Chapter 7
EMPC Systems: Computational Efficiency and Real-Time Implementation

7.1 Introduction

While the two-layer EMPC structures of Chap. 6 were shown to successfully reduce the on-line computation time relative to that required for a centralized, one-layer EMPC scheme, EMPC optimization problems typically found within the context of chemical processes are nonlinear and non-convex because a nonlinear dynamic model is embedded in the optimization problem. Although many advances have been made in solving such problems and modern computers may efficiently perform complex calculations, it is possible that computation delay will occur that may approach or exceed the sampling time. If the computational delay is significant relative to the sampling period, closed-loop performance degradation and/or closed-loop instability may occur.

In this chapter, three EMPC design methodologies are presented to further address computational efficiency. In the first section, a composite control structure featuring EMPC is designed for systems with explicit two-time-scale dynamic behavior. Owing to the fact that the class of dynamic models describing such systems are typically stiff, a sufficiently small time step is required for forward numerical integration with explicit methods, which subsequently affects the computation time required to solve the EMPC problem. On the other hand, the composite control structure allows for larger time steps because it avoids the use of the stiff dynamic model embedded in the MPC problems of the composite control structure. In the second section, distributed EMPC (DEMPC), which computes the control actions by solving a series of distributed EMPC problems, is considered. Specifically, an application study whereby several DEMPC schemes are applied to a benchmark chemical process example is presented to evaluate the ability of the resulting DEMPC schemes to reduce the computation time relative to a centralized EMPC system. The closed-loop performance under DEMPC is compared with that achieved under a centralized EMPC approach. In the third section, a real-time implementation strategy for Lyapunov-based EMPC (LEMPC) is presented which addresses potentially unknown and time-varying computational time for control action calculation. Closed-loop stability under the real-time LEMPC strategy is rigorously analyzed.

7.2 Economic Model Predictive Control of Nonlinear Singularly Perturbed Systems

The development of optimal process control, automation, and management methodologies while addressing time-scale multiplicity due to the strong coupling of slow and fast phenomena occurring at different time-scales is an important issue in the context of chemical process control. For multiple-time-scale systems, closed-loop stability as well as controller design are usually addressed through explicit separation of fast and slow states in a standard singular perturbation setting [1–3] or by taking advantage of change of coordinates for two-time-scale systems in nonstandard singularly perturbed form [4]. In our previous work, we developed methods for slow time-scale (tracking) MPC as well as composite fast-slow (tracking) MPC for nonlinear singularly perturbed systems [5, 6]. In this section, these ideas are extended to EMPC of nonlinear singularly perturbed systems.

Specifically, in this section, an EMPC method for a broad class of nonlinear singularly perturbed systems is presented whereby a “fast” Lyapunov-based MPC (LMPC) using a standard tracking quadratic cost is employed to stabilize the fast closed-loop dynamics at their equilibrium slow manifold while a “slow” Lyapunov-based EMPC (LEMPC) is utilized for the slow dynamics to address economic performance considerations. Multi-rate sampling of the states is considered involving fast-sampling of the fast states and slow-sampling of the slow states. The states are subsequently used in the fast and slow MPC systems, respectively. Closed-loop stability of the control scheme is addressed through singular perturbation theory. The control method is demonstrated through a chemical process example which exhibits two-time-scale behavior.

7.2.1 Class of Nonlinear Singularly Perturbed Systems

Nonlinear singularly perturbed systems in standard form are considered in this section with the following state-space description:

$$\dot{x} = f(x, z, u_s, w, \varepsilon), \tag{7.1a}$$

$$\varepsilon \dot{z} = g(x, z, u_f, w, \varepsilon), \tag{7.1b}$$

where $x \in \mathbb{R}^{n_x}$ and $z \in \mathbb{R}^{n_z}$ denote the state vectors, $u_s \in \mathbb{U}_s \subset \mathbb{R}^{m_s}$ and $u_f \in \mathbb{U}_f \subset \mathbb{R}^{m_f}$ are the control (manipulated) inputs, $w \in \mathbb{R}^l$ denotes the vector of process disturbances, and $\varepsilon > 0$ is a small positive parameter. The sets $\mathbb{U}_s$ and $\mathbb{U}_f$ are assumed to be compact sets. The disturbance vector is assumed to be an absolutely continuous function of time and bounded in a sense that there exists a $\theta > 0$ such that $|w(t)| \leq \theta$ for all $t \geq 0$. To this end, let $\mathbb{W} = \{w \in \mathbb{R}^l : |w| \leq \theta\}$.

Owing to the multiplication of the small parameter ε with $\dot{z}$ in Eq. 7.1, there exists a time-scale separation in the two systems of differential equations of Eq. 7.1. Moreover, the system of Eq. 7.1 is said to be in singularly perturbed form. Through the rest of the section, x and z will be referred to as the slow and fast states, respectively. Furthermore, the vector functions f and g are assumed to be sufficiently smooth on $\mathbb{R}^{n_x} \times \mathbb{R}^{n_z} \times \mathbb{U}_s \times \mathbb{W} \times [0, \bar{\varepsilon})$ and $\mathbb{R}^{n_x} \times \mathbb{R}^{n_z} \times \mathbb{U}_f \times \mathbb{W} \times [0, \bar{\varepsilon})$, respectively, for some $\bar{\varepsilon} > 0$. The origin is assumed to be an equilibrium point of the unforced nominal system; that is, system of Eq. 7.1 with $u_s = 0$, $u_f = 0$, and $w \equiv 0$ possesses an equilibrium point at $(x, z) = (0, 0)$.

The fast states are sampled synchronously and are available at time instants indicated by the time sequence $\{t_{k_f}\}_{k_f \geq 0}$ with $t_{k_f} = t_0 + k_f \Delta_f$, $k_f = 0, 1, \ldots$ where t_0 is the initial time and $\Delta_f > 0$ is the measurement sampling period of the fast states. Similarly, the slow states are sampled synchronously and are available at time instants indicated by the time sequence $\{t_{k_s}\}_{k_s \geq 0}$ with $t_{k_s} = t_0 + k_s \Delta_s$, $k_s = 0, 1, \ldots$ where $\Delta_s > 0$ ($\Delta_s > \Delta_f$) is the measurement sampling period of the slow states. The initial time is taken to be zero, i.e., $t_0 = 0$. With respect to the control problem formulation, the controls u_f and u_s, which are responsible for the fast and slow dynamics, are computed every Δ_f and Δ_s, respectively. For the sake of simplicity, Δ_s / Δ_f is assumed to be a positive integer.

7.2.2 *Two-Time-Scale Decomposition*

The explicit separation of slow and fast states in the system of Eq. 7.1 allows for decomposing the system into two separate reduced-order systems evolving in different time-scales. To proceed with such a two-time-scale decomposition and to simplify the notation of the subsequent development, the issue of controlling the fast dynamics is addressed first. Similar to [5], a "fast" model predictive controller will be designed that renders the fast dynamics asymptotically stable in a sense to be made precise in Assumption 7.2. Moreover, u_f does not modify the open-loop equilibrium slow manifold for the fast dynamics. By setting $\varepsilon = 0$, the dimension of the state-space of the system of Eq. 7.1 reduces from $n_x + n_z$ to n_x because the differential equation of Eq. 7.1b degenerates into an algebraic equation. The following model is obtained that describes the slow dynamics with $u_f = 0$:

$$\dot{\bar{x}} = f(\bar{x}, \bar{z}, u_s, w, 0) \tag{7.2a}$$

$$0 = g(\bar{x}, \bar{z}, 0, w, 0) \tag{7.2b}$$

where the bar notation in $\bar{x}$ and $\bar{z}$ is used to indicate that these variables have been obtained by setting $\varepsilon = 0$.

Systems of the form of Eq. 7.1 satisfying the following assumption are said to be in *standard singularly perturbed form*, e.g. [1].

Assumption 7.1 The equation $g(\bar{x}, \bar{z}, 0, w, 0) = 0$ possesses a unique isolated root

$$\bar{z} = \tilde{g}(\bar{x}, w) \tag{7.3}$$

where $\tilde{g} : \mathbb{R}^{n_x} \times \mathbb{R}^{l} \rightarrow \mathbb{R}^{n_z}$ and its partial derivatives $\partial \tilde{g}/\partial \bar{x}$, $\partial \tilde{g}/\partial w$ are sufficiently smooth and $|\partial \tilde{g}/\partial w| \leq L_{\tilde{g}}$.

Assumption 7.1 ensures that the system of Eq. 7.1 has an isolated equilibrium manifold for the fast dynamics. While on this manifold, $\bar{z}$ may be expressed in terms of $\bar{x}$ and w using an algebraic expression. It should be emphasized that $g(\bar{x}, \bar{z}, 0, w, 0)$ is, in this case, independent of the expression of the "fast" input, u_f. Assumption 7.1 does not pose any significant limitations in practical applications, and it is a necessary one in the singular perturbation framework to construct a well-defined slow subsystem. Utilizing $\bar{z} = \tilde{g}(\bar{x}, w)$, the system of Eq. 7.2 may be re-written as follows:

$$\dot{\bar{x}} = f(\bar{x}, \tilde{g}(\bar{x}, w), u_s, w, 0) =: f_s(\bar{x}, u_s, w). \tag{7.4}$$

The system of Eq. 7.4 is referred to as the slow subsystem or the reduced system.

Introducing the fast time-scale $\tau = t/\varepsilon$ and the deviation variable $y = z - \tilde{g}(x, w)$, i.e., the deviation of the fast state from the equilibrium manifold, the nonlinear singularly perturbed system of Eq. 7.1 may be written in the (x, y) coordinates with respect to the fast time-scale as follows:

$$\begin{aligned} \frac{dx}{d\tau} &= \varepsilon f(x, y + \tilde{g}(x, w), u_s, w, \varepsilon) \\ \frac{dy}{d\tau} &= g(x, y + \tilde{g}(x, w), u_f, w, \varepsilon) \\ &\quad - \varepsilon \left(\frac{\partial \tilde{g}}{\partial x} f(x, y + \tilde{g}(x, w), u_f, w, \varepsilon) + \frac{\partial \tilde{g}}{\partial w} \dot{w} \right) \end{aligned} \tag{7.5}$$

Setting $\varepsilon = 0$, the following fast subsystem is obtained:

$$\frac{d\bar{y}}{d\tau} = g(x, \bar{y} + \tilde{g}(x, w), u_f, w, 0) \tag{7.6}$$

where the notation $\bar{y}$ is again used to indicate that its dynamics have been derived by setting $\varepsilon = 0$. In the system of Eq. 7.6, x and w are considered to be "frozen" to their initial values in the fast time-scale since their change in this time-scale is of order ε.

Remark 7.1 The difference between y and $\bar{y}$ is: y is the deviation between the singularly perturbed system state z (Eq. 7.1 with $\varepsilon > 0$) and the solution to the algebraic equation $g(x, \bar{z}, 0, w, 0) = 0$ for $x \in \mathbb{R}^{n_x}$ and $w \in \mathbb{W}$, which is denoted as $\bar{z} = \tilde{g}(x, w)$. The variable $\bar{y}$ is used to denote the solution to the fast subsystem obtained from Eq. 7.6 where x and w are frozen to their initial values and $\varepsilon = 0$; the initial condition of the ODE of Eq. 7.6 at some time $t_0 \geq 0$ is $\bar{y}(t_0) = y(t_0) = z(t_0) - \tilde{g}(x(t_0), w(t_0))$.

7.2.3 Stabilizability Assumption

A stabilizability assumption is imposed on the slow subsystem in the sense that the existence of a Lyapunov-based locally Lipschitz feedback controller $u_s = h_s(\bar{x})$ is assumed which, under continuous implementation, renders the origin of the nominal closed-loop slow subsystem of Eq. 7.4 asymptotically stable while satisfying the input constraints for all the states $\bar{x}$ inside a given stability region. Using converse Lyapunov theorems [7–9], this assumption implies that there exists a continuously differentiable Lyapunov function $V_s : \mathbb{D}_s \to \mathbb{R}_+$ for the nominal closed-loop slow subsystem that satisfy the following inequalities:

$$\alpha_{s_1}(|\bar{x}|) \leq V_s(\bar{x}) \leq \alpha_{s_2}(|\bar{x}|) \tag{7.7a}$$

$$\frac{\partial V_s(\bar{x})}{\partial \bar{x}} f_s(\bar{x}, h_s(\bar{x}), 0) \leq -\alpha_{s_3}(|\bar{x}|) \tag{7.7b}$$

$$\left| \frac{\partial V_s(\bar{x})}{\partial \bar{x}} \right| \leq \alpha_{s_4}(|\bar{x}|) \tag{7.7c}$$

$$h_s(\bar{x}) \in \mathbb{U}_s \tag{7.7d}$$

for all $\bar{x} \in \mathbb{D}_s \subseteq \mathbb{R}^{n_x}$ where D_s is an open neighborhood of the origin and the functions $\alpha_{s_i}(\cdot)$, $i = 1, 2, 3, 4$ are class $\mathscr{K}$ functions. The region $\Omega_{\rho^s} \subset \mathbb{D}_s$ denotes the stability region of the closed-loop slow subsystem under the Lyapunov-based controller $h_s(\bar{x})$.

Similarly, a stabilizability assumption is placed on the fast subsystem.

Assumption 7.2 There exists a feedback controller $u_f = p(x)\bar{y} \in \mathbb{U}_f$ where $p(x)$ is a sufficiently smooth vector function in x, such that the origin of the closed-loop fast subsystem:

$$\frac{d\bar{y}}{d\tau} = g(x, \bar{y} + \tilde{g}(x, w), p(x)\bar{y}, w, 0) \tag{7.8}$$

is globally asymptotically stable, uniformly in $x \in \mathbb{R}^{n_x}$ and $w \in \mathbb{W}$, in the sense that there exists a class $\mathscr{KL}$ function β_y such that for any $\bar{y}(0) \in \mathbb{R}^{n_z}$:

$$|\bar{y}(\tau)| \leq \beta_y(|\bar{y}(0)|, \tau) \tag{7.9}$$

for $\tau \geq 0$.

This assumption implies that there exist functions $\alpha_{f_i}(\cdot)$, $i = 1, 2, 3$ of class $\mathscr{K}$ and a continuously differentiable Lyapunov function $V_f(\bar{y})$ for the nominal closed-loop fast subsystem which satisfy the following inequalities:

$$\begin{gathered}
\alpha_{f_1}(|\bar{y}|) \leq V_f(\bar{y}) \leq \alpha_{f_2}(|\bar{y}|) \\
\frac{\partial V_f(\bar{y})}{\partial \bar{y}} g(x, \bar{y} + \tilde{g}(x, 0), p(x)\bar{y}, 0, 0) \leq -\alpha_{f_3}(|\bar{y}|) \\
p(x)\bar{y} \in \mathbb{U}_f
\end{gathered} \tag{7.10}$$

for all $\bar{y} \in \mathbb{D}_f \subseteq \mathbb{R}^{n_z}$ and $x \in D_s$ where $\mathbb{D}_f$ is an open neighborhood of the origin. The region $\Omega_{\rho^f} \subseteq \mathbb{D}_f$ is used to denote the stability region of the closed-loop fast subsystem under the nonlinear controller $p(x)\bar{y}$.

Remark 7.2 The sets Ω_{ρ^s} and Ω_{ρ^f} denote the stability regions for the closed-loop slow and fast subsystems under the controllers $u_s = h_s(\bar{x})$ and $u_f = p(x)\bar{y}$, respectively, in the sense that the closed-loop states of the fast and slow subsystems, starting in Ω_{ρ^s} and Ω_{ρ^f}, remain in these sets thereafter. Regarding the construction of Ω_{ρ^s}, we have estimated it through the following procedure: $\dot{V}_s(\bar{x})$ is evaluated for different values of $\bar{x}$ while $h_s(\bar{x})$ is applied to the nominal system subject to the input constraint $h_s(\bar{x}) \in \mathbb{U}_s$. Then, we estimated Ω_{ρ^s} as the largest level set of the Lyapunov function $V_s(\bar{x})$ where $\dot{V}_s(\bar{x}) < 0$. The region Ω_{ρ^f} may be estimated in a similar fashion using the fast subsystem and the controller $p(x)\bar{y}$.

7.2.4 LEMPC of Nonlinear Singularly Perturbed Systems

In this section, the design of a composite control structure featuring an EMPC for nonlinear singularly perturbed systems is presented. In the control structure, a tracking Lyapunov-based MPC is used to stabilize the fast dynamics in a sense to be made precise below. To control the slow dynamics, a two-mode LEMPC of Sect. 4.2 is used to address economic considerations as well as address closed-loop stability of the slow subsystem.

Over the time interval $[0, t_s)$, the LEMPC operates in mode 1 operation to dictate a potentially time-varying operating policy of the slow subsystem. After t_s, the LEMPC operates in mode 2 operation to enforce convergence of the closed-loop subsystem state to the steady-state. In operation period $[0, t_s)$, the predicted slow subsystem state along the finite prediction horizon is constrained to be maintained in the region $\Omega_{\rho_e} \subset \Omega_{\rho^s}$. If the current slow subsystem state is in the region $\Omega_{\rho^s} \setminus \Omega_{\rho_e}$, the LEMPC drives the slow subsystem state to the region Ω_{ρ_e} and once the state converges to the set Ω_{ρ_e}, the LEMPC may dictate a time-varying operating policy. Under mode 2 operation of the LEMPC ($t \geq t_s$), LEMPC computes control actions such that the slow subsystem state is driven to a neighborhood of the steady-state.

7.2.4.1 Implementation Strategy

The above described implementation strategy corresponding to the fast subsystem under the fast LMPC may be described as follows:

1. The LMPC receives the fast subsystem state $\bar{y}(t_{k_f})$.
2. The LMPC obtains its manipulated input trajectory which ensures that the fast subsystem state is steered to a neighborhood of the equilibrium slow manifold of the fast state.
3. The LMPC sends the first step value of the computed input trajectory to the corresponding actuators.
4. Go to Step 1 ($k_f \leftarrow k_f + 1$).

Similarly, the implementation strategy corresponding to the slow subsystem under the slow LEMPC can be described as follows:

1. The LEMPC receives $x(t_{k_s})$ from the sensors.
2. If $t_{k_s} < t_s$, go to Step 3. Else, go to Step 4.
3. If $x(t_{k_s}) \in \Omega_{\rho_e}$, go to Step 3.1. Else, go to Step 3.2.
   - 3.1. The LEMPC computes an input trajectory over a finite-time prediction horizon that optimizes the economic cost function while maintains the predicted state trajectory to be within Ω_{ρ_e}. Go to Step 5.
   - 3.2. The LEMPC computes an input trajectory that forces the state closer to the region Ω_{ρ_e}. Go to Step 5.
4. The LEMPC computes an input trajectory that drives the slow subsystem state to a small neighborhood of the origin.
5. The LEMPC sends the first step value of the computed input trajectory to the corresponding actuators.
6. Go to Step 1 ($k_s \leftarrow k_s + 1$).

7.2.4.2 Fast LMPC Formulation

Referring to the fast subsystem of Eq. 7.6, the fast LMPC at sampling time t_{k_f} is formulated as follows

$$\min_{u_f \in S(\Delta_f)} \int_{t_{k_f}}^{t_{k_f}+N_f\Delta_f} [\tilde{y}^T(\hat{\tau})Q_f\tilde{y}(\hat{\tau}) + u_f^T(\hat{\tau})R_f u_f(\hat{\tau})]\, d\hat{\tau} \tag{7.11a}$$

$$\text{s.t.} \quad \frac{d\tilde{y}(\hat{\tau})}{d\hat{\tau}} = g(x(t_{k_s}), \tilde{y}(\hat{\tau}) + \tilde{g}(x(t_{k_s}), 0), u_f(\hat{\tau}), 0, 0) \tag{7.11b}$$

$$\tilde{y}(t_{k_f}) = \bar{y}(t_{k_f}) \tag{7.11c}$$

$$u_f(\hat{\tau}) \in \mathbb{U}_f,\ \forall\, \hat{\tau} \in [t_{k_f}, t_{k_f} + N_f\Delta_f) \tag{7.11d}$$

$$\begin{aligned} &\frac{\partial V_f(\bar{y}(t_{k_f}))}{\partial \bar{y}} g(x(t_{k_s}), \bar{y}(t_{k_f}) + \tilde{g}(x(t_{k_s}), 0), u_f(t_{k_f}), 0, 0) \\ &\le \frac{\partial V_f(\bar{y}(t_{k_f}))}{\partial \bar{y}} g(x(t_{k_s}), \bar{y}(t_{k_f}) + \tilde{g}(x(t_{k_s}), 0), p(x(t_{k_s}))\bar{y}(t_{k_f}), 0, 0) \end{aligned} \tag{7.11e}$$

where $S(\Delta_f)$ is the family of piece-wise constant functions with sampling period Δ_f, N_f is the prediction horizon of LMPC, Q_f and R_f are positive definite weighting matrices that penalize the deviation of the fast subsystem state and manipulated input from their corresponding values at the equilibrium slow manifold, $\bar{y}(t_{k_f})$ is the fast subsystem state measurement obtained at t_{k_f}, $\tilde{y}$ denotes the predicted fast subsystem state trajectory of the nominal fast subsystem model of Eq. 7.11b over the prediction horizon. The input trajectory is subject to the manipulated input constraint of Eq. 7.11d. The constraint of Eq. 7.11e indicates that the amount of reduction in the value of the Lyapunov function when the manipulated input u_f computed by the LMPC of Eq. 7.11 is applied is at least the amount of reduction when the Lyapunov-based controller $p(x)y$ is applied in a sample-and-hold fashion. Since the LMPC of Eq. 7.11 obtains the manipulated input trajectory u_f every Δ_f, $x(t_{k_s})$ is the last available measurement of the slow process state, i.e., $t_{k_s} \leq t_{k_f}$. The optimal solution to this optimization problem is defined by $u_f^*(\hat{\tau}|t_{k_f})$ for all $\hat{\tau} \in [t_{k_f}, t_{k_f} + N_f \Delta_f)$ and the manipulated input of the closed-loop fast subsystem under the LMPC of Eq. 7.11 is defined as follows:

$$u_f(t) = u_f^*(t|t_{k_f}), \ \forall\, t \in [t_{k_f}, t_{k_f} + \Delta_f). \tag{7.12}$$

To analyze the closed-loop stability of the fast subsystem under the fast LMPC, a few properties are needed. By continuity, the smoothness property assumed for the vector function $g(\cdot)$ and taking into account that the manipulated input u_f and the disturbance w are bounded in compact sets, there exists a positive constant M_f such that

$$|g(x, \bar{y} + \tilde{g}(x, w), u_f, w, 0)| \leq M_f \tag{7.13}$$

for all $\bar{y} \in \Omega_{\rho^f}$, $u_f \in \mathbb{U}_f$, $x \in \Omega_{\rho^s}$ and $w \in \mathbb{W}$. Furthermore, by the continuous differentiable property of the Lyapunov function $V_f(\cdot)$ and the smoothness property assumed for the vector function $g(\cdot)$, there exist positive constants $L_{\bar{y}}$ and L_{w_f} such that

$$\begin{aligned} \left| \frac{\partial V_f(\bar{y})}{\partial \bar{y}} g(x, \bar{y}, u_f, w, 0) - \frac{\partial V_f(\bar{y}')}{\partial \bar{y}} g(x, \bar{y}', u_f, w, 0) \right| &\leq L_{\bar{y}} |\bar{y} - \bar{y}'| \\ \left| \frac{\partial V_f(\bar{y})}{\partial \bar{y}} g(x, \bar{y}, u_f, w, 0) - \frac{\partial V_f(\bar{y})}{\partial \bar{y}} g(x, \bar{y}, u_f, w', 0) \right| &\leq L_{w_f} |w - w'| \end{aligned} \tag{7.14}$$

for all $\bar{y}, \bar{y}' \in \Omega_{\rho^f}$, $u_f \in \mathbb{U}_f$, and $w, w' \in \mathbb{W}$.

Proposition 7.1 characterizes the closed-loop stability properties of the LMPC of Eq. 7.11. The analysis is similar to that of the LMPC presented in [10].

Proposition 7.1 *Consider the fast subsystem of Eq. 7.6 in closed-loop under the LMPC of Eq. 7.11 based on the feedback controller $p(x)y$ that satisfies the conditions of Eq. 7.10. Let $\varepsilon_{w_f} > 0$, $\Delta_f > 0$ and $\rho^f > \rho_s^f > 0$, $\theta > 0$ satisfy:*

$$-\alpha_{f_3}(\alpha_{f_2}^{-1}(\rho_s^f)) + L_{\bar{y}} M_f \Delta_f + (L_{\bar{y}} L_{\tilde{g}} + L_{w_f})\theta \leq -\varepsilon_{w_f}/\Delta_f. \tag{7.15}$$

Then, there exists a class $\mathscr{KL}$ function β_y and a class $\mathscr{K}$ function γ_y such that if $\bar{y}(0) \in \Omega_{\rho^f}$, then $\bar{y}(t) \in \Omega_{\rho^f}$ for all $t \geq 0$ and

$$|\bar{y}(t)| \leq \beta_y \left(|\bar{y}(0)|, \frac{t}{\varepsilon} \right) + \gamma_y(\rho_f^*) \tag{7.16}$$

with $\rho_f^* = \max_{s \in [0, \Delta_f]} \{V_f(\bar{y}(s)) : V_f(\bar{y}(0)) \leq \rho_s^f\}$, uniformly in $x \in \Omega_{\rho^s}$ and $w \in W$.

Proof The time derivative of the Lyapunov function along the state trajectory $\bar{y}(t)$ of fast subsystem of Eq. 7.6 for $t \in [t_{k_f}, t_{k_f} + \Delta_f)$ is written as follows:

$$\dot{V}_f(\bar{y}(t)) = \frac{\partial V_f(\bar{y}(t))}{\partial \bar{y}} g(x(t_{k_s}), \bar{y}(t) + \tilde{g}(x(t_{k_s}), w), u_f^*(t_{k_f}|t_{k_f}), w, 0). \tag{7.17}$$

Adding and subtracting the term:

$$\frac{\partial V_f(\bar{y}(t_{k_f}))}{\partial \bar{y}} g(x(t_{k_s}), \bar{y}(t_{k_f}) + \tilde{g}(x(t_{k_s}), 0), u_f^*(t_{k_f}|t_{k_f}), 0, 0)$$

to the right-hand-side of Eq. 7.17 and taking Eq. 7.10 into account, we obtain the following inequality:

$$\begin{aligned}\dot{V}_f(\bar{y}(t)) \leq -\alpha_{f_3}(|\bar{y}(t_{k_f})|) &+ \frac{\partial V_f(\bar{y}(t))}{\partial \bar{y}} g(x(t_{k_s}), \bar{y}(t) + \tilde{g}(x(t_{k_s}), w), u_f^*(t_{k_f}|t_{k_f}), w, 0) \\ &- \frac{\partial V_f(\bar{y}(t_{k_f}))}{\partial \bar{y}} g(x(t_{k_s}), \bar{y}(t_{k_f}) + \tilde{g}(x(t_{k_s}), 0), u_f^*(t_{k_f}|t_{k_f}), 0, 0)\end{aligned} \tag{7.18}$$

for $t \in [t_{k_f}, t_{k_f} + \Delta_f)$. From Eq. 7.14, Assumption 7.1 and the inequality of Eq. 7.18, the following inequality is obtained for all $\bar{y}(t_{k_f}) \in \Omega_{\rho^f} \backslash \Omega_{\rho_s^f}$:

$$\dot{V}_f(\bar{y}(t)) \leq -\alpha_{f_3}(\alpha_{f_2}^{-1}(\rho_s^f)) + L_{\bar{y}}|\bar{y}(t) - \bar{y}(t_{k_f})| + (L_{\bar{y}} L_{\tilde{g}} + L_{w_f})\theta. \tag{7.19}$$

for $t \in [t_{k_f}, t_{k_f} + \Delta_f)$. Taking into account Eq. 7.13 and the continuity of $\bar{y}(t)$, the following bound can be written for all $t \in [t_{k_f}, t_{k_f} + \Delta_f)$, $|\bar{y}(t) - \bar{y}(t_{k_f})| \leq M_f \Delta_f$. Using this expression, the following bound is obtained on the time derivative of the Lyapunov function for $t \in [t_{k_f}, t_{k_f} + \Delta_f)$:

$$\dot{V}_f(\bar{y}(t)) \leq -\alpha_{f_3}(\alpha_{f_2}^{-1}(\rho_s^f)) + L_{\bar{y}} M_f \Delta_f + (L_{\bar{y}} L_{\tilde{g}} + L_{w_f})\theta.$$

for all $\bar{y}(t_{k_f}) \in \Omega_{\rho^f} \backslash \Omega_{\rho_s^f}$. If the condition of Eq. 7.15 is satisfied, then

$$\dot{V}_f(\bar{y}(t)) \leq -\varepsilon_{w_f}/\Delta_f, \ \forall\, t \in [t_{k_f}, t_{k_f} + \Delta_f)$$

for $\bar{y}(t_{k_f}) \in \Omega_{\rho^f} \setminus \Omega_{\rho_s^f}$. Integrating this bound over $t \in [t_{k_f}, t_{k_f} + \Delta_f)$, the following is obtained:

$$V_f(\bar{y}(t_{k_f} + \Delta_f)) \leq V_f(\bar{y}(t_{k_f})) - \varepsilon_{w_f} \tag{7.20a}$$
$$V_f(\bar{y}(t)) \leq V_f(\bar{y}(t_{k_f})),\ \forall\, t \in [t_{k_f}, t_{k_f} + \Delta_f) \tag{7.20b}$$

for all $\bar{y}(t_{k_f}) \in \Omega_{\rho^f} \setminus \Omega_{\rho_s^f}$. Using Eq. 7.20 recursively, it can be proved that, if $x(t_{k_f}) \in \Omega_{\rho^f} \setminus \Omega_{\rho_s^f}$, the state converges to $\Omega_{\rho_s^f}$ in a finite number of sampling times without leaving the stability region. Once the state converges to $\Omega_{\rho_s^f} \subseteq \Omega_{\rho_f^*}$, it remains inside $\Omega_{\rho_f^*}$ for all times. This statement holds because of the definition of ρ_f^*. This proves that the closed-loop system under the fast LMPC design is ultimately bounded in $\Omega_{\rho_f^*}$. Thus, due to the continuity of the Lyapunov function $V_f(\cdot)$, it can be concluded that there exists a class $\mathscr{KL}$ function β_y and a class $\mathscr{K}$ function γ_y such that if $\bar{y}(0) \in \Omega_{\rho^f}$, then $\bar{y}(t) \in \Omega_{\rho^f}$ for all $t \geq 0$ and

$$|\bar{y}(t)| \leq \beta_y\left(|\bar{y}(0)|, \frac{t}{\varepsilon}\right) + \gamma_y(\rho_f^*). \tag{7.21}$$

Remark 7.3 The purpose of the fast LMPC scheme is to stabilize the fast subsystem dynamics while economic considerations are addressed through the slow LEMPC. Depending on the application and certain optimality specifications, the fast LMPC is desired in processes where the fast time-scale is large enough to warrant the use of MPC to achieve optimal performance compared to explicit fast feedback controllers that achieve fast stabilizability without necessarily achieving optimal performance. However, when the fast time-scale is short, an explicit feedback controller may be needed to ensure sufficiently fast computation of the "fast" control action; please see the example section. Additionally, since the closed-loop stability analysis does not require certain conditions on the stage cost of the LMPC, i.e., the stage cost does not need to be a quadratic stage cost, one can readily use an economic stage cost in the fast LMPC formulation.

7.2.4.3 Slow LEMPC Formulation

Referring to the slow subsystem of Eq. 7.4, the slow LEMPC at sampling time t_{k_s} is formulated as follows

$$\max_{u_s \in S(\Delta_s)} \int_{t_{k_s}}^{t_{k_s}+N_s\Delta_s} l_e(\tilde{x}(\tilde{\tau}), u_s(\tilde{\tau}))\, d\tilde{\tau} \tag{7.22a}$$
$$\text{s.t.}\quad \frac{d\tilde{x}(\tilde{\tau})}{d\tilde{\tau}} = f_s(\tilde{x}(\tilde{\tau}), u_s(\tilde{\tau}), 0) \tag{7.22b}$$
$$u_s(\tilde{\tau}) \in \mathbb{U}_s,\ \forall\, \tilde{\tau} \in [t_{k_s}, t_{k_s} + N_s\Delta_s) \tag{7.22c}$$
$$\tilde{x}(t_{k_s}) = \bar{x}(t_{k_s}) \tag{7.22d}$$

$$V_s(\tilde{x}(\tilde{\tau})) \leq \rho_e, \ \forall\, \tilde{\tau} \in [t_{k_s}, t_{k_s} + N_s \Delta_s), \ \text{ if } \ t_{k_s} \leq t_s \ \text{ and } \ V_s(\bar{x}(t_{k_s})) \leq \rho_e \tag{7.22e}$$

$$\frac{\partial V_s(\bar{x}(t_{k_s}))}{\partial \bar{x}} f_s(\bar{x}(t_{k_s}), u_s(t_{k_s}), 0) \leq \frac{\partial V_s(\bar{x}(t_{k_s}))}{\partial \bar{x}} f_s(\bar{x}(t_{k_s}), h_s(\bar{x}(t_{k_s})), 0),$$
$$\text{if } \ t_{k_s} > t_s \text{ or } V_s(x(t_{k_s})) > \rho_e \tag{7.22f}$$

where $S(\Delta_s)$ is the family of piece-wise constant functions with sampling period Δ_s, N_s is the prediction horizon of LEMPC and $l_e(\cdot)$ denotes an economic stage cost function. The symbol $\tilde{x}$ denotes the predicted slow subsystem state trajectory obtained by utilizing the nominal slow subsystem model of Eq. 7.22b initialized by the state feedback of Eq. 7.22d. For mode one operation which corresponds to sampling times $t_{k_s} \leq t_s$, the constraint of Eq. 7.22e maintains the predicted slow state within Ω_{ρ_e} if $V_s(x(t_{k_s})) \leq \rho_e$. If $V_s(x(t_{k_s})) > \rho_e$ or the LEMPC operates under mode two, the constraint of Eq. 7.22f ensures that the amount of reduction in the value of the Lyapunov function $V_s(\cdot)$ when u_s computed by the LEMPC of Eq. 7.22 is applied, is at least at the level when the feedback controller $h_s(\cdot)$ is applied in a sample-and-hold fashion. The optimal solution to the optimization problem of Eq. 7.22 is defined by $u_s^*(\tilde{\tau}|t_{k_s})$ for $\tilde{\tau} \in [t_{k_s}, t_{k_s} + N_s \Delta_s)$ and the manipulated input of the closed-loop slow subsystem under the LEMPC of Eq. 7.22 is defined as follows:

$$u_s(t) = u_s^*(t|t_{k_s}), \ \forall\, t \in [t_{k_s}, t_{k_s} + \Delta_s). \tag{7.23}$$

Next, the stability properties of the slow subsystem under the slow LEMPC is analyzed. By continuity, the smoothness property assumed for the vector field $f_s(\cdot)$ and taking into account that the manipulated input u_s and the disturbance w are bounded in compact sets, there exists a positive constant M_s such that

$$|f_s(\bar{x}, u_s, w)| \leq M_s \tag{7.24}$$

for all $\bar{x} \in \Omega_{\rho^s}$, $u_s \in \mathbb{U}_s$, and $w \in \mathbb{W}$. In addition, by the continuous differentiable property of the Lyapunov function $V_s(\cdot)$ and the smoothness property assumed for the vector field $f_s(\cdot)$, there exist positive constants $L_{\bar{x}}$ and L_{w_s} such that

$$\left| \frac{\partial V_s(\bar{x})}{\partial \bar{x}} f_s(\bar{x}, u_s, w) - \frac{\partial V_s(\bar{x}')}{\partial \bar{x}} f_s(\bar{x}', u_s, w) \right| \leq L_{\bar{x}} |\bar{x} - \bar{x}'| \tag{7.25a}$$

$$\left| \frac{\partial V_s(\bar{x})}{\partial \bar{x}} f_s(\bar{x}, u_s, w) - \frac{\partial V_s(\bar{x})}{\partial \bar{x}} f_s(\bar{x}, u_s, w') \right| \leq L_{w_s} |w - w'| \tag{7.25b}$$

for all $\bar{x}, \bar{x}' \in \Omega_{\rho^s}$, $u_s \in \mathbb{U}_s$, and $w, w' \in \mathbb{W}$. Also, by the smoothness property assumed for the vector function $f(x, z, u_s, w, \varepsilon)$, there exist positive constants L_z, L_ε, L_x, $\bar{L}_z$, $\bar{L}_\varepsilon$, and $\bar{L}_w$ such that

$$\left| f(x, z, u_s, w, \varepsilon) - f(x, z', u_s, w, \varepsilon') \right| \leq L_z |z - z'| + L_\varepsilon |\varepsilon - \varepsilon'| \tag{7.26}$$

$$\left| \frac{\partial V_s(x)}{\partial \bar{x}} f(x, z, u_s, w, \varepsilon) - \frac{\partial V_s(x')}{\partial \bar{x}} f(x', z', u_s, w', \varepsilon') \right| \leq L_x |x - x'| + \bar{L}_z |z - z'| + \bar{L}_\varepsilon |\varepsilon - \varepsilon'| + \bar{L}_w |w - w'| \tag{7.27}$$

for all $x, x' \in \Omega_{\rho^s}$, $z - \tilde{g}(x, w)$, $z' - \tilde{g}(x, w') \in \Omega_{\rho^f}$, $u_f \in \mathbb{U}_f$, and $w, w' \in \mathbb{W}$.

Proposition 7.2 characterizes the closed-loop stability properties of the LEMPC of Eq. 7.11.

Proposition 7.2 (Theorem 4.1) *Consider the slow subsystem of Eq. 7.4 under the LEMPC design of Eqs. 7.22 based on a controller $h_s(\cdot)$ that satisfies the conditions of Eq. 7.7. Let $\varepsilon_{w_s} > 0$, $\Delta_s > 0$, $\rho^s > \rho_e > 0$ and $\rho^s > \rho_s^s > 0$ satisfy*

$$\rho_e \leq \rho^s - f_V(f_W(\Delta_s)) \tag{7.28}$$

where

$$f_V(s) = \alpha_{s_4}(\alpha_{s_1}^{-1}(\rho^s))s + M_v s^2 \tag{7.29}$$

with M_v being a positive constant and

$$f_W(s) = \frac{L_{w_s}\theta}{L_{\bar{x}}}(e^{L_{\bar{x}}s} - 1) \tag{7.30}$$

and

$$-\alpha_{s_3}(\alpha_{s_2}^{-1}(\rho_s^s)) + L_{\bar{x}} M_s \Delta_s + L_{w_s}\theta \leq -\frac{\varepsilon_{w_s}}{\Delta_s}. \tag{7.31}$$

If $\bar{x}(0) \in \Omega_{\rho^s}$, $\rho_s^s < \rho_e$, $\rho_s^* < \rho^s$ and $N_s \geq 1$ then the state $\bar{x}(t)$ of the closed-loop slow subsystem is always bounded in Ω_{ρ^s}. Furthermore, there exists a class $\mathcal{KL}$ function β_x and a class $\mathcal{K}$ function γ_x such that

$$|\bar{x}(t)| \leq \beta_x(|\bar{x}(t^*)|, t - t^*) + \gamma_x(\rho_s^*) \tag{7.32}$$

with $\rho_s^* = \max_{s \in [0, \Delta_s]}\{V_s(\bar{x}(s)) : V_s(\bar{x}(0)) \leq \rho_s^s\}$, for all $\bar{x}(t^*) \in B_\delta \subset \Omega_{\rho^s}$ and for $t \geq t^* > t_s$ where t^* is chosen such that $\bar{x}(t^*) \in B_\delta$ and $B_\delta = \{x \in \mathbb{R}^{n_x} : |x| \leq \delta\}$.

The proof includes similar steps as the proof of Proposition 7.1. For the specific details, refer to Theorem 4.1.

7.2.4.4 Closed-Loop Stability

The closed-loop stability of the system of Eq. 7.1 under the LMPC of Eq. 7.11 and LEMPC of Eq. 7.22 is established in the following theorem under appropriate conditions.

Theorem 7.1 *Consider the system of Eq. 7.1 in closed-loop with u_f and u_s computed by the LMPC of Eq. 7.11 and LEMPC of Eq. 7.22 based on the Lyapunov-based controllers $p(x)y$ and $h_s(\cdot)$ that satisfy the conditions of Eqs. 7.7 and 7.10, respectively. Let Assumptions 7.1 and 7.2 and the conditions of Propositions 7.1 and 7.2 hold and*

$$\rho_e + (M_s + L_z M_y + L_\varepsilon \varepsilon)\Delta_s \alpha_{s_4}(\alpha_{s_1}^{-1}(\rho^s)) < \rho^s \tag{7.33}$$

and

$$-\alpha_{s_3}(\alpha_{s_1}^{-1}(\rho_s^s)) + d_1 < 0 \tag{7.34}$$

where L_z, M_y and d_1 are positive constants to be defined in the proof. Then there exist functions β_x of class $\mathscr{KL}$ and γ_x of class $\mathscr{K}$, a pair of positive real numbers (δ_x, d) and $\varepsilon^* > 0$ such that if

$$\max\{|x(0)|, |y(0)|, \|w\|, \|\dot{w}\|\} \leq \delta_x$$

where $\|w\|$ denotes $\text{ess sup}_{t\geq 0}|w(t)|$ and $\varepsilon \in (0, \varepsilon^*]$, then $x(t) \in \Omega_{\rho^s}$ and $y(t) \in \Omega_{\rho^f}$ for $t \geq 0$ and

$$|x(t)| \leq \beta_x(|x(t^*)|, t - t^*) + \gamma_x(\rho_s^*) + d \tag{7.35}$$

for all $t \geq t^* > t_s$ where t^* has been defined in Proposition 7.2.

Proof When $u_f = u_f^*$ and $u_s = u_s^*$ are determined by the LMPC of Eq. 7.11 and LEMPC of Eq. 7.22, respectively, the closed-loop system takes the following form:

$$\begin{aligned} \dot{x} &= f(x, z, u_s^*, w, \varepsilon) \\ \varepsilon\dot{z} &= g(x, z, u_f^*, w, \varepsilon). \end{aligned} \tag{7.36}$$

We will first compute the slow and fast closed-loop subsystems.

Setting $\varepsilon = 0$ in Eq. 7.36 and taking advantage of the fact that $u_f^* = 0$ when $\varepsilon = 0$, one obtains that:

$$\begin{aligned} \frac{d\bar{x}}{dt} &= f(\bar{x}, \bar{z}, u_s^*, w, 0) \\ 0 &= g(\bar{x}, \bar{z}, 0, w, 0). \end{aligned} \tag{7.37}$$

Using Assumption 7.1, Eq. 7.37 can be written as follows:

$$\frac{d\bar{x}}{dt} = f(\bar{x}, \tilde{g}(\bar{x}, w), u_s^*, w, 0) = f_s(\bar{x}, u_s^*, w) \tag{7.38}$$

Let $\tau = t/\varepsilon$ and $y = z - \tilde{g}(x, w)$, and the closed-loop system of Eq. 7.36 can be written as:

$$\begin{aligned}\frac{dx}{d\tau} &= \varepsilon f(x, y + \tilde{g}(x, w), u_s^*, w, \varepsilon) \\ \frac{dy}{d\tau} &= g(x, y + \tilde{g}(x, w), u_f^*, w, \varepsilon) - \varepsilon \frac{\partial \tilde{g}}{\partial w}\dot{w} - \varepsilon \frac{\partial \tilde{g}}{\partial x} f(x, y + \tilde{g}(x, w), u_s^*, w, \varepsilon)\end{aligned} \tag{7.39}$$

Setting $\varepsilon = 0$, the following closed-loop fast subsystem is obtained:

$$\frac{d\bar{y}}{d\tau} = g(x, \bar{y} + \tilde{g}(x, w), u_f^*, w, 0) \tag{7.40}$$

Now, we focus on the singularly perturbed system of Eq. 7.39. Considering the fast subsystem state $y(t)$ of Eq. 7.39 and assuming that $x(t)$ is bounded in Ω_{ρ^s} (which will be proved later), it can be obtained using a Lyapunov argument that there exist positive constants δ_{x_1} and ε_1 such that if $\max\{|x(0)|, |y(0)|, \|w\|, \|\dot{w}\|\} \leq \delta_{x_1}$ and $\varepsilon \in (0, \varepsilon_1]$, there exists a positive constant k_1 such that:

$$|z - \tilde{g}(x, w)| = |y(t)| \leq \beta_y \left(\delta_{x_1}, \frac{t}{\varepsilon}\right) + \gamma_y(\rho_f^*) + k_1 \tag{7.41}$$

for all $t \geq 0$. We consider $t \in (0, \Delta_s]$ and $t \geq \Delta_s$ separately and prove that if the conditions stated in Theorem 7.1 are satisfied, the boundedness of the states of the system of Eq. 7.39 is ensured. When $x(0) \in B_{\delta_{x_2}} \subset \Omega_{\rho_e} \subset \Omega_{\rho^s}$, where δ_{x_2} is a positive real number, considering the closed-loop slow subsystem of Eq. 7.39 state trajectory:

$$\dot{x}(t) = f(x, y + \tilde{g}(x, w), u_s^*, w, \varepsilon), \;\forall\, t \in (0, \Delta_s]$$

and considering the facts that

$$\begin{aligned}|f(x, z, u_s^*, w, \varepsilon)| &\leq |f_s(x, u_s^*, w)| + |f(x, z, u_s^*, w, \varepsilon) - f_s(x, u_s^*, w)|, \\ |f_s(x, u_s^*, w)| &\leq M_s, \\ |y(t)| &\leq \beta_y(\delta_{x_2}, 0) + \gamma_y(\rho_f^*) + k_1 < M_y\end{aligned}$$

where M_y is a positive constant such that

$$\begin{aligned}|f(x, z, u_s^*, w, \varepsilon) - f_s(x, u_s^*, w)| &= |f(x, z, u_s^*, w, \varepsilon) - f(x, \tilde{g}(x, w), u_s^*, w, 0)| \\ &\leq L_z|z - \tilde{g}(x, w)| + L_\varepsilon \varepsilon \leq L_z M_y + L_\varepsilon \varepsilon\end{aligned} \tag{7.42}$$

and

$$\begin{aligned}V_s(x(t)) &= V_s(x(0)) + \int_0^t \dot{V}_s(x(s))\, ds \\ &= V_s(x(0)) + \int_0^t \frac{\partial V_s(x(s))}{\partial x}\dot{x}(s)\, ds\end{aligned}$$

$$\leq \rho_e + (M_s + L_z M_y + L_\varepsilon \varepsilon)\Delta_s \alpha_{s_4}(\alpha_{s_1}^{-1}(\rho^s)) \tag{7.43}$$

Thus, there exists Δ_1 and ε_2 such that if $\Delta_s \in (0, \Delta_1]$ and $\varepsilon \in (0, \varepsilon_2]$, Eq. 7.33 holds and

$$V_s(x(t)) < \rho^s, \ \forall\, t \in (0, \Delta_s] \tag{7.44}$$

Picking $\varepsilon_3 = \min\{\varepsilon_1, \varepsilon_2\}$ ensures that $x(t) \in \Omega_{\rho^s}$ and $y(t) \in \Omega_{\rho^f}$ for all $t \in [0, \Delta_s)$.

For $t \geq \Delta_s$, considering Eq. 7.41, there exists a positive real number $\bar{M}_y$ such that

$$|y(t)| \leq \beta_y\left(\delta_{x_2}, \frac{\Delta_s}{\varepsilon}\right) + \gamma_y(\rho_f^*) + k_1 \leq \bar{M}_y \tag{7.45}$$

and we can write the time derivative of the Lyapunov function $V_s(\cdot)$ along the closed-loop system state of Eq. 7.1 under the LEMPC of Eq. 7.22 for $t \in [t_{k_s}, t_{k_s+1})$ (assuming without loss of generality that $t_{k_s} = \Delta_s$) as follows

$$\dot{V}_s(x(t)) = \frac{\partial V_s(x(t))}{\partial x} f(x(t), z(t), u_s^*(t_{k_s}), w(t), \varepsilon) \tag{7.46}$$

Adding/subtracting the terms:

$$\frac{\partial V_s(x(t_{k_s}))}{\partial x} f_s(x(t_{k_s}), u_s(t_{k_s}), 0) \text{ and } \frac{\partial V_s(x(t))}{\partial x} f_s(x(t), u_s(t_{k_s}), w(t))$$

to/from the above inequality and taking advantage of Eqs. 7.7 and 7.22f and the fact that

$$\begin{aligned}
&|f(x(t), z(t), u_s^*(t_{k_s}), w(t), \varepsilon)| \\
&\quad \leq |f_s(x(t), u_s^*(t_{k_s}), 0)| \\
&\qquad + |f_s(x(t), u_s^*(t_{k_s}), w(t)) - f_s(x(t_{k_s}), u_s^*(t_{k_s}), 0)| \\
&\qquad + |f(x(t), z(t), u_s^*(t_{k_s}), w(t), \varepsilon) - f_s(x(t), u_s^*(t_{k_s}), w(t))|
\end{aligned}$$

and considering

$$\left|\frac{\partial V_s(x(t))}{\partial x} f_s(x, u_s^*, w)\right| \leq \alpha_{s_4}(\alpha_{s_1}^{-1}(\rho^s))M_s, \tag{7.47}$$

$$\begin{aligned}
&\left|\frac{\partial V_s(x(t))}{\partial x} f(x(t), z(t), u_s^*(t_{k_s}), w(t), \varepsilon) - \frac{\partial V_s(x(t))}{\partial x} f_s(x(t), u_s^*(t_{k_s}), w(t))\right| \\
&\quad \leq L_{\bar{z}}|z - \tilde{g}(x, w)| + L_{\bar{\varepsilon}}\varepsilon,
\end{aligned} \tag{7.48}$$

$$|z - \tilde{g}(x, w)| \leq \bar{M}_y, \tag{7.49}$$

$$\left| \frac{\partial V_s(x(t))}{\partial x} f_s(x(t), u_s^*(t_{k_s}), w(t)) - \frac{\partial V_s(x(t_{k_s}))}{\partial x} f_s(x(t_{k_s}), u_s^*(t_{k_s}), 0) \right| \\ \leq L_{\bar{x}} |x(t) - x(t_{k_s})| + L_{w_s}\theta, \tag{7.50}$$

and

$$|x(t) - x(t_{k_s})| \leq M_s \Delta_s, \tag{7.51}$$

we can obtain

$$\dot{V}_s(x(t)) \leq -\alpha_{s_3}(\alpha_{s_1}^{-1}(\rho_s^s)) + d_1 \tag{7.52}$$

where

$$d_1 = \alpha_{s_4}(\alpha_{s_1}^{-1}(\rho^s))M_s + L_{\bar{z}}\bar{M}_y + L_{\bar{\varepsilon}}\varepsilon + L_{\bar{x}} M_s \Delta_s + L_{w_s}\|w\| \tag{7.53}$$

where d_1 is a positive constant. Picking δ_{x_2}, ε_4 and Δ_2 such that for any $\varepsilon \in (0, \varepsilon_4]$, $\max\{|x(0)|, |y(0)|, \|w\|, \|\dot{w}\|\} \leq \delta_{x_2}$ and for all $\Delta_s \in (0, \Delta_2]$, Eq. 7.34 is satisfied, the closed-loop system state $x(t)$ is bounded in Ω_{ρ^s} for all $t \geq \Delta_s$. Finally, using similar arguments to the proof of Theorem 1 in [11], there exist a class $\mathscr{K}\mathscr{L}$ function β_x and a class $\mathscr{K}$ function γ_x, positive real numbers (δ_x, d_x) where $\delta_x < \min\{\delta_{x_1}, \delta_{x_2}\}$, and $0 < \varepsilon^* < \min\{\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4\}$ and $0 < \Delta^* < \min\{\Delta_1, \Delta_2\}$ such that if $\max\{|x(0)|, |y(0)|, \|w\|, \|\dot{w}\|\} \leq \delta_x$, $\varepsilon \in (0, \varepsilon^*]$ and $\Delta_s \in (0, \Delta^*]$, then, the bound of Eq. 7.35 holds for all $t \geq t^* > t_s$.

Remark 7.4 It should be emphasized that in Theorem 7.1, it has been indicated that for operation periods corresponding to LEMPC mode 1 operation, both of fast and slow reduced order subsystem states are bounded in the invariant sets $\Omega_{\tilde{\rho}^s}$ and $\Omega_{\tilde{\rho}^f}$ to ensure that the actual states of the system are bounded in certain stability regions, i.e., Ω_{ρ^s} and Ω_{ρ^f} through restricting their corresponding initial states. On the other hand, for operation periods corresponding to LEMPC mode 2 operation, both of system states are asymptotically bounded in a small invariant set containing the origin.

Remark 7.5 While the present work focuses on nonlinear singularly perturbed systems and general economic stage cost functions, the results of this work are novel and apply to the case of linear singularly perturbed systems; however, in the linear case, the verification of the assumption that there is an isolated equilibrium manifold for the fast dynamics (Assumption 1), the construction of the explicit control laws for the slow and fast subsystems imposed in Sect. 2.4 and the computation of the associated closed-loop stability regions, and the solution of the LEMPC and LMPC optimization problems when convex economic cost functions are used simplify significantly, given the availability of robust and efficient tools for matrix calculations and convex optimization.

7.2.5 Application to a Chemical Process Example

Consider a well-mixed, non-isothermal continuous stirred tank reactor (CSTR) where an irreversible, second-order, endothermic reaction $A \rightarrow B$ takes place, where A is the reactant and B is the desired product. The feed to the reactor consists of the reactant A and an inert gas at flow rate F, temperature T_0 and molar concentration C_{A0}. Due to the non-isothermal nature of the reactor, a jacket is used to provide heat to the reactor. The dynamic equations describing the behavior of the reactor, obtained through material and energy balances under standard modeling assumptions, are given below:

$$\frac{dC_A}{dt} = \frac{F}{V_R}(C_{A0} - C_A) - k_0 e^{-E/RT} C_A^2 \tag{7.54a}$$

$$\rho_R C_p \frac{dT}{dt} = \frac{F \rho_R C_p}{V_R}(T_0 - T) - \Delta H k_0 e^{-E/RT} C_A^2 + \frac{Q}{V_R} \tag{7.54b}$$

where C_A denotes the concentration of the reactant A, T denotes the temperature of the reactor, Q denotes the rate of heat supply to the reactor, V_R represents the volume of the reactor, ΔH, k_0, and E denote the enthalpy, pre-exponential constant and activation energy of the reaction, respectively, and C_p and ρ_R denote the heat capacity and the density of the fluid in the reactor, respectively. The values of the process parameters used in the simulations are shown in Table 7.1. The process model of Eq. 7.54 is numerically simulated using an explicit Euler integration method with integration step $h_c = 1.0 \times 1.0^{-6}$ h.

The process model has one unstable steady-state in the operating range of interest. The control objective is to optimize the process operation in a region around the unstable steady-state (C_{As}, T_s) to maximize the average production rate of B through manipulation of the concentration of A in the inlet to the reactor. The steady-state input value associated with the steady-state point is denoted by C_{A0s}. Defining $\varepsilon = \rho_R c_P$, the following nonlinear state-space model can be obtained

$$\dot{x}(t) = f(x(t), z(t), u_s(t), 0, \varepsilon)$$
$$\varepsilon \dot{z}(t) = g(x(t), z(t), u_f(t), 0, \varepsilon)$$

Table 7.1 Parameter values

$T_0 = 300$	K	$F = 5$	$\mathrm{m^3\,h^{-1}}$
$V_R = 1.0$	$\mathrm{m^3}$	$E = 5 \times 10^4$	$\mathrm{kJ\,kmol^{-1}}$
$k_0 = 8.46 \times 10^6$	$\mathrm{m^3\,kmol^{-1}\,h^{-1}}$	$\Delta H = -19.91$	$\mathrm{kJ\,kmol^{-1}}$
$C_p = 0.02$	$\mathrm{kJ\,kg^{-1}\,K^{-1}}$	$R = 8.314$	$\mathrm{kJ\,kmol^{-1}\,K^{-1}}$
$\rho_R = 20$	$\mathrm{kg\,m^{-3}}$	$C_{As} = 1.95$	$\mathrm{kmol\,m^{-3}}$
$T_s = 401.87$	K	$C_{A0s} = 4$	$\mathrm{kmol\,m^{-3}}$
$Q_s = 0$	$\mathrm{kJ\,h^{-1}}$		

where $x = C_A - C_{As}$ and $z = T - T_s$ are the states, $u_s = C_{A0} - C_{A0s}$ and $u_f = Q - Q_s$ are the inputs and f and g are scalar functions. The inputs are subject to constraints as follows: $|u_s| \leq 3.5\,\mathrm{kmol\,m^{-3}}$ and $|u_f| \leq 5.0 \times 10^5\,\mathrm{kJ\,h^{-1}}$. The economic measure considered in this example is as follows:

$$\frac{1}{t_N} \int_0^{t_N} k_0 e^{-E/RT(\tau)} C_A^2(\tau)\, d\tau \tag{7.55}$$

where $t_N = 1.0\,\mathrm{h}$ is the time duration of a reactor operating period. This economic objective function highlights the maximization of the average production rate over a process operation period for $t_N = 1.0\,\mathrm{h}$ (of course, different, yet finite, values of t_N can be chosen). A limitation on the amount of reactant material that may be used over the period t_N is considered. Specifically, the control input trajectory of u_s should satisfy the following constraint:

$$\frac{1}{t_N} \int_0^{t_N} u_s(\tau)\, d\tau = 1.0\,\mathrm{kmol\,m^{-3}}. \tag{7.56}$$

This constraint means that the available amount of reactant material over one period is fixed. For the sake of simplicity, the constraint of Eq. 7.56 is referred for the material constraint for the remainder.

In terms of the Lyapunov-based controllers, feedback linearization techniques are utilized for the design of explicit controllers for the fast and slow reduced-order subsystems subject to input constraints and quadratic Lyapunov functions $V_s(x) = x^2$ and $V_f(y) = y^2$ are used to compute the stability regions. Through feedback linearization and evaluating $\dot{V}_s(\cdot)$ subject to the input constraint, $\dot{V}_s(x) \leq 0$ when $x \in \Omega_{\rho^s}$ and $\rho^s = 4$. Furthermore, to guarantee that $C_A > 0$ and $T \leq 480$ K, the corresponding stability Ω_{ρ^s} is defined as $\Omega_{\rho^s} = \{x | -1.15 \leq x \leq 3.95\}$.

In this example, a slow LEMPC is designed to regulate the slow subsystem state, which maximizes the average production rate of the desired product B, and a fast feedback linearizing controller is designed to stabilize the fast subsystem state. With respect to the fast feedback linearizing controller, the deviation variable $y(t)$ is defined as $z(t) - \tilde{z}(t)$ where $\tilde{z}(t)$ is the unique root of the algebraic equation $g(x(t), \tilde{z}(t), 0, 0, 0) = 0$ given $x(t)$. For the purpose of simulation, this unique root has been approximated through a tenth-order polynomial. Furthermore, we assume that the state measurements are available every $\Delta_f = 1.0 \times 1.0^{-6}$ h and the manipulated input u_f is obtained every Δ_f such that

$$g(x, y + \tilde{g}(x, 0), 0, u_f, 0) = -\mu y \tag{7.57}$$

where $\mu = 100$. Regarding the slow dynamics, the LEMPC obtains its manipulated input trajectory u_s every $\Delta_s = 1.0 \times 10^{-2}$ h by optimizing the objective function of Eq. 7.55 using the one dimensional slow subsystem which is independent of ε. As a result, the slow subsystem used in the LEMPC is well conditioned and is integrated with time step 1.0×10^{-3} h resulting in a nearly-three order of magnitude improvement in the computational time needed to compute the control action for u_s. The LEMPC operates only under mode 1 operation to highlight the effect of economic optimization. Considering the material constraint which needs to be satisfied through one period of process operation, a shrinking prediction horizon sequence is employed in the LEMPC in the sense that $N_0, \ldots, N_{99}$ where $N_i = 100 - i$ denotes the prediction horizon at time step i and $i = 0, \ldots, 99$.

Figures 7.1, 7.2, 7.3, 7.4 and 7.5 illustrate closed-loop state and manipulated input trajectories of the chemical process of Eq. 7.54 under the mode one operation of the LEMPC design of Eq. 7.22 and feedback linearization of Eq. 7.57 for an initial condition of $(C_A(0), T(0)) = (3.0\,\text{kmol}\,\text{m}^{-3}, 400\,\text{K})$. From these figures, the economic cost of Eq. 7.55 is optimized by dictating time-varying operation by the LEMPC when considering the material constraint of Eq. 7.56. Furthermore, u_f through feedback linearization ensures that the fast subsystem state $y(t)$ converges to zero. We point out that either the open-loop or closed-loop dynamics can evolve on different time-scales. In this example, feedback linearization is used with a gain chosen to drive the deviation variable y to zero fast relative to the slow state C_A as observed in Fig. 7.3. Therefore, this illustrative example possesses two-time-scale behavior.

A set of simulations is performed to compare the economic closed-loop performance of the method versus the case that the input material is fed to the reactor uniformly in time, i.e., $u_s(t) = 1.0\,\text{kmol}\,\text{m}^{-3}$ for all $t \in [0, 1.0\,\text{h}]$. To carry out this comparison, the total cost of each scenario is computed based on the index of the following form:

$$J = \frac{1}{t_M} \sum_{i=0}^{M-1} \left(k_0 e^{-E/RT(t_i)} C_A^2(t_i) \right)$$

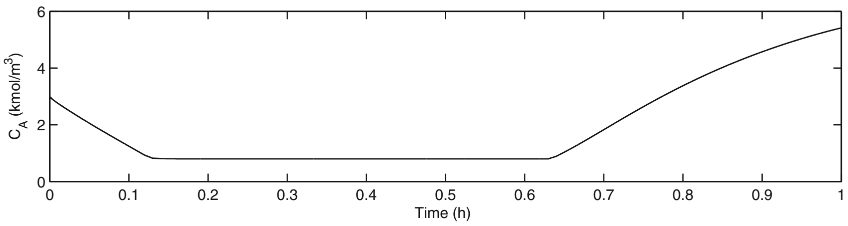


Fig. 7.1 The closed-loop reactant concentration profile under the composite control structure

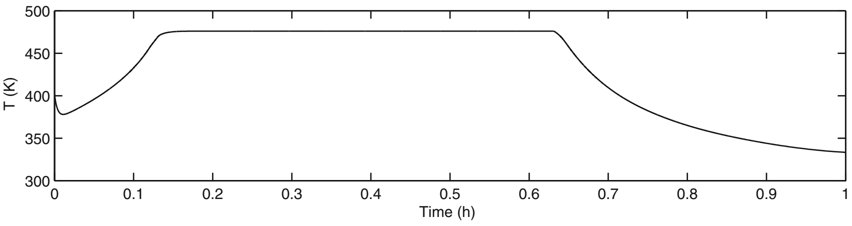


Fig. 7.2 The closed-loop temperature profile under the composite control structure

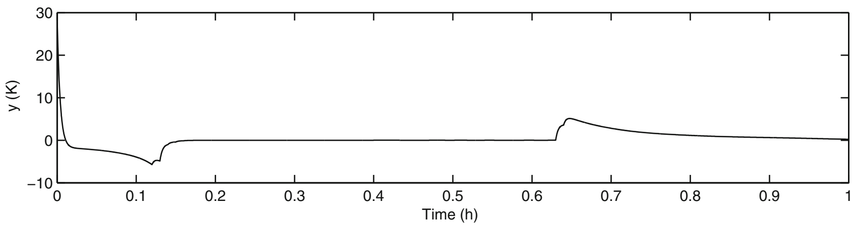


Fig. 7.3 The closed-loop profile of $y = z - \tilde{g}(x, 0)$

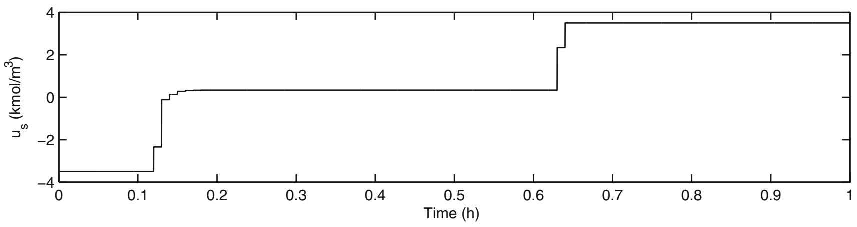


Fig. 7.4 The manipulated input u_s profile under the slow LEMPC, which optimizes the production rate of the desired product B

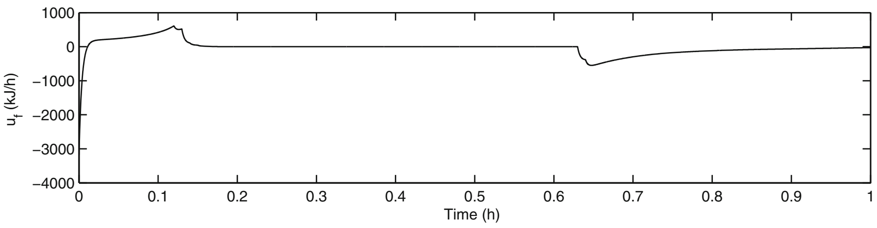


Fig. 7.5 The manipulated input u_f profile under the feedback linearizing controller of Eq. 7.57

where $t_0 = 0.0$ h, $t_i = i0.01$ h and $M = 100$. By comparing the performance index J for these two cases, the LEMPC through a time-varying operation achieves a greater cost value, i.e., larger average production rate, compared to the case that the reactant material is fed to the reactor uniformly in time (13.12 versus 5.92, respectively).

7.3 Distributed EMPC: Evaluation of Sequential and Iterative Architectures

It is possible that significant computation delay may result, which may affect closed-loop stability and performance, when computing control actions for large-scale process systems with many states and inputs. In the context of control of large-scale nonlinear chemical process networks, an alternative is to employ a distributed

MPC (DMPC) architecture (see, for example, the review [12] and references contained therein). DMPC has the ability to control large-scale process systems subject to input and state constraints while remaining computationally feasible to be implemented on-line through a distributed implementation of the computations. Numerous formulations, implementation strategies, and theoretical results have been developed within the context of tracking DMPC, e.g., [13, 14]; see, also, the reviews of [12, 15] and the references therein. In the context of distributed EMPC (DEMPC), some work has been completed including DEMPC for linear systems [16, 17] and for nonlinear systems [18, 19].

In this section, sequential and iterative DEMPC strategies are developed and applied to the benchmark catalytic reactor to produce ethylene oxide from ethylene, which was first presented in Sect. 1.3.1. Recall, in Sect. 3.2, the application of EMPC to the catalytic reactor resulted in improved average yield of ethylene oxide compared to the yield of steady-state operation and to the yield achieved under an open-loop optimal periodic switching of the inputs considered in [20]. Here, several EMPC implementation strategies (centralized and distributed) are applied to the catalytic reactor. A description of the DEMPC implementation strategies is provided. Several closed-loop simulations are performed to evaluate the approaches. Two key performance metrics are considered in the evaluation: the closed-loop economic performance under the various DEMPC strategies and the on-line computation time required to solve the EMPC optimization problems.

Regarding the implementation details of the EMPC systems below, a sampling period of $\Delta = 1.0$ (dimensionless time units) was used. The optimization problems were solved using the interior point solver Ipopt [21]. To account for real-time computation considerations, the solver was forced to terminate after 100 iterations and/or after 100 seconds of computation time. The tolerance of the solver was set to 10^{-5}. To satisfy the constraint on the amount of ethylene that may be fed to the reactor, this constraint was enforced over operating windows of length $t_p = 47$, that is the average molar flow rate of ethylene must be equal to 0.175 at the end of each operating window (refer to Sect. 1.3.1 for more details regarding this average material constraint). A shrinking horizon approach was used within EMPC: at the beginning of the jth operating window, the prediction horizon was set to $N_k := t_p/\Delta$ and the horizon was decreased by one at every subsequent sampling time ($N_k = N_{k-1} - 1$ at the sampling instance t_k). At the beginning of the $(j + 1)$th operating window, the prediction horizon was set to t_p/Δ.

The closed-loop simulations below were programmed using C++ on a desktop computer with an Ubuntu Linux operating system and an Intel® Core™ i7 3.4 GHz processor. To recursively solve the catalytic reactor dynamic model, the explicit Euler method was used. A step size of 0.00001 was used to simulate the closed-loop dynamics of the reactor, while a step size of 0.005 was used to solve the model within the EMPC problem; both led to stable numerical integration.

7.3.1 Centralized EMPC

For this computational study, a centralized EMPC (C-EMPC) strategy is considered to compare against the two distributed implementation strategies. Recall, the catalytic reactor that produces ethylene oxide from ethylene has three inputs: the inlet flow rate, u_1, the ethylene concentration in the inlet, u_2, and the coolant temperature in the reactor jacket. Also, the reactor model has the form of:

$$\dot{x} = f(x, u_1, u_2, u_3) \tag{7.58}$$

where the state vector $x \in \mathbb{R}^4$ includes the reactor content density, x_1, the reactor ethylene concentration, x_2, the reactor ethylene oxide concentration, x_3, and the reactor temperature, x_4. The C-EMPC formulation with an economic stage cost function that maximizes the yield of ethylene oxide is given by:

$$\max_{u_1,u_2,u_3 \in S(\Delta)} \int_{t_k}^{t_k+N_k\Delta} u_1(\tau)\tilde{x}_4(\tau)\tilde{x}_3(\tau)\, d\tau \tag{7.59a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = f(\tilde{x}(t), u_1(t), u_2(t), u_3(t)) \tag{7.59b}$$

$$u_i(t) \in \mathbb{U}_i, \ \forall\, t \in [t_k, t_k + N_k\Delta), \ i = 1, 2, 3 \tag{7.59c}$$

$$\frac{1}{t_p} \int_{t_k}^{t_k+N_k\Delta} u_1(t)u_2(t)\, dt = 0.175 - \frac{1}{t_p} \int_{t_0+jt_p}^{t_k} u_1^*(t)u_2^*(t)\, dt \tag{7.59d}$$

where $\mathbb{U}_i$ denotes the set of admissible values of the ith input (refer to Sect. 1.3.1 for more details) and u_1^* and u_2^* denote the optimal control actions applied to the reactor from the beginning of the current operating window to current sampling time, t_k. The EMPC problem of Eq. 7.59 maximizes the yield of ethylene oxide (or more precisely, the numerator of the yield) over the prediction horizon (Eq. 7.59a) subject to the dynamic process model to predict the future behavior of the reactor (Eq. 7.59b), the inputs constraints (Eq. 7.59c), and the average constraint on amount of ethylene that may be fed to the reactor (Eq. 7.59d).

Figures 7.6 and 7.7 depict the closed-loop state and input trajectories under the C-EMPC scheme over ten operating windows. Similar to the results of [22], the C-EMPC distributes the ethylene in a non-uniform fashion with respect to time to optimize the yield of ethylene oxide. The average yield of ethylene oxide of the reactor under the C-EMPC is 10.22 yield of ethylene oxide of the reactor over the same length of operation under constant steady-state input values is 6.38 and the average yield under EMPC is 60 percent better than that achieved under steady-state operation.

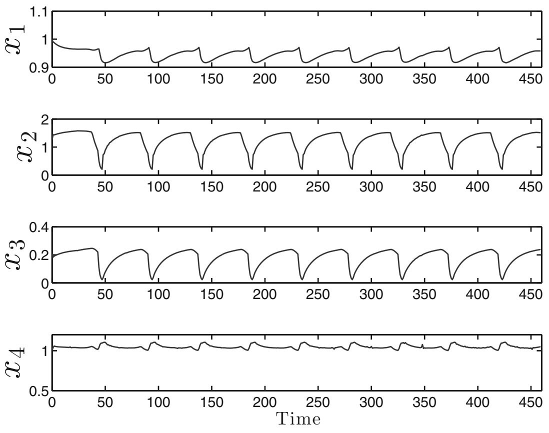


Fig. 7.6 State trajectories under C-EMPC

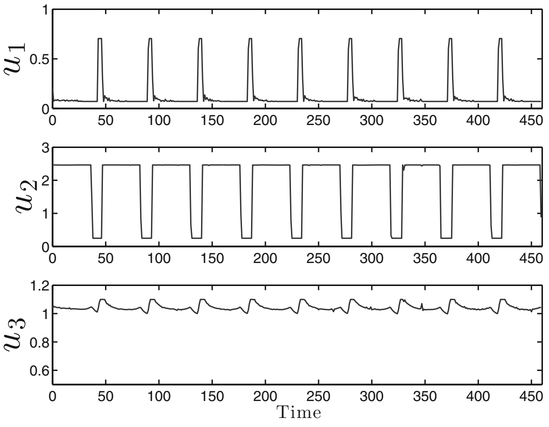


Fig. 7.7 Input trajectories computed by the C-EMPC

7.3.2 Sequential DEMPC

A sequential DEMPC implementation strategy computes the control actions by sequentially solving a series of DEMPC problems. One-way communication is used between controllers to send the computed input trajectories from one EMPC to the next EMPC. The next EMPC also receives the input trajectory from all other previously solved EMPCs. Once all the input trajectories are received, the EMPC is solved utilizing this information. The resulting trajectories are then sent to the subsequent EMPC. The process is repeated until all EMPCs are solved and the control actions for all inputs computed by the sequential DEMPC approach are obtained.

For the catalytic reactor example, which has three inputs, a reasonable choice of input grouping can be made as a consequence of the integral input constraint. The inputs u_1 and u_2 should be computed by the same EMPC, while it is worth investigating if the input u_3 can be placed on another EMPC system. This input pairing will be used in all the DEMPC schemes below. The formulation of the EMPC problem that computes control actions for u_1 and u_2, which is denoted as EMPC-1, is given by

$$\max_{u_1,u_2 \in S(\Delta)} \int_{t_k}^{t_k+N_k\Delta} u_1(\tau)\tilde{x}_4(\tau)\tilde{x}_3(\tau)\, d\tau \tag{7.60a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = f(\tilde{x}(t), u_1(t), u_2(t), \hat{u}_3(t)) \tag{7.60b}$$

$$u_i(t) \in \mathbb{U}_i,\ \forall\, t \in [t_k, t_k + N_k\Delta),\ i = 1, 2 \tag{7.60c}$$

$$\frac{1}{t_p}\int_{t_k}^{t_k+N_k\Delta} u_1(t)u_2(t)\, dt = 0.175 - \frac{1}{t_p}\int_{t_0+jt_p}^{t_k} u_1^*(t)u_2^*(t)\, dt \tag{7.60d}$$

and the formulation of the EMPC that computes control actions for u_3, which is denoted as EMPC-2 is given by:

$$\max_{u_3 \in S(\Delta)} \int_{t_k}^{t_k+N_k\Delta} u_1(\tau)\tilde{x}_4(\tau)\tilde{x}_3(\tau)\, d\tau \tag{7.61a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = f(\tilde{x}(t), \hat{u}_1(t), \hat{u}_2(t), u_3(t)) \tag{7.61b}$$

$$u_3(t) \in \mathbb{U}_3,\ \forall\, t \in [t_k, t_k + N_k\Delta) \tag{7.61c}$$

In the problems of Eqs. 7.60–7.61, the input trajectories denoted by $\hat{u}_i$ must be provided before the problems may be solved. The input trajectory $\hat{u}_3$ must be assumed if EMPC-1 is solved first. In general, the assumed input trajectory may be a constant input trajectory, an input trajectory computed by an explicit controller, or the input trajectory of EMPC-2 computed at the previous sampling time. Similarly, if EMPC-2 is solved first in the sequential DEMPC architecture, the input trajectories $\hat{u}_1$ and $\hat{u}_2$ must be assumed before solving EMPC-2.

7.3.2.1 Sequential DEMPC 1-2

The first configuration considered, which is referred to as the sequential DEMPC 1-2 and abbreviated to S-DEMPC 1-2, first solves the EMPC-1 problem for the optimal input trajectories $u_1^*(t|t_k)$ and $u_2^*(t|t_k)$ for $t \in [t_k, t_{k+N})$. Then, the EMPC-2 problem is solved to compute the input trajectory $u_3^*(t|t_k)$ after receiving $u_1^*(t|t_k)$ and $u_2^*(t|t_k)$ from EMPC-1. A block diagram of the resulting control architecture showing the communication between the controllers is given in Fig. 7.8. Since the input trajectory $\hat{u}_3(t)$ for $t \in [t_k, t_{k+N})$ has not been determined when the EMPC-1 problem is solved, it is set to be the resulting input trajectory under a proportional-integral (PI) controller implemented in a sample-and-hold fashion over the prediction horizon (other methods for the assumed profile of $\hat{u}_3(t)$ within EMPC-1 could be considered).

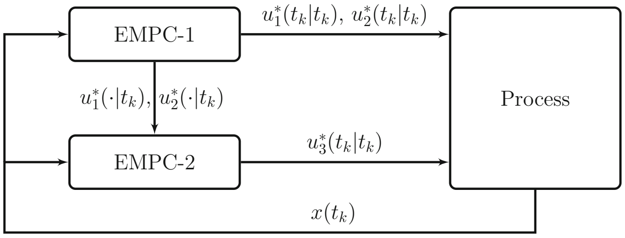

EMPC-1
EMPC-2
Process
$u_1^*(t_k|t_k),\ u_2^*(t_k|t_k)$
$u_1^*(\cdot|t_k), u_2^*(\cdot|t_k)$
$u_3^*(t_k|t_k)$
$x(t_k)$

Fig. 7.8 A block diagram of the S-DEMPC 1-2 scheme

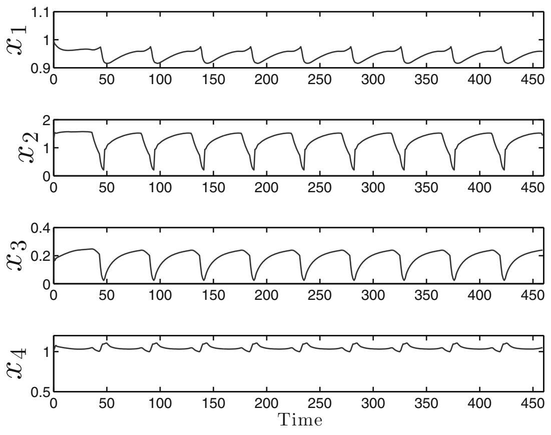


Fig. 7.9 Closed-loop state trajectories of the catalytic reactor under the S-DEMPC 1-2

The input constraints are accounted for in the computed PI input trajectory, e.g., if the PI controller computes a control action greater than the upper bound on u_3, it is set to $u_{3,\max}$. For the input trajectories $\hat{u}_1$ and $\hat{u}_2$ in the EMPC-2 problem, the optimal input trajectories computed by the EMPC-1 problem are used.

Figures 7.9 and 7.10 show the closed-loop state and input trajectories under the S-DEMPC 1-2, respectively. The trajectories are similar to those under the C-EMPC (Figs. 7.6 and 7.7). For the closed-loop simulation, the average yield was 10.20 (recall, the average yield under the C-EMPC was 10.22 differences in the state trajectories are observed from Figs. 7.6 and 7.9, e.g., in the x_1 and x_4 trajectories. It is important to note that given the nonlinear nature of the process considered, there is no guarantee, in general, that the centralized EMPC and sequential EMPC scheme will lead to the same or even similar optimal input trajectories.

7.3.2.2 Sequential DEMPC 2-1

Another sequential distributed implementation of EMPC-1 and EMPC-2 may be considered by reversing the execution of EMPC-1 and EMPC-2. In this DEMPC approach, which is shown in Fig. 7.11, EMPC-2 computes its optimal input trajectory

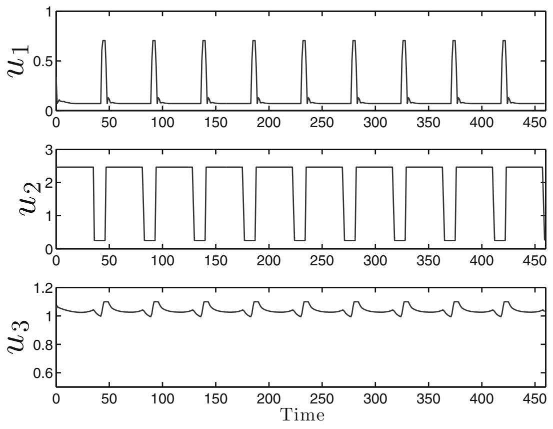


Fig. 7.10 Closed-loop input trajectories computed by the S-DEMPC 1-2

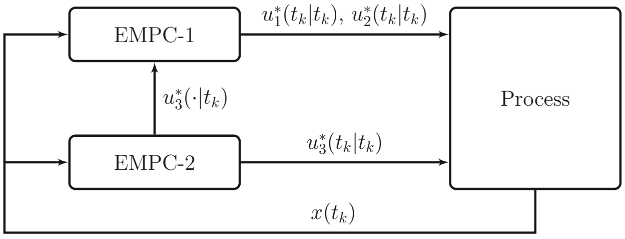


Fig. 7.11 A block diagram of the S-DEMPC 1-2 scheme

$u_3^*(t|t_k)$ for $t \in [t_k, t_{k+N})$ first. The sequential DEMPC approach is referred to as the sequential DEMPC 2-1 (S-DEMPC 2-1). To solve EMPC-2, the trajectories $\hat{u}_1(t)$ and $\hat{u}_2(t)$ for $t \in [t_k, t_{k+N})$ are set to the input trajectories resulting from two PI controllers implemented in sample-and-hold fashion. While the bounds on admissible input values are accounted for in the PI input trajectories, the input average constraint is not accounted for in the PI input trajectories. Figures 7.12 and 7.13 give the closed-loop state and input trajectories under the S-DEMPC 2-1 approach. Noticeable differences are observed between the closed-loop trajectories under the S-DEMPC 2-1 approach and those under the C-EMPC approach (Figs. 7.6 and 7.7).

7.3.3 *Iterative DEMPC*

Instead of sequential computation of the distributed EMPC problems, parallel computation may be employed in the sense that each problem may be solved simultaneously. Given the control actions are computed without the knowledge of the control actions computed by the other distributed EMPC schemes, an iterative approach

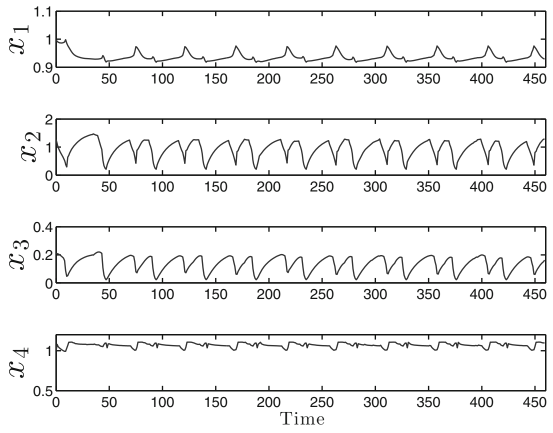


Fig. 7.12 Closed-loop state trajectories of the catalytic reactor under the S-DEMPC 2-1

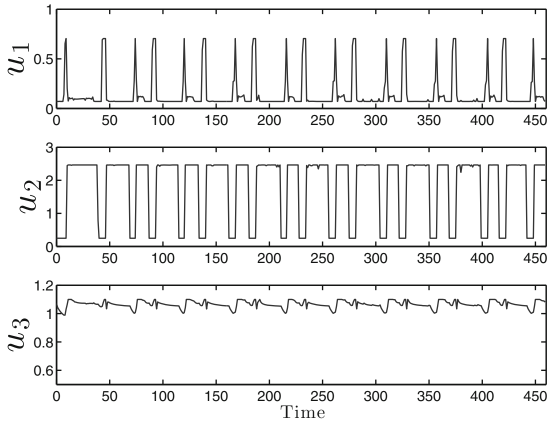


Fig. 7.13 Closed-loop input trajectories of computed by the S-DEMPC 2-1

may be used to (ideally) compute control actions closer to the centralized solution. Again, it is important to emphasize that given the nonlinearity and non-convexity of the optimization problems, it is difficult, in general, to guarantee that an iterative DEMPC strategy will converge to the centralized solution even after infinite iterations. Moreover, there is no guarantee that the input solution computed at each iteration improves upon the closed-loop performance over the previous iteration.

An iterative DEMPC (I-DEMPC) scheme is designed for the catalytic reactor and a block diagram of the I-DEMPC control architecture is given in Fig. 7.14. The computed input trajectories at each iteration of the I-DEMPC is denoted as $u_i^{*,c}(t|t_k)$ for $t \in [t_k, t_{k+N})$, $i = 1,\ 2,\ 3$ where c is the iteration number. At the first iteration, the input trajectory $\hat{u}_3(t)$ for $t \in [t_k, t_{k+N})$ in EMPC-1 is initialized with the sample-and-hold input trajectory computed from the same PI controller used in the S-DEMPC 1-2 scheme, and similarly, the input trajectories $\hat{u}_2(t)$ and $\hat{u}_3(t)$ for $t \in [t_k, t_{k+N})$ in EMPC-2 are initialized with the input trajectories computed from

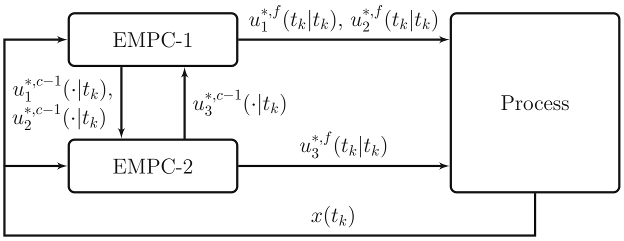

EMPC-1
EMPC-2
Process
$u_1^{*,f}(t_k|t_k)$, $u_2^{*,f}(t_k|t_k)$
$u_1^{*,c-1}(\cdot|t_k)$, $u_2^{*,c-1}(\cdot|t_k)$
$u_3^{*,c-1}(\cdot|t_k)$
$u_3^{*,f}(t_k|t_k)$
$x(t_k)$

Fig. 7.14 A block diagram of the I-DEMPC scheme

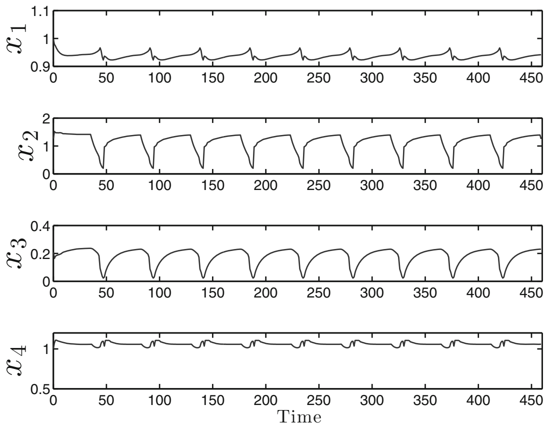


Fig. 7.15 Closed-loop state trajectories of the catalytic reactor under the I-DEMPC (1 iteration)

the PI controllers of the S-DEMPC 2-1 scheme. The control action applied to the reactor is denoted as $u_i^{*,f}(t_k|t_k)$ for $i = 1, 2, 3$ where f is the total number of iterations of the iterative DEMPC scheme (f is a design parameter of the scheme). When $f = 1$, the I-EMPC scheme is a decentralized DEMPC approach in the sense that there is no communication between EMPC-1 and EMPC-2 and each problem are solved independently of each other.

For this example, no closed-loop performance benefit was observed after iterating more than once through the I-DEMPC scheme. In fact, using the previous iterate solution to compute the next iterative gave worse closed-loop performance than applying the first computed iteration to the process. One method considered to compensate for this problem was to use the best computed input solution over all iterations to compute the next iteration. However, minimal closed-loop performance benefit was observed with this method. Thus, $f = 1$, which corresponds to a decentralized DEMPC approach, was selected for this case given that using more than one iteration did not improve the closed-loop performance. The resulting closed-loop trajectories are given in Figs. 7.15 and 7.16. The trajectories have similar characteristics as the centralized case.

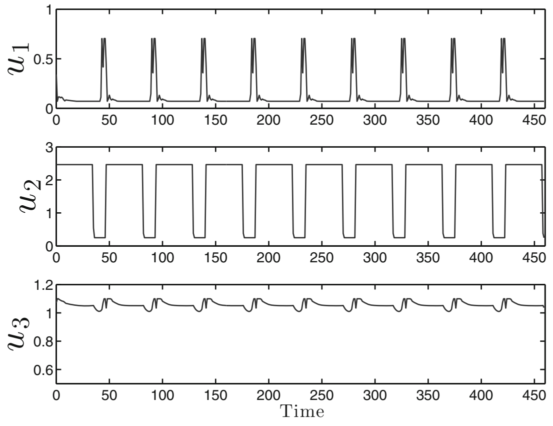


Fig. 7.16 Closed-loop input trajectories of computed by the I-DEMPC (1 iteration)

7.3.4 Evaluation of DEMPC Approaches

The average yield and average computation time required to solve the optimization problem at each sampling time over the entire simulation were considered for all the cases. The sequential DEMPC computation time is computed as the sum of the computation time of EMPC-1 and EMPC-2 at each sampling time because the sequential DEMPC schemes are computed sequentially. The iterative DEMPC computation time is the maximum computation time over all EMPCs at each sampling time (recall only one iteration was used). The average yield and average computation time for all the cases is given in Table 7.2. The closed-loop performance under the centralized EMPC, the sequential DEMPC 1-2, and the iterative DEMPC schemes was similar. The sequential DEMPC 1-2 and iterative DEMPC resulted in approximately a 70 reduction in computation time over the centralized EMPC. The sequential DEMPC 2-1 scheme not only had the worst performance of all the strategies considered (albeit still better than steady-state operation), but also, required a comparable amount of time to solve the optimization problems as the centralized case, thereby implying a strong dependence of closed-loop performance on controller calculation sequence. DEMPC was able to yield comparable closed-loop performance while substantially reducing the on-line computation time. This demonstrates that a distributed implementation may allow EMPC to be used on processes where centralized control is not feasible due to the solve time.

This example illustrates another key point within the context of DEMPC. Specifically, the inclusion of integral constraint in EMPC may be an important consideration for input selection in DEMPC. From the sequential DEMPC results, the computed u_3 profile is impacted by the assumed input profiles $\bar{u}_1$ and $\bar{u}_2$ (Fig. 7.13), while u_1 and u_2 are not affected as much by the assumed profile $\bar{u}_3$ (Fig. 7.10) compared to the centralized EMPC case (Fig. 7.7). This behavior may be due to the enforcement of the integral input constraint, and for this example, there may only be one method

Table 7.2 The average yield and computation time under the EMPC strategies

Strategy	Yield (%)	Comp. time (s)
Sequential DEMPC 1-2	10.20	1.039
Sequential DEMPC 2-1	9.92	2.969
Iterative DEMPC ($f = 1$)	10.05	0.832
Centralized EMPC	10.22	4.244

to distribute a fixed amount of ethylene to the reactor that maximizes the yield that is independent of u_3.

7.4 Real-Time Economic Model Predictive Control of Nonlinear Process Systems

Besides designing EMPC strategies that improves the computational efficiency such as the use of two-layer EMPC or distributed EMPC implementations, it may also be important to consider an EMPC implementation strategy that explicitly addresses potential computational delay. Some of the early work addressing computational delay within tracking MPC includes developing an implementation strategy of solving the MPC problem intermittently to account for the computational delay [23] and predicting the future state after an assumed constant computational delay to compute an input trajectory to be implemented after the optimization problem is solved [24, 25]. Nominal feasibility and stability has been proved for tracking MPC subject to computational delay formulated with a positive definite stage cost (with respect to the set-point or steady-state), a terminal cost, and a terminal region constraint [24, 25]. Another option to handle computational delay would be to force the optimization solver to terminate after a pre-specified time to ensure that the solver returns a solution by the time needed to ensure closed-loop stability. This concept is typically referred to as suboptimal MPC [26] because the returned solution will likely be suboptimal. It was shown that when the returned solution of the MPC with a terminal constraint is any feasible solution, the operating steady-state of the closed-loop system is asymptotically stable [26].

More recently, more advanced strategies have been proposed. Particularly, nonlinear programming (NLP) sensitivity analysis has demonstrated to be a useful tool to handle computational delay by splitting the MPC optimization problem into two parts: (1) solving a computationally intensive nonlinear optimization problem which is completed before feedback is received and (2) performing a fast on-line update of the precomputed input trajectories using NLP sensitivities (when the active-set does not change) after the current state measurement is obtained, e.g., [27, 28]; see, also, the review [29]. If the active-set changes, various methods have been proposed to cope with changing active-sets, e.g., solving a quadratic program like that proposed in [30]. In this direction, the advanced-step MPC [28] has been proposed which computes the solution of the optimization problem one sampling period in advance using a prediction of the state at the next sampling period. At the next sampling period (when the precomputed control action will be applied), the optimal solution is updated employing NLP sensitivities after state feedback is received. The advanced-step (tracking) MPC has been extended to handle computation spanning multiple sampling periods [31] and to EMPC [32]. Another related approach involves a hierarchical control structure [33, 34]. The upper layer is the full optimization problem which is solved infrequently. In the lower layer, NLP sensitivities are used to update the control actions at each sampling period that are applied to the system. The aforementioned schemes solve an optimization problem to (local) optimality using a prediction of the state at the sampling time the control action is to be applied to the system.

As another way, the so-called real-time nonlinear MPC (NMPC) scheme [35] only takes one Newton-step of the NLP solver instead of solving the optimization problem to optimality at each sampling period. To accomplish this, the structure of the resulting dynamic optimization program, which is solved using a direct multiple shooting method, is exploited to divide the program into a preparation phase and a feedback phase. In the preparation phase, the computationally expensive calculations are completed before feedback is received. In the feedback phase, a measurement is received and the remaining fast computations of the Newton-step are completed on-line to compute the control action to apply to the system. The advantage of such a strategy is that the on-line computation after a feedback measurement is obtained is insignificant compared to solving the optimization problem to optimality. The disadvantage is one would expect to sacrifice at least some closed-loop performance as a result of not solving the problem to optimality.

Clearly, the available computing power has significantly increased since the early work on computational delay of MPC and if this trend continues, one may expect a significant increase in computing power over the next decade. Moreover, more efficient solution strategies for nonlinear dynamic optimization problems continue to be developed (see, for example, the overview paper [36] and the book [37] for results in this direction). However, the ability to guarantee that a solver will converge within the time needed for closed-loop stability remains an open problem especially for nonlinear, non-convex dynamic optimization problems and systems with fast dynamics. Additionally, EMPC is generally more computationally intensive compared to tracking MPC given the additional possible nonlinearities in the stage cost of EMPC.

In this section, a real-time implementation strategy for LEMPC, referred to as real-time LEMPC, is developed to account for possibly unknown and time-varying computational delay. The underlying implementation strategy is inspired by event-triggered control concepts [38] since the LEMPC is only recomputed when stability conditions dictate that it must recompute a new input trajectory. If the precomputed control action satisfies the stability conditions, the control action is applied to the closed-loop system. If not, a back-up explicit controller, which has negligible computation time, is used to compute the control action for the system at the current sampling instance. This type of implementation strategy has the advantage of being easy to implement and the strategy avoids potential complications of active-set changes because the re-computation condition is only formulated to account for closed-loop stability considerations. Closed-loop stability under the real-time LEMPC scheme is analyzed and specific stability conditions are derived. The real-time LEMPC scheme is applied to an illustrative chemical process network to demonstrate closed-loop stability under the control scheme. The example also demonstrates that real-time LEMPC improves closed-loop economic performance compared to operation at the economically optimal steady-state.

7.4.1 Class of Systems

The class of nonlinear systems considered has the following state-space form:

$$\dot{x}(t) = f(x(t), u(t), w(t)) \tag{7.62}$$

where $x(t) \in \mathbb{R}^n$ is the state vector, $u(t) \in \mathbb{U} \subset \mathbb{R}^m$ is the manipulated input vector, $\mathbb{U}$ is a compact set, $w(t) \in \mathbb{W} \subset \mathbb{R}^l$ is the disturbance vector, and f is a locally Lipschitz vector function. The disturbance vector is bounded in the following set:

$$\mathbb{W} := \{w \in \mathbb{R}^l : |w| \leq \theta\} \tag{7.63}$$

where $\theta > 0$ bounds the norm of the disturbance vector. Without loss of generality, the origin of the unforced system is assumed to be the equilibrium point of Eq. 7.62, i.e., $f(0, 0, 0) = 0$.

The following stabilizability assumption further qualifies the class of systems considered and is similar to the assumption that the pair (A, B) is stabilizable in linear systems.

Assumption 7.3 There exists a feedback controller $h(x) \in \mathbb{U}$ with $h(0) = 0$ that renders the origin of the closed-loop system of Eq. 7.62 with $u(t) = h(x(t))$ and $w \equiv 0$ asymptotically stable for all $x \in D_0$ where D_0 is an open neighborhood of the origin.

Applying converse theorems [7, 9], Assumption 7.3 implies that there exists a continuously differentiable Lyapunov function, $V : D \to \mathbb{R}_+$, for the closed-loop system of Eq. 7.62 with $u = h(x) \in \mathbb{U}$ and $w \equiv 0$ such that the following inequalities hold:

$$\alpha_1(|x|) \leq V(x) \leq \alpha_2(|x|), \tag{7.64a}$$

$$\frac{\partial V(x)}{\partial x} f(x, h(x), 0) \leq -\alpha_3(|x|), \tag{7.64b}$$

$$\left| \frac{\partial V(x)}{\partial x} \right| \leq \alpha_4(|x|) \tag{7.64c}$$

for all $x \in D$ where D is an open neighborhood of the origin and $\alpha_i, i = 1, 2, 3, 4$ are functions of class $\mathscr{K}$. A level set of the Lyapunov function Ω_ρ, which defines a subset of D (ideally the largest subset contained in D), is taken to be the stability region of the closed-loop system under the controller $h(x)$.

Measurements of the state vector of Eq. 7.62 are assumed to be available synchronously at sampling instances denoted as $t_k := k\Delta$ where $\Delta > 0$ is the sampling period and $k = 0, 1, \ldots$. As described below, the EMPC computes sample-and-hold control actions and thus, the resulting closed-loop system, which consists of the continuous-time system of Eq. 7.62 under a sample-and-hold controller, is a sampled-data system. If the controller $h(x)$ is implemented in a sample-and-hold fashion, it possesses a certain degree of robustness to uncertainty in the sense that the origin of the closed-loop system is rendered practically stable when a sufficiently small sampling period is used and the bound θ on the disturbance vector is sufficiently small; see, for example, [10] for more discussion on this point.

7.4.2 Real-Time LEMPC Formulation

The overall objective of the real-time LEMPC is to account for the real-time computation time required to solve the optimization problem for a (local) solution. Particularly, the case when the average computation time, which is denoted as $\bar{t}_s$, is greater than one sampling period is considered, i.e., $N_s = \lceil \bar{t}_s/\Delta \rceil \geq 1$ where N_s is the average number of sampling periods required to solve the optimization problem. During the time the solver is solving the optimization problem, the control actions computed at a previous sampling period are applied to the system if there are precomputed control actions available and if the stability conditions described below are satisfied. If no precomputed control actions are available or the stability conditions are violated, the explicit controller $h(x)$ is used to compute and apply control actions during the time that the real-time LEMPC is computing. In this fashion, the LEMPC is used to compute control actions to improve the economic performance when possible.

Specifically, when the closed-loop state is in the subset of the stability region $\Omega_{\rho_e} \subset \Omega_\rho$, the control actions of the precomputed LEMPC problem may be applied to the system. When the state is outside the subset, the explicit controller is used because maintaining the closed-loop state in Ω_ρ is required for guaranteeing the existence of a feasible input trajectory that maintains closed-loop stability (in the sense that the closed-loop state trajectory is always bounded in Ω_ρ). To force the state back to the subset of the stability region Ω_{ρ_e}, the Lyapunov function must decrease over each sampling period in the presence of uncertainty. This requires the incorporation of feedback, i.e., recomputing the control action at each sampling period using a measurement of the current state. Owing to the computational burden of solving the LEMPC optimization problem, it may not be possible to achieve convergence of the optimization solver within one sampling period. Hence, the controller $h(x)$ is used when the state is outside of Ω_{ρ_e}.

For real-time implementation, only mode 1 of the LEMPC of Eq. 4.3 is used and the LEMPC is solved infrequently (not every sampling period) which will be made clear when the implementation strategy is discussed. The real-time LEMPC is formulated as follows:

$$\min_{u \in S(\Delta)} \int_{t_{j+1}}^{t_{j+N}} l_e(\tilde{x}(t), u(t))\, dt \tag{7.65a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = f(\tilde{x}(t), u(t), 0) \tag{7.65b}$$

$$\tilde{x}(t_j) = x(t_j) \tag{7.65c}$$

$$u(t) = \tilde{u}(t_j),\ \forall\, t \in [t_j, t_{j+1}) \tag{7.65d}$$

$$u(t) \in \mathbb{U},\ \forall\, t \in [t_{j+1}, t_{j+N}) \tag{7.65e}$$

$$V(\tilde{x}(t)) \leq \rho_e,\ \forall\, t \in [t_{j+1}, t_{j+N}) \tag{7.65f}$$

where the notation and constraints are similar to that used in LEMPC of Eq. 4.3 except for an additional constraint of Eq. 7.65d. This additional constraint is used because a predetermined control action is applied to the system over the first sampling period of the prediction horizon. The predetermined control action is either the control action computed by the LEMPC at a previous sampling period or the control action from the explicit controller $h(x)$, i.e., the input trajectory over the first sampling period of the prediction horizon is not a degree of freedom in the optimization problem. The LEMPC of Eq. 7.65 may dictate a time-varying operating policy to optimize the economic cost as long as the predicted evolution is maintained in the level set $\Omega_{\rho_e} \subset \Omega_\rho$. The notation t_j denotes the sampling time at which the LEMPC problem is initialized with a state measurement and the solver begins solving the resulting optimization problem. The optimal solution of the LEMPC is denoted as $u^*(t|t_j)$ and is defined for $t \in [t_{j+1}, t_{j+N})$. Feasibility of the optimization problem is considered in Sect. 7.4.4. However, it is important to point out that $x(t_j) \in \Omega_{\rho_e}$ and $\tilde{x}(t_{j+1}) \in \Omega_{\rho_e}$ owing to the real-time implementation strategy, and thus, the real-time LEMPC has a feasible solution (refer to the proof of Theorem 7.2).

7.4.3 Implementation Strategy

Before the implementation strategy is presented, the following discrete-time signals are defined to simplify the presentation of the implementation strategy. The first signal is used to keep track of whether the solver is currently solving an LEMPC optimization problem:

$$s_1(k) = \begin{cases} 1, & \text{solving the LEMPC} \\ 0, & \text{not solving the LEMPC} \end{cases} \tag{7.66}$$

where k denotes the k-th sampling period, i.e., t_k. The second signal keeps track if there is a previously computed input trajectory currently stored in memory:

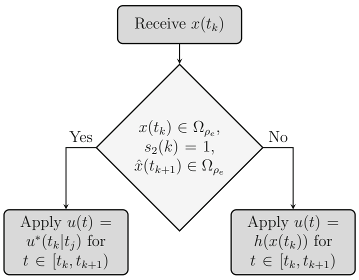

Receive $x(t_k)$
$x(t_k) \in \Omega_{\rho_e}$, $s_2(k) = 1$, $\hat{x}(t_{k+1}) \in \Omega_{\rho_e}$
Yes
No
Apply $u(t) = u^*(t_k|t_j)$ for $t \in [t_k, t_{k+1})$
Apply $u(t) = h(x(t_k))$ for $t \in [t_k, t_{k+1})$

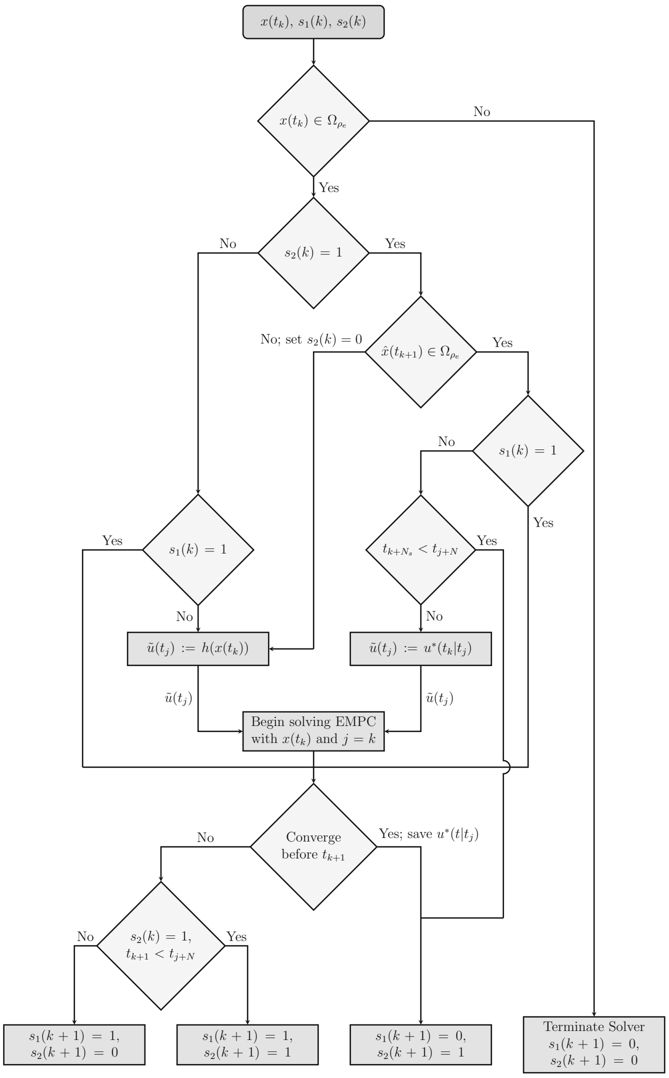
Fig. 7.17 Implementation strategy for determining the control action at each sampling period. The notation $u^*(t_k|t_j)$ is used to denote the control action to be applied over the sampling period t_k to t_{k+1} from the precomputed input solution of the real-time LEMPC of Eq. 7.65 solved at time step t_j

$$s_2(k) = \begin{cases} 1, & \text{previous input solution stored} \\ 0, & \text{no previous input solution stored} \end{cases} \tag{7.67}$$

At each sampling period, a state measurement $x(t_k)$ is received from the sensors and three conditions are used to determine if a precomputed control action from LEMPC or if the control action from the explicit controller $h(x)$ is applied to the system. If the following three conditions are satisfied the control action applied to the system in a sample-and-hold fashion is the precomputed control action from the LEMPC: (1) the current state must be in Ω_{ρ_e} ($x(t_k) \in \Omega_{\rho_e}$), (2) there must be a precomputed control action available for the sampling instance t_k, i.e., $s_2(k) = 1$, and (3) the predicted state under the precomputed control action must satisfy: $\hat{x}(t_{k+1}) \in \Omega_{\rho_e}$ where $\hat{x}(t_{k+1})$ denotes the predicted state. To obtain a prediction of the state at the next sampling period, the nominal model of Eq. 7.62 with $w \equiv 0$ is recursively solved with the input $u(t) = u^*(t_k|t_j)$ for $t \in [t_k, t_{k+1})$ (the on-line computation time to accomplish this step is assumed to be negligible). The control action decision at a given sampling instance t_k is summarized by the flow chart of Fig. 7.17.

A series of decisions are made at each sampling period to determine if the LEMPC should begin resolving, continue solving, or terminate solving the optimization problem and is illustrated in the flow chart of Fig. 7.18. The computation strategy is summarized in the following algorithm. To initialize the algorithm at $t_0 = 0$, get the state measurement $x(0) \in \Omega_\rho$. If $x(0) \in \Omega_{\rho_e}$, begin solving the LEMPC problem with $k = j = 0$ and $x(0)$. Set $s_1(0) = 1$, $s_2(0) = 0$, and $\tilde{u}(t_j) = h(x(0))$. Go to Step 8. Else, set $s_1(0) = s_1(1) = s_2(0) = s_2(1) = 0$ and go to Step 9.

1. Receive a measurement of the current state $x(t_k)$ from the sensors; go to Step 2.
2. If $x(t_k) \in \Omega_{\rho_e}$, then go to Step 2.1. Else, go to Step 2.2.
   - 2.1 If $s_2(k) = 1$, go to Step 3. Else, go to Step 6.
   - 2.2 Terminate solver if $s_1(k) = 1$, set $s_1(k+1) = 0$ and $s_2(k+1) = 0$, and go to Step 9.



$x(t_k)$, $s_1(k)$, $s_2(k)$
$x(t_k) \in \Omega_{\rho_e}$
No
Yes
$s_2(k) = 1$
No
Yes
$\hat{x}(t_{k+1}) \in \Omega_{\rho_e}$
No; set $s_2(k) = 0$
Yes
$s_1(k) = 1$
No
Yes
$s_1(k) = 1$
Yes
No
$t_{k+N_s} < t_{j+N}$
Yes
No
$\tilde{u}(t_j) := h(x(t_k))$
$\tilde{u}(t_j) := u^*(t_k|t_j)$
$\tilde{u}(t_j)$
$\tilde{u}(t_j)$
Begin solving EMPC with $x(t_k)$ and $j = k$
Converge before t_{k+1}
No
Yes; save $u^*(t|t_j)$
$s_2(k) = 1$, $t_{k+1} < t_{j+N}$
No
Yes
$s_1(k+1) = 1$, $s_2(k+1) = 0$
$s_1(k+1) = 1$, $s_2(k+1) = 1$
$s_1(k+1) = 0$, $s_2(k+1) = 1$
Terminate Solver $s_1(k+1) = 0$, $s_2(k+1) = 0$

Fig. 7.18 Computation strategy for the real-time LEMPC scheme

3. If $\hat{x}(t_{k+1}) \in \Omega_{\rho_e}$, go to Step 4. Else, set $s_2(k) = 0$ and $\tilde{u}(t_j) = h(x(t_k))$; go to Step 7.
4. If $s_1(k) = 1$, go to Step 8. Else, go to Step 5.
5. If $t_{k+N_s} < t_{j+N}$, set $s_1(k+1) = 0$ and $s_2(k+1) = 1$, and go to Step 9. Else, set $\tilde{u}(t_j) = u^*(t_k|t_j)$; go to Step 7.
6. If $s_1(k) = 1$, go to Step 8. Else, set $\tilde{u}(t_j) = h(x(t_k))$; go to Step 7.

7. If the solver is currently solving a problem ($s_1(k) = 1$), terminate the solver. Begin solving the LEMPC problem with $j = k$ and $x(t_j) = x(t_k)$. Go to Step 8.
8. If the solver converges before t_{k+1}, then go to Step 8.1. Else, go to Step 8.2.
   - 8.1 Save $u^*(t|t_j)$ for $t \in [t_k, t_{j+N})$. Set $s_1(k+1) = 0$ and $s_2(k+1) = 1$. Go to Step 9.
   - 8.2 Set $s_1(k+1) = 1$. If $s_2(k) = 1$ and $t_{k+1} < t_{j+N}$, the go to Step 8.2.1. Else, go to Step 8.2.2.
     - 8.2.1 Set $s_2(k+1) = 1$. Go to Step 9.
     - 8.2.2 Set $s_2(k+1) = 0$. Go to Step 9.
9. Go to Step 1 ($k \leftarrow k + 1$).

In practice, N_s may be unknown or possibly time varying. If N_s is unknown, then one may specify the number of sampling periods that the real-time LEMPC may apply a precomputed input trajectory before it must start re-computing a new input trajectory as a design parameter. This condition may be used instead of Step 5 of the algorithm above. Additionally, it may be beneficial from a closed-loop performance perspective to force the LEMPC to recompute its solution more often than prescribed by the implementation strategy described above.

A possible input trajectory resulting under the real-time LEMPC scheme is given in Fig. 7.19. In the illustration, the solver begins to solve an LEMPC optimization problem at t_0 and returns a solution at t_5. It is assumed that the closed-loop state is maintained in Ω_{ρ_e} from t_0 to t_5 so that the solver is not terminated. Over the time the solver is solving, the explicit controller is applied to the system since a precomputed LEMPC input trajectory is not available. The precomputed LEMPC solution is applied from t_5 to t_{13}. At t_{10}, the solver begins to solve a new LEMPC problem. The solver returns a solution at t_{13}. At t_{16}, the stability conditions are

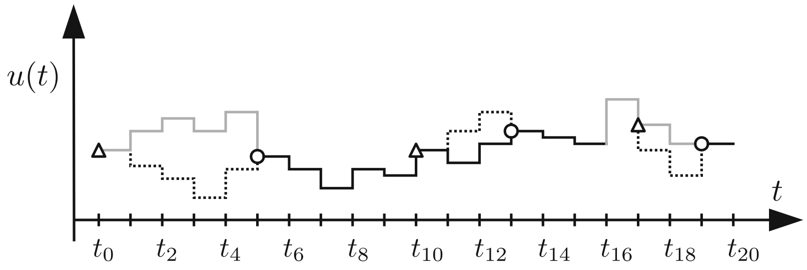


Fig. 7.19 An illustration of an example input trajectory resulting under the real-time LEMPC scheme. The *triangles* are used to denote the time instances when the LEMPC begins to solve the optimization problem, while the *circles* are used to denote when the solver converges to a solution. The *solid black* trajectory represents the control actions computed by the LEMPC which are applied to the system, the *dotted* trajectory represents the computed input trajectory by the LEMPC (not applied to the system), and the *solid gray* trajectory is the input trajectory of the explicit controller which is applied to the system

not satisfied for the precomputed LEMPC input trajectory, so the explicit controller computes a control action and applies it to the system.

7.4.4 Stability Analysis

In this section, sufficient conditions such that the closed-loop state under the real-time LEMPC is bounded in Ω_ρ are presented which make use of the following properties. Since f is a locally Lipschitz vector function of its arguments and the Lyapunov function V is a continuously differentiable function, there exist positive constants L_x, L_w, L'_x, and L'_w such that the following bounds hold:

$$|f(x_a, u, w) - f(x_b, u, 0)| \leq L_x |x_a - x_b| + L_w |w| \tag{7.68}$$

$$\left| \frac{\partial V(x_a)}{\partial x} f(x_a, u, w) - \frac{\partial V(x_b)}{\partial x} f(x_b, u, 0) \right| \leq L'_x |x_a - x_b| + L'_w |w| \tag{7.69}$$

for all $x_a, x_b \in \Omega_\rho$, $u \in \mathbb{U}$ and $w \in \mathbb{W}$. Furthermore, there exists $M > 0$ such that

$$|f(x, u, w)| \leq M \tag{7.70}$$

for all $x \in \Omega_\rho$, $u \in \mathbb{U}$ and $w \in \mathbb{W}$ owing to the compactness of the sets Ω_ρ, $\mathbb{U}$, and $\mathbb{W}$ and the locally Lipschitz property of the vector field.

The following proposition bounds the difference between the actual state trajectory of the system of Eq. 7.62 ($w \not\equiv 0$) and the nominal state trajectory ($w \equiv 0$).

Proposition 7.3 (Proposition 4.1) *Consider the state trajectories $x(t)$ and $\hat{x}(t)$ with dynamics:*

$$\dot{x}(t) = f(x(t), u(t), w(t)), \tag{7.71}$$

$$\dot{\hat{x}}(t) = f(\hat{x}(t), u(t), 0), \tag{7.72}$$

input trajectory $u(t) \in \mathbb{U}$, $w(t) \in \mathbb{W}$, and initial condition $x(0) = \hat{x}(0) \in \Omega_\rho$. If $x(t)$, $\hat{x}(t) \in \Omega_\rho$ for all $t \in [0, T]$ where $T \geq 0$, then the difference between $x(T)$ and $\hat{x}(T)$ is bounded by the function $\gamma_e(\cdot)$:

$$\left|x(T) - \hat{x}(T)\right| \leq \gamma_e(T) := \frac{L_w \theta}{L_x} \left(e^{L_x T} - 1\right). \tag{7.73}$$

Owing to the compactness of the set Ω_ρ, the difference in Lyapunov function values for any two points in Ω_ρ may be bounded by a quadratic function which is stated in the following proposition.

Proposition 7.4 (Proposition 4.2) *Consider the Lyapunov function V of the closed-loop system of Eq. 7.62 under the controller $h(x)$. There exists a scalar-valued quadratic function $f_V(\cdot)$ such that*

$$V(x_a) \le V(x_b) + f_V(|x_a - x_b|) \tag{7.74}$$

for all x_a, $x_b \in \Omega_\rho$ where

$$f_V(s) := \alpha_4(\alpha_1^{-1}(\rho))s + \beta s^2 \tag{7.75}$$

and β is a positive constant.

Theorem 7.2 provides sufficient conditions such that the real-time LEMPC renders the closed-loop state trajectory bounded in Ω_ρ for all times. The conditions such that the closed-loop state trajectory is maintained in Ω_ρ are independent of the computation time required to solve the LEMPC optimization problem. From the perspective of closed-loop stability, computational delay of arbitrary size may be handled with the real-time LEMPC methodology. In the case where the computational delay is always greater than the prediction horizon, the real-time LEMPC scheme would return the input trajectory under the explicit controller applied in a sample-and-hold fashion.

Theorem 7.2 *Consider the system of Eq. 7.62 in closed-loop under the real-time LEMPC of Eq. 7.65 based on a controller $h(x)$ that satisfies the conditions of Eq. 7.64 that is implemented according to the implementation strategy of Fig. 7.17. Let $\varepsilon_w > 0$, $\Delta > 0$ and $\rho > \rho_e \ge \rho_{\min} > \rho_s > 0$ satisfy*

$$-\alpha_3(\alpha_2^{-1}(\rho_s)) + L'_x M\Delta + L'_w\theta \le -\varepsilon_w/\Delta, \tag{7.76}$$

$$\rho_{\min} = \max\{V(x(t+\Delta)) \mid V(x(t)) \le \rho_s\}, \tag{7.77}$$

and

$$\rho_e < \rho - f_V(\gamma_e(\Delta)). \tag{7.78}$$

If $x(t_0) \in \Omega_\rho$ and $N \ge 1$, then the state trajectory $x(t)$ of the closed-loop system is always bounded in Ω_ρ for $t \ge t_0$.

Proof If the real-time LEMPC is implemented according to the implementation strategy of Fig. 7.17, the control action to be applied over the sampling period has been (pre)computed by the LEMPC or the explicit controller $h(x)$. To prove that the closed-loop state is bounded in Ω_ρ, we will show that when the control action is computed from the explicit controller and $x(t_k) \in \Omega_\rho$, then the state at the next sampling period will be contained in Ω_ρ. If the control action comes from a precomputed LEMPC solution, we will show that if $x(t_k) \in \Omega_{\rho_e}$, then $x(t_{k+1}) \in \Omega_\rho$ owing to the stability conditions imposed on applying the precomputed LEMPC solution. The proof consists of two parts. In the first part, the closed-loop properties when the control action is computed by the explicit controller $h(x)$ are analyzed. This part of the proof is based on the proof of [10] which considers the stability properties of an explicit controller of the form assumed for $h(x)$ implemented in a sample-and-hold fashion. In the second part, the closed-loop stability properties of the precomputed control actions by the LEMPC are considered. In both cases, the closed-loop state trajectory is shown to be maintained in Ω_ρ for $t \geq t_0$ when $x(t_0) \in \Omega_\rho$.

Part 1: First, consider the properties of the control action computed by the explicit controller $h(x)$ applied to the system of Eq. 7.62 in a sample-and-hold fashion. Let $x(t_k) \in \Omega_\rho \setminus \Omega_{\rho_s}$ for some $\rho_s > 0$ such that the conditions of Theorem 7.2 are satisfied, i.e., Eq. 7.76. The explicit controller $h(x)$ computes a control action that has the following property (from condition of Eq. 7.64):

$$\frac{\partial V(x(t_k))}{\partial x} f(x(t_k), h(x(t_k)), 0) \leq -\alpha_3(|x(t_k)|) \leq -\alpha_3(\alpha_2^{-1}(\rho_s)) \tag{7.79}$$

for any $x(t_k) \in \Omega_\rho \setminus \Omega_{\rho_s}$. Over the sampling period, the time-derivative of the Lyapunov function is:

$$\begin{aligned}\dot{V}(x(t)) = {} & \frac{\partial V(x(t_k))}{\partial x} f(x(t_k), h(x(t_k)), 0) + \frac{\partial V(x(t))}{\partial x} f(x(t), h(x(t_k)), w(t)) \\ & - \frac{\partial V(x(t_k))}{\partial x} f(x(t_k), h(x(t_k)), 0)\end{aligned} \tag{7.80}$$

for all $t \in [t_k, t_{k+1})$. From the bound on the time-derivative of Lyapunov function of Eq. 7.79, the Lipschitz bound of Eq. 7.69, and the bound on the norm of the disturbance vector, the time-derivative of the Lyapunov function is bounded for $t \in [t_k, t_{k+1})$ as follows:

$$\begin{aligned}\dot{V}(x(t)) \leq {} & -\alpha_3(\alpha_2^{-1}(\rho_s)) \\ & + \left| \frac{\partial V(x(t))}{\partial x} f(x(t), h(x(t_k)), w(t)) - \frac{\partial V(x(t_k))}{\partial x} f(x(t_k), h(x(t_k)), 0) \right| \\ \leq {} & -\alpha_3(\alpha_2^{-1}(\rho_s)) + L'_x \, |x(t) - x(t_k)| + L'_w \, |w(t)| \\ \leq {} & -\alpha_3(\alpha_2^{-1}(\rho_s)) + L'_x \, |x(t) - x(t_k)| + L'_w \theta\end{aligned} \tag{7.81}$$

for all $t \in [t_k, t_{k+1})$. Taking into account of Eq. 7.70 and the continuity of $x(t)$, the following bound may be written for all $t \in [t_k, t_{k+1})$:

$$|x(t) - x(t_k)| \leq M\Delta. \tag{7.82}$$

From Eqs. 7.81 and 7.82, the bound below follows:

$$\dot{V}(x(t)) \leq -\alpha_3(\alpha_2^{-1}(\rho_s)) + L'_x M\Delta + L'_w \theta \tag{7.83}$$

for all $t \in [t_k, t_{k+1})$. If the condition of Eq. 7.76 is satisfied, i.e., Δ and θ is sufficiently small, then there exists $\varepsilon_w > 0$ such that:

$$\dot{V}(x(t)) \leq -\varepsilon_w/\Delta \tag{7.84}$$

for all $t \in [t_k, t_{k+1})$. Integrating the above bound, yields:

$$V(x(t)) \leq V(x(t_k)), \quad \forall\, t \in [t_k, t_{k+1}), \tag{7.85}$$

$$V(x(t_{k+1})) \leq V(x(t_k)) - \varepsilon_w. \tag{7.86}$$

For any state $x(t_k) \in \Omega_\rho \backslash \Omega_{\rho_s}$, the state at the next sampling period will be in a smaller level set when the control action $u(t) = h(x(t_k))$ is applied for $t \in [t_k, t_{k+1})$. Also, the state will not come out of Ω_ρ over the sampling period owing to Eq. 7.84. Once the closed-loop state under the explicit controller $h(x)$ implemented in a sample-and-hold fashion has converged to Ω_{ρ_s}, the closed-loop state trajectory will be maintained in $\Omega_{\rho_{\min}}$ if $\rho_{\min} \leq \rho$ and $\rho_{\min}$ is defined according to Eq. 7.77. Thus, the sets Ω_ρ and $\Omega_{\rho_{\min}}$ are forward invariant sets under the controller $h(x)$ and if $x(t_k) \in \Omega_\rho$, then $x(t_{k+1}) \in \Omega_\rho$ under the explicit controller $h(x)$.

Part 2: In this part, the closed-loop stability properties of the input precomputed by the LEMPC for the sampling period t_k to t_{k+1} are considered. For clarity of presentation, the notation $\hat{x}(t)$ denotes the prediction of closed-loop state at time t, i.e., this prediction used in the implementation strategy to determine which control action to apply to the system, while the notation $\tilde{x}(t)$ will be reserved to denote the predicted state in the LEMPC of Eq. 7.65. The predicted state in the LEMPC of Eq. 7.65 at t_{j+1}, which is denoted as $\tilde{x}(t_{j+1})$, satisfies $\hat{x}(t_{j+1}) = \tilde{x}(t_{j+1})$ because both predicted states use the nominal model with the same initial condition and same piecewise constant input applied from t_j to t_{j+1}.

First, feasibility of the optimization problem is considered. Owing to the formulation of the LEMPC of Eq. 7.65, the optimization problem is always feasible if ρ_e satisfies: $\rho > \rho_e \geq \rho_{\min}$. Recall, the input over the sampling period t_j to t_{j+1} is not a degree of freedom in the optimization problem. If this control action is precomputed from a previous LEMPC solution, it must have the property that $\hat{x}(t_{j+1}) = \tilde{x}(t_{j+1}) \in \Omega_{\rho_e}$ which is imposed as a condition of the implementation strategy of Fig. 7.17. If the control action is computed by the explicit controller, the control action over the sampling period t_j to t_{j+1} will maintain $\tilde{x}(t_{j+1}) \in \Omega_{\rho_e}$. Thus, $\tilde{x}(t_{j+1}) \in \Omega_{\rho_e}$ in the LEMPC of Eq. 7.65. Feasibility of the optimization problem follows from the fact that the input trajectory obtained from the explicit controller $h(x)$ over the prediction horizon is a feasible solution, that is $u(t) = h(\hat{x}(t_i))$ for $t \in [t_i, t_{i+1}), i = j+1,\ j+2,\ \ldots,\ j+N-1$ where $\hat{x}(t)$ is obtained by recursively solving the model:

$$\dot{\hat{x}}(t) = f(\hat{x}(t), h(\hat{x}(t_i)), 0) \tag{7.87}$$

for $t \in [t_i, t_{i+1})$ and $i = j+1,\ j+1\ \ldots,\ j+N-1$ with the initial condition $\hat{x}(t_{j+1}) = \tilde{x}(t_{j+1})$. Furthermore, the set Ω_{ρ_e} is forward invariant under the controller

$h(x)$ (the proof is analogous to Part 1 where the set Ω_{ρ_e} is used instead of Ω_ρ). Thus, the LEMPC of Eq. 7.65 is always feasible for any $x(t_j) \in \Omega_{\rho_e}$.

If the LEMPC is implemented according to the implementation strategy of Fig. 7.17, then the precomputed input for t_k by the LEMPC is only used when $x(t_k) \in \Omega_{\rho_e}$ and the predicted state at the next sampling period $\hat{x}(t_{k+1}) \in \Omega_{\rho_e}$. When $x(t) \in \Omega_\rho$ for $t \in [t_k, t_{k+1})$, i.e., a sufficiently small sampling period is used, the following bound on the Lyapunov function value at the next sampling period t_{k+1} may be derived from Propositions 7.3–7.4:

$$V(x(t_{k+1})) \leq V(\hat{x}(t_{k+1})) + f_V(\gamma_e(\Delta)). \tag{7.88}$$

Since $\hat{x}(t_{k+1}) \in \Omega_{\rho_e}$ and if the condition of Eq. 7.78 is satisfied, $x(t_{k+1}) \in \Omega_\rho$.

To summarize, if the control action to be applied over the sampling period t_k to t_{k+1} is $u(t_k) = h(x(t_k))$, the state at the next sampling period will be in Ω_ρ ($x(t_{k+1}) \in \Omega_\rho$). If the control action to be applied over the sampling period t_k to t_{k+1} is from a precomputed LEMPC input, the state at the next sampling period will also be contained in Ω_ρ which completes the proof of boundedness of the closed-loop state trajectory $x(t) \in \Omega_\rho$ under the real-time LEMPC for $t \geq t_0$.

Remark 7.6 No closed-loop performance guarantees may be made because performance constraints, e.g., terminal constraints, are not imposed on the LEMPC and the closed-loop performance may be adversely affected with greater computation time. The latter point is associated with the fact that the LEMPC problem allows for the input trajectory from t_{j+1} to t_{j+N_s}, i.e., the time the solver converges, to be degrees of freedom in the optimization problem. However, the actual closed-loop input trajectory applied over this period may be different from that computed by the LEMPC over the same time period. Potentially, one may also employ sensitivity-based corrections to the precomputed control actions after receiving state feedback like that employed in [27, 28] to improve closed-loop performance. However, active set changes must be handled appropriately which may introduce additional on-line computation. It is important to point out that the computed solution of the LEMPC may dictate a time-varying operating policy to optimize the process economics. Even in the presence of uncertainty, the closed-loop performance under the real-time LEMPC may be substantially better (with respect to the economic cost) than traditional control methods, which is the case for the chemical process network considered in Sect. 7.4.5.

Remark 7.7 In the current section, unknown and possibly time-varying computational delay is considered for operation affected by unknown bounded disturbance. If, instead of the computation algorithm described above, a hard cap was placed on the solver to terminate and return a (suboptimal) solution by a certain number of sampling times, one could account for the control actions that are applied to the system over the computation time by setting the input trajectory in the LEMPC problem over the specified number of sampling periods of the prediction horizon be equal to a predetermined input trajectory. This potential strategy, however, does not account for the fact that the solver may return a solution before the end of specified number of sampling periods.

Remark 7.8 From the proof of Theorem 7.2, recursive feasibility of the LEMPC in the presence of bounded uncertainty is guaranteed if the initial state is in Ω_ρ. It is difficult in general to characterize the feasible set under EMPC formulated with a terminal constraint, i.e., the set of points where recursive feasibility is maintained in the presence of uncertainty. Thus, it may be difficult to ensure that the closed-loop state is maintained in the feasible set under EMPC with a terminal constraint in the presence of uncertainty and computational delay. In this respect, LEMPC has a unique advantage for real-time implementation compared to EMPC with a terminal constraint in that LEMPC maintains the closed-loop state inside Ω_ρ where recursive feasibility is guaranteed.

Remark 7.9 The number of times that the explicit controller is applied to the closed-loop system may be a factor in the closed-loop economic performance. Whether the control action is from a precomputed LEMPC problem or the explicit controller is mainly influenced by how close the state measurement is to the boundary of Ω_{ρ_e}. To decrease the number of times that the explicit controller is applied to the system, one could potentially add penalization terms to the stage cost of the LEMPC to penalize the closeness of the state to the boundary of Ω_{ρ_e}.

7.4.5 Application to a Chemical Process Network

Consider a chemical process network consisting of two continuous stirred-tank reactors (CSTRs) in series followed by a flash separator shown in Fig. 7.20. In each of the reactors, the reactant A is converted to the desired product B through an exothermic and irreversible reaction of the form $A \rightarrow B$. A fresh feedstock containing a dilute solution of the reactant A in an inert solvent D is fed to each reactor. The reaction rate is second-order in the reactant concentration. The CSTRs are denoted as CSTR-1 and CSTR-2, respectively. A flash separator, which is denoted as SEP-1, is used to recover some unreacted A. The overhead vapor from the flash tank is condensed and recycled back to CSTR-1. The bottom stream is the product stream of the process network which contains the desired product B. In the separator, a negligible amount of A is assumed to be converted to B through the reaction. The two reactors have both heating and cooling capabilities and the rate of heat supplied to or removed from the reactors is denoted as Q_j, $j = 1,\ 2$. While the heat supplied to or removed from the vessel contents is modeled with one variable, two different actuators may be used in practice for supplying heat to and removing heat from each vessel. To vaporize some contents of the separator, heat is supplied to the separator at a rate of Q_3. The liquid holdup of each vessel is assumed to be constant and the liquid density throughout the process network is also assumed to be constant.

Applying first principles, a dynamic model of the process network may be obtained (neglecting the dynamics of the condenser and the solvent) and is given by the following ordinary differential equations (ODEs) (see Table 7.3 for parameter notation and values):

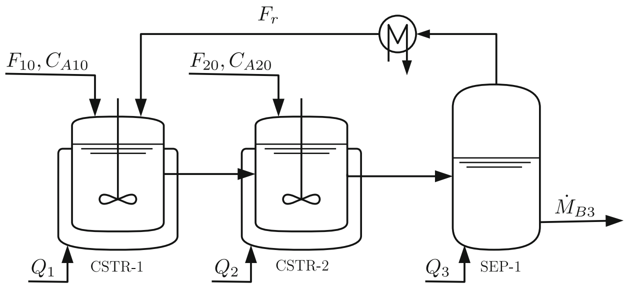


Fig. 7.20 Process flow diagram of the reactor and separator process network

Table 7.3 Process parameters of the reactor and separator process network

Symbol/Value	Description	Symbol/Value	Description
$T_{10} = 300\,\mathrm{K}$	Temp.: CSTR-1 inlet	$k_0 = 1.9 \times 10^9\,\mathrm{m}^3\,\mathrm{kmol}^{-1}\,\mathrm{h}^{-1}$	Pre-exponential factor
$T_{20} = 300\,\mathrm{K}$	Temp.: CSTR-2 inlet	$E = 7.1 \times 10^4\,\mathrm{kJ\,kmol}^{-1}$	Activation energy
$F_{10} = 5.0\,\mathrm{m}^3\,\mathrm{h}^{-1}$	Flow: CSTR-1 inlet	$\Delta H = -7.8 \times 10^3\,\mathrm{kJ\,kmol}^{-1}$	Heat of reaction
$F_{20} = 5.0\,\mathrm{m}^3\,\mathrm{h}^{-1}$	Flow: CSTR-2 inlet	$\Delta H_{\mathrm{vap}} = 4.02 \times 10^4\,\mathrm{kJ\,kmol}^{-1}$	Heat of vaporization
$F_r = 2.0\,\mathrm{m}^3\,\mathrm{h}^{-1}$	Flow: SEP-1 vapor	$C_p = 0.231\,\mathrm{kJ\,kg}^{-1}K^{-1}$	Heat capacity
$V_1 = 5.0\,\mathrm{m}^3$	Volume: CSTR-1	$R = 8.314\,\mathrm{kJ\,kmol}^{-1}\,\mathrm{K}^{-1}$	Gas constant
$V_2 = 5.0\,\mathrm{m}^3$	Volume: CSTR-2	$\rho_L = 1000\,\mathrm{kg\,m}^{-3}$	Liquid solution density
$V_3 = 3.0\,\mathrm{m}^3$	Volume: SEP-1	$MW_A = 18\,\mathrm{kg\,kmol}^{-1}$	Molecular weight: A
$\alpha_A = 3.0$	Relative volatility: A	$MW_B = 18\,\mathrm{kg\,kmol}^{-1}$	Molecular weight: B
$\alpha_B = 0.8$	Relative volatility: B	$MW_D = 40.0\,\mathrm{kg\,kmol}^{-1}$	Molecular weight: D
$\alpha_D = 1.0$	Relative volatility: D		

$$\frac{dT_1}{dt} = \frac{F_{10}}{V_1}T_{10} + \frac{F_r}{V_1}T_3 - \frac{F_1}{V_1}T_1 - \frac{\Delta H k_0}{\rho_L C_p}e^{-E/RT_1}C_{A1}^2 + \frac{Q_1}{\rho_L C_p V_1} \tag{7.89a}$$

$$\frac{dC_{A1}}{dt} = \frac{F_{10}}{V_1}C_{A10} + \frac{F_r}{V_1}C_{Ar} - \frac{F_1}{V_1}C_{A1} - k_0 e^{-E/RT_1}C_{A1}^2 \tag{7.89b}$$

$$\frac{dC_{B1}}{dt} = \frac{F_r}{V_1}C_{Br} - \frac{F_1}{V_1}C_{B1} + k_0 e^{-E/RT_1}C_{A1}^2 \tag{7.89c}$$

$$\frac{dT_2}{dt} = \frac{F_{20}}{V_2}T_{20} + \frac{F_1}{V_2}T_1 - \frac{F_2}{V_2}T_2 - \frac{\Delta H k_0}{\rho_L C_p}e^{-E/RT_2}C_{A2}^2 + \frac{Q_2}{\rho_L C_p V_2} \tag{7.89d}$$

$$\frac{dC_{A2}}{dt} = \frac{F_{20}}{V_2}C_{A20} + \frac{F_1}{V_2}C_{A1} - \frac{F_2}{V_2}C_{A2} - k_0 e^{-E/RT_2}C_{A2}^2 \tag{7.89e}$$

$$\frac{dC_{B2}}{dt} = \frac{F_1}{V_2}C_{B1} - \frac{F_2}{V_2}C_{B2} + k_0 e^{-E/RT_2}C_{A2}^2 \tag{7.89f}$$

$$\frac{dT_3}{dt} = \frac{F_2}{V_3}(T_2 - T_3) - \frac{\Delta H_{\mathrm{vap}} \dot{M}_r}{\rho_L C_p V_3} + \frac{Q_3}{\rho_L C_p V_3} \tag{7.89g}$$

$$\frac{dC_{A3}}{dt} = \frac{F_2}{V_3} C_{A2} - \frac{F_r}{V_3} C_{Ar} - \frac{F_3}{V_3} C_{A3} \tag{7.89h}$$

$$\frac{dC_{B3}}{dt} = \frac{F_2}{V_3} C_{B2} - \frac{F_r}{V_3} C_{Br} - \frac{F_3}{V_3} C_{B3} \tag{7.89i}$$

where T_j denotes the temperature of the j-th vessel ($j = 1$ denotes CSTR-1, $j = 2$ denotes CSTR-2, and $j = 3$ denotes SEP-1), C_{ij} denotes the concentration of the i-th species ($i = A,\ B$) in the j-th vessel, and $\dot{M}_r$ denotes the molar flow rate of the recycle stream.

The relative volatility of each species is assumed to be constant within the operating temperature range of the flash tank. The following algebraic equations are used to model the composition of the recycle stream:

$$C_{D3} = (\rho_L - C_{A3} MW_A - C_{B3} MW_B) / MW_D \tag{7.90a}$$

$$C_{ir} = \frac{\alpha_i \rho_L C_{i3}}{\sum_{j \in \{A,B,D\}} \alpha_j C_{j3} MW_j}, \quad i = A,\ B,\ D \tag{7.90b}$$

$$\dot{M}_r = F_r \left(C_{Ar} + C_{Br} + C_{Dr} \right) \tag{7.90c}$$

where C_{ir} is the overhead vapor concentration of the separator. Given the assumption of constant liquid hold-up and constant liquid density, the volumetric flow rates are given by the following equations:

$$F_1 = F_r + F_{10} \tag{7.91a}$$

$$F_2 = F_1 + F_{20} \tag{7.91b}$$

$$F_3 = F_2 - F_r \tag{7.91c}$$

where F_j is the volumetric flow rate of the outlet stream of the j-th vessel.

The process network has five manipulated inputs: the three heat rates Q_j, $j = 1,\ 2,\ 3$ and the inlet concentration of the reactant A in the feedstock to each reactor (C_{A10} and C_{A20}). The bounds on the available control action are $Q_j \in [-1.0,\ 1.0] \times 10^5$ kJ h^{-1} for $j = 1,\ 2$, $Q_3 \in [2.2,\ 2.5] \times 10^6$ kJ h^{-1}, and $C_{Aj0} \in [0.5,\ 7.5]$ kmol m^{-3} $j = 1,\ 2$. In addition to the input constraints, the reactions take place within the temperature range from 370.0 to 395.0 K and thus, the reactors are to be operated within this temperature range. The separation occurs at 390.0 K.

The real-time economics of the process network are assumed to be described by the molar flow rate of desired product B leaving the process network which is denoted as $\dot{M}_{B3}$. The time-averaged amount of reactant that may be fed to each reactor is constrained to an average amount of 20.0 kmol h^{-1} which gives rise to the following two input average constraints:

$$\frac{1}{t_f - t_0}\int_{t_0}^{t_f} F_{j0}C_{Aj0}(t)\,dt = 20.0\,\text{kmol}\,\text{h}^{-1} \tag{7.92}$$

for $j = 1,\ 2$ where t_0 and t_f are the initial and final time of the operation of the process network. Since the inlet flow rates F_{10} and F_{20} are constant, the average input constraint may be written in terms of the inlet concentration of A only such that the time-averaged value of C_{Aj0} must be equal to 4.0 kmol m^{-3}.

The economically optimal steady-state (which is simply referred to as the optimal steady-state for the remainder) will be used in the design of a real-time LEMPC, i.e., the stability region for the optimal steady-state will be used in the LEMPC formulation. Since the reaction rate is maximized at high temperature, computing the optimal steady-state with the exact acceptable temperature operating range will give an optimal steady-state with the greatest acceptable reactor operating temperature. Much like current practice, the optimal steady-state is computed with a degree of conservativeness or "back-off" introduced in the acceptable operating temperature range, so that the reactor temperature is maintained within the acceptable operating range over the length of operation in the presence of uncertainty and disturbances (see [39] and the references therein, for instance, for more details on the back-off methodology). Thus, the optimal steady-state must satisfy a restricted temperature range of $T_{js} \in [370.0, 380.0]$ K for $j = 1,\ 2$. The steady-state optimization problem is given by:

$$\begin{aligned}
\max_{x_s, u_s} \quad & F_3 C_{B3s} \\
\text{s.t.} \quad & f(x_s, u_s) = 0 \\
& 370.0\,\text{K} \le T_{1s} \le 380.0\,\text{K} \\
& 370.0\,\text{K} \le T_{2s} \le 380.0\,\text{K} \\
& T_{3s} = 390.0\,\text{K} \\
& -1.0 \times 10^5\,\text{kJh}^{-1} \le Q_{1s} \le 1.0 \times 10^5\,\text{kJh}^{-1} \\
& -1.0 \times 10^5\,\text{kJh}^{-1} \le Q_{2s} \le 1.0 \times 10^5\,\text{kJh}^{-1} \\
& 2.2 \times 10^6\,\text{kJh}^{-1} \le Q_{3s} \le 2.5 \times 10^6\,\text{kJh}^{-1} \\
& C_{A10s} = C_{A20s} = 4.0\,\text{kmol}\,\text{m}^{-3}
\end{aligned} \tag{7.93}$$

where $f(x_s, u_s) = 0$ represents the steady-state model. The optimal steady-state vector (omitting units) is:

$$\begin{aligned}
x_s^* &= \left[\, T_{1s}^* \; C_{A1s}^* \; C_{B1s}^* \; T_{2s}^* \; C_{A2s}^* \; C_{B2s}^* \; T_{3s}^* \; C_{A3s}^* \; C_{B3s}^* \,\right]^T \\
&= \left[\, 380.0 \; 2.67 \; 2.15 \; 380.0 \; 2.42 \; 2.06 \; 390.0 \; 1.85 \; 2.15 \,\right]^T,
\end{aligned} \tag{7.94}$$

and the optimal steady-state input vector is

$$\begin{aligned} u_s^* &= \left[\, Q_{1s}^* \; Q_{2s}^* \; Q_{3s}^* \; C_{A10s}^* \; C_{A20s}^* \,\right]^T \\ &= \left[-4.21 \times 10^3 \; 1.70 \times 10^4 \; 2.34 \times 10^6 \; 4.0 \; 4.0\right]^T . \end{aligned} \tag{7.95}$$

The optimal steady-state is open-loop unstable.

The control objective of the process network is to optimize the economics through real-time operation while maintaining the closed-loop state trajectory inside a well-defined state-space set. To accomplish this objective, the real-time LEMPC scheme is applied to the process network. In stark contrast to traditional tracking control that forces the closed-loop state to converge to the (optimal) steady-state, applying LEMPC to the process network is not expected to achieve convergence to the optimal steady-state. Instead, LEMPC may force the process network to operate in a consistently transient manner to achieve better closed-loop performance compared to the closed-loop performance at the optimal steady-state.

For the implementation of the LEMPC, the acceptable temperature range is not treated as a hard constraint. Instead, the acceptable temperature range is accounted for by imposing quadratic penalty terms in the stage cost of the LEMPC. Thus, the stage cost used in the objective function of the LEMPC is

$$l_e(x, u) = -F_3 C_{B3} + \sum_{i=1}^{3} Q_{c,i} (T_i - T_{is}^*)^2 \tag{7.96}$$

where T_{is}^*, $i = 1,\ 2,\ 3$ are the optimal steady-state temperatures. The stage cost of Eq. 7.96 includes the economics and the quadratic penalty terms for the temperature. The weight coefficients are $Q_{c,1} = 0.018$, $Q_{c,2} = 0.022$, and $Q_{c,3} = 0.01$ and have been tuned such that the closed-loop temperatures are maintained near the optimal steady-state temperature. Since no hard or soft constraints are imposed on the temperature in the LEMPC, it is emphasized that there is no guarantee that the temperatures are maintained within the acceptable temperature range described above ($T_j \in [370.0,\ 395.0]$ K for $j = 1,\ 2$ and $T_3 \approx 390.0$ K). In this example, small violations over a short period are considered acceptable. If maintaining the operation within the acceptable operating temperature range is considered critical, one may add various modifications to the LEMPC to achieve this objective such as decreasing the size of Ω_{ρ_e}, adding hard or soft constraints on the temperature in the LEMPC, or adding a contractive constraint on the temperature ODEs.

An explicit stabilizing controller is designed using feedback linearization techniques to make the dynamics of the temperature ODEs linear (in a state-space region where the input constraints are satisfied) under the explicit controller. Specifically, the temperature ODEs are input-affine in the heat rate input and have the form:

$$\dot{T}_j = f_j(x) + b_j Q_j \tag{7.97}$$

where $f_j(x)$ is a nonlinear scalar-valued function, b_j is constant and $j = 1,\ 2,\ 3$. The controller that makes the closed-loop temperature dynamics linear is:

$$Q_j = -\frac{1}{b_j}\left(f_j(x) + K_j(T_j - T_{js}^*)\right) \tag{7.98}$$

where K_j denotes the controller gain. In this case, the controller gains are $K_1 = 5$, $K_2 = 5$, and $K_3 = 7$, respectively. The inlet concentration input values are fixed to the average values (4.0 kmol m^{-3}). Through extensive closed-loop simulations under the state feedback controller, a quadratic Lyapunov function for the process network under the feedback controller $h(x)$ is determined. An estimate of the stability region of the process network under the feedback controller was characterized by computing the state-space points where $\dot{V} < 0$ and taking the stability region to be a level set of the Lyapunov function containing only state-space points where the time-derivative of the Lyapunov function is negative. The quadratic Lyapunov function has the form:

$$V(x) = (x - x_s^*)^T P (x - x_s^*) \tag{7.99}$$

where P is the following positive definite matrix:

$$P = \text{diag}\left[0.001\ 1.5\ 0.05\ 0.001\ 1.5\ 0.05\ 0.001\ 1.5\ 0.05\right]. \tag{7.100}$$

The estimated stability region Ω_ρ is the level set of the Lyapunov function where $V(x) \leq 11.0$, i.e., $\rho = 11.0$. The subset of the stability region which defines the mode 1 constraint of the LEMPC is $\rho_e = 10.0$ and has been determined through extensive closed-loop simulation under LEMPC as the subset of the stability region Ω_ρ where the closed-loop state under LEMPC is maintained in Ω_ρ.

The input average constraint is imposed over successive, finite-length operating periods. Specifically, the average constraint must be satisfied over each operating period $t_M = M\Delta$ where M is the number of sampling periods in the operating period. This ensures that over the entire length of operation the average constraint will be satisfied. For this example, the operating period was chosen to be $t_M =$ 2.4 h which leads to better asymptotic average economic performance under LEMPC (assuming no computational delay) than the asymptotic average performance at the economically optimal steady-state.

To solve the dynamic optimization problem of the LEMPC, orthogonal collocation with three Radau collocation points per sampling period is employed for discretization of the ODEs (see, for instance, [37] for details on solving a dynamic optimization problem using a simultaneous approach). The open-source nonlinear optimization solver Ipopt [21] was employed owing to its ability to exploit the high degree of sparsity of the resulting optimization problem. Analytical first and second-order derivative information was provided to the solver. The closed-loop simulations were coded in C++ and performed on an Intel® Core™ 2 Quad 2.66 GHz processor running an Ubuntu Linux operating system. The sampling period of the LEMPC used in the simulations below is $\Delta = 0.01$ h. To simulate forward in time the closed-loop process network, the fourth-order Runge-Kutta method was used with a time step of 0.0001 h.

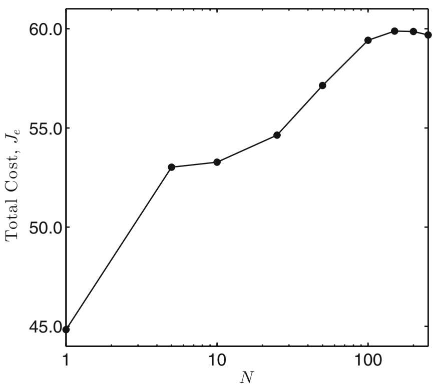


Fig. 7.21 The total economic cost J_e over one operating window length of operation (2.4 h) of the process network under LEMPC with the prediction horizon length

In the first set of simulations, nominal operation of the process network under LEMPC implemented in a typical receding horizon fashion is considered under ideal computation, i.e., assuming no computational delay. The closed-loop economic performance under LEMPC is assessed using the economic performance index which is defined as:

$$J_e = \int_0^{t_f} F_3 C_{B3}\, dt. \tag{7.101}$$

Since the LEMPC does not directly optimize the molar flow rate of product out of the process network, the stage cost index will also be considered as a measure of the closed-loop performance and is given by:

$$L_e = -\int_0^{t_f} l_e(x, u)\, dt. \tag{7.102}$$

First, the effect of the prediction horizon on the closed-loop economic performance over one operating period (2.4 h) is considered. The closed-loop performance index of Eq. 7.101 plotted against the prediction horizon length is given in Fig. 7.21. A significant increase in closed-loop performance is observed initially with increasing prediction horizon length until the closed-loop performance becomes approximately constant. Owing to this fact, a prediction horizon of $N = 200$ is used in all subsequent simulations. A simulation over many operating periods such that the effect of the initial condition on closed-loop performance becomes negligible is performed (with $N = 200$). The asymptotic average closed-loop economic performance, which is the time-averaged economic cost after the effect of the initial condition becomes negligible, is determined from this simulation to be 25.0 kmol h^{-1} (in this case, the time-averaged production rate over each operating window becomes constant after a sufficiently long length of operation). The optimal steady-state production rate of B is 21.5 kmol h^{-1}. Thus, the asymptotic production rate of the process network under the LEMPC is 16.3 % better than the production rate at the economically optimal steady-state.

The effect of computational delay is considered in the next set of simulations, and two scenarios are considered: (1) the closed-loop process network under LEMPC implemented in a typical receding horizon fashion where the control action is subject to real-time computational delay (for the sake of simplicity, this case will be referred to as the closed-loop process network under LEMPC for the remainder) and (2) the closed-loop process network under the real-time LEMPC scheme (also, subject to real-time computational delay). For the former scenario, the LEMPC begins to compute a control action at each sampling instance after receiving a state measurement. Owing to the computational delay, the control action applied to the process network is the most up-to-date control action. For example, if it takes 0.002 h to compute the control action at the sampling instance t_k, then $u(t_{k-1})$ is applied to the process network from t_k to t_k + 0.002h (assuming $u(t_{k-1})$ is available at t_k) and applies $u(t_k)$ to the process network from t_k + 0.002 h to $t_{k+1} = t_k + \Delta$. For each scenario, a 12.0 h length of closed-loop operation is simulated. For the real-time LEMPC case, the LEMPC is forced to recompute a new solution after three sampling periods have elapsed since the last time an LEMPC solution was computed, i.e., the solver starts computing a new solution at the beginning of the fourth sampling period.

The average computation time required to solve the LEMPC (of scenario (1)) at each sampling time was 11.2 s (31.2 % of the sampling period) with a standard deviation of 7.42 s. The maximum computation time over the simulation is 61.9s which is almost double the sampling period. The computation time exceeds the sampling period ten out of the 1,200 sampling periods in the simulation. Over the course of both simulations, the closed-loop state is maintained in Ω_ρ. The closed-loop trajectories under the real-time LEMPC scheme are given in Fig. 7.22 (the closed-loop behavior under the LEMPC subject to real-time computational delay was similar). The difference between the performance index of the two cases is less

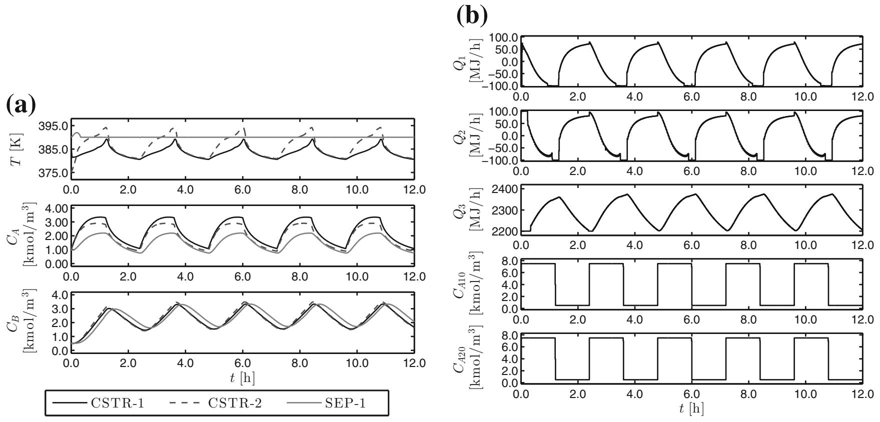


Fig. 7.22 The closed-loop **a** state and **b** input trajectories of the nominally operated process network under the real-time LEMPC scheme

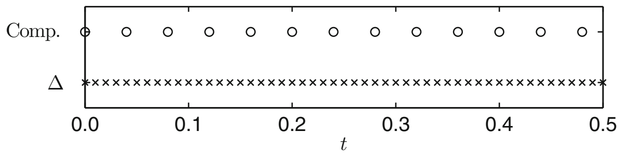


Fig. 7.23 The number of times the LEMPC problem was solved (Comp.) as dictated by the real-time implementation strategy compared to the sampling period (Δ) over the first 0.5 h of operation

than 0.5 % (the performance indexes for case (1) and case (2) were 284.3 and 283.3, respectively).

While little difference between the two cases in terms of closed-loop performance is observed, it is important to note that an a priori guarantee on closed-loop stability under the real-time LEMPC may be made. Also, the total on-line computation time to solve the LEMPC over the two simulations is 3.74 and 0.94 h, respectively. The real-time LEMPC reduces the total on-line computation requirement by 75 % compared to LEMPC implemented in a receding horizon fashion because the real-time LEMPC does not recompute a control action at each sampling instance, while LEMPC, implemented in a receding horizon fashion, recomputes a control action at each sampling instance. To better illustrate this point, Fig. 7.23 shows the frequency the LEMPC problem is solved under the real-time implementation strategy with respect to the sampling period over the first 0.5 h of operation. Over this time, the LEMPC optimization problem is solved at a rate of one out of every four sampling periods. This trend continues over the remainder of the 12.0 h length of operation and hence, the 75 % reduction in total computational time.

Since the computational delay depends on many factors, e.g., model dimension, prediction horizon, solution strategy used to solve the dynamic optimization problem, the nonlinear optimization solver used, and computer hardware, it is also important to consider computational delay greater than one sampling period to demonstrate that the real-time LEMPC scheme may handle computation delay of arbitrary length. Therefore, another set of simulations is considered where longer computational delay is simulated. The computation delay is modeled as a bounded uniformly-distributed random number and the maximum computational delay is assumed to be less than 10 sampling periods. Both the LEMPC (receding horizon implementation) and the real-time LEMPC scheme are considered. To make the comparison as consistent as possible, the computational delay, at the time steps the real-time LEMPC is solved, is simulated to be the same as the computation delay to solve the LEMPC at the same time step (recall the real-time LEMPC is not solved at each sampling period). Given the computational delay is much greater for this set of simulations than in the previous set of simulations, the real-time LEMPC is forced to recompute a new solution after 15 sampling periods have elapsed since the last time it computed a solution.

Several simulations are performed, each starting at a different initial condition, and the performance indexes of these simulations are given in Table 7.4. Applying

Table 7.4 The performance indices of the process network under the back-up explicit controller, under the LEMPC subject to computational delay, and under the real-time LEMPC for several simulations

Sim.	Back-up controller		LEMPC		Real-time LEMPC	
	J_e	L_e	J_e	L_e	J_e	L_e
1	225.5	225.4	277.0	245.0	295.1	216.5
2	254.2	254.1	318.7	278.6	307.3	279.6
3	260.5	260.4	319.9	286.3	318.1	294.7
4	232.7	230.6	290.7	255.7	299.2	266.4
5	250.0	250.0	308.7	276.9	322.8	282.9

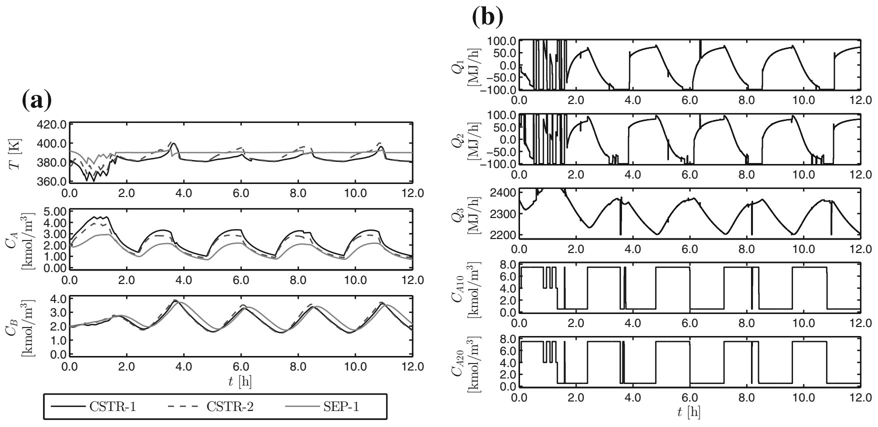


Fig. 7.24 The closed-loop **a** state and **b** input trajectories of process network under the real-time LEMPC scheme where the computational delay is modeled as a bounded random number

the back-up explicit controller $h(x)$ implemented in a sample-and-hold fashion to the chemical process network is also considered and the performance indexes of these simulations are given in Table 7.4 as well. The average improvement in economic performance compared to the process network under the back-up controller is 26.1 % under the real-time LEMPC scheme and 23.9 % under the LEMPC (implemented in a receding horizon). Thus, a substantial economic benefit is achieved by applying LEMPC to the process network. While the real-time LEMPC does not always achieve better performance (either measured in terms of the economic performance index or stage cost index) compared to the performance under LEMPC, the closed-loop trajectories between the two cases are significantly different. Figures 7.24 and 7.25 give the closed-loop trajectories of simulation 2 (as labeled in Table 7.4). The input trajectory computed by the real-time LEMPC has chattering initially over the first operating period because of the effect of the initial condition, but after the first operating period when the effect of the initial condition dissipates, the computed input trajectory is significantly smoother. On the other hand, chattering in the input profiles

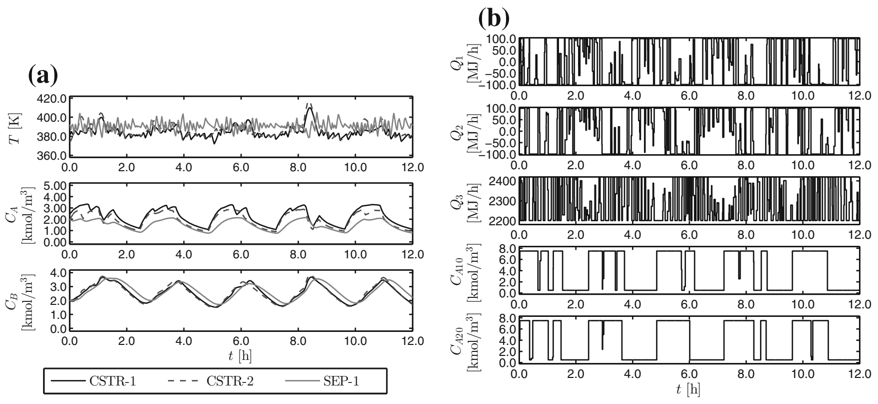


Fig. 7.25 The closed-loop **a** state and **b** input trajectories of process network under LEMPC subject to computational delay where the computational delay is modeled as a bounded random number

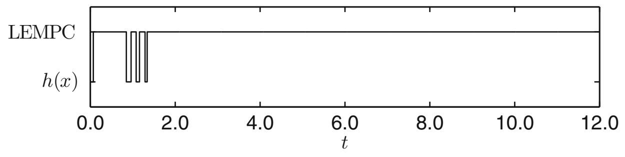


Fig. 7.26 A discrete trajectory depicting when the control action applied to the process network over each sampling period is from a precomputed LEMPC solution or from the back-up controller for the closed-loop simulation of Fig. 7.24

is observed throughout the entire simulation under the LEMPC. If we compare the performance index of operation from $t = 2.4$ h to $t = 12.0$ h (after the first operating period) for simulation 2, the indexes are $J_e = 249.8$ and $L_e = 227.9$ for operation under the real-time LEMPC and $J_e = 248.5$ and $L_e = 217.4$ for operation under the LEMPC; the performance under the real-time LEMPC is better over this period than under LEMPC.

Over the five simulations under the real-time LEMPC strategy, the explicit controller was applied on average 19 out of 1200 sampling periods. For the simulation of Fig. 7.24, a discrete trajectory showing when the control action applied to the process network under the real-time LEMPC strategy is from a precomputed LEMPC solution or from the back-up controller is given in Fig. 7.26. For this case, the back-up controller is used 31 out of 1200 sampling periods (2.7 % of the sampling periods). From Fig. 7.26, the back-up controller is only applied over the first operating period and is not used in any subsequent sampling period. Thus, the source of performance degradation for this case (Sim. 2 in Table 7.4) is due to applying the explicit back-up controller to maintain the closed-loop state in Ω_ρ. Again, it is emphasized that there is no a priori guarantee that the LEMPC implemented in a receding horizon fashion subject to computational delay may maintain the closed-loop state inside Ω_ρ.

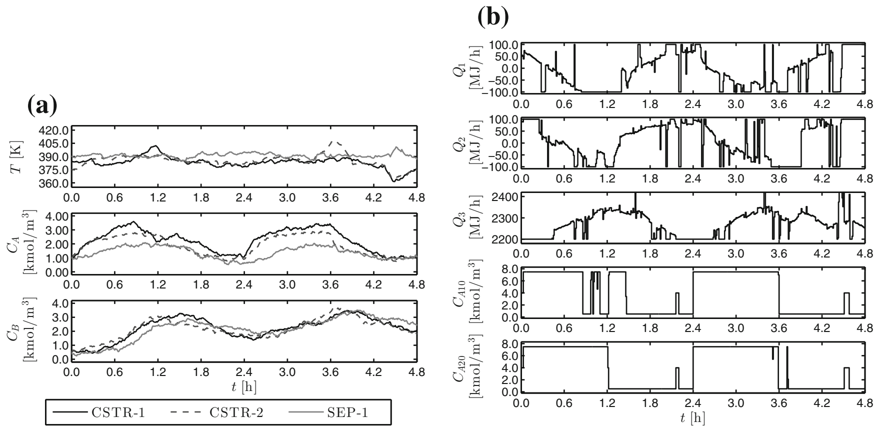


Fig. 7.27 The closed-loop **a** state and **b** input trajectories of process network under the real-time LEMPC scheme with bounded process noise

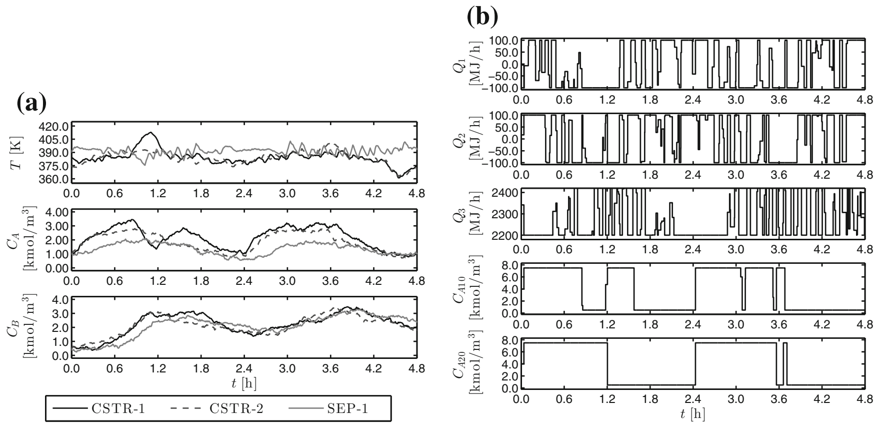


Fig. 7.28 The closed-loop **a** state and **b** input trajectories of process network under LEMPC subject to computational delay with bounded process noise

In the last set of simulations, significant bounded Gaussian process noise with zero mean is added to the model states. The standard deviations of the noise added to the temperature and concentration states are 5.0 and 0.5, respectively and the bounds on the noise are 2.0 and 15.0, respectively. Two closed-loop simulations over 12.0 h length of operation are completed with the same realization of the process noise. In the first simulation, the process network is controlled by the real-time LEMPC and the closed-loop trajectories are given in Fig. 7.27 over the first two operating periods. For this case, the back-up controller is applied 69 out of 1200 sampling periods (5.8 % of the sampling periods). In the second simulation, the process network is controlled by LEMPC subject to computation delay (trajectories shown in Fig. 7.28).

From Fig. 7.28, a significant degree of chattering and bang-bang type actuation in the input trajectory is observed. This behavior tries to combat the effect of the added process noise and is due to not penalizing control actions in the stage cost and not imposing rate of change constraints on the control actions. In practice, one could add one or both of these elements to the LEMPC if the computed input trajectory is not implementable. On the other hand, the real-time LEMPC implements a much smoother input trajectory (Fig. 7.27) because the precomputed input trajectory of the real-time LEMPC has a degree of smoothness like the closed-loop trajectory of the nominally operated process network (Fig. 7.22). If the precomputed input trajectory satisfies the stability conditions, it will be applied to the closed-loop process network with disturbances. The closed-loop system under the real-time LEMPC has guaranteed stability properties, but is not recomputed at each sampling period like the receding horizon implementation of LEMPC which will try to combat the effect of the disturbance on performance. In both cases, the state is maintained in Ω_ρ. The performance indexes of the two cases are 301.6 under the real-time LEMPC and 295.5 under the LEMPC; the closed-loop performance under the real-time LEMPC scheme is 2.0 % better than applying LEMPC without accounting for the computational delay. Moreover, the back-up controller is also applied to the process network subject to the same realization of the process noise. The economic performance index for this case is 242.3. For operation with process noise, the economic performance improvement over the process network under the back-up controller is 24.4 % under the real-time LEMPC strategy and 21.9 % under the receding horizon LEMPC for the same initial condition.

7.5 Conclusions

In this chapter, three EMPC designs were considered. The first section focused on the design of economic MPC for a class of nonlinear singularly perturbed systems. Under appropriate stabilizability assumptions, fast sampling of the fast states and slow sampling of the slow states, the presented composite control system featuring an EMPC may dictate a possible time-varying operation to address economic considerations while guaranteeing closed-loop stability. Closed-loop stability was addressed through singular perturbation arguments.

In the second section, an application study of several distributed EMPC strategies was presented. Two important performance metrics were considered to assess the EMPC strategies including the closed-loop performance and the computational time required to solve the optimization problem(s) at each sampling time. From the closed-loop simulation results of application study, a distributed EMPC strategy was capable of delivering similar closed-loop performance as a centralized EMPC approach while reducing the on-line computation time required to solve the optimization problems.

In the final section, a strategy for implementing Lyapunov-based economic model predictive control (LEMPC) in real-time with computation delay was developed. The implementation strategy uses a triggering condition to precompute an input trajectory from LEMPC over a finite-time horizon. At each sampling period, if a certain stability (triggering) condition is satisfied, then the precomputed control action by LEMPC is applied to the closed-loop system. If the stability condition is violated, then a backup explicit stabilizing controller is used to compute the control action for the sampling period. In this fashion, the LEMPC is used when possible to optimize the economics of the process. Conditions such that the closed-loop state under the real-time LEMPC is always bounded in a compact set were derived.

References

1. Kokotovic P, Khalil HK, O'Reilly J (1999) Singular perturbation methods in control: analysis and design, vol. 25. SIAM
2. Christofides PD, Daoutidis P (1996) Feedback control of two-time-scale nonlinear systems. Int J Control 63:965–994
3. Kumar A, Daoutidis P (2002) Nonlinear dynamics and control of process systems with recycle. J Process Control 12:475–484
4. Baldea M, Daoutidis P (2012) Control of integrated chemical process systems using underlying DAE models. In: Biegler LT, Campbell SL, Mehrmann V (eds) Control and optimization with differential-algebraic constraints. SIAM, pp 273–291
5. Chen X, Heidarinejad M, Liu J, Christofides PD (2012) Composite fast-slow MPC design for nonlinear singularly perturbed systems. AIChE J 58:1802–1811
6. Chen X, Heidarinejad M, Liu J, Muñoz de la Peña D, Christofides PD (2011) Model predictive control of nonlinear singularly perturbed systems: application to a large-scale process network. J Process Control 21:1296–1305
7. Massera JL (1956) Contributions to stability theory. Ann Math 64:182–206
8. Lin Y, Sontag E, Wang Y (1996) A smooth converse Lyapunov theorem for robust stability. SIAM J Control Optim 34:124–160
9. Khalil HK (2002) Nonlinear Systems, 3rd edn. Prentice Hall, Upper Saddle River, NJ
10. Muñoz de la Peña D, Christofides PD (2008) Lyapunov-based model predictive control of nonlinear systems subject to data losses. IEEE Trans Autom Control 53:2076–2089
11. Christofides P, Teel A (1996) Singular perturbations and input-to-state stability. IEEE Trans Autom Control 41:1645–1650
12. Christofides PD, Scattolini R, Muñoz de la Peña D, Liu J (2013) Distributed model predictive control: a tutorial review and future research directions. Comput Chem Eng 51:21–41
13. Liu J, Muñoz de la Peña D, Christofides PD (2009) Distributed model predictive control of nonlinear process systems. AIChE J 55:1171–1184
14. Liu J, Chen X, Muñoz de la Peña D, Christofides PD (2010) Sequential and iterative architectures for distributed model predictive control of nonlinear process systems. AIChE J 56:2137–2149
15. Scattolini R (2009) Architectures for distributed and hierarchical model predictive control—a review. J Process Control 19:723–731
16. Müller MA, Allgöwer F (2014) Distributed economic MPC: a framework for cooperative control problems. In: Proceedings of the 19th world congress of the international federation of automatic control. Cape Town, South Africa, pp 1029–1034
17. Driessen PAA, Hermans RM, van den Bosch PPJ (2012) Distributed economic model predictive control of networks in competitive environments. In: Proceedings of the 51st IEEE conference on decision and control, Maui, HI, pp 266–271
18. Chen X, Heidarinejad M, Liu J, Christofides PD (2012) Distributed economic MPC: application to a nonlinear chemical process network. J Process Control 22:689–699

19. Lee J, Angeli D (2012) Distributed cooperative nonlinear economic MPC. In: Proceedings of the 20th international symposium on mathematical theory of networks and systems. Melbourne, Australia
20. Özgülşen F, Adomaitis RA, Çinar A (1992) A numerical method for determining optimal parameter values in forced periodic operation. Chem Eng Sci 47:605–613
21. Wächter A, Biegler LT (2006) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math Program 106:25–57
22. Ellis M, Christofides PD (2014) Optimal time-varying operation of nonlinear process systems with economic model predictive control. Ind Eng Chem Res 53:4991–5001
23. Ronco E, Arsan T, Gawthrop PJ (1999) Open-loop intermittent feedback control: practical continuous-time GPC. IEE Proc—Control Theory Appl 146:426–434
24. Chen WH, Ballance DJ, O'Reilly J (2000) Model predictive control of nonlinear systems: computational burden and stability. IEE Proc—Control Theory Appl 147:387–394
25. Findeisen R, Allgöwer F (2004) Computational delay in nonlinear model predictive control. In: Proceedings of the IFAC international symposium of advanced control of chemical processes, Hong Kong, pp 427–432
26. Scokaert POM, Mayne DQ, Rawlings JB (1999) Suboptimal model predictive control (feasibility implies stability). IEEE Trans Autom Control 44:648–654
27. Würth L, Hannemann R, Marquardt W (2009) Neighboring-extremal updates for nonlinear model-predictive control and dynamic real-time optimization. J Process Control 19:1277–1288
28. Zavala VM, Biegler LT (2009) The advanced-step NMPC controller: optimality, stability and robustness. Automatica 45:86–93
29. Biegler LT, Yang X, Fischer GAG (2015) Advances in sensitivity-based nonlinear model predictive control and dynamic real-time optimization. J Process Control 30:104–116
30. Ganesh N, Biegler LT (1987) A reduced Hessian strategy for sensitivity analysis of optimal flowsheets. AIChE J 33:282–296
31. Yang X, Biegler LT (2013) Advanced-multi-step nonlinear model predictive control. J Process Control 23:1116–1128
32. Jäschke J, Yang X, Biegler LT (2014) Fast economic model predictive control based on NLP-sensitivities. J Process Control 24:1260–1272
33. Würth L, Hannemann R, Marquardt W (2011) A two-layer architecture for economically optimal process control and operation. J Process Control 21:311–321
34. Wolf IJ, Muñoz DA, Marquardt W (2014) Consistent hierarchical economic NMPC for a class of hybrid systems using neighboring-extremal updates. J Process Control 24:389–398
35. Diehl M, Bock H, Schlöder J (2005) A real-time iteration scheme for nonlinear optimization in optimal feedback control. SIAM J Control Optim 43:1714–1736
36. Diehl M, Ferreau HJ, Haverbeke N (2009) Efficient numerical methods for nonlinear MPC and moving horizon estimation. In: Magni L, Raimondo DM, Allgöwer F (eds) Nonlinear model predictive control, vol 384., Lecture Notes in Control and Information SciencesSpringer, Berlin Heidelberg, pp 391–417
37. Biegler LT (2010) Nonlinear programming: concepts, algorithms, and applications to chemical processes. SIAM, Philadelphia, PA
38. Tabuada P (2007) Event-triggered real-time scheduling of stabilizing control tasks. IEEE Trans Autom Control 52:1680–1685
39. Kookos IK, Perkins JD (2002) An algorithmic method for the selection of multivariable process control structures. J Process Control 12:85–99