated using the falsified state measurement, y, which contains the cyberattack matrix Φ that captures the size of the attack. This formulation implies that the model state is directly impacted by the cyberattack only at the update instances.

The impact of the falsified model state updates on the stability of the closed-loop system is analyzed in the next section.

9.3 Closed-loop Stability Analysis

The objective of this section is to characterize the stability properties of the networked closed-loop system in terms of the sensor-controller communication rate and the size of the measurement falsification attack. This characterization will serve as the basis for the development of cyberattack mitigation strategies. We begin by formulating the networked closed-loop system and then proceed to present a sufficient condition for closed-loop stability.

9.3.1 Networked closed-loop system formulation

To analyze closed-loop stability, the networked closed-loop system must be first formulated. To this end, we define the estimation error:

$$e(t) = x(t) - \widehat{x}(t)$$

where e represents the difference between the process state and the model state. It can be shown that the closed-loop system of (9.1) subject to (9.11) can be formulated as a combined discrete-continuous system of the following general form:

$$\begin{aligned} \dot{x}(t) &= F(x(t), e(t)) \\ \dot{e}(t) &= \widehat{F}(x(t), e(t)),\ t \in (t_k, t_{k+1}) \\ e(t_k) &= (I - \Phi)x(t_k),\ k \in \{0, 1, 2, \cdots\} \end{aligned} \tag{9.12}$$

where the process state evolves continuously in time, while the model estimation error is reset at each update instance, and $F(\cdot)$ and $\widehat{F}(\cdot)$ are smooth nonlinear functions that depend on the process and model dynamics, the plant-model mismatch, and the control law. It is important to note that in the absence of cyberattacks, the cyberattack matrix is the identity matrix, i.e., $\Phi = I$, and the model estimation error is reset to zero.

9.3.2 Sufficient condition for closed-loop stability

To achieve closed-loop stability with minimal communication, it is important to characterize the maximum allowable update period for the control system. This corresponds

to the minimum rate at which state measurements need to transmitted by the sensor to the controller. Owing to the presence of falsified measurements, it is expected that this parameter will be dependent on the size of the measurement error introduced by the data falsification attack. The following theorem provides a sufficient condition for the stability of the networked closed-loop system in terms of all the relevant system and controller parameters, including the model state update period, the model uncertainty, the cyberattack matrix and the choice of the control law.

Theorem 9.1. *Consider the closed-loop system of* (9.1) *subject to the control law of* (9.11) *and the initial condition* $x(t_0) = \widehat{x}(t_0) \in \Omega$. *If the update period is chosen to satisfy the following condition:*

$$F_1(h) := 1 - \alpha \parallel \Phi \parallel \left[e^{-\beta h} + \Big(e^{L_e h} - e^{-\beta h} \Big) \frac{L_\theta}{\beta + L_e} \right] - \parallel I - \Phi \parallel e^{L_e h} > 0, \tag{9.13}$$

where $L_e = L_f + L_w$, $L_\theta = L_w + L_m L_k$, *and* $L_0 = \parallel w(0) \parallel + \parallel m(0) \parallel$, *then the state of the networked closed-loop system is bounded and its norm satisfies the decreasing sequence:*

$$\parallel x(t_{k+1}^-) \parallel \; < \; \parallel x(t_k) \parallel, \;\; \forall \parallel x(t_k) \parallel \; > r(h) \tag{9.14}$$

for $k \in \{0, 1, 2, \cdots\}$, *where* $x(t_{k+1}^-) = \lim_{t \to t_{k+1}^-} x(t_{k+1})$, $r(h) = F_2(h)/F_1(h)$ *and* $F_2(h)$ *is defined as:*

$$F_2(h) = \frac{L_0}{L_e} \Big(e^{L_e h} - 1 \Big) \tag{9.15}$$

Remark 9.1. *According to Theorem 9.1, if the update period for the model state is chosen such that the stability test function is positive, i.e.,* $F_1(h) > 0$, *the norm of the closed-loop state is guaranteed to decrease at successive update times, as long as it is outside some terminal neighborhood of the origin (i.e.,* $\parallel x(t_k) \parallel \; > F_2(h)/F_1(h)$*). The size of this terminal neighborhood is dependent on the choice of the update period. The immediate implication of this result is that the sampled closed-loop state,* $x(t_k)$, *is guaranteed to converge, in finite time, to the terminal neighborhood, where it remains confined for all future times. Note that the continuous closed-loop state, i.e.,* $x(t)$, *is also guaranteed to be ultimately bounded and converge in finite-time to a terminal set around the origin.*

However, the terminal set in this case is larger than that for the sampled state owing to the fact that between consecutive update times, the closed-loop state can grow a certain amount (see (B.9) *in Appendix B). This growth, however, is bounded since the update period is finite.*

Remark 9.2. *The stability test function* $F_1(\cdot)$*, which is defined in* (9.13)*, is dependent on several key parameters that influence closed-loop stability. These include the size of the plant-model mismatch (which is captured by* L_w *and* L_m*), the magnitude of the cyberattack (which is captured by* $\| \Phi \|$ *and* $\| I - \Phi \|$*), the choice of the update period (*h*), and the choice of the controller design parameters (which is reflected in the values of* α *and* β*). The stability condition of* (9.13) *can be viewed as a constraint that links all these parameters together, and as such can be used to analyze the various interplays between these parameters in shaping the stability region of the closed-loop system. For example, for a given model-based controller design and a given update period, the condition in* (9.13) *can be used to estimate the range of cyberattack magnitudes that can be tolerated by the closed-loop system (i.e., attacks that do not cause instability). Conversely, for a fixed model-based controller design and a fixed cyberattack magnitude, one can use the condition in* (9.13) *to estimate the range of stabilizing update periods. Once can also use the stability condition to explore a host of other important interplays, including the interplay between the size of the plant-model mismatch size and the update period, the interplay between the magnitude of the cyberattack magnitude and the choice of control parameters.*

Remark 9.3. *To gain some insight into how the stability condition of* (9.13) *captures the influence of cyberattacks on the range of allowable update periods, we note that as the magnitude of the cyberattack increases (i.e., as either* $\| \Phi \|$ *or* $\| I - \Phi \|$ *becomes larger), the range of update periods that ensure positivity of the stability test function shrinks. In the limit as the bound on the cyberattack matrix* $\| \Phi \|$ *shrinks and approaches 1 (or as* Φ *approaches* I*), the range of stabilizing update periods increases until it reaches its maximum feasible range, which is ultimately limited at that point by the size of the plant-model mismatch. Note that in the absence of cyberattacks (i.e, when* $\Phi = I$*) and in the absence of any plant-model mismatch (i.e., when* $L_w = L_m = 0$*), and for the case*

when $\alpha = 1$, any update period, no matter how large, satisfies (9.13). *Note also that the parameter β, which quantifies the response speed of the closed-loop model state, also influences the stabilizing range of update periods, where larger values of β could provide some leverage to help expand the feasible range. However, this influence is ultimately limited by how large the magnitudes of the cyberattack and the plant-model mismatch are.*

Remark 9.4. *In relation to the problem of handling cyberattacks, a key significance of the result of Theorem 9.1 is that it suggests ways by which the influence of cyberattacks on closed-loop stability could be mitigated. By inspecting the stability condition of* (9.13)*, for example, it can be seen that for large cyberattack magnitudes, which may destabilize the closed-loop system, reducing the update period and/or increasing the control parameter β (i.e., making the controller more aggressive) tend to have a restorative stabilizing effect. These measures, however, are ultimately limited by the the magnitude of the cyberattack as well as the size of the plant-model mismatch.*

Remark 9.5. *The result of Theorem 9.1 applies to the general case where the plant-model mismatch is assumed to be non-vanishing in the sense that the plant and the model do not have the same equilibrium point. In the special case where the uncertainty is vanishing (i.e., $w(0) = m(0) = 0$), the terminal set collapses to the origin. This follows from that fact that $F_2(h) = 0$ when $L_0 = 0$. In this case, the stability condition of* (9.13) *guarantees not only boundedness of the closed-loop trajectories, but also asymptotic convergence of the closed-loop state to the origin.*

9.3.3 Application to the illustrative example

To gain an understanding of how cyberattacks influence the stability of the networked closed-loop system, we apply in this section the results of Theorem 9.1 to the illustrative example introduced in Section 9.1.2. This will allow us to determine the feasible range of operating parameters that stabilize the networked closed-loop system close to the desired steady-state subject to cyberattacks.

Following our framework outline in Figure 9.7, we must first design the model-based

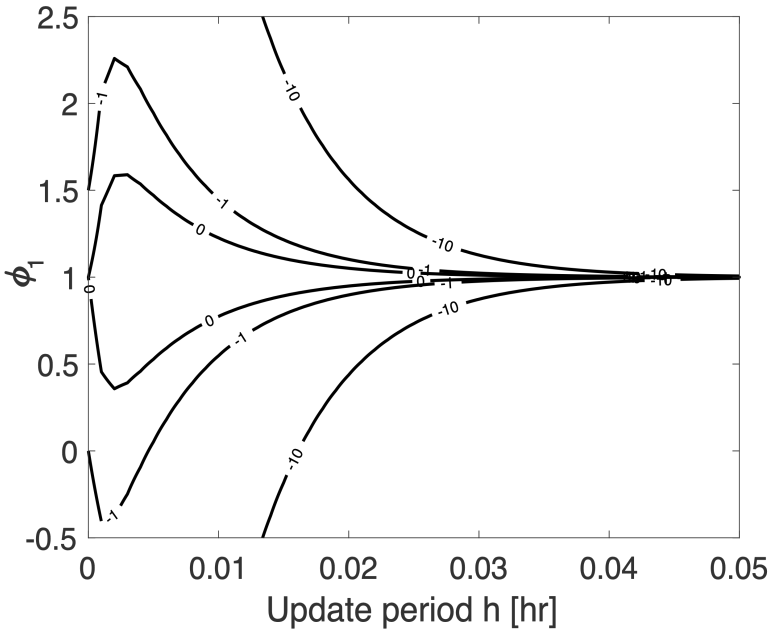

Figure 9.2. Stability region contour plot that demonstrates the dependence of $F_1(h)$ on ϕ_1 and h

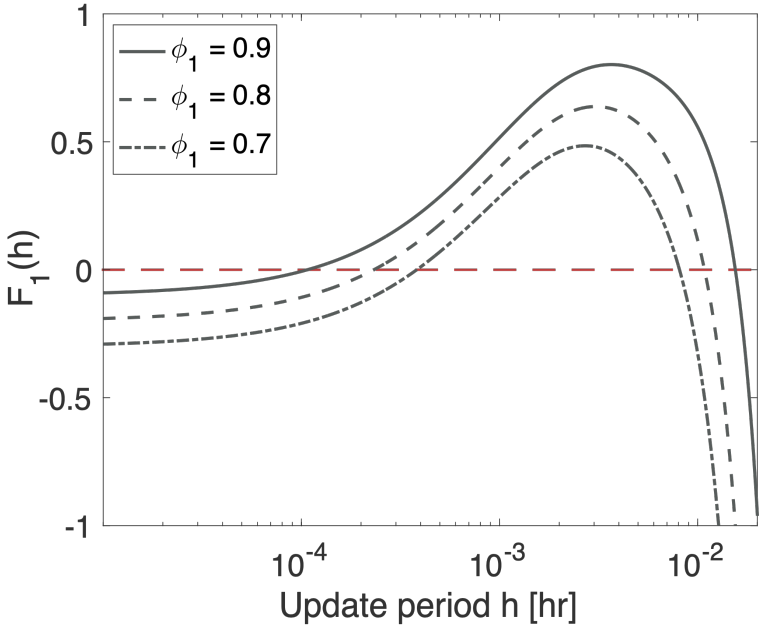

Figure 9.3. Dependence of $F_1(h)$ on h for specific cyberattack magnitudes ϕ_1

controller. To that end, the plant proposed in (9.5) can be cast in the following form:

$$\dot{x} = f(x) + Gu \tag{9.16}$$

where x and u are dimensionless state and manipulated input vectors, respectively, $f(\cdot)$ is a sufficiently smooth nonlinear function, and G is a constant matrix. In our application, each CSTR has a local controller, within each controller an uncertain model of the plant is embedded and takes the following form:

$$\dot{\hat{x}} = f(\hat{x}) + G\hat{u} \tag{9.17}$$

where $\hat{u}$ is chosen such that the plant model is exponentially stabilized and takes the following form:

$$\hat{u} = -G^{-1}\big(f(\hat{x}) + \lambda\hat{x}\big) \tag{9.18}$$

where $\lambda > 0$ is a controller design parameter that enables us to place the closed-loop eigenvalues at $-\lambda$. Note that for the purposes of this example, the plant-model mismatch, $w(0)$, is chosen to be zero and λ is placed at -10. The next step is to characterize the stability region to investigate the tolerable range of cyberattacks and the appropriate range of update period values necessary for mitigation. The results of this characterization are detailed in Section 9.3.3.

Figure 9.2 shows a contour plot of the stability test function, $F_1(\cdot)$ as a function of the the cyberattack magnitude, ϕ_1 and the update period, h. This plot was generated using the following values: $\alpha = 1$, $\beta = 100$, and the following Lipschitz constants: $L_{f1} = 110.3$, $L_{f2} = 38.4$, $L_{m1} = 162.0$, $L_{m2} = 106.0$ and $L_w = L_0 = 0$.

This plot demonstrates the dependence of $F_1(h)$ on both ϕ_1 and h. Values enclosed by the zero contour lines represent the estimated region within which the closed-loop nonlinear plant can be stabilized. It can be seen that, generally, as the update period increases (the communication frequency decreases) the range of tolerable cyberattacks becomes narrower. The same trend can be further confirmed in Figure 9.3 where the dependence of $F_1(h)$ on h for different ϕ_1 values is shown. Values above the zero (red-dashed) line represent the estimated values by which the closed-loop nonlinear plant can

be stabilized; the larger the cyberattack magnitude, the smaller the stabilizing region of update periods.

Remark 9.6. *Investigating Figure 9.2 one would expect that at update period values close to zero (i.e., high sensor-controller communication frequencies) the range of tolerable cyberattack magnitudes should be the greatest, however, this is not reflected in the contour plot. This is due to the fact that the stability region analysis performed in Section 9.3 is conservative, after all the stability criterion is sufficient but not necessary.*

9.4 Cyberattack Classification and Mitigation

While modeling the effect of the cyberattack on system stability is helpful, knowledge of the existence of a cyberattack and its magnitude is crucial for the mitigation of these attacks. To that end, in this work we utilize machine learning classification methods to determine the existence and magnitude of a cyberattack. Specifically, we utilize NN algorithms to classify the operating condition of the system; whether it be nominally operating or being subjected to cyberattacks, and we couple this with the knowledge of the system's stability regions to mitigate these attacks.

9.4.1 Neural network based classification

Neural network algorithms generate a general class of nonlinear functions using input and output data. An example of a basic feed-forward NN with one hidden layer is shown in Figure 9.4. The NN structure is divided into three main layers: (1) an input layer with n neurons, each representing an input variable $u_i,\ i = \{1, 2, \cdots, n\}$, (2) a hidden layer (which can consist of a single or multiple layers) with m neurons, N_i^l, where $i = \{1, 2, \cdots, m\}$, $l = \{1, 2, \cdots, L\}$ and L is the number of hidden layers, and (3) an output layer with an output neuron y_{NN} which is the classification outcome of this network, based on the input values. The hidden layer neurons are fed the weighted sum of the input values from each neuron in the input layer and using an activation function $\sigma_1(x)$, such as the Sigmoid function $\big(\sigma(x) = 1/(1 + e^{-x})\big)$ or the Rectifier (ReLu) function $\big(\sigma(x) = \max(0, x)\big)$, a nonlinear correlation between the inputs and outputs is established such that the output is not a simple linear combination of the inputs. This is represented

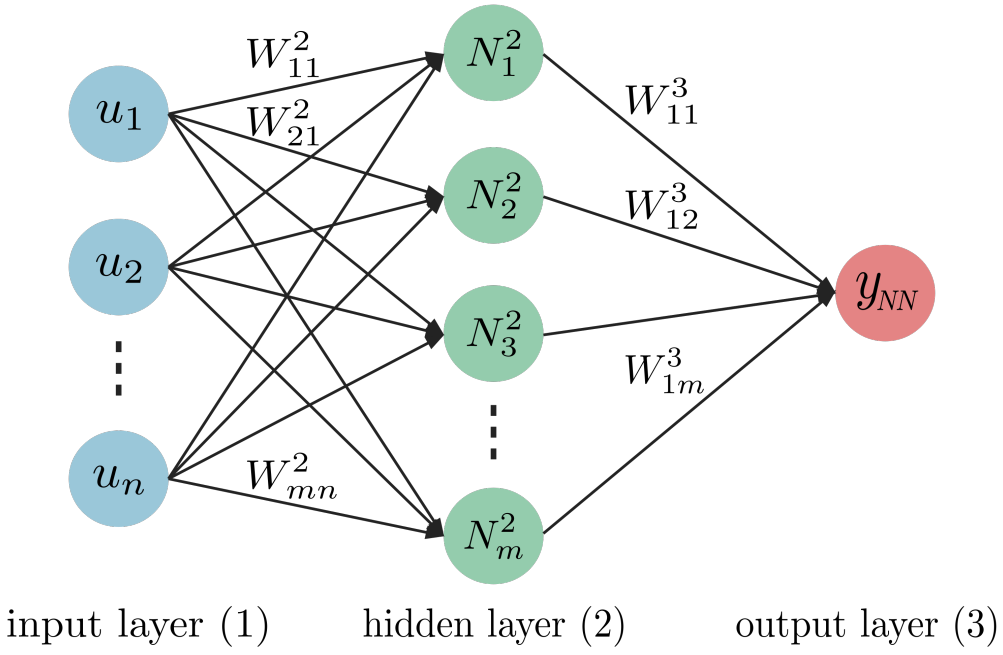

Figure 9.4. An example of a basic feed-forward NN with one hidden layer

by the following equation:

$$N_m^{(2)} = \sigma_1 \left(\sum_{i=1}^{n} W_{mi}^{(2)} u_i + b_i^{(2)} \right) \tag{9.19}$$

where $W_{mi}^{(2)}$ is the weight of the input value from input neuron i to hidden layer neuron m in the second layer and $b_i^{(2)}$ is the bias of the i-th neuron in the second layer. Note that we use the notation $W_{jk}^{(l)}$ to denote the weight for the signal from the k-th neuron in the $(l-1)$-th layer to the j-th neuron in the l-th layer. The output neuron y_{NN} generates a classification label based on the linear combination of the hidden layer neurons fed to an activation function $\sigma_2(x)$, such as the Sigmoid function for binary classification or the Softmax function $\left(\sigma_i(x) = e^{x_i} / \sum_{j=1}^{c} e^{x_j}\right.$ where c is the number of classes$\left.\right)$ for non-binary classification, using the following equation:

$$y_{NN} = \sigma_2 \left(\sum_{i=1}^{m} W_{1i}^{(3)} N_i^{(2)} + b_i^{(3)} \right) \tag{9.20}$$

where W_{1i} is the weight of the hidden layer neuron value from neuron i to the output layer neuron.

In order for the NN to generate accurate classifications, the weights and biases must be optimized through training. To achieve this, a training dataset that includes input

vectors $\mathbf{u_i}$, $i = \{1, 2, \cdots, Q\}$, and the expected classification labels vector, $\hat{\mathbf{y}}_{\mathbf{NNi}}$, is fed to the NN. A model is generated by minimizing the following cost function (cross-entropy loss function):

$$C = -\frac{1}{Q}\sum_{i=1}^{n}\left[\hat{y}_{NNi}\, ln(y_{NNi}) + (1-\hat{y}_{NNi})\, ln(1-y_{NNi})\right] \tag{9.21}$$

where n is the number of input neurons, using the stochastic gradient descent (SGD) method (which does not require the cost function to be convex) with backpropagation in order to guarantee that the global minimum is found. Note that we utilize the binary cross-entropy loss function to optimize the NN learning process. During each iteration of this training process, the weights, $W_{jk}^{(l)}$, and biases, $b_k^{(l)}$, are updated in one sweep according to the following equation:

$$\mathbf{W} = \mathbf{W} - \eta\frac{\partial C}{\partial \mathbf{W}} \tag{9.22}$$

where $\mathbf{W}$ is a matrix that represents all the weights and biases and η is the learning rate that allows us to control the rate of convergence.

To improve the NN model performance and accuracy, a k-fold cross-validation algorithm is utilized to randomly split the dataset into k-1 subsets for training and 1 subset for testing. Once the model is trained and tested using the assigned training and testing subsets, the model classification accuracy (test accuracy) is calculated using the following equation $A = N_c/Q$, where N_C is the number of correctly predicted classes. The dataset is then randomly split again and the process is repeated. This ensures that all of the dataset inputs are used for testing at some point during the training (note that the testing subsets are not used for training). Furthermore, to avoid over-fitting the data, a dropout regularization technique is utilized to randomly deactivate selected neurons during training. This effectively temporarily removes these deactivated neurons' contributions to the activation of downstream neurons. The result of this is a NN model that is more generalized and not specialized for the training dataset.

9.4.2 Neural network model training

To detect the existence and magnitude of cyberattacks, we train a feed-forward NN using data obtained from our illustrative example proposed in Section 9.1.2. The feed-forward

NN is constructed using the Keras Python library. To enable us to detect the magnitude of the cyberattack, the system is simulated for different values of ϕ_1 and the input values, u, are recorded. A dataset consisting of 5500 data samples of the input values for four different ϕ_1 values: 0.25, 0.5 0.75 and 1 (see Remark 9.7 for a discussion about this discretization of ϕ during training) are fed to the NN which consists of an input layer, two hidden layers, and an output layer. The number of neurons and activation functions used for each layer are summarized in Table 9.2. By conducting a sensitivity analysis test on

Table 9.2. Neural network structure

	Neurons	Activation Function
First Hidden Layer	60	ReLu
Second Hidden Layer	100	ReLu
Output Layer	1	Softmax

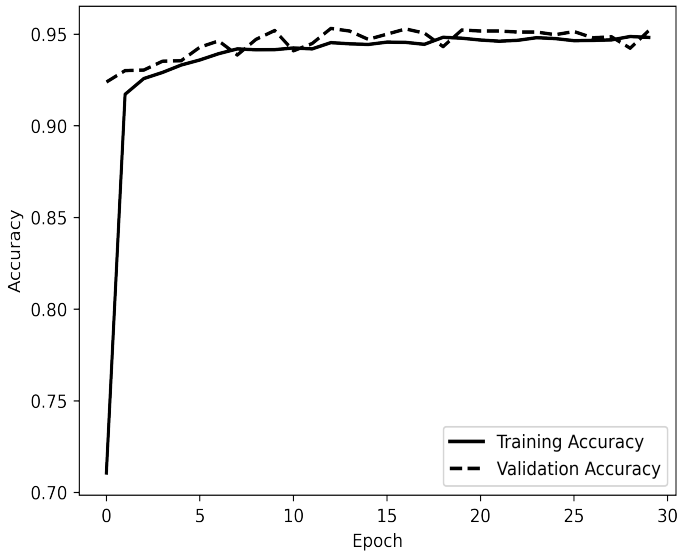

Figure 9.5. Neural network training model accuracy

the size of the dataset fed to the NN with respect to the resulting classification accuracy it was concluded that a 5500 sample dataset results in the optimal accuracy; smaller datasets resulted in a lower accuracy and larger datasets did not significantly increase

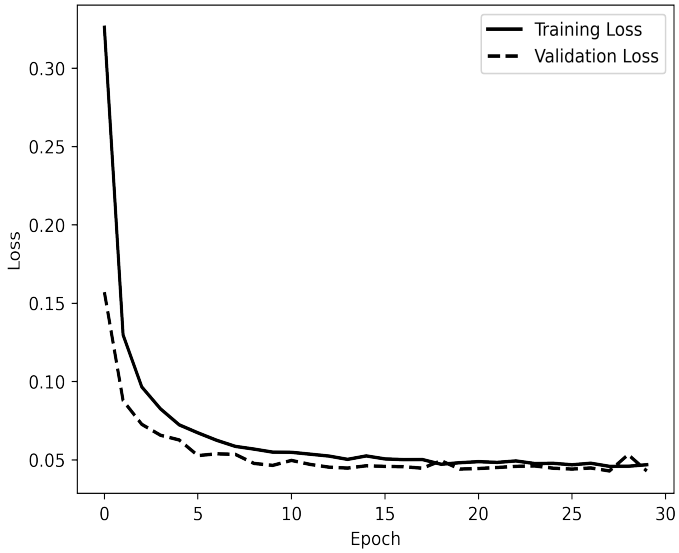

Figure 9.6. Neural network training model loss function

Table 9.3. Confusion matrix

	Actual Class 0 $\phi_1 = 0.25$ (Cyberattack)	Actual Class 1 $\phi_1 = 0.50$ (Cyberattack)	Actual Class 2 $\phi_1 = 0.75$ (Cyberattack)	Actual Class 3 $\phi_1 = 1.0$ (Nominal)
Predicted Class 0	1067	11	3	0
Predicted Class 1	18	1023	41	44
Predicted Class 2	0	39	1019	32
Predicted Class 3	0	34	20	1050

accuracy. Before training the model, the dataset is split into a 4455 training subset, a 495 validation set and a 550 testing set. Figure 9.5 shows a summary of the training and validation accuracy of the NN model throughout the training process. The final training accuracy is $94.98\% \pm 0.56\%$ and the validation accuracy is 94.25%. Additionally, Figure 9.6 shows how the loss function is minimized throughout the training process. Finally, the confusion matrix for the NN is shown in Table 9.3 for the four different classes. The results in the confusion matrix further confirm the obtained accuracy.

9.4.3 Cyberattack mitigation

Once the NN model is trained, it can be used to generate classifications based on the input vectors fed to the model. In our application, the NN will be trained based on input data for different cyberattack magnitudes, ϕ_i. The resulting classification will help us determine the existence of a cyberattack and its magnitude. Once this value is obtained, mitigation is achieved by modifying the update period value h based on the stability region analysis performed on the plant as described in Section 9.3.3. Figure 9.7 summarizes the proposed mitigation strategy for a cyberattack.

Initially, and offline, a model-based controller is designed for the system under investigation and the system stability in terms of cyberattack magnitude ϕ_i is analyzed. Based on this stability region analysis, and a suitable update period value h_0 is chosen. A NN model is then trained for different cyberattack magnitude values ϕ_i. The system operation then commences and the process is placed online and the NN based classification takes place every sampling instance. While the classification result obtained from the NN model provides us with the magnitude of ϕ_i, the accuracy of this result is only as good as the accuracy of the NN model. Thus, if a ϕ_i value that is not equal to 1 is detected; meaning that there is a potential cyberattack, we could not, with a 100% accuracy, conclude that a cyberattack is taking place. To counteract this, a moving window detection algorithm is activated when a $\phi_i \neq 1$ value is detected at t_a wherein the frequency of ϕ_i is tracked over a range of update periods. If the value of ϕ_i is persistent over the horizon of the moving window (a value determined by the user) then we can with greater certainty conclude that a cyberattack of magnitude ϕ_i is taking place. Note that although the cyberattack only manifests itself at update times h, monitoring the cyberattacks at sampling instances allows us to track the persistence of the attack and to take mitigation actions far ahead of any corrupt state updates.

With the knowledge of the magnitude of ϕ_i and the system stability criterion, we can now determine if there is an update period value that will stabilize the system subject to this cyberattack. If a stabilizing h_m exists, the controller update period is modified to h_m at t_m, and the process of classification continues to track for possible additional

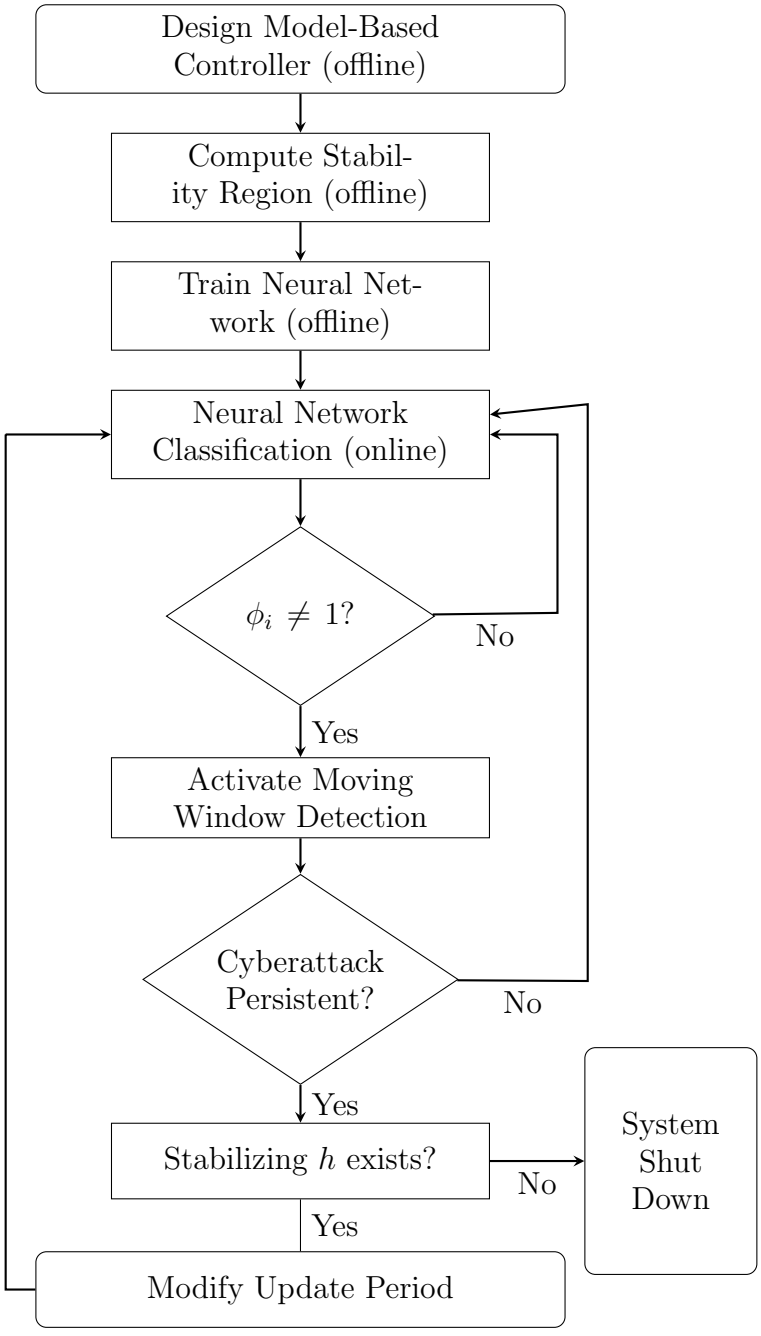

Figure 9.7. Summary of the proposed framework for integrated model-based control and NN based classification

cyberattacks. If no stabilizing h exists, however, the system is shut down and alternate methods need to be taken to secure the system.

Remark 9.7. *In this framework, there is an interplay between the classification scheme and the mitigation scheme. To reduce the computational cost associated with NN training, in this work we consider a finite set of cyberattack magnitude values when training the NN model. Due to this discretized nature of the NN model training, the resulting NN model effectiveness depends on the resolution of the discretization. If a high resolution discretization of the cyberattack magnitudes is performed (i.e., the NN model is trained using more values of ϕ_i) the NN model becomes more effective at classifying finer values of ϕ_i; albeit at the possible cost of model accuracy. However, if a low resolution discretization is chosen, the NN model becomes less effective at accurately classifying finer values of ϕ_i. In this case, if an attack takes place of a magnitude that is outside of this discretized set, the result of NN classification will be the nearest trained ϕ value and there will be a mismatch between the actual cyberattack magnitude and the classified value. Thus, when determining the range of stabilizing update periods during the mitigation step, one must take into account this possible cyberattack-classification mismatch and identify the update period values that will fully contain, not just the classification of the ϕ_i value but, a confidence interval of ϕ_i values determined by the resolution of the discretization.*

9.5 Simulation Study: Application to Chemical Reactors

In this section, we will bring together all of the previous concepts to illustrate how our proposed model-based networked cyberattack classification and mitigation strategy works using our motivating example detailed in Section 9.1.2. Following our framework outline in Figure 9.7, we must first design the model-based controller. To that end, the plant proposed in (9.5) can be cast in the following form:

$$\dot{x} = f(x) + Gu \tag{9.23}$$

where x and u are dimensionless state and manipulated input vectors, respectively, $f(\cdot)$ is a sufficiently smooth nonlinear function, and G is a constant matrix. In our application,

each CSTR has a local controller, within each controller an uncertain model of the plant is embedded and takes the following form:

$$\dot{\hat{x}} = f(\hat{x}) + G\hat{u} \tag{9.24}$$

where $\hat{u}$ is chosen such that the plant model is exponentially stabilized and takes the following form:

$$\hat{u} = -G^{-1}\big(f(\hat{x}) + \lambda\hat{x}\big) \tag{9.25}$$

where $\lambda > 0$ is a controller design parameter that enables us to place the closed-loop eigenvalues at $-\lambda$. Note that for the purposes of this example, the plant-model mismatch, $w(0)$, is chosen to be zero and λ is placed at -10. The next step is to characterize the stability region to investigate the tolerable range of cyberattacks and the appropriate range of update period values necessary for mitigation. The results of this characterization are detailed in Section 9.3.3. After the NN model is trained (see Section 9.4.2), the system operation commences and the NN classification takes place every sampling instance.

Initially, the plant is initialized with an update period $h_0 = 0.13hr$ under no cyberattack, $\phi_1 = 1$. Figure 9.8 shows the closed-loop state stable profiles for the temperatures and concentrations in both CSTRs. At time $t_a = 0.4hr$ a false data injection cyberattack of magnitude $\phi_1 = 0.6$ is introduced to the temperature sensor in CSTR 1, T_1. To understand the effect of this cyberattack on the system if no detection or mitigation takes place, we simulate the closed-loop state portfolio of T_1 in CSTR 1. Figure 9.9 shows the effect of this attack on the sensor measurements (solid line) in comparison to the actual state measurement (dashed line). It can be observed that the normal sensor measurement trajectory is stable, however, the data is being falsified and the sensor measurements subject to this falsification attacks appear to diverge from steady-state. Not knowing that they are false, the controller calculates control actions based on these sensor measurements which lead to actuator manipulations that are otherwise not necessary. Likewise, the input values for each manipulated input variable, Q_1, C_{A0}, Q_2 and C_{A3}, follow suit and their trajectories can be seen in Figure 9.10. The dashed lines represent the control actions that would have been calculated under no false sensor measurements and the

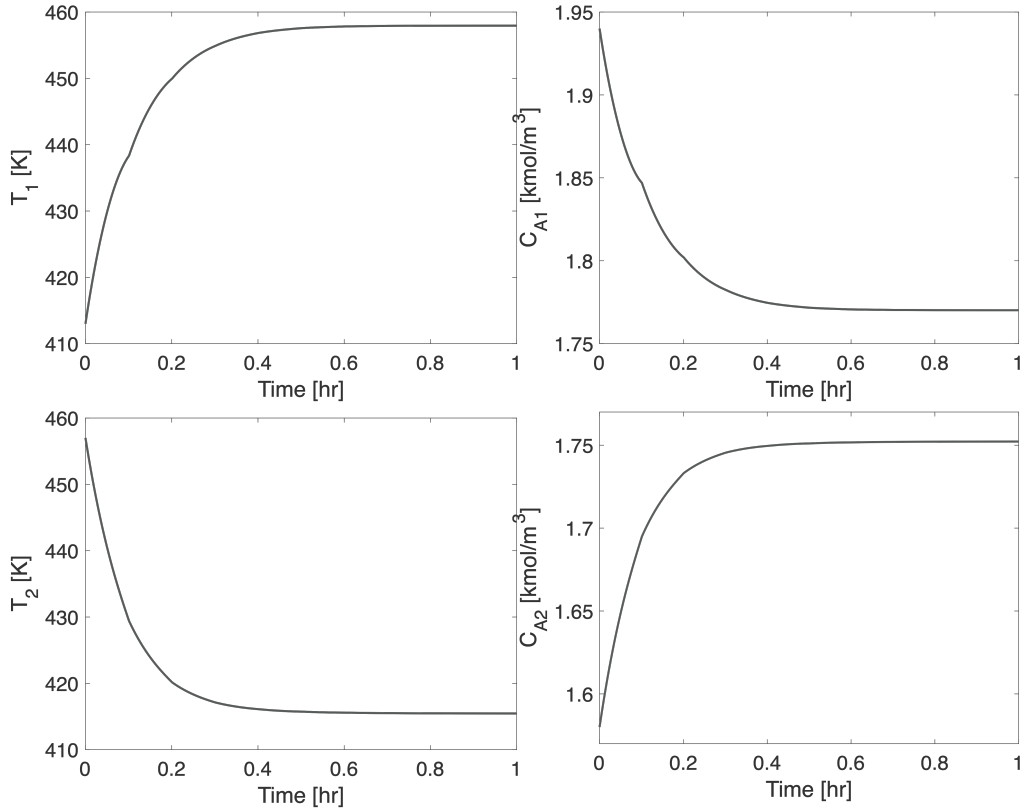

Figure 9.8. Closed-loop state profiles for T_1, T_2, C_{A1}, and C_{A2} under nominal conditions with no cyberattack $\phi_1 = 1$ and an update period $h_0 = 0.13hr$

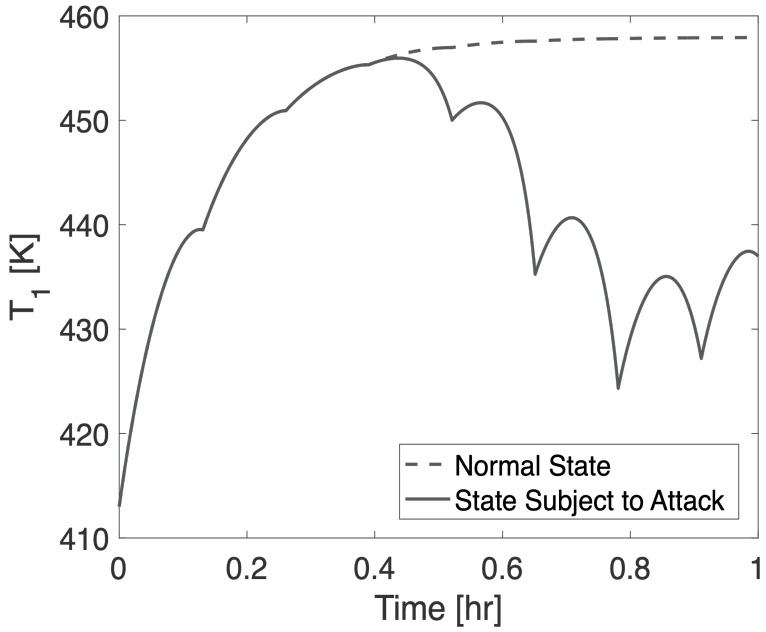

Figure 9.9. Actual state profile vs. false sensor measurements for T_1 subject to a $\phi_1 = 0.6$ magnitude cyberattack at $t_a = 0.4hr$ and an update period $h_0 = 0.13hr$

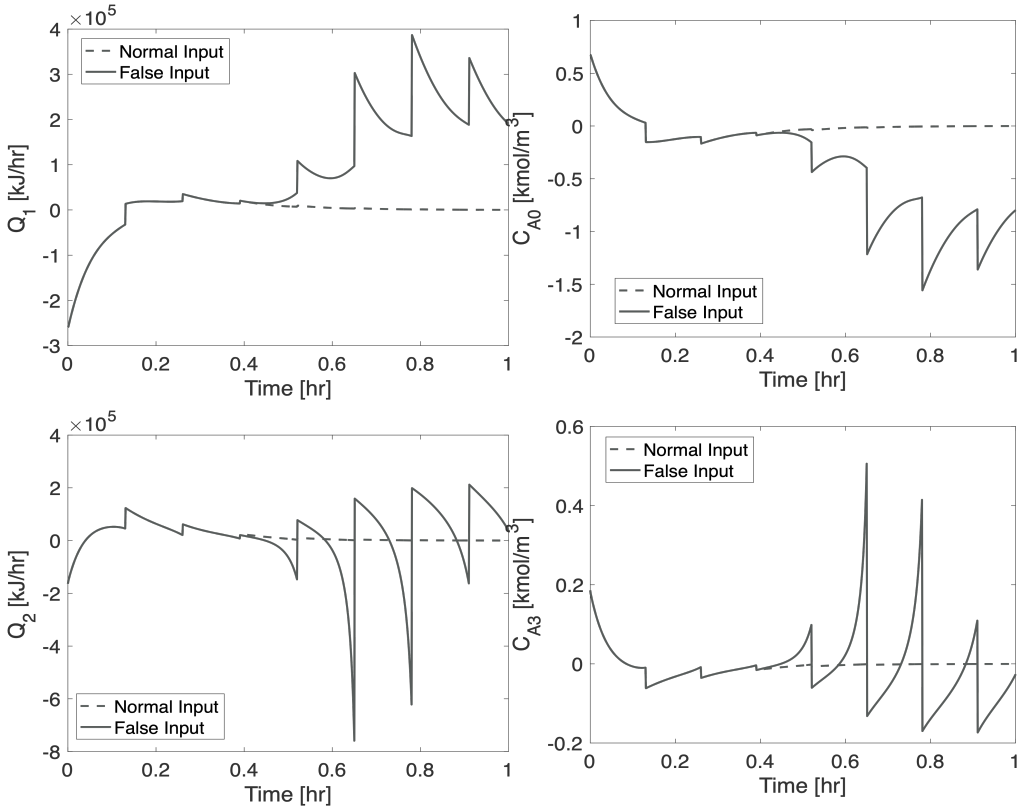

Figure 9.10. Closed-loop manipulated input profiles for Q_1, Q_2, C_{A0}, and C_{A3} subject to a $\phi_1 = 0.6$ magnitude cyberattack at $t_a = 0.4hr$ and an update period $h_0 = 0.13hr$

solid lines are the false inputs. If left unaddressed, this attack renders the plant unstable as demonstrated in Figure 9.11 where the dashed lines represent the normal closed-loop state profiles for the temperatures and concentrations and the solid lines represent the closed-loop state profiles resulting from the attack.

At this point, the NN based classification comes into play to detect the attack. Remember that the controller input values, u, are continuously fed to the NN model (established in Section 9.4.2) to obtain a classification at each sampling instance. If an attack is detected, the moving window detection algorithm is activated and the attack magnitude frequency is monitored over a horizon to confirm the persistence of the attack. It is important to note that in this example, the cyberattack introduced to the T_1 sensor at $t_a = 0.4hr$ has a magnitude of $\phi_1 = 0.6$ which is not included in the set of discretized ϕ values used during the NN training step. We consider this attack value so that we can test the robustness of the proposed detection and mitigation framework to possible clas-

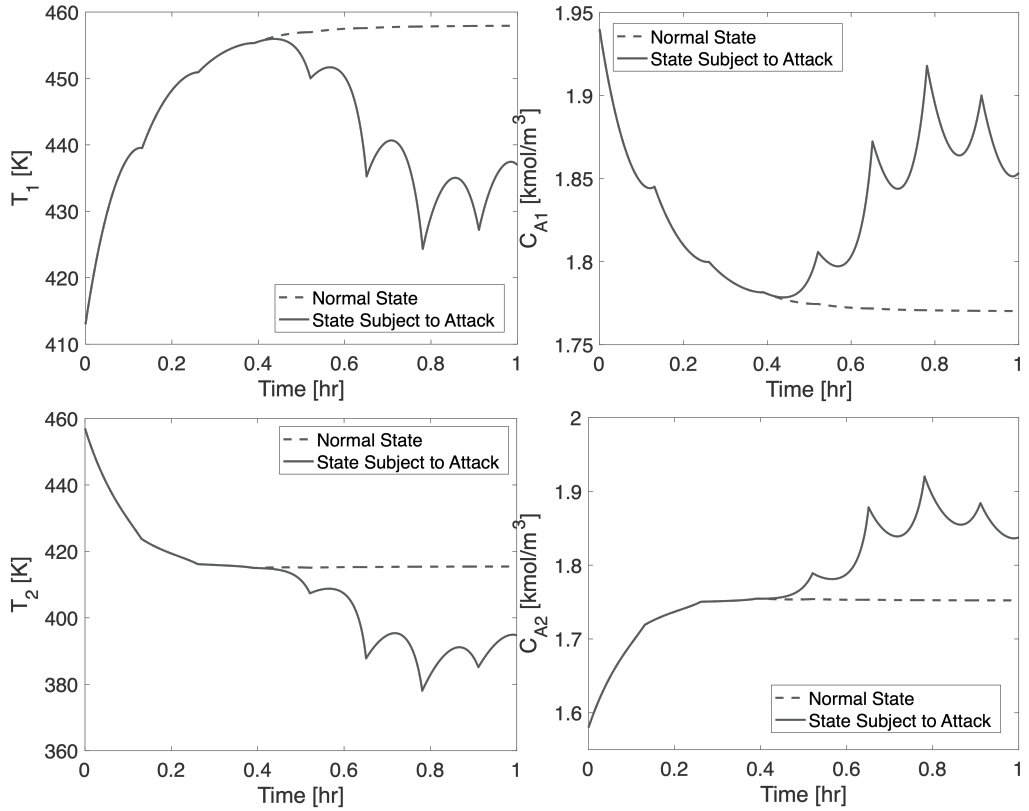

Figure 9.11. Closed-loop state profiles for T_1, T_2, C_{A1}, and C_{A2} subject to a $\phi_1 = 0.6$ magnitude cyberattack at $t_a = 0.4hr$ and an update period $h_0 = 0.13hr$

sification mismatches that may be present. The NN classification, however, classified this cyberattack as having a magnitude of $\phi_1 = 0.5$ and as a process operator, we would not have prior knowledge of this classification mismatch. Figure 9.12 summarizes the actual cyberattack magnitude value ϕ_1 vs the classified cyberattack value over time and helps visualize the mismatch between these two values. Due to this mismatch, when attempting to mitigate this attack, one must assign a confidence interval to the resultant classification value before investigating the possible stabilizing update period values. In this case, since the discretization of the ϕ values is specified in Section 9.4.2 as 0.25, a confidence interval of 0.125 is chosen for analysis purposes. Figure 9.13 depicts the stability region contour plot with the resultant classification and its confidence intervals; the solid red line shows where the value of the resultant classification lies and the dashed red lines are the upper and lower limits of the confidence interval. To select a update period value that will stabilize this attack subject to a classification mismatch, one must find an update period value

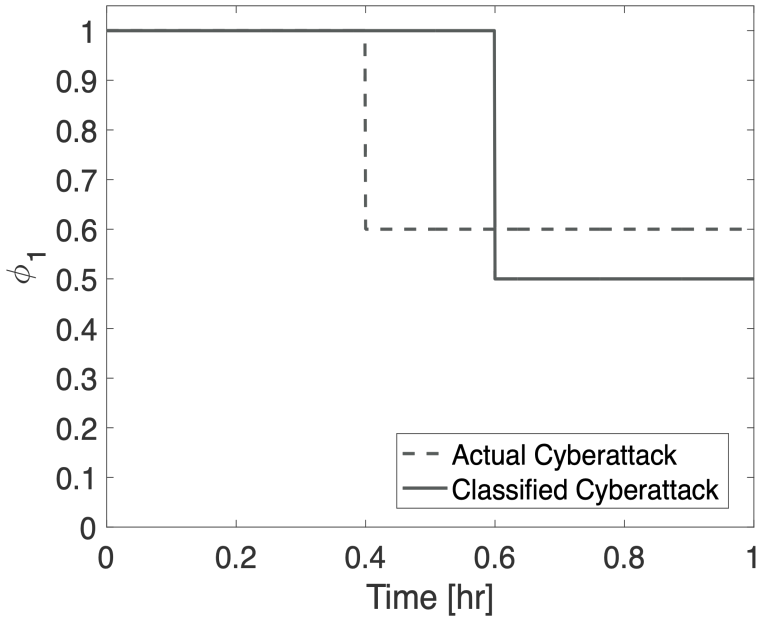

Figure 9.12. Actual cyberattack magnitude value ϕ_1 vs the classified cyberattack value over time

that encloses, not only the classification result value $\phi = 0.05$ but also, the confidence interval in the stability region. A brief visual inspection of Figure 9.13 reveals that for our given attack and confidence intervals, an update period of $h_m = 0.002$ would stabilize the system even if the actual cyberattack within the range of attacks $\phi_1 = 0.5 \pm 0.125$. Now that a stabilizing update period value is identified, the last step is to modify the update period to stabilize the system subject to the cyberattack. Figure 9.14 summarizes the result of this mitigation step. It can be seen that at $t_a = 0.4hr$ the states start to diverge from steady-state due to the cyberattack, however, the cyberattack is not confirmed and mitigated until $t_m = 0.6hr$ due to the moving window horizon algorithm. Once the update period is modified to $h_m = 0.002$, the states converge again and the system is stabilized successfully.

9.6 Conclusions

In this chapter, we propose a framework for the classification and mitigation of cyberattacks in model-based networked process control systems using machine learning methods. We first designed a model-based controller and analysed the closed-loop stability of the

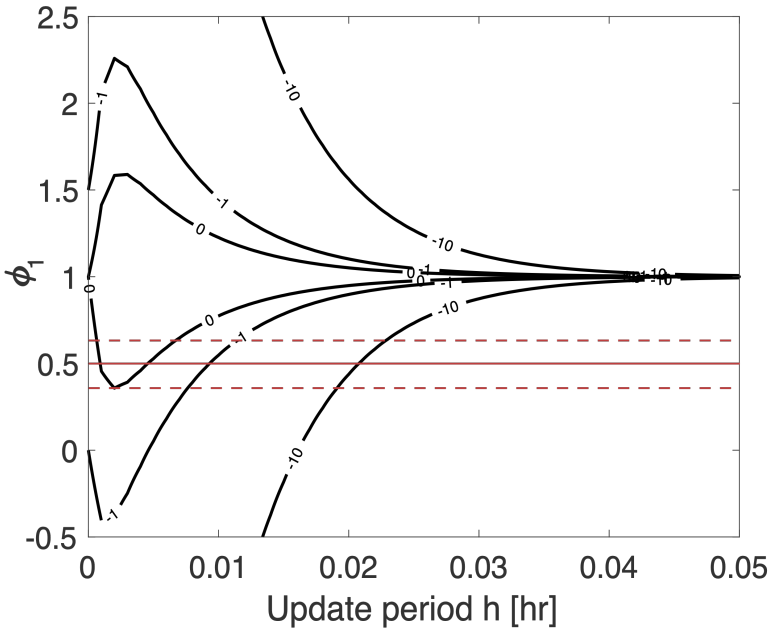

Figure 9.13. Stability region contour plot with ϕ_1 confidence intervals

system to characterize the stability region within which the system can tolerate different magnitudes of false data injection cyberattacks. A NN model was then trained offline using data from the simulated process for different cyberattack magnitudes as classes to enable us to classify both the existence and magnitude of a cyberattack. The NN model was then utilized online during the process operation and a classification was generated at every update instance. If an attack is detected, moving window detection algorithm was utilized to track the persistence of the attack. If the attack was indeed persistent, the updated period is modified and a stabilizing value is sought using the previously generated stability region. A chemical process example subject to a cyberattack was used to demonstrate the proposed framework and it was shown that using the trained NN model we were able to detect the cyberattack existence and determine its magnitude and, using the stability region analysis, were able to determine a feasible update period value that stabilized the system. A few considerations for future work include investigating the robustness of this approach to simultaneous, yet different magnitude, false data injection cyberattacks occurring to the same sensor as well as multiple false data injection cyberattacks occurring to different sensors at once, and exploring the effect of disturbances on

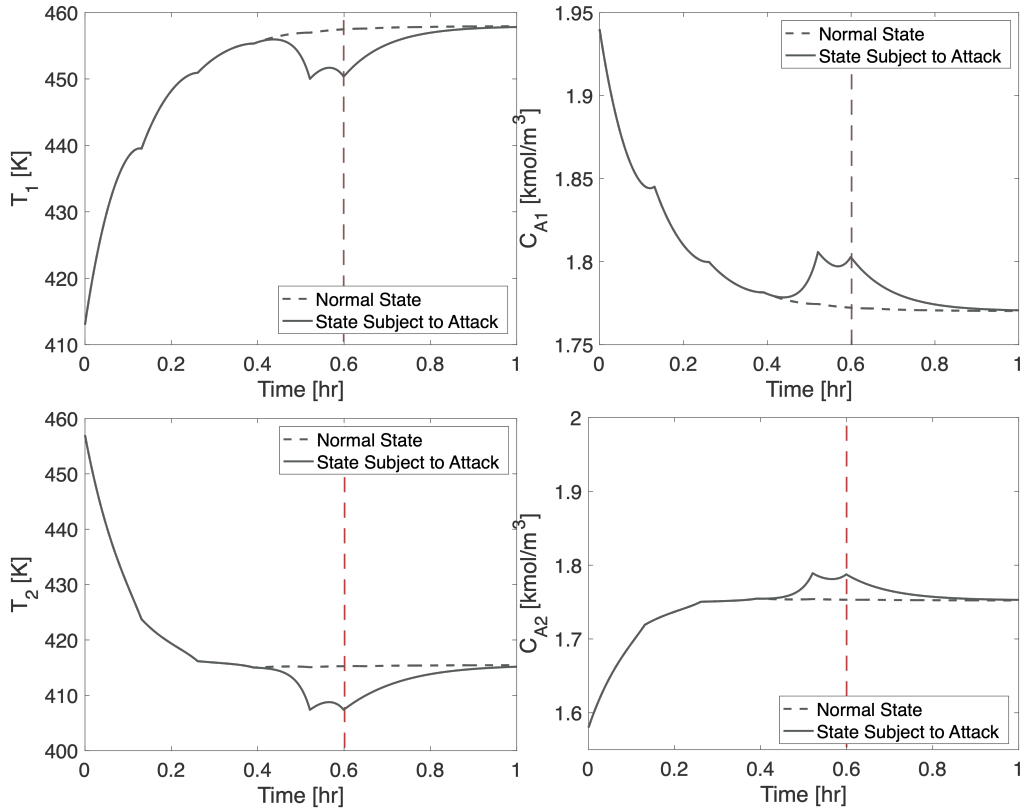

Figure 9.14. Closed-loop state profiles for T_1, T_2, C_{A1}, and C_{A2} subject to a $\phi_1 = 0.6$ magnitude cyberattack at $t_a = 0.4hr$, an update period $h_m = 0.002hr$ and mitigation at $t_m = 0.6hr$

the NN model training and classification.

Appendix A

Proofs of Chapter 8

A.1 Proof of Theorem 8.1

Proof. We proceed by analyzing the behavior of the norm of the closed-loop state between consecutive model updates. The stability of the closed-loop system can be established if $\|a_s(t)\|$ decreases such that $\|a_s(t_k)\| > \left\|a_s(t_{k+1}^-)\right\|$, where t_k and t_{k+1} are sampling times with $t_{k+1} - t_k = \Delta$. From the definition of the model estimation error and the triangular inequality, we have that for any $t \in [t_k, t_{k+1})$:

$$\|a_s(t)\| \quad \leq \quad \|\widehat{a}_s(t)\| + \|e(t)\| \tag{A.1}$$

and therefore $\|a_s(t)\|$ will decrease over the period $[t_k, t_{k+1})$ if $\|\widehat{a}_s(t)\| + \|e(t)\|$ decreases over the same period. We now establish a bound on the norm of the error as a function of the sampling period Δ. To this end, we can solve the error equation in (8.14) to obtain $\forall t \in [t_k, t_{k+1})$:

$$\begin{aligned} e(t) &= e(t_k) + \int_{t_k}^{t} \left(A_s e + \widetilde{A}_s \widehat{a}_s + \widetilde{B}_s(z_a) k(\widehat{a}_s) \right) ds \\ &+ \int_{t_k}^{t} \left(\widehat{f}_s(e + a_s) - \widehat{f}_s(\widehat{a}_s) + \theta(a_s) \right) ds \end{aligned} \tag{A.2}$$

Taking the norm of both sides and using the fact that $e(t_k) = (I - \Xi_M)a_s(t_k)$, we have that $\forall t \in [t_k, t_{k+1})$:

$$\begin{aligned} \|e(t)\| &\leq \|I - \Xi_M\|\|a_s(t_k)\| \\ &+ \int_{t_k}^{t} \left(\|A_s\|\|e\| + \left\|\widetilde{A}_s\right\|\|\widehat{a}_s\| \right) ds \\ &+ \int_{t_k}^{t} \left(\left\|\widetilde{B}_s(z_a)\right\|\|k(\widehat{a}_s)\| \right) ds \\ &+ \int_{t_k}^{t} \left(\left\|\widehat{f}_s(e + a_s) - \widehat{f}_s(\widehat{a}_s)\right\| + \|\theta(a_s)\| \right) ds \end{aligned} \tag{A.3}$$

Substituting the growth bounds of (8.9) and (8.12), as well as the inequality of (A.1), into (A.3) and noting that

$$\|\theta(a_s) - \theta(0)\| \leq L_\theta \|a_s\| \Longrightarrow \|\theta(a_s)\| \leq L_\theta \|a_s\| + \|\theta(0)\| \tag{A.4}$$

the following bound can be obtained $\forall t \in [t_k, t_{k+1})$:

$$\begin{aligned} \|e(t)\| &\leq \|I - \Xi_M\|\|a_s(t_k)\| \\ &+ \int_{t_k}^{t} (L_F\|e(s)\| + L_w\|\widehat{a}_s(s)\| + L_0)\, ds \end{aligned} \tag{A.5}$$

Applying the Gronwall-Bellman inequality yields:

$$\begin{aligned} \|e(t)\| &\leq \|I - \Xi_M\|\|a_s(t_k)\| e^{L_F(t-t_k)} \\ &+ \|\widehat{a}_s(t_k)\| \frac{\alpha L_w}{\beta + L_F} (e^{L_F(t-t_k)} - e^{-\beta(t-t_k)}) \\ &+ \frac{L_0}{L_F}(e^{L_F(t-t_k)} - 1),\ \forall t \in [t_k, t_{k+1}) \end{aligned} \tag{A.6}$$

With this bound on the estimation error and the bound on the closed-loop model state in (8.11), we can proceed to calculate a bound on the system state using (A.1), where it can be shown after some algebraic manipulations that $\forall\, t \in [t_k, t_{k+1})$:

$$\|a_s(t)\| \leq \|\widehat{a}_s(t_k)\|(1 - F_1(t - t_k)) + F_2(t - t_k) \tag{A.7}$$

where $F_1(\cdot)$ and $F_2(\cdot)$ are defined in (8.17) and (8.19), respectively. Using the above estimate to calculate $\|a_s(t_{k+1}^-)\|$ and noting that $\|\widehat{a}_s(t_k)\| \leq \|\Xi_M\|\|a_s(t_k)\|$, we finally obtain:

$$\left\|a_s(t_{k+1}^-)\right\| - \|a_s(t_k)\| \leq F_2(h) - F_1(h)\|a_s(t_k)\| \tag{A.8}$$

$\forall\ t \in [t_k, t_{k+1})$. Clearly, if $F_1(\Delta) > 0$ and $\|a_s(t_k)\| > F_2(\Delta)/F_1(\Delta)$, then $\left\|a_s(t_{k+1}^-)\right\| - \|a_s(t_k)\| < 0$ and (8.24) holds. This implies that $\lim_{k\to\infty} \|a_s(t_k)\| \leq r(\Delta)$. Substituting this estimate into (A.7), we have that $\lim_{t\to\infty} \|a_s(t)\| \leq \psi_1 r(\Delta) + \psi_2$.

To ensure that the bounds in (8.9) and (8.12) are valid, we must verify that $\|a_s(t)\| \leq M$ for all $t \geq t_0$. From (A.7), it can be verified that if $\|a_s(t_0)\| \leq M'$, then for all $t \in [t_0, t_1)$, $\|a_s(t)\| \leq M'\psi_1 + \psi_2 = M$. Considering that $\left\|a_s(t_{k+1}^-)\right\| < \|a_s(t_k)\|$, we have that if $\|a_s(t_0)\| \leq M'$ is satisfied, then $\|a_s(t)\| \leq M$ for all $t \geq t_0$. This completes the proof of the theorem. $\square$

Appendix B

Proofs of Chapter 9

B.1 Proof of Theorem 9.1

Proof. *To establish closed-loop stability, we need to show that the norm of the state of the closed-loop system* $\| x(t) \|$ *decreases over successive update times such that* $\| x(t_k) \| > \| x(t_{k+1}^-) \|$ *for all* $k \in \{0, 1, 2, \cdots\}$. *To show this, we start with the triangular inequality, which establishes that, over any period of time* $[t_k, t_{k+1})$, *the following relationship holds:*

$$\| x(t) \| \leq \| \widehat{x}(t) \| + \| e(t) \| \tag{B.1}$$

Based on this, to establish that $\| x(t) \|$ *is decreasing over the period* $[t_k, t_{k+1})$, *it is sufficient to show that* $\| \widehat{x}(t) \| + \| e(t) \|$ *is decreasing over the same period.*

To obtain a suitable bound on the norm of the model estimation error in terms of the model state update period h, *we use the definition of the estimation error, together with the system and model dynamics, to describe the time evolution of the estimation error as follows:*

$$\begin{aligned} \dot{e}(t) &= \dot{x}(t) - \dot{\widehat{x}}(t) \\ &= \widehat{f}\big(x(t)\big) - \widehat{f}\big(\widehat{x}(t)\big) + w\big(x(t)\big) + m\big(k\big(x(t)\big)\big) \end{aligned} \tag{B.2}$$

Solving for $e(t)$, *we have that* $\forall t \in [t_k, t_{k+1})$:

$$e(t) = e(t_k) + \int_{t_k}^{t} \Big(\widehat{f}\big(x(s)\big) - \widehat{f}\big(\widehat{x}(s)\big) + w\big(x(s)\big) \Big)\, ds \; + \int_{t_k}^{t} m\big(k\big(\widehat{x}(s)\big)\big)\, ds \tag{B.3}$$

Taking the norm of both sides, we have that $\forall t \in [t_k, t_{k+1})$*:*

$$\begin{aligned} \| e(t) \| \leq \| e(t_k) \| + \int_{t_k}^{t} \| \widehat{f}\big(x(s)\big) - \widehat{f}\big(\widehat{x}(s)\big) \| \, ds + \int_{t_k}^{t} \| w\big(x(s)\big) \| \, ds \\ + \int_{t_k}^{t} \| m\big(k\big(\widehat{x}(s)\big)\big) \| \, ds \end{aligned} \tag{B.4}$$

Substituting the bounds in (9.3) *into* (B.4)*, and noting that:*

$$\| w(x) - w(0) \| \leq L_w \| x \| \implies \| w(x) \| \leq L_w \| x \| + \| w(0) \|,$$

$$\| m(u) - m(0) \| \leq L_m \| u \| \implies \| m(u) \| \leq L_m \| u \| + \| m(0) \|,$$

the following bound can be obtained $\forall t \in [t_k, t_{k+1})$*:*

$$\begin{aligned} \| e(t) \| \leq \| e(t_k) \| + \int_{t_k}^{t} L_f \| \mathbf{e}(s) \| \, ds + \int_{t_k}^{t} \big(L_w \| x(s) \| + \| w(0) \| \big) \, ds \\ + \int_{t_k}^{t} \big(L_m L_k \| \widehat{x}(s) \| + \| m(0) \| \big) \, ds \end{aligned} \tag{B.5}$$

Using the fact that $\| x \| \leq \| e \| + \| \widehat{x} \|$*, the above bound can be expressed as follows:*

$$\begin{aligned} \| e(t) \| \leq \| e(t_k) \| + \int_{t_k}^{t} (L_f + L_w) \| \mathbf{e}(s) \| \, ds + \int_{t_k}^{t} \big(L_m L_k + L_w \big) \| \widehat{x}(s) \| \, ds \\ + \int_{t_k}^{t} \big(\| w(0) \| + \| m(0) \| \big) \, ds \end{aligned} \tag{B.6}$$

$$\| e(t) \| \leq \| e(t_k) \| + L_e \int_{t_k}^{t} \| \mathbf{e}(s) \| \, ds + L_\theta \int_{t_k}^{t} \| \hat{\mathbf{x}}(s) \| \, ds + L_0 (t - t_k) \tag{B.7}$$

where $L_e = L_f + L_w$*,* $L_\theta = L_m L_k + L_w$ *and* $L_0 = \| w(0) \| + \| m(0) \|$*. Substituting the bound on the closed-loop model state given in* (9.10) *into* (B.7) *and applying the Gronwall-Bellman inequality, it can be shown that* $\forall t \in [t_k, t_{k+1})$*:*

$$\| e(t) \| \leq \| \mathbf{e}(t_k) \| \, e^{L_e (t - t_k)} + \frac{L_0}{L_e} \Big[e^{L_e (t - t_k)} - 1 \Big] + \frac{\alpha L_\theta}{\beta + L_e} \| \widehat{\mathbf{x}}(t_k) \| \Big[e^{L_e (t - t_k)} - e^{-\beta (t - t_k)} \Big] \tag{B.8}$$

Now that a bound on the estimation error has been established, we can calculate a bound on the closed-loop state, $\forall t \in [t_k, t_{k+1})$*, using the bound in* (9.10)*, the triangular inequality*

in (B.1) *and the fact that* $\| \widehat{x}(t_k) \| = \| \Phi x(t_k) \|$ *and* $\| e(t_k) \| = \| \big(I - \Phi\big) x(t_k) \|$. *This bound can be expressed as follows:*

$$\| x(t) \| \leq \| x(t_k) \| \big(1 - F_1(t - t_k)\big) + F_2(t - t_k) \tag{B.9}$$

where the forms of the functions $F_1(\cdot)$ *and* $F_2(\cdot)$ *are given in* (8.17) *and* (8.19)*, respectively. Finally, using* (B.9)*, we can calculate* $\| x(t_{k+1}^-) \|$ *to obtain:*

$$\| x(t_{k+1}^-) \| - \| x(t_k) \| \leq F_2(h) - F_1(h) \| x(t_k) \| \tag{B.10}$$

where we have used the fact that $h = t_{k+1}^- - t_k$. *It can be seen from the result in* (B.10) *that if* $F_1(h) > 0$ *and* $\| x(t_k) \| > F_2(h)/F_1(h)$*, then* $\| x(t_{k+1}^-) \| - \| x(t_k) \| < 0$ *and* (9.14) *holds.*

References

[1] T. Takamatsu, "The nature and role of process systems engineering," *Computers & Chemical Engineering*, vol. 7, no. 4, pp. 203–218, 1983. [Online]. Available: https://www.sciencedirect.com/science/article/pii/009813548380012X

[2] P. D. Christofides, J. F. Davis, N. H. El-Farra, D. Clark, K. R. D. Harris, and J. N. Gipson, "Smart plant operations: Vision, progress and challenges," *AIChE Journal*, vol. 53, no. 11, pp. 2734–2741, 2007.

[3] J. Alford and G. Buckbee, "Industrial process control systems: A new approach to education," *Chemical Engineering Progress*, vol. 116, no. 12, pp. 35–42, 2020.

[4] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," in *The Institute of Electrical and Electronics Engineers (IEEE), 2007.* IEEE Xplore, 2007, pp. 138–162.

[5] P. Antsaklis and J. Baillieul, "Special issue on technology of networked control systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 5–8, 2007.

[6] K.-Y. You and L.-H. Xie, "Survey of recent progress in networked control systems," *Acta Automatica Sinica*, vol. 39, pp. 101–117, 2013.

[7] X. Zhang, Q. Han, and X. Yu, "Survey on recent advances in networked control systems," *IEEE Trans. on Industrial Informatics*, vol. 12, no. 5, pp. 1740–1752, 2015.

[8] D. Zhang, P. Shi, Q.-G. Wang, and L. Yu, "Analysis and synthesis of networked control systems: A survey of recent advances and challenges," *ISA Trans.*, vol. 66, pp. 376–392, 2017.

[9] K. R. Jillson and B. E. Ydstie, "Process networks with decentralized inventory and flow control," *Journal of Process Control*, vol. 17, pp. 399–413, 2007.

[10] J. B. Rawlings and B. T. Stewart, "Coordinating multiple optimization-based controllers: New opportunities and challenges," *Computers & Chemical Engineering*, vol. 18, pp. 839–845, 2008.

[11] S. S. Jogwar, M. Baldea, and P. Daoutidis, "Dynamics and control of process networks with large energy recycle," *Industrial & Engineering Chemistry Research*, vol. 48, pp. 6087–6097, 2009.

[12] P. D. Christofides, J. Liu, and D. Munoz de la Pena, *Networked and Distributed Predictive Control: Methods and Nonlinear Process Netwrok Applications.* Springer-Verlag, 2011.

[13] X. Cai, M. J. Tippett, L. Xie, and J. Bao, "Fast distributed mpc based on active set method," *Computers & Chemical Engineering*, vol. 71, pp. 158–170, 2014.

[14] X. Ge, F. Yang, and Q.-L. Han, "Distributed networked control systems: A brief overview," *Information Sciences*, vol. 380, pp. 117–131, 2017.

[15] Y. Sun and N. H. El-Farra, "Quasi-decentralized model-based networked control of process systems," *Computers & Chemical Engineering*, vol. 32, no. 9, pp. 2016–2029, 2008.

[16] ——, "Resource-aware quasi-decentralized control of networked process systems over wireless sensor networks," *Chemical Engineering Science*, vol. 69, no. 1, pp. 93–106, 2012.

[17] E. Garcia, P. J. Antsaklis, and L. A. Montestruque, *Model-Based Control of Networked Systems,* Systems & Control: Foundations & Applications. Switzerland: Springer International Publishing, 2014.

[18] D. Xue and N. H. El-Farra, "Supervisory logic for control of networked process systems with event-based communication," in *Proceedings of American Control Conference*, 2016, pp. 7135–7140.

[19] Y. Sun, S. Ghantasala, and N. H. El-Farra, "Networked control of spatially distributed processes with sensor-controller communication constraints," in *Proceedings of American Control Conference*, 2009, pp. 2489–2494.

[20] Z. Yao and N. H. El-Farra, "Model-based networked control of spatially distributed systems with measurement delays," in *Proceedings of American Control Conference*, 2012, pp. 2990–2995.

[21] Y. Sun and N. H. El-Farra, "A quasi-decentralized approach for networked state estimation and control of process systems," *Industrial & Engineering Chemistry Research*, vol. 49, no. 17, pp. 7957–7971, 2010.

[22] ——, "Robust quasi-decentralized control of uncertain process networks," *Industrial Engineering and Chemistry Research*, vol. 53, pp. 7421—-7433, 2014.

[23] S. Amin, A. A. Cárdenas, and S. S. Sastry, "Safe and secure networked control systems under denial-of-service attacks," in *Hybrid Systems: Computation and Control*, R. Majumdar and P. Tabuada, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 31–45.

[24] A. Teixeira, S. Amin, H. Sandberg, K. H. Johansson, and S. S. Sastry, "Cyber security analysis of state estimators in electric power systems," in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 5991–5998.

[25] Y. Mo and B. Sinopoli, "Secure control against replay attacks," in *2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2009, pp. 911–918.

[26] R. S. Smith, "A decoupled feedback structure for covertly appropriating networked control systems," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 90 – 95, 2011, 18th IFAC World Congress. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667016435925

[27] Y. Liu, P. Ning, and M. K. Reiter, "False data injection attacks against state estimation in electric power grids," in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, ser. CCS '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 21–32. [Online]. Available: https://doi.org/10.1145/1653662.1653666

[28] F. Khorrami, P. Krishnamurthy, and R. Karri, "Cybersecurity for control systems: A process-aware perspective," *IEEE Design and Test*, vol. 33, pp. 75–83, 2016.

[29] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," *IEEE Security and Privacy*, vol. 9, pp. 49–51, 2011.

[30] J. P. Farwell and R. Rohozinski, "Stuxnet and the future of cyber war," *Survival*, vol. 53, no. 1, pp. 23–40, 2011. [Online]. Available: https://doi.org/10.1080/00396338.2011.555586

[31] A. Greenberg, "How an entire nation became russia's test lab for cyberwar," *Available online: https://www.wired.com/story/russian-hackers-attack-ukraine/*, 2017.

[32] J. P. Conti, "The day the samba stopped," *Engineering Technology*, vol. 5, no. 4, pp. 46–47, 2010.

[33] R. M. Clark, S. Panguluri, T. D. Nelson, and R. P. Wyman, "Protecting drinking water utilities from cyberthreats," *Journal American Water Works Association*, vol. 109, pp. 50–58, 2017.

[34] J. Slay and M. Miller, "Lessons learned from the maroochy water breach," in *Critical Infrastructure Protection*, E. Goetz and S. Shenoi, Eds. Boston, MA: Springer US, 2008, pp. 73–82.

[35] Z.-H. Pang and G.-P. Liu, "Design and implementation of secure networked predictive control systems under deception attacks," *IEEE Transactions on Control Systems Technology*, vol. 20, pp. 1334–1342, 2011.

[36] C.-W. Ten, C.-C. Liu, and G. Manimaran, "Vulnerability assessment of cybersecurity for scada systems," *IEEE Transactions on Power Systems*, vol. 23, pp. 1836–1846, 2008.

[37] O. Linda, M. Manic, and M. McQueen, "Improving control system cyber-state awareness using known secure sensor measurements," in *Critical Information Infrastructures Security*, B. M. Hämmerli, N. Kalstad Svendsen, and J. Lopez, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 46–58.

[38] A. S. Rawat, P. Anand, H. Chen, and P. K. Varshney, "Collaborative spectrum sensing in the presence of byzantine attacks in cognitive radio networks," *IEEE Transactions on Signal Processing*, vol. 59, no. 2, pp. 774–786, 2011.

[39] B. Kailkhura, Y. S. Han, S. Brahma, and P. K. Varshney, "Distributed bayesian detection in the presence of byzantine data," *IEEE Transactions on Signal Processing*, vol. 63, no. 19, pp. 5250–5263, 2015.

[40] L. Liu, M. Esmalifalak, Q. Ding, V. A. Emesih, and Z. Han, "Detecting false data injection attacks on power grid by sparse optimization," *IEEE Transactions on Smart Grid*, vol. 5, no. 2, pp. 612–621, 2014.

[41] O. Kosut, L. Jia, R. J. Thomas, and L. Tong, "Malicious data attacks on the smart grid," *IEEE Transactions on Smart Grid*, vol. 2, no. 4, pp. 645–658, 2011.

[42] K. Manandhar, X. Cao, F. Hu, and Y. Liu, "Detection of faults and attacks including false data injection attack in smart grid using kalman filter," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 4, pp. 370–379, 2014.

[43] Y. Mo, R. Chabukswar, and B. Sinopoli, "Detecting integrity attacks on scada systems," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 4, pp. 1396–1407, 2014.

[44] D. Ding, Q.-L. Han, Y. Xiang, X. Ge, and X.-M. Zhang, "A survey on security control and attack detection for industrial cyber-physical systems,"

Neurocomputing, vol. 275, pp. 1674 – 1683, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231217316351

[45] I. Kiss, B. Genge, and P. Haller, "A clustering-based approach to detect cyber attacks in process control systems," in *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, 2015, pp. 142–148.

[46] B. Satchidanandan and P. R. Kumar, "Dynamic watermarking: Active defense of networked cyber–physical systems," *Proceedings of the IEEE*, vol. 105, no. 2, pp. 219–240, 2017.

[47] V. S. Dolk, P. Tesi, C. De Persis, and W. P. M. H. Heemels, "Event-triggered control systems under denial-of-service attacks," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 1, pp. 93–105, 2017.

[48] M. Pajic, J. Weimer, N. Bezzo, O. Sokolsky, G. J. Pappas, and I. Lee, "Design and implementation of attack-resilient cyberphysical systems: With a focus on attack-resilient state estimators," *IEEE Control Systems Magazine*, vol. 37, no. 2, pp. 66–81, 2017.

[49] H. Durand, "A nonlinear systems framework for cyberattack prevention for chemical process control systems," *Mathematics*, vol. 6, no. 9, 2018. [Online]. Available: http://www.mdpi.com/2227-7390/6/9/169

[50] S. Chen, Z. Wu, and P. D. Christofides, "Cyber-attack detection and resilient operation of nonlinear processes under economic model predictive control," *Computers & Chemical Engineering*, vol. 136, p. 106806, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0098135420300855

[51] A. Zedan, D. Xue, and N. H. El-Farra, "Integrating model identification and model-based control of networked process systems," in *13th International Symposium on Process Systems Engineering (PSE 2018)*, ser. Computer Aided Chemical Engineering, M. R. Eden, M. G. Ierapetritou, and G. P.

Towler, Eds. Elsevier, 2018, vol. 44, pp. 715 – 720. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780444642417501142

[52] A. Zedan and N. H. El-Farra, "Model-based networked control of spatially-distributed processes with event-triggered parameter re-identification," in *Proceedings of 58th IEEE Conference on Decision and Control*, 2019, pp. 1207–1212.

[53] B. I. Seraphim, S. Palit, K. Srivastava, and E. Poovammal, "A survey on machine learning techniques in network intrusion detection system," in *2018 4th International Conference on Computing Communication and Automation (ICCCA)*, 2018, pp. 1–5.

[54] M. Ozay, I. Esnaola, F. T. Yarman Vural, S. R. Kulkarni, and H. V. Poor, "Machine learning methods for attack detection in the smart grid," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 8, pp. 1773–1786, 2016.

[55] Z. Wu, F. Albalawi, J. Zhang, Z. Zhang, H. Durand, and P. D. Christofides, "Detecting and handling cyber-attacks in model predictive control of chemical processes," *Mathematics*, vol. 6, no. 10, 2018. [Online]. Available: https://www.mdpi.com/2227-7390/6/10/173

[56] Z. Wu, A. Tran, D. Rincon, and P. D. Christofides, "Machine-learning-based predictive control of nonlinear processes. part ii: Computational implementation," *AIChE Journal*, vol. 65, no. 11, p. e16734, 2019. [Online]. Available: https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.16734

[57] A. Zedan, H. Huang, and N. H. El-Farra, "Enhancing model-based networked control of uncertain process systems through error-triggered parameter re-identification," in *Proceedings of American Control Conference*, 2021.

[58] W. Favoreel, B. D. Moor, and P. V. Overschee, "Subspace state space system identification for industrial processes," *Journal of Process Control*, vol. 10, pp. 149–155, 2000.

[59] L. Ljung, *System Identification*. Prentice Hall, 1999.

[60] B. Corbett and P. Mhaskar, "Subspace identification for data-driven modeling and quality control of batch processes," *AIChE Journal*, vol. 62, pp. 1581–1601, 2016.

[61] A. Alanqar, H. Durand, and P. D. Christofides, "Handling plant variation via error-triggered on-line model identification: Application to economic model predictive control," in *International Federation for Automatic Control (IFAC), 2016*. Elsevier, 2016, pp. 802–807.

[62] A. Smyshlyaev and M. Krstic, *Adaptive Control of Parabolic PDEs*. Princeton University Press, 2010.

[63] P. D. Christofides, *Nonlinear and Rhobust Control of PDE systems*. Birkhäuser, 2001.

[64] I. Karafyllis, N. Bekiaris-Liberis, and M. Papageorgiou, "Feedback control of nonlinear hyperbolic PDE systems inspired by traffic flow models," *IEEE Trans. Automat. Contr.*, 2018.

[65] P. D. Christofides and P. Daoutidis, "Finite-dimensional control of parabolic pde systems using approximate inertial manifolds," *Journal of Mathematical Analysis and Applications*, vol. 216, pp. 398—-420, 1997.

[66] M. J. Balas, "Feedback control of linear diffusion processes," *Int. J. Contr.*, vol. 29, pp. 523—-534, 2007.

[67] A. Varshney, S. Pitchaiah, and A. Armaou, "Feedback control of dissipative pde systems using adaptive model reduction," *AIChE J.*, vol. 55, no. 4, pp. 906–918, 2009.

[68] L. Biao, W. Huai-Ning, and L. Han-Xiong, "Adaptive optimal control of highly dissipative nonlinear spatially distributed processes with neuro-dynamic programming," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, pp. 684–696, 2015.

[69] H. Wu, H. Wang, and L. Guo, "Finite dimensional disturbance observer based control for nonlinear parabolic pde systems via output feedback," *J. Proc. Contr.*, vol. 48, pp. 25–40, 2016.

[70] R. F. Curtain and H. Zwart, *An Introduction to Infinite-Dimensional Linear Systems Theory*. Springer-Verlag, 1978.

[71] A. Zedan and N. H. El-Farra, "An event-based approach for model-based control and parameter identification in networked distributed processes," in *Proceedings of American Control Conference*, 2020, pp. 3425–3430.

[72] ——, "Enhancing resource utilization efficiency in networked control of distributed processes with parameter variations and limited measurements," in *Proceedings of American Control Conference*, 2021.

[73] D. Xue and N. H. El-Farra, "Output feedback-based event-triggered control of distributed processes with communication constraints," in *Proceedings of 55th IEEE Conference on Decision and Control*, 2016, pp. 4296–4301.

[74] Y. Sun and N. H. El-Farra, "Resource-aware quasi-decentralized control of nonlinear plants over communication networks," in *Proceedings of American Control Conference*, 2009, pp. 154–159.