Real-Time Economic Model Predictive Control of Nonlinear Process Systems

Matthew Ellis
Dept. of Chemical and Biomolecular Engineering, University of California, Los Angeles, CA 90095–1592

Panagiotis D. Christofides
Dept. of Chemical and Biomolecular Engineering, University of California, Los Angeles, CA 90095–1592

Dept. of Electrical Engineering, University of California, Los Angeles, CA 90095–1592

DOI 10.1002/aic.14673
Published online November 18, 2014 in Wiley Online Library (wileyonlinelibrary.com)

Closed-loop stability of nonlinear systems under real-time Lyapunov-based economic model predictive control (LEMPC) with potentially unknown and time-varying computational delay is considered. To address guaranteed closed-loop stability (in the sense of boundedness of the closed-loop state in a compact state-space set), an implementation strategy is proposed which features a triggered evaluation of the LEMPC optimization problem to compute an input trajectory over a finite-time prediction horizon in advance. At each sampling period, stability conditions must be satisfied for the precomputed LEMPC control action to be applied to the closed-loop system. If the stability conditions are not satisfied, a backup explicit stabilizing controller is applied over the sampling period. Closed-loop stability under the real-time LEMPC strategy is analyzed and specific stability conditions are derived. The real-time LEMPC scheme is applied to a chemical process network example to demonstrate closed-loop stability and closed-loop economic performance improvement over that achieved for operation at the economically optimal steady state. © 2014 American Institute of Chemical Engineers *AIChE J*, 61: 555–571, 2015
Keywords: process control, process optimization, chemical processes, model predictive control, process economics, nonlinear systems

Introduction

Nonlinear model predictive control (NMPC) is a control methodology that has attracted substantial attention especially within the chemical process control community because it can stabilize multiple-input multiple-output nonlinear systems while accounting for constraints and performance considerations. Within the last 5 years, significant research effort has focused on economic (nonlinear) model predictive control (EMPC) which attempts to unify process/system economics and real-time feedback control[1–3] (see also Ref. 4, for an overview of recent results on EMPC). Using a general cost functional that reflects the process/system economics, the overall goal of EMPC is to translate economic and stability considerations into control actions that not only maintain stability of the closed-loop system, but also, obtain economic optimality that cannot be achieved through tracking NMPC. The optimization problem of EMPC consists of three main parts: a cost functional with a stage cost that accounts for the economics to be optimized, system constraints including state and input constraints as well as other constraints like stability and performance constraints, and a nonlinear dynamic model to predict the future evolution of the system (and thus, be able to select the optimal input profile with respect to the economic cost over a finite-time prediction horizon). The main difference between tracking NMPC and EMPC is that tracking NMPC uses a positive definite stage cost with respect to a set-point or reference trajectory (typically, a quadratic stage cost is used that is zero at the economically optimal steady state), while EMPC uses a general stage cost that does not need to be positive definite with respect to a set-point, steady state, or reference trajectory.

To highlight some of the recent theoretical work on EMPC, the performance of EMPC with a generalized terminal constraint and self-tuning terminal cost was analyzed,[5] the asymptotic stability and transient optimality of EMPC without terminal conditions was considered,[6] a Lyapunov-based EMPC (LEMPC) scheme with performance constraints generated from an auxiliary Lyapunov-based MPC (LMPC) was proposed that has guaranteed finite-time and infinite-time closed-loop economic performance improvement over the auxiliary LMPC,[7] an LEMPC with triggered evaluations was proposed for both state and output feedback implementations,[8] and a multistage scenario-based EMPC scheme designed to handle uncertainties was developed.[9] Additionally, several applications of EMPC to energy systems have been studied including applying EMPC to: a commercial building to account for demand and energy price,[10] buildings with thermal energy storage,[11] create an electricity pricing structure that minimizes peak electricity demand,[12] and handle energy management in a chlor-alkai process with hybrid

Correspondence concerning this article should be addressed to: P. D. Christofides at pdc@seas.ucla.edu.

renewable energy generation.[13] Many of the aforementioned works have illustrative applications that demonstrate the potential of EMPC on closed-loop economic performance.

Within the chemical process industries, nonlinearities are common owing to Arrhenius reaction rate dependencies and nonlinear kinetic relationships, for example. As the nonlinearities may be significant, the use of a nonlinear dynamic model which is capable of capturing nonlinearities within model predictive control (MPC) has the potential to yield better closed-loop performance in general compared to controlling a nonlinear system with an MPC based on a linear model. However, computational delay that may approach or exceed the sampling time may result as the NMPC optimization problem is nonlinear and potentially nonconvex. If the computational delay is significant relative to the sampling period, closed-loop performance degradation and/or closed-loop instability may occur. To this end, it is important to point out that computational delay is just one type of delay that may cause performance degradation or instability. Other types of delay include measurement and input delay. However, each type of delay is typically handled using different methods (see, e.g. Refs. 14 and 15, for some results on measurement and input delay handling).

Some of the early work addressing computational delay within NMPC includes developing an implementation strategy of solving the MPC problem intermittently to account for the computational delay[16] 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.[17,18] Nominal feasibility and stability has been proved for 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.[17,18] Another option to handle computational delay would be to force the optimization solver to terminate after a prespecified 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[19] 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 origin of the closed-loop system is asymptotically stable.[19]

Since the early work on computational delay of NMPC, more involved strategies have been proposed. Particularly, nonlinear programming (NLP) sensitivity analysis has demonstrated to be a useful tool to handle computational delay by splitting the NMPC optimization problem into two parts: (1) solving a computationally intensive nonlinear optimization problem which is completed before state 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.,[20,21]). If the active-set changes, various methods have been proposed to cope with changing active-sets, for example, solving a quadratic program like that proposed in Ref. 22. In this direction, the advanced-step NMPC[21] 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 using NLP sensitivities after state feedback is received. The advanced-step NMPC has been extended to handle computation spanning multiple sampling periods[23] and to EMPC.[24] Another related approach involves a hierarchical control structure.[25,26] 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 NMPC scheme[27] 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 the state feedback is received. In the feedback phase, a state 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 NMPC and if this trend continues, one can 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, e.g., the overview paper[28] and the book[29] 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, nonconvex dynamic optimization problems and systems with fast dynamics. Additionally, EMPC is generally more computationally intensive compared to tracking NMPC given the additional possible nonlinearities in the stage cost of EMPC. To date, only limited work on explicitly accounting for computational delay within the context of EMPC has been completed (e.g.,[24,26]). In this work, a real-time implementation strategy for LEMPC, referred to as real-time LEMPC, is proposed to account for possibly unknown and time-varying computational delay. The underlying implementation strategy is inspired by event-triggered control concepts[30] as 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 recomputation 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. Lastly, the real-time LEMPC scheme is applied to an illustrative chemical process network to demonstrate closed-loop stability under the proposed scheme. The example also demonstrates that real-time LEMPC improves closed-loop economic

performance compared to operation at the economically optimal steady state.

Preliminaries

Notation

The operator $|\cdot|$ is used to denote the Euclidean norm of a vector. The symbol $S(\Delta)$ denotes the family of piecewise constant functions with period Δ. A continuous function $\alpha : [0, a) \to [0, \infty)$ belongs to class $\mathcal{K}$ if it is strictly increasing and $\alpha(0)=0$. The symbol Ω_ρ denotes a level set of a scalar-valued function $V : \mathbf{R}^{n_x} \to \mathbf{R}_{\geq 0}$ (i.e., $\Omega_\rho := \{x \in \mathbf{R}^{n_x} | V(x) \leq \rho\}$).

Class of systems

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

$$\dot{x}(t) = f(x(t), u(t), w(t)) \tag{1}$$

where $x(t) \in \mathbf{R}^{n_x}$ is the state vector, $u(t) \in U \subset \mathbf{R}^{n_u}$ is the manipulated input vector, $w(t) \in W \subset \mathbf{R}^{n_w}$ is the disturbance vector, and $f(\cdot,\cdot,\cdot)$ is a locally Lipschitz vector function. The input and disturbance vectors are bounded in the following sets

$$U := \{u \in \mathbf{R}^{n_u} \mid u_{\min,i} \leq u_i \leq u_{\max,i},\ i=1,\ \ldots,\ n_u\}, \tag{2}$$

$$W := \{w \in \mathbf{R}^{n_w} \mid |w| \leq \theta\}\ , \tag{3}$$

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 (1) (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 1. *There exists a feedback controller $h(x) \in U$ with $h(0)=0$ that renders the origin of the closed-loop system (1) with $u(t)=h(x(t))$ and $w(t) \equiv 0$ asymptotically stable for all $x \in D$ where D is an open neighborhood of the origin.*

Applying converse theorems,[31,32] Assumption 1 implies that there exists a continuously differentiable Lyapunov function, $V(x)$, for the closed-loop system (1) with $u=h(x) \in U$ and $w(t) \equiv 0$ such that the following inequalities hold

$$\alpha_1(|x|) \leq V(x) \leq \alpha_2(|x|), \tag{4a}$$

$$\frac{\partial V(x)}{\partial x} f(x, h(x), 0) \leq -\alpha_3(|x|), \tag{4b}$$

$$\left|\frac{\partial V}{\partial x}\right| \leq \alpha_4(|x|) \tag{4c}$$

for all $x \in D \subseteq \mathbf{R}^{n_x}$ where D is an open neighborhood of the origin (not necessarily the same open neighborhood as that introduced in Assumption 1) and $\alpha_i(\cdot), i=1,\ 2,\ 3,\ 4$ are functions of class $\mathcal{K}$. 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 $h(x)$. Various techniques have been developed for designing explicit stabilizing controllers for the system (1) including Lyapunov-based techniques and geometric methods.[33–35]

Measurements of the state vector of (1) 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 (1) 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, Ref. 36 for more discussion on this point.

Remark 1. *Currently, there are no general methods for constructing Lyapunov functions for broad classes of nonlinear systems with constraints. However, for certain classes of systems, there exist some general methods for constructing Lyapunov functions (e.g., Zubov's method[37] and the sum of squares decomposition[38]). Within the context of chemical process control, quadratic Lyapunov functions have been widely used and have been demonstrated to be effective for estimating the region of attraction of a given equilibrium point of a system (see, e.g., the numerous examples in Ref.* 35*).*

Lyapunov-based economic model predictive control

Lyapunov-based economic model predictive control (LEMPC) is an EMPC scheme that uses the explicit stabilizing controller $h(x)$ to design two regions of operation where closed-loop stability of the system (1) under the LEMPC and recursive feasibility of the optimization problem are guaranteed for operation in the presence of uncertainty. LEMPC is a two-mode controller where the various modes are defined by two Lyapunov-based constraints. The formulation of the LEMPC optimization is

$$\underset{u \in S(\Delta)}{\text{minimize}} \quad \int_{t_k}^{t_{k+N}} l_e(\tilde{x}(\tau), u(\tau))\ d\tau \tag{5a}$$

$$\text{subject to} \quad \dot{\tilde{x}}(t) = f(\tilde{x}(t), u(t), 0) \tag{5b}$$

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

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

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

$$\frac{\partial V(x(t_k))}{\partial x} f(x(t_k), u(t_k), 0) \leq \frac{\partial V(x(t_k))}{\partial x} f(x(t_k), h(x(t_k)), 0),\ \text{if } x(t_k) \notin \Omega_{\rho_e} \tag{5f}$$

where the decision variable of the optimization problem is the piecewise constant input trajectory over the prediction horizon $N\Delta$. The notation $\tilde{x}(t)$ is used to denote the predicted state trajectory.

The LEMPC dynamic optimization problem minimizes a cost functional (5a) with a stage cost $l_e : \mathbf{R}^{n_x} \times \mathbf{R}^{n_u} \to \mathbf{R}$ chosen to reflect the system economics. The nominal process model is the constraint (5b) and is used to predict the evolution of the system under the computed input trajectory over the prediction horizon. The dynamic model is initialized with a state measurement obtained at the current sampling period [constraint (5c)]. The constraint (5d) limits the computed control actions to be in the set of the available control actions.

Mode 1 operation of the LEMPC is defined by the constraint (5e) and is active when the current state is inside a predefined subset of the stability region $\Omega_{\rho_e} \subset \Omega_\rho$. As an economic performance benefit may be realized when operating the system (1) in a dynamic fashion compared to operating the system at the economically optimal steady state (i.e., the steady state that minimizes the stage cost amongst all of the admissible steady states), mode 1 of the LEMPC is used to allow the controller to force a potentially dynamic or transient operating policy. However, to maintain boundedness of the closed-loop state within a well-defined state-space set (in this case boundedness of the state trajectory in Ω_ρ) and thus, guarantee closed-loop stability, a second mode is used which is defined by the constraint of (5f). This constraint forces the computed control action by the LEMPC to decrease the Lyapunov function by at least the rate given by the explicit stabilizing controller $h(x)$ for the first sampling period in the prediction horizon. Owing to the properties of the explicit controller $h(x)$ implemented in a sample-and-hold fashion, the Lyapunov function value under the LEMPC operating in mode 2 will decrease at the next sampling period with the constraint (5f).

The LEMPC is implemented in a receding horizon fashion meaning that at each sampling instance, the LEMPC receives a state measurement $x(t_k)$, solves the optimization problem, and sends the control action for the first sampling period of the prediction horizon to be implemented by the control actuators for t_k to t_{k+1}. The process repeats at the next sampling period by rolling the horizon one sampling period into the future. The optimal input trajectory computed by the LEMPC at a given sampling instance t_k is denoted as $u^*(t|t_k)$ and is defined for $t \in [t_k, t_{k+N})$. The control action that is sent at time t_k to the control actuators to be applied over the sampling period from t_k to t_{k+1} is denoted as $u^*(t_k|t_k)$. For more details on LEMPC including the complete closed-loop stability and recursive feasibility proof, the interested reader is referred to Ref. 3.

Remark 2. *A clarification is in order regarding LEMPC. Within chemical process industries, many of the processes are safety critical, and maintaining safe and stable operation is of the highest priority. Given that EMPC may operate a process system in a dynamic fashion to optimize the economics, maintaining the closed-loop state trajectory in a well-defined state-space set, where a certain degree of robustness to uncertainty is achieved, is one method to achieve safe and stable operation under EMPC. This objective is the main motivating factor in designing LEMPC with a two-mode strategy which allows for time-varying operation while maintaining the closed-loop state in Ω_ρ. For points outside of Ω_ρ, no guarantees on stability and robustness can be made a priori. If it is desirable to force the closed-loop state trajectory to the origin at any point over the length of operation, one could force the LEMPC to operate in mode 2 operation only which will steer the closed-loop state to a small neighborhood of the origin. Thus, the two mode strategy of the LEMPC is used for the aforementioned reasons, and the potential use of two-mode operation does not represent two artificial operational stages.*

Real-Time Economic Model Predictive Control

In this section, the formulation and implementation strategy of the real-time LEMPC is presented along with sufficient conditions such that the closed-loop system under the real-time LEMPC renders the closed-loop state trajectory bounded in Ω_ρ.

EMPC formulation

The overall objective of the proposed 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 (5) 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

$$\underset{u\in S(\Delta)}{\text{minimize}} \quad \int_{t_{j+1}}^{t_{j+N}} l_e(\tilde{x}(t), u(t))\ dt \tag{6a}$$

$$\text{subject to} \quad \dot{\tilde{x}}(t) = f(\tilde{x}(t), u(t), 0) \tag{6b}$$

$$\tilde{x}(t_j) = x(t_j) \tag{6c}$$

$$u(t) = \tilde{u}(t_j), \ \forall\ t \in [t_j, t_{j+1}) \tag{6d}$$

$$u(t) \in U, \ \forall\ t \in [t_{j+1}, t_{j+N}) \tag{6e}$$

$$V(\tilde{x}(t)) \leq \rho_e, \ \forall\ t \in [t_{j+1}, t_{j+N}) \tag{6f}$$

where the notation and constraints are similar to that used in LEMPC (5) except for an additional constraint of (6d). 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 (6) 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 the "Stability Analysis" Subsection. 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 proposed real-time implementation strategy, and thus, the real-time LEMPC has a feasible solution (refer to the proof of Theorem 1).

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

$$s_1(k)=\begin{cases} 1, & \text{solving the LEMPC} \\ 0, & \text{not solving the LEMPC} \end{cases} \tag{7}$$

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

$$s_2(k)=\begin{cases} 1, & \text{previous input solution stored} \\ 0, & \text{no previous input solution stored} \end{cases} \tag{8}$$

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 (1) with $w(t) \equiv 0$ 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 Figure 1.

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 Figure 2. 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 $j=0$ 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.

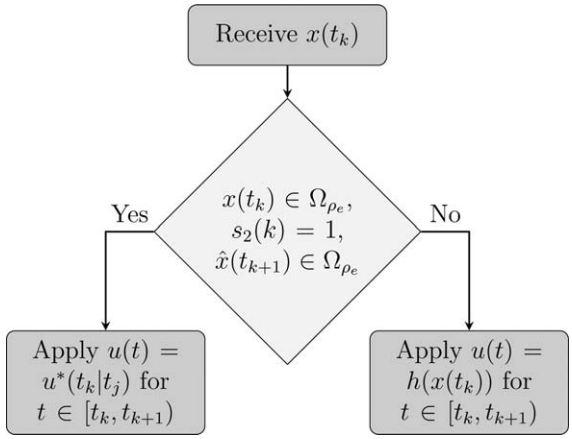

Figure 1. Implementation strategy for determining the control action at each sampling period.

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 (6) solved at time step t_j.

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.

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 can specify the number of sampling periods that the real-time LEMPC may apply a precomputed input trajectory before it must start recomputing 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 proposed real-time LEMPC scheme is given in Figure 3. 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 as 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

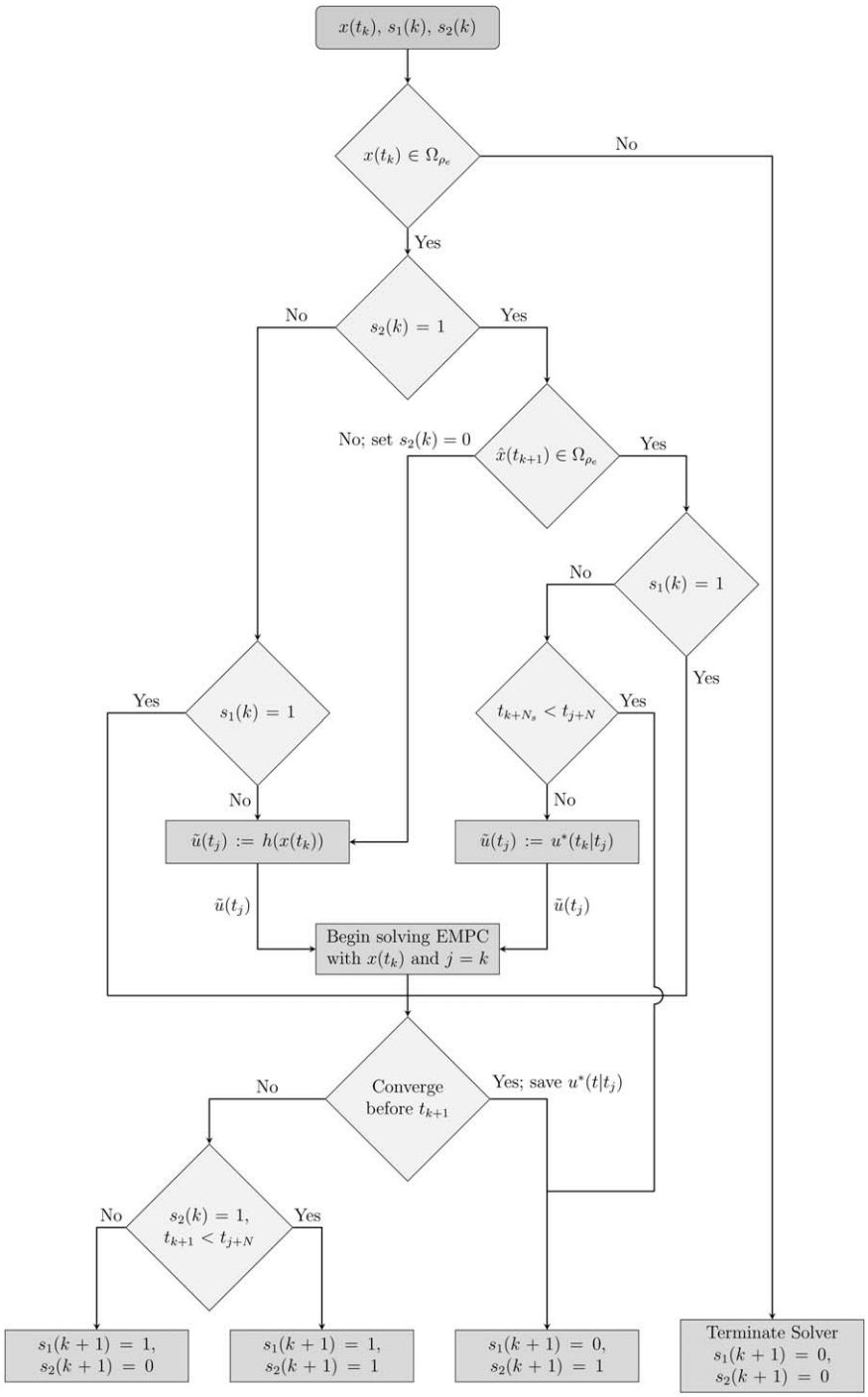

Figure 2. Computation strategy for the real-time LEMPC scheme.

solver returns a solution at t_{13}. At t_{16}, the stability conditions are not satisfied for the precomputed LEMPC input trajectory, so the explicit controller computes a control action and apply it to the system.

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. As $f(\cdot,\cdot,\cdot)$ is a locally Lipschitz vector function and the Lyapunov function $V(\cdot)$ is a continuously differentiable function, there exist positive constants L_x, L_w, $L_x^{'}$, and $L_w^{'}$ such that the following bounds hold

$$|f(x_a,u,w)-f(x_b,u,0)| \le L_x|x_a-x_b|+L_w|w| \tag{9}$$

$$|\frac{\partial V(x_a)}{\partial x}f(x_a,u,w)-\frac{\partial V(x_b)}{\partial x}f(x_b,u,0)| \le L_x^{'}|x_a-x_b|+L_w^{'}|w| \tag{10}$$

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

$$|f(x,u,w)| \le M \tag{11}$$

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

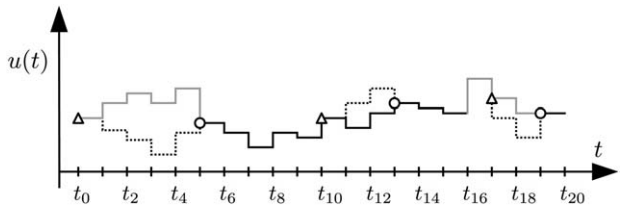

Figure 3. An illustration of an example input trajectory resulting under the real-time LEMPC scheme.

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.

The following proposition bounds the difference between the actual state trajectory of the system (1) ($w(t) \not\equiv 0$) and the nominal state trajectory ($w(t) \equiv 0$).

Proposition 1. (c.f.[36]) *Consider the state trajectories x(t) and $\hat{x}(t)$ with dynamics*

$$\dot{x}(t) = f(x(t), u(t), w(t)), \tag{12}$$

$$\dot{\hat{x}}(t) = f(\hat{x}(t), u(t), 0), \tag{13}$$

input trajectory $u(t) \in U, w(t) \in W$, 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 $\gamma_e(\cdot)$

$$|x(T) - \hat{x}(T)| \leq \gamma_e(T) := \frac{L_w \theta}{L_x}\left(e^{L_x T} - 1\right) . \tag{14}$$

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

Proposition 2. (c.f.[36]) *Consider the Lyapunov function $V(\cdot)$ of the closed-loop system (1) under the controller h(x). There exists a scalar-valued quadratic function $f_V(\cdot)$ such that*

$$V(x_a) \leq V(x_b) + f_V(|x_a - x_b|) \tag{15}$$

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

$$f_V(s) := \alpha_4(\alpha_1^{-1}(\rho))s + \beta s^2 \tag{16}$$

and β is a positive constant.

Theorem 1 below 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 can be handled with the proposed real-time LEMPC methodology. In the case where the computational delay is always greater than the prediction horizon, the proposed real-time LEMPC scheme would return the input trajectory under the explicit controller applied in a sample-and-hold fashion.

Theorem 1. *Consider the system (1) in closed-loop under the real-time LEMPC (6) based on a controller h(x) that satisfies the conditions (4) that is implemented according to the implementation strategy of Figure 1. Let $\varepsilon_w > 0, \Delta > 0$ and $\rho > \rho_e \geq \rho_{\min} \geq \rho_s > 0$ satisfy*

$$-\alpha_3(\alpha_2^{-1}(\rho_s)) + L_x' M \Delta + L_w' \theta \leq -\varepsilon_w/\Delta , \tag{17}$$

$$\rho_{\min} = \max\{V(x(t+\Delta) \mid V(x(t)) \leq \rho_s\} , \tag{18}$$

and

$$\rho_e \leq \rho - f_V(\gamma_e(\Delta)) . \tag{19}$$

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

Proof. If the real-time LEMPC is implemented according to the implementation strategy of Figure 1, the control action to be applied over the sampling period either comes from the precomputed LEMPC input trajectory 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[36] 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 $t \geq 0$ 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 (1) in a sample-and-hold fashion. Let $x(t_k) \in \Omega_\rho \backslash \Omega_{\rho_s}$ for some $\rho_s > 0$ such that the conditions of Theorem 1 are satisfied (i.e., (17)). The explicit controller $h(x)$ computes a control action that has the following property [from condition (4)]

$$\begin{aligned} \frac{\partial V(x(t_k))}{\partial x} f(x(t_k), h(x(t_k)), 0) &\leq -\alpha_3(|x(t_k)|) \\ &\leq -\alpha_3(\alpha_2^{-1}(\rho_s)) \end{aligned} \tag{20}$$

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

$$\begin{aligned} \dot{V}(x(t)) = &\frac{\partial V(x(t_k))}{\partial x} f(x(t_k), h(x(t_k)), 0) \\ &+ \frac{\partial V(x(t))}{\partial x} f(x(t), h(x(t_k)), w(t)) \\ &- \frac{\partial V(x(t_k))}{\partial x} f(x(t_k), h(x(t_k)), 0) \end{aligned} \tag{21}$$

for all $t \in [t_k, t_{k+1})$. From the bound on the time-derivative of Lyapunov function (20), the Lipschitz bound (10), and the bound on the norm of the disturbance vector, the time-derivative of the Lyapunov function can be bounded for $t \in [t_k, t_{k+1})$ as follows:

$$\begin{aligned}\dot{V}(x(t)) &\le -\alpha_3(\alpha_2^{-1}(\rho_s)) \\ &+\left|\frac{\partial V(x(t))}{\partial x} f(x(t), h(x(t_k)), w(t))\right. \\ &\left.-\frac{\partial V(x(t_k))}{\partial x} f(x(t_k), h(x(t_k)), 0)\right| \\ &\le -\alpha_3(\alpha_2^{-1}(\rho_s)) + L_x'|x(t)-x(t_k)| + L_w'|w(t)| \\ &\le -\alpha_3(\alpha_2^{-1}(\rho_s)) + L_x'|x(t)-x(t_k)| + L_w'\theta\end{aligned} \tag{22}$$

for all $t \in [t_k, t_{k+1})$. Taking into account (11) and the continuity of $x(t)$, the following bound can be written for all $t \in [t_k, t_{k+1})$

$$|x(t)-x(t_k)| \le M\Delta \ . \tag{23}$$

From (22) and (23), the following bound can be derived

$$\dot{V}(x(t)) \le -\alpha_3(\alpha_2^{-1}(\rho_s)) + L_x'M\Delta + L_w'\theta \tag{24}$$

for all $t \in [t_k, t_{k+1})$. If the condition (17) is satisfied (i.e., Δ is sufficiently small), then there exists $\varepsilon_w > 0$ such that

$$\dot{V}(x(t)) \le -\varepsilon_w/\Delta \tag{25}$$

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

$$V(x(t)) \le V(x(t_k)), \quad \forall\ t \in [t_k, t_{k+1}), \tag{26}$$

$$V(x(t_{k+1})) \le V(x(t_k)) - \varepsilon_w \ . \tag{27}$$

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 (25). 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 $\rho_{\min} \le \rho$ and $\rho_{\min}$ is defined according to (18). 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, while the notation $\tilde{x}(t)$ will be reserved to denote the predicted state in the LEMPC (6). The predicted state in the LEMPC (6) 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 (6), the optimization problem is always feasible if ρ_e satisfies: $\rho > \rho_e \ge \rho_{\min}$. 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 Figure 1. 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 (6). 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

$$\dot{\hat{x}}(t)=f(\hat{x}(t), h(\hat{x}(t_i)), 0) \tag{28}$$

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 (6) is always feasible for any $x(t_j) \in \Omega_{\rho_e}$.

If the LEMPC is implemented according to the proposed implementation strategy of Figure 1, 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} can be derived from Propositions 1 and 2

$$V(x(t_{k+1})) \le V(\hat{x}(t_{k+1})) + f_V(\gamma_e(\Delta)) \ . \tag{29}$$

As $\hat{x}(t_{k+1}) \in \Omega_{\rho_e}$ and if the condition (19) 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 $t \ge t_0$. □

Remark 3. *No closed-loop performance guarantees can be made with the proposed real-time LEMPC because no performance constraints (e.g., terminal constraints) are 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 of time may be different than that computed by the LEMPC over the same time period. Potentially, one may also use sensitivity-based corrections to the precomputed control actions after receiving state feedback like that used in Refs.* 20 *and* 21 *to improve closed-loop performance (e.g., a first-order correction). However, active set changes must be handled appropriately which may introduce additional on-line computation. A complete and rigorous discussion of this point is beyond the scope of this work. 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 time-varying operating policy dictated by the real-time LEMPC may be substantially better (with respect to the economic cost) than steady state operation which is the case for the chemical process network considered below.*

Remark 4. *In this work, unknown and possibly time-varying computational delay is considered for operation affected by unknown bounded disturbance. If, instead of the proposed computation algorithm, 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. Moreover, the predetermined input trajectory is a strictly open-loop input trajectory and thus, closed-loop stability under the influence of unknown disturbances may become an issue under the resulting controller. Conversely, the proposed real-time LEMPC strategy incorporates a degree of feedback because a state measurement is received at each sampling time. The precomputed control action is checked for closed-loop stability at each sampling period and may be substituted by the control action computed by the back-up controller to ensure stability.

Remark 5. *Regarding handling unreachable set-points (set-points that are not steady states) within LEMPC, recall that the main objective of LEMPC is to optimize the operation within Ω_ρ. A steady state, possibly the optimal steady state, of the model is used when characterizing the stability region Ω_ρ. Thus, to handle unreachable set-points within LEMPC, one can handle them like[39]; that is, use the unreachable set-point in the stage cost (a quadratic cost was used in Ref.* 39*) and define the stability region with respect to optimal steady state (the optimal steady state was used as the terminal constraint in Ref.* 39*).*

Remark 6. *The real-time LEMPC method presented in this work can be also applied with minor modifications to periodically-operated processes. To address this case, a few definitions are needed. As is the case in Ref.* 2 *(generalizing to continuous-time), the dynamic behavior of a nominally operated periodic process evolves with dynamics*

$$\dot{x}^*(t)=f(x^*(t),u^*(t),0) \tag{30}$$

where $x^*(t)$ and $u^*(t)$ are periodic trajectories; that is, if τ is the period, then $x^*(t)$ and $u^*(t)$ satisfy the periodic conditions

$$x^*(t)=x^*(t+\tau)\ , \tag{31a}$$

$$u^*(t)=u^*(t+\tau) \tag{31b}$$

for all t. In the context of periodically-operated processes, the periodic trajectories $x^*(t)$ and $u^*(t)$ will be available a priori and may correspond to the economically optimal operating policy. In Ref. 2*, the trajectory $x^*(t)$ is used in the formulation of a periodic terminal constraint and it was shown that the closed-loop state trajectory under the resulting EMPC will converge to the periodic trajectory when certain properties on the stage cost are satisfied. To satisfy the assumptions on the stage cost, regularization terms (i.e., tracking terms) may need to be added to the stage cost. In this context, the overall control objective may be considered to force the state to track the periodic trajectory. As this is a tracking control problem, an important consideration is whether it is applicable to use EMPC to accomplish this control objective because EMPC with a general stage cost and with or without terminal constraints is not guaranteed in general to force the closed-loop state to track the periodic trajectory.*

If it is applicable to apply EMPC, one could use LEMPC to accomplish the desired control objective. To accomplish this objective, the dynamic model to use in LEMPC is the deviation dynamics: $\dot{z}(t)=\dot{x}(t)-\dot{x}^*(t)$. For this case, the dynamic model is time-varying (i.e., the vector field depends explicitly on time). The assumption of the existence of a Lyapunov function needs to be made for the deviation system and with similar bounds as in (4) (i.e., the bounds should not explicitly depend on time). Furthermore, an explicit controller needs to be assumed that can force the closed-loop system to the periodic trajectory. Using the Lyapunov function and explicit controller in the Lyapunov-based constraints, a mode 2 type constraint like (5f) can be added to the formulation of the real-time LEMPC (6). The real-time LEMPC will force the system to track the periodic trajectory if the resulting mode two constraint is enforced over each sampling period in the prediction horizon. If, conversely, a periodic trajectory is not available a priori, then using LEMPC with a sufficiently long horizon will result in the LEMPC forcing the system along the optimal time-varying trajectory whether the resulting trajectory be a periodic trajectory or some other more general time-varying trajectory (see Ref. 6*, for some closed-loop performance results on EMPC without terminal conditions).*

Remark 7. *As pointed out in the proof of Theorem* 1*, 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 8. *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}.*

Remark 9. *Within the context of EMPC, output feedback may be handled using similar principles used in Refs.* 40 *and* 41 *which use a high-gain observer and moving horizon estimation scheme, respectively, to carry out state estimation. For the case of significant measurement noise, a moving horizon estimation scheme, such as the one designed in Ref.* 41*, may give better estimation performance compared to a high-gain observer which tends to be more sensitive to measurement noise. Regarding the application of real-time EMPC to large-scale systems, a distributed EMPC strategy may be more suited to handle the computation of many control actions for large-scale systems. However, addressing the rigorous design of a distributed EMPC strategy for real-time implementation is beyond the scope of this work.*

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 Figure 4. 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 of the 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 of the 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 assumed to be constant.

Applying first principles, a dynamic model of the process network can be obtained (neglecting the dynamics of the condenser and the solvent) and is given by the following ordinary differential equations (ODEs) (see Table 1 for parameter notation and values)

$$\frac{dT_1}{dt}=\frac{F_{10}}{V_1}T_{10}+\frac{F_r}{V_1}T_3-\frac{F_1}{V_1}T_1-\frac{\Delta H k_0}{\rho_L C_p}e^{-E/RT_1}C_{A1}^2+\frac{Q_1}{\rho_L C_p V_1} \tag{32a}$$

$$\frac{dC_{A1}}{dt}=\frac{F_{10}}{V_1}C_{A10}+\frac{F_r}{V_1}C_{Ar}-\frac{F_1}{V_1}C_{A1}-k_0 e^{-E/RT_1}C_{A1}^2 \tag{32b}$$

$$\frac{dC_{B1}}{dt}=\frac{F_r}{V_1}C_{Br}-\frac{F_1}{V_1}C_{B1}+k_0 e^{-E/RT_1}C_{A1}^2 \tag{32c}$$

$$\frac{dT_2}{dt}=\frac{F_{20}}{V_2}T_{20}+\frac{F_1}{V_2}T_1-\frac{F_2}{V_2}T_2-\frac{\Delta H k_0}{\rho_L C_p}e^{-E/RT_2}C_{A2}^2+\frac{Q_2}{\rho_L C_p V_2} \tag{32d}$$

$$\frac{dC_{A2}}{dt}=\frac{F_{20}}{V_2}C_{A20}+\frac{F_1}{V_2}C_{A1}-\frac{F_2}{V_2}C_{A2}-k_0 e^{-E/RT_2}C_{A2}^2 \tag{32e}$$

$$\frac{dC_{B2}}{dt}=\frac{F_1}{V_2}C_{B1}-\frac{F_2}{V_2}C_{B2}+k_0 e^{-E/RT_2}C_{A2}^2 \tag{32f}$$

$$\frac{dT_3}{dt}=\frac{F_2}{V_3}(T_2-T_3)-\frac{\Delta H_{\text{vap}}\dot{M}_r}{\rho_L C_p V_3}+\frac{Q_3}{\rho_L C_p V_3} \tag{32g}$$

$$\frac{dC_{A3}}{dt}=\frac{F_2}{V_3}C_{A2}-\frac{F_r}{V_3}C_{Ar}-\frac{F_3}{V_3}C_{A3} \tag{32h}$$

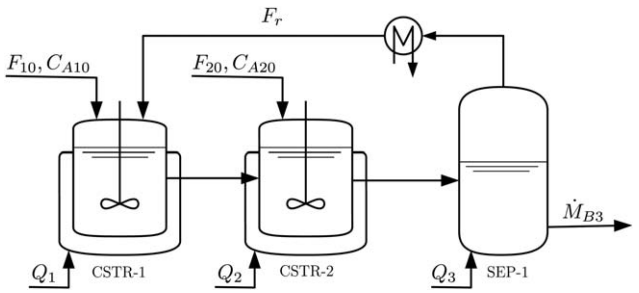

Figure 4. Process flow diagram of the reactor and separator process network.

Table 1. Process Parameters of the Reactor and Separator Process Network

Symbol / Value	Description
$T_{10}=300\,\text{K}$	Temp.: CSTR-1 inlet
$T_{20}=300\,\text{K}$	Temp.: CSTR-2 inlet
$F_{10}=5.0\,\text{m}^3\text{h}^{-1}$	Flow rate: CSTR-1 inlet
$F_{20}=5.0\,\text{m}^3\text{h}^{-1}$	Flow rate: CSTR-2 inlet
$F_r=2.0\,\text{m}^3\text{h}^{-1}$	Flow rate: SEP-1 vapor
$V_1=5.0\,\text{m}^3$	Volume: CSTR-1
$V_2=5.0\,\text{m}^3$	Volume: CSTR-2
$V_3=3.0\,\text{m}^3$	Volume: SEP-1
$k_0=1.9\times10^9\,\text{m}^3\,\text{kmol}^{-1}\text{h}^{-1}$	Pre-exponential factor
$E=7.1\times10^4\,\text{kJ kmol}^{-1}$	Activation energy
$\Delta H=-7.8\times10^3\,\text{kJ kmol}^{-1}$	Heat of reaction
$\Delta H_{\text{vap}}=4.02\times10^4\,\text{kJ kmol}^{-1}$	Heat of vaporization
$C_p=0.231\,\text{kJ kg}^{-1}\,\text{K}^{-1}$	Heat capacity
$R=8.314\,\text{kJ kmol}^{-1}\,\text{K}^{-1}$	Gas constant
$\rho_L=1000\,\text{kgm}^{-3}$	Liquid Solution Density
$\alpha_A=3.0$	Relative volatility: A
$\alpha_B=0.8$	Relative volatility: B
$\alpha_D=1.0$	Relative volatility: D
$MW_A=18\,\text{kg kmol}^{-1}$	Molecular weight: A
$MW_B=18\,\text{kg kmol}^{-1}$	Molecular weight: B
$MW_D=40.0\,\text{kg kmol}^{-1}$	Molecular weight: D

$$\frac{dC_{B3}}{dt}=\frac{F_2}{V_3}C_{B2}-\frac{F_r}{V_3}C_{Br}-\frac{F_3}{V_3}C_{B3} \tag{32i}$$

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

$$C_{D3}=(\rho_L-C_{A3}MW_A-C_{B3}MW_B)/MW_D \tag{33a}$$

$$C_{ir}=\frac{\alpha_i\rho_L C_{i3}}{\sum_{j\in\{A,B,D\}}\alpha_j C_{j3}MW_j},\quad i=A,\ B,\ D \tag{33b}$$

$$\dot{M}_r=F_r(C_{Ar}+C_{Br}+C_{Dr}) \tag{33c}$$

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

$$F_1=F_r+F_{10} \tag{34a}$$

$$F_2=F_1+F_{20} \tag{34b}$$

$$F_3=F_2-F_r \tag{34c}$$

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 $Q_j\in[-1.0,\ 1.0]\times10^5$ kJ h^{-1} for $j=1,\ 2$, $Q_3\in[2.2,\ 2.5]\times10^6$ 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

$$\frac{1}{t_f}\int_{t_0}^{t_f} F_{j0}C_{Aj0}(t)\ dt = 20.0\,\text{kmol h}^{-1} \tag{35}$$

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 can 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). As 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 Ref. 42 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

$$\begin{aligned}
&\max_{x_s,u_s} F_3C_{B3s}\\
&\text{s.t.} f(x_s,u_s)=0\\
&370.0\,\text{K} \le T_{1s} \le 380.0\,\text{K}\\
&370.0\,\text{K} \le T_{2s} \le 380.0\,\text{K}\\
&T_{3s}=390.0\,\text{K}\\
&-1.0\times10^5\,\text{kJ h}^{-1} \le Q_{1s} \le 1.0\times10^5\,\text{kJ h}^{-1}\\
&-1.0\times10^5\,\text{kJh}^{-1} \le Q_{2s} \le 1.0\times10^5\,\text{kJ h}^{-1}\\
&2.2\times10^6\,\text{kJ h}^{-1} \le Q_{3s} \le 2.5\times10^6\,\text{kJ h}^{-1}\\
&C_{A10s}=C_{A20s}=4.0\,\text{kmol m}^{-3}
\end{aligned} \tag{36}$$

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

$$\begin{aligned}
x_s^* &= [\,T_{1s}^* \quad C_{A1s}^* \quad C_{B1s}^* \quad T_{2s}^* \quad C_{A2s}^* \quad C_{B2s}^* \quad T_{3s}^* \quad C_{A3s}^* \quad C_{B3s}^*\,]^T\\
&= [\,380.0 \quad 2.67 \quad 2.15 \quad 380.0 \quad 2.42 \quad 2.06 \quad 390.0 \quad 1.85 \quad 2.15\,]^T,
\end{aligned} \tag{37}$$

and the optimal steady-state input vector is

$$\begin{aligned}
u_s^* &= [\,Q_{1s}^* \quad Q_{2s}^* \quad Q_{3s}^* \quad C_{A10s}^* \quad C_{A20s}^*\,]^T\\
&= \left[\,-4.21\times10^3 \quad 1.70\times10^4 \quad 2.34\times10^6 \quad 4.0 \quad 4.0\,\right]^T.
\end{aligned} \tag{38}$$

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

$$l_e(x,u) = -F_3C_{B3} + \sum_{i=1}^{3} Q_{c,i}(T_i - T_{is}^*)^2 \tag{39}$$

where $T_{is}^*, i=1,\ 2,\ 3$ are the optimal steady-state temperatures. The stage cost (39) 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 of time 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 (see Ref. 4, for discussion on imposing hard and soft state constraints within the context of LEMPC), or adding a contractive constraint on the temperature ODEs (see the example of Ref. 3, for an illustrative example).

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 not violated) under the explicit controller. Specifically, the temperature ODEs are input-affine in the heat rate input and have the form

$$\dot{T}_j = f_j(x) + b_jQ_j \tag{40}$$

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:

$$Q_j = -\frac{1}{b_j}\left(f_j(x) + K_j(T_j - T_{js}^*)\right) \tag{41}$$

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 $h(x)$ was 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

$$V(x) = (x - x_s^*)^T P (x - x_s^*) \tag{42}$$

where P is the following positive definite matrix

$$P = \text{diag}[0.001 \quad 1.5 \quad 0.05 \quad 0.001 \quad 1.5 \quad 0.05 \quad 0.001 \quad 1.5 \quad 0.05]. \tag{43}$$

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 (see Ref. 4 for details on the implementation of this type of constraint). 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, a simultaneous approach is used with collocation discretization of the ODEs, and three Radau collocation points are used per sampling period (see, for instance, Ref. 29 for details on solving a dynamic optimization problem using a simultaneous approach). The open-source nonlinear optimization solver Ipopt[29,43] was used owing to its ability to exploit the high degree of sparsity of the resulting optimization problem. Exact (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.

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

$$J_e = \int_0^{t_f} F_3 C_{B3} \, dt \ . \tag{44}$$

As 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

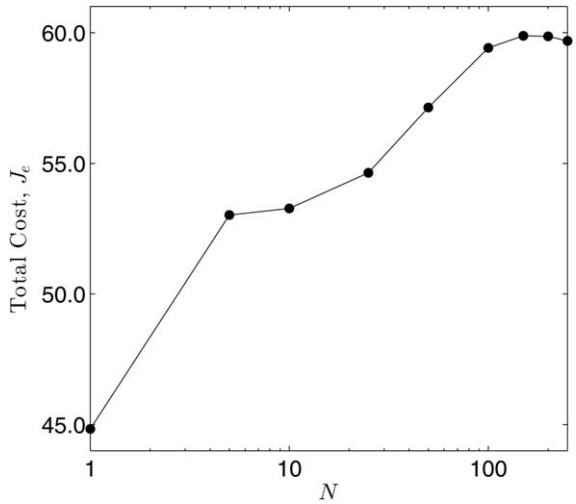

Figure 5. The total economic cost J_e over one operating window length of operation (2.4 h) of the process network under LEMPC with the prediction horizon length.

$$L_e = -\int_0^{t_f} l_e(x, u) \, dt \ . \tag{45}$$

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 (44) plotted against the prediction horizon length is given in Figure 5. 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, was 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

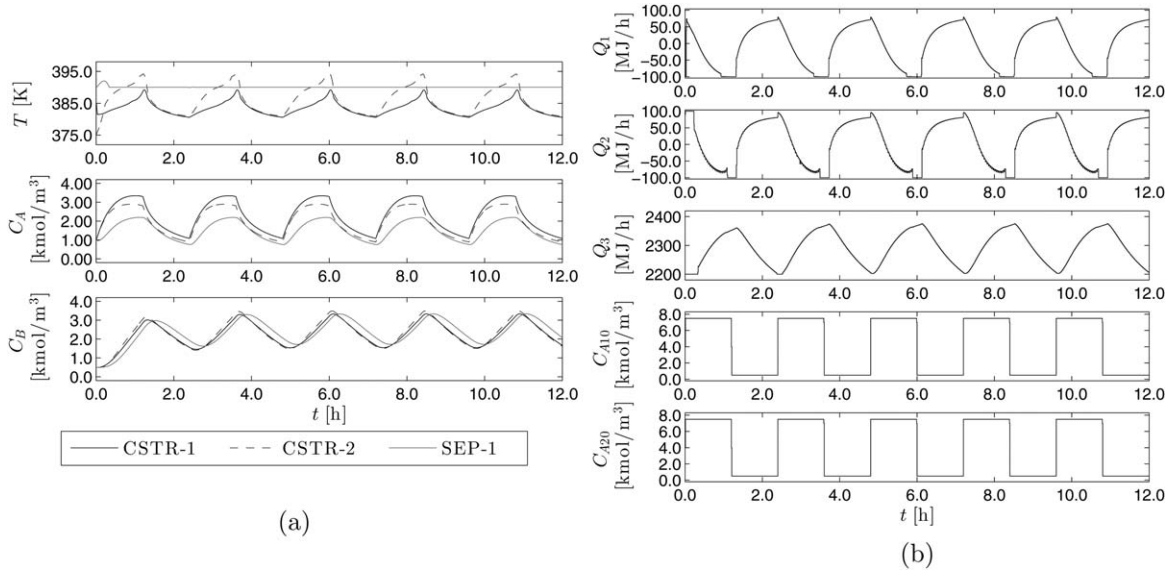

Figure 6. The closed-loop (a) state and (b) input trajectories of the nominally operated process network under the real-time LEMPC scheme.

$t_k + 0.002$ h (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 was simulated. For the real-time LEMPC, 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 was 61.9 s which is almost double the sampling period. The computation time exceeded the sampling period 10 out of the 1200 sampling periods in the simulation. Over the course of both simulations, the closed-loop state was maintained in Ω_ρ. The closed-loop trajectories under the real-time LEMPC scheme are given in Figure 6 (the closed-loop behavior under the LEMPC subject to real-time computational delay was similar). The difference between the performance indices of the two cases was less than 0.5% (the performance indices 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 was realized, it is important to note that an *a priori* guarantee on closed-loop stability under the real-time LEMPC can be made, while, under the LEMPC with computational delay, it is difficult to guarantee closed-loop stability *a priori*. Also, the total on-line computation time to solve the LEMPC over the two simulations was 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, Figure 7 shows the frequency the LEMPC problem was 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 was 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.

As 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 can 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.

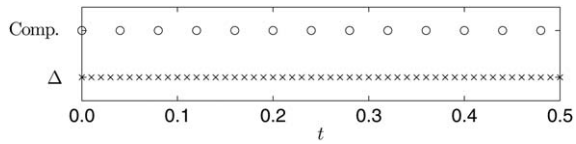

Figure 7. The number of times the LEMPC optimization problem was solved (Comp.) as dictated by the real-time implementation strategy compared to the sampling period (Δ) over the first 0.5 h of operation.

Table 2. 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

	Back-Up Controller		LEMPC		Real-Time LEMPC	
Sim.	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

Several simulations were performed, each starting at a different initial condition, and the performance indices of these simulations are given in Table 2. Applying the back-up explicit controller $h(x)$ implemented in a sample-and-hold fashion to the chemical process network was also considered and the performance indices of these simulations are given in Table 2 as well. The average improvement in economic performance compared to the process network under the back-up controller was 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 did 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 8 and 9 give the closed-loop trajectories of simulation 2 (as labeled in Table 2). The input trajectory computed by the real-time LEMPC has chattering initially over the first operating period because

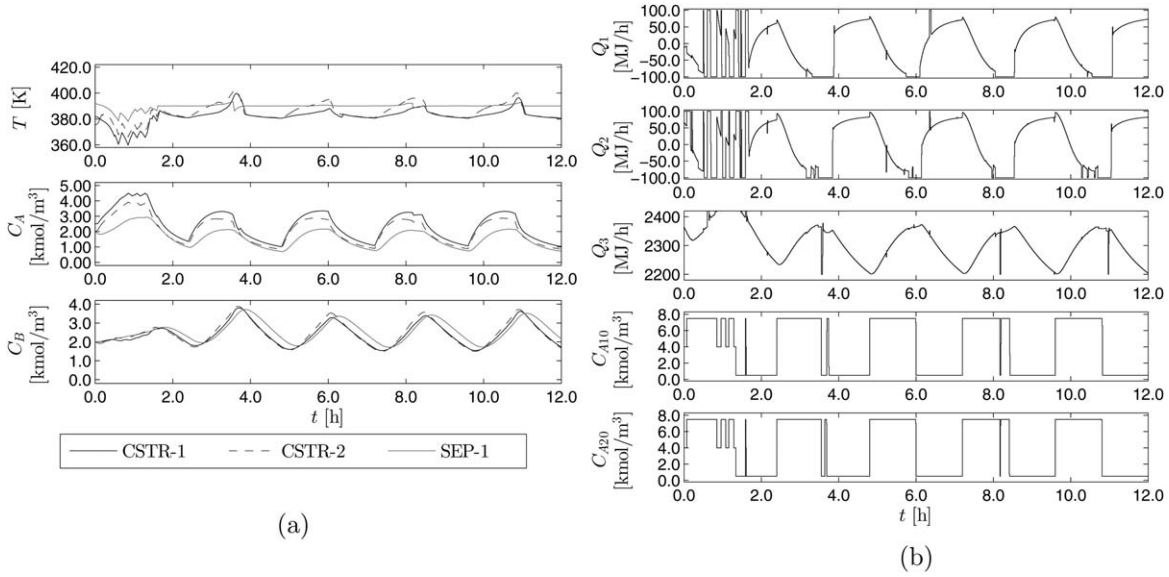

Figure 8. The closed-loop (a) state and (b) input trajectories of process network under the real-time LEMPC scheme where the computational delay is modeled as a bounded random number.

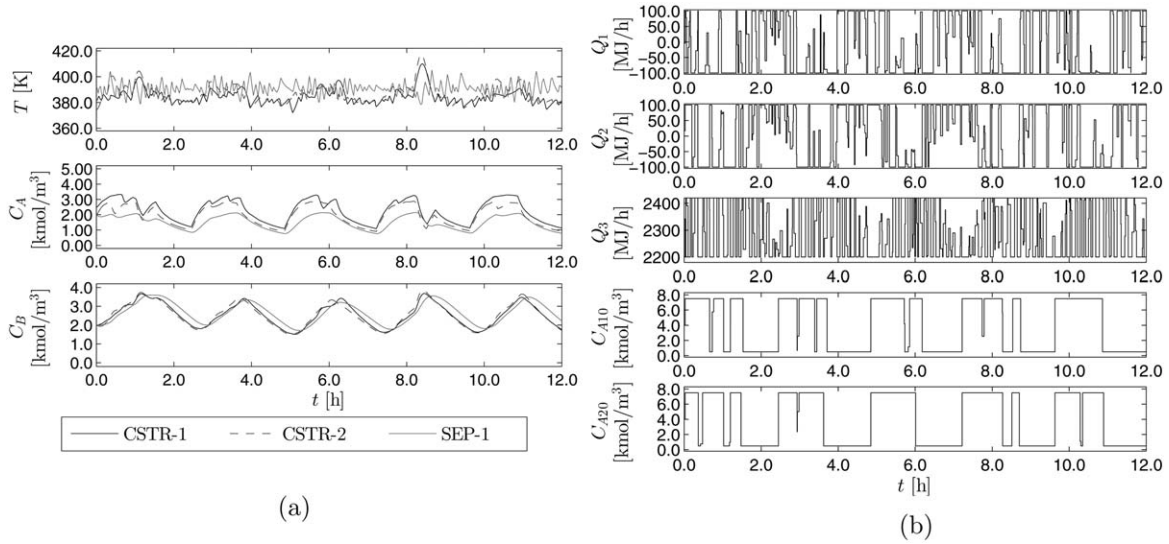

Figure 9. The closed-loop (a) state and (b) input trajectories of process network under LEMPC subject to computational delay where the computational delay is modeled as a bounded random number.

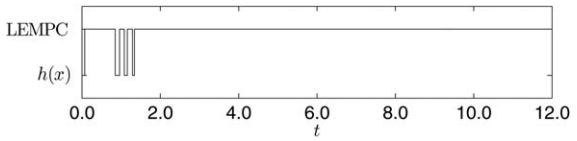

Figure 10. A discrete trajectory depicting when the control action applied to the process network over each sampling period was from a precomputed LEMPC solution or from the back-up controller for the closed-loop simulation of Figure 8.

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. Conversely, chattering in the input profiles 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 indices 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 Figure 8, a discrete trajectory showing when the control action applied to the process network under the real-time LEMPC strategy was from a precomputed LEMPC solution or from the back-up controller is given in Figure 10. For this case, the back-up controller is used 31 out of 1200 sampling periods (2.7% of the sampling periods). From Figure 10, the

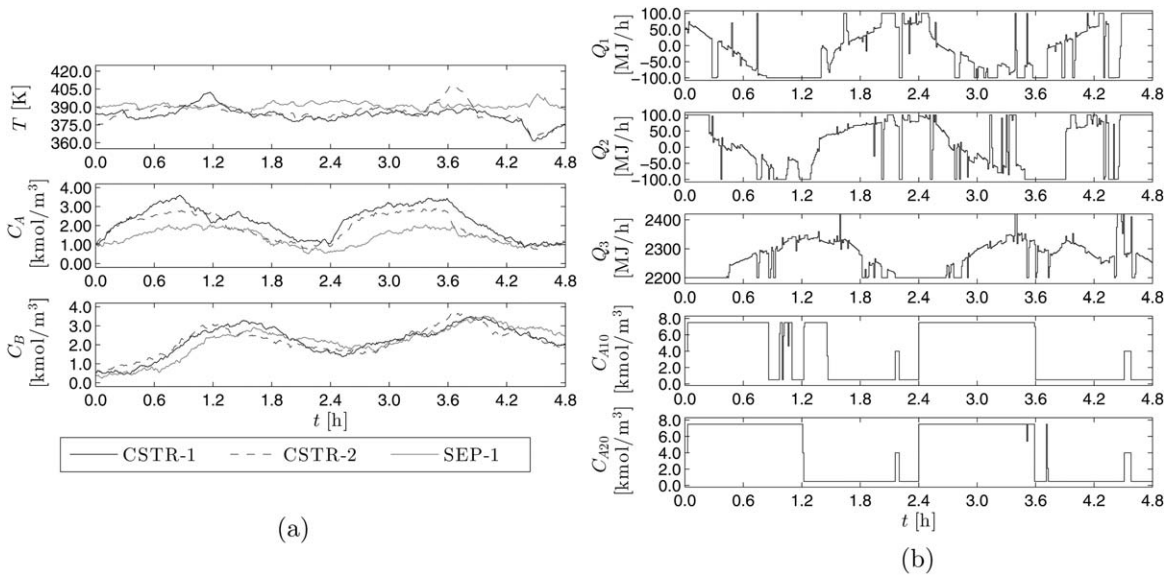

Figure 11. The closed-loop (a) state and (b) input trajectories of process network under the real-time LEMPC scheme with bounded process noise.

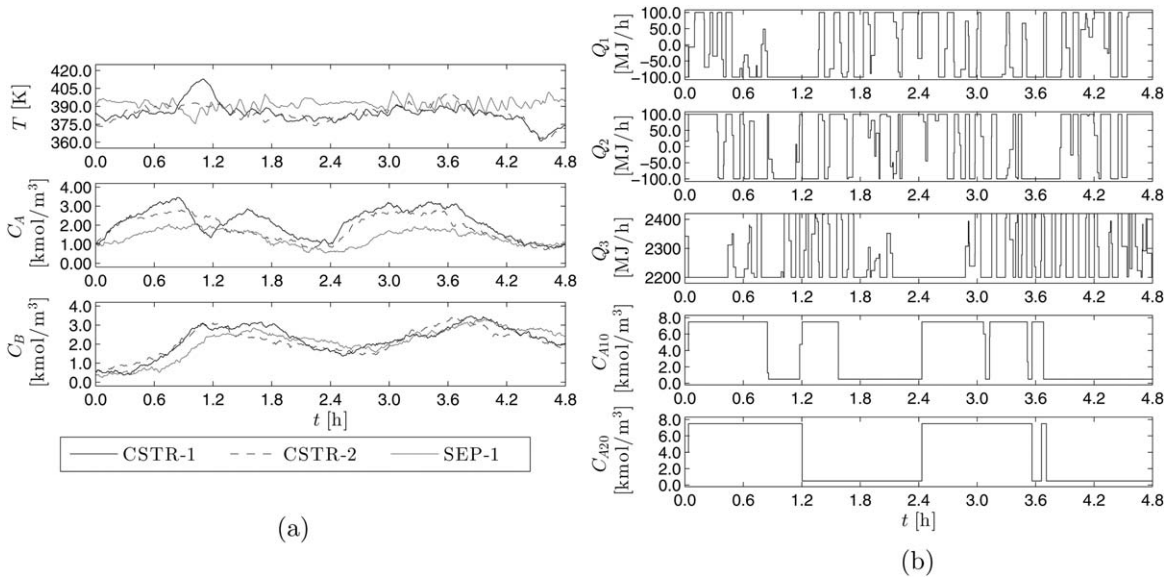

Figure 12. The closed-loop (a) state and (b) input trajectories of process network under LEMPC subject to computational delay with bounded process noise.

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 2) 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 could maintain the closed-loop state inside Ω_ρ.

In the last set of simulations, significant bounded Gaussian process noise with zero mean was added to the model states. The standard deviations of the noise added to the temperature and concentration states were 5.0 and 0.5, respectively, and the bounds on the noise were 2.0 and 15.0, respectively. Two closed-loop simulations over 12.0 h length of operation were completed with the same realization of the process noise. In the first simulation, the process network was controlled by the real-time LEMPC and the closed-loop trajectories are given in Figure 11 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 was controlled by LEMPC subject to computation delay (trajectories shown in Figure 12).

From Figure 12, 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. Conversely, the real-time LEMPC implements a much smoother input trajectory (Figure 11) 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 (Figure 6). 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 indices 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 was also applied to the process network subject to the same realization of the process noise. The economic performance index for this case was 242.3. For operation with process noise, the economic performance improvement over the process network under the back-up controller was 24.4% under the real-time LEMPC strategy and 21.9% under the receding horizon LEMPC for the same initial condition.

Conclusions

In this work, a strategy for implementing LEMPC in real-time with computation delay was proposed. 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. Lastly, the real-time LEMPC scheme was applied to a chemical process network and demonstrated that it can maintain closed-loop stability in the presence of significant computation delay and process noise while also, improving the closed-loop economic performance compared to the economic performance at the economically optimal steady state.

Acknowledgment

Financial support from the National Science Foundation and the Department of Energy is gratefully acknowledged.

Literature Cited

1. Angeli D, Amrit R, Rawlings JB. On average performance and stability of economic model predictive control. *IEEE Trans Automat Control*. 2012;57:1615–1626.
2. Huang R, Harinath E, Biegler LT. Lyapunov stability of economically oriented NMPC for cyclic processes. *J Process Control*. 2011; 21:501–509.
3. Heidarinejad M, Liu J, Christofides PD. Economic model predictive control of nonlinear process systems using Lyapunov techniques. *AIChE J*. 2012;58:855–870.
4. Ellis M, Durand H, Christofides PD. A tutorial review of economic model predictive control methods. *J Process Control*. 2014;24:1156–1178.
5. Müller MA, Angeli D, Allgöwer F. On the performance of economic model predictive control with self-tuning terminal cost. *J Process Control*. 2014;24:1179–1186.
6. Grüne L, Stieler M. Asymptotic stability and transient optimality of economic MPC without terminal conditions. *J Process Control*. 2014;24:1187–1196.
7. Ellis M, Christofides PD. On finite-time and infinite-time cost improvement of economic model predictive control for nonlinear systems. *Automatica*. 2014;50:2561–2569.
8. Zhang J, Liu S, Liu J. Economic model predictive control with triggered evaluations: state and output feedback. *J Process Control*. 2014;24:1197–1206.
9. Lucia S, Andersson JAE, Brandt H, Diehl M, Engell S. Handling uncertainty in economic nonlinear model predictive control: a comparative case study. *J Process Control*. 2014;24:1247–1259.
10. Ma J, Qin SJ, Salsbury T. Application of economic MPC to the energy and demand minimization of a commercial building. *J Process Control*. 2014;24:1282–1291.
11. Touretzky CR, Baldea M. Integrating scheduling and control for economic MPC of buildings with energy storage. *J Process Control*. 2014;24:1292–1300.
12. Cole WJ, Morton DP, Edgar TF. Optimal electricity rate structures for peak demand reduction using economic model predictive control. *J Process Control*. 2014;24:1311–1317.
13. Wang X, Teichgraeber H, Palazoglu A, El-Farra NH. An economic receding horizon optimization approach for energy management in the chlor-alkali process with hybrid renewable energy generation. *J Process Control*. 2014;24:1318–1327.
14. Liu J, Muñoz de la Peña D, Christofides PD, Davis JF. Lyapunov-based model predictive control of nonlinear systems subject to time-varying measurement delays. *Int J Adapt Control Signal Process*. 2009;23:788–807.
15. Karafyllis I, Krstic M. Nonlinear stabilization under sampled and delayed measurements, and with inputs subject to delay and zero-order hold. *IEEE Trans Automat Control*. 2012;57:1141–1154.
16. Ronco E, Arsan T, Gawthrop PJ. Open-loop intermittent feedback control: practical continuous-time GPC. *IEE Proc—Control Theory Appl*. 1999;146:426–434.

17. Chen WH, Ballance DJ, O'Reilly J. Model predictive control of nonlinear systems: computational burden and stability. *IEE Proc—Control Theory Appl*. 2000;147:387–394.
18. Findeisen R, Allgöwer F. Computational delay in nonlinear model predictive control. In: *Proceedings of the IFAC International Symposium of Advanced Control of Chemical Processes*. Hong Kong, 2004;427–432.
19. Scokaert POM, Mayne DQ, Rawlings JB. Suboptimal model predictive control (feasibility implies stability). *IEEE Trans Automat Control*. 1999;44:648–654.
20. Würth L, Hannemann R, Marquardt W. Neighboring-extremal updates for nonlinear model-predictive control and dynamic real-time optimization. *J Process Control*. 2009;19:1277–1288.
21. Zavala VM, Biegler LT. The advanced-step NMPC controller: optimality, stability and robustness. *Automatica*. 2009;45:86–93.
22. Ganesh N, Biegler LT. A reduced Hessian strategy for sensitivity analysis of optimal flowsheets. *AIChE J*. 1987;33:282–296.
23. Yang X, Biegler LT. Advanced-multi-step nonlinear model predictive control. *J Process Control*. 2013;23:1116–1128.
24. Jäschke J, Yang X, Biegler LT. Fast economic model predictive control based on NLP-sensitivities. *J Process Control*. 2014;24:1260–1272.
25. Würth L, Hannemann R, Marquardt W. A two-layer architecture for economically optimal process control and operation. *J Process Control*. 2011;21:311–321.
26. Wolf IJ, Muñoz DA, Marquardt W. Consistent hierarchical economic NMPC for a class of hybrid systems using neighboring-extremal updates. *J Process Control*. 2014;24:389–398.
27. Diehl M, Bock H, Schlöder J. A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM J Control Optim*. 2005;43:1714–1736.
28. Diehl M, Ferreau HJ, Haverbeke N. Efficient numerical methods for nonlinear MPC and moving horizon estimation. In: Magni L, Raimondo DM, Allgöwer F, editors. Nonlinear Model Predictive Control, *Vol. 384: Lecture Notes in Control and Information Sciences*, Berlin, Heidelberg: Springer-Verlag, 2009:391–417.
29. Biegler LT. *Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes*. Philadelphia, PA: SIAM, 2010.
30. Tabuada P. Event-triggered real-time scheduling of stabilizing control tasks. *IEEE Trans Automat Control*. 2007;52:1680–1685.
31. Massera JL. Contributions to stability theory. *Ann Math*. 1956;64:182–206.
32. Khalil HK. *Nonlinear Systems*, 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2002.
33. Kravaris C, Kantor JC. Geometric methods for nonlinear process control. 2. Controller synthesis. *Ind Eng Chem Res*. 1990;29:2310–2323.
34. Kokotović P, Arcak M. Constructive nonlinear control: a historical perspective. *Automatica*. 2001;37:637–662.
35. Christofides PD, El-Farra NH. *Control of Nonlinear and Hybrid Process Systems: Designs for Uncertainty, Constraints and Time-Delays*. Berlin, Germany: Springer-Verlag, 2005.
36. Muñoz de la Peña D, Christofides PD. Lyapunov-based model predictive control of nonlinear systems subject to data losses. *IEEE Trans Automat Control*. 2008;53:2076–2089.
37. Dubljević S, Kazantzis N. A new Lyapunov design approach for nonlinear systems based on Zubov's method. *Automatica*. 2002;38:1999–2007.
38. Papachristodoulou A, Prajna S. On the construction of Lyapunov functions using the sum of squares decomposition. In: *Proceedings of the 41st IEEE Conference on Decision and Control*, Vol. 3. Las Vegas, NV, 2002;3482–3487.
39. Rawlings JB, Bonné D, Jørgensen JB, Venkat AN, Jørgensen SB. Unreachable setpoints in model predictive control. *IEEE Trans Automat Control*. 2008;53:2209–2215.
40. Heidarinejad M, Liu J, Christofides PD. State-estimation-based economic model predictive control of nonlinear systems. *Syst Control Lett*. 2012;61:926–935.
41. Ellis M, Zhang J, Liu J, Christofides PD. Robust moving horizon estimation based output feedback economic model predictive control. *Syst Control Lett*. 2014;68:101–109.
42. Kookos IK, Perkins JD. An algorithmic method for the selection of multivariable process control structures. *J Process Control*. 2002;12:85–99.
43. Wächter A, Biegler LT. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math Program*. 2006;106:25–57.

Manuscript received Aug. 26, 2014, and revision received Oct. 16, 2014.