Integrating Data-Based Modeling and Nonlinear Control Tools for Batch Process Control

Siam Aumi and Prashant Mhaskar
Dept. of Chemical Engineering, McMaster University, Hamilton, ON, Canada L8S 4L7

DOI 10.1002/aic.12720
Published online August 8, 2011 in Wiley Online Library (wileyonlinelibrary.com).

A data-based multimodel approach is developed in this work for modeling batch systems in which multiple local linear models are identified using latent variable regression and combined using an appropriate weighting function that arises from fuzzy c*-means clustering. The resulting model is used to generate empirical reverse-time reachability regions (RTRRs) (defined as the set of states from where the data-based model can be driven inside a desired end-point neighborhood of the system), which are subsequently incorporated in a predictive control design. Simulation results of a fed-batch reactor system under proportional-integral (PI) control and the proposed RTRR-based design demonstrate the superior performance of the RTRR-based design in both a fault-free and faulty environment. The data-based modeling methodology is then applied on a nylon-6,6 batch polymerization process to design a trajectory tracking predictive controller. Closed-loop simulation results illustrate the superior tracking performance of the proposed predictive controller over PI control.*
© 2011 American Institute of Chemical Engineers *AIChE J,* 58: 2105–2119, 2012
Keywords: process control, control, optimization, batch control, data-based modeling

Introduction

Batch and fed-batch processes play an important role in the production of high-quality, low-volume products such as bio-chemicals and polymers. Typical examples include fermentation, crystallization, and emulsion polymerization. The main control objective in batch processes is to meet final product quality specifications. Key characteristics of batch processes, such as the absence of equilibrium conditions coupled with strong nonlinear and time-varying dynamics, complicate the batch control problem and limit the control performance associated with the implementation of control approaches developed for continuous processes (characterized by control at a steady-state). Another operational issue with batch process control is the occurrence of faults that can ruin an entire batch or even damage expensive equipment. The ability to handle faults, therefore, is an intrinsic requirement of the modeling (maintaining model validity for a wide range of operating conditions) and control design.

The first of the two main approaches in the development of models (for controlling systems at steady states) includes the use of first-principles to derive (typically) ordinary differential equations with some parameters to be determined from experimental data (the so-called deterministic or mechanistic model development). The second approach (the so-called empirical model development) typically uses a much simpler model structure (often linear) and determines the model parameters entirely from experimental data. One of the limitations with the development of mechanistic models is the lack of sufficient measurements to uniquely determine the key model parameters, and even when available, many of the simplifying assumptions taken during model development may be violated in specific situations in practice. Identification experiments (to build empirical models), such as those in which a pseudo-random binary signal (PRBS) is applied on the process, while suitable for identification at steady states, are often too expensive to justify for batch systems since they result in expensive wasted batches. Furthermore, batch process behavior is highly nonlinear and time-varying, making conventional system identification approaches, where a single linear model is identified, ill-suited for identifying accurate dynamic models. The high expenses associated with every batch dictate the need for the development of dedicated modeling tools for batch processes that minimize wasted batches in the model-development process and yet provide a model that captures the essential nonlinear and complex nature of the process.

In response to this, input–output based multimodel approaches, such as piece-wise affine (PWA) models, have been developed in which multiple linear models are used to capture local dynamics in the data and then combined, yielding a single model capable of describing the major nonlinearities. Note however that PWA models are inherently discontinuous since the identification of the correct cluster is a discrete decision. In the Takagi, Sugeno, and Kang (TSK) modeling approach, the state-input space is not partitioned using clustering algorithms, but instead, each input variable is individually partitioned using fuzzy rule sets thereby ignoring state-input interactions, and a weighted combination

Correspondence concerning this article should be addressed to P. Mhaskar at mhaskar@mcmaster.ca.

of the models (based on so-called membership functions) is used for prediction.

The availability of measurements (beyond the designated inputs and outputs), and possible correlations among the data has motivated the application of latent variable modeling methods, particularly partial least squares (PLS) regression, to identify dynamic batch process models, albeit limited by the assumption of a linear relationship between the variables. In the approaches to incorporate nonlinear relationships, the model's predictive capability depends on the choice of the nonlinear mapping (quadratic functions, neural networks, etc). Another example of a modeling technique (where the clustering is done essentially along time points, or appropriate markers, in the batch duration) is that using principal component analysis (PCA) models, where the process is described via deviations from a nominal process trajectory.[1]

Existing batch control approaches (model-based or otherwise) either specify the desired end point quality indirectly, through first determining a process trajectory that terminates at the desired end-point and then using classical or advanced control tools (such as model predictive control (MPC)) to track the desired trajectory, or directly in an end-point based MPC framework. The inapplicability of step test based tuning techniques in batch processes negatively impacts the control performance of trajectory tracking approaches using proportional-integral-derivative controllers. In trajectory tracking predictive control approaches,[1–8] with modeling errors and process noise, a process trajectory deemed optimal in off-line calculations may be significantly sub-optimal, or even infeasible in online implementation. In end-point based MPC, a dynamic optimization (DO) problem that incorporates the final product quality specifications (in the objective function and/or constraints) is solved at each sampling instance until batch termination. In these approaches, as the input trajectory from the current time to the end of the batch has to be optimized, the size of the optimization problem can become intractable specifically during the early stages in the batch. Input parametrization techniques[9,10] constitute a fairly recently developed method for making nonlinear dynamic optimization problems computationally tractable for real-time application. The impact of plant-model mismatch and disturbances, however, remains significant in this approach (owing, in part, to use of approximate first-principles models with parametric errors and un-modeled disturbances).

The majority of the existing work addressing faults in batch systems focuses primarily on fault detection and isolation while the design of explicit fault tolerant control structures (FTCS) for batch systems have received limited attention. Existing FTCS for batch processes are mostly robust control designs that treat faults as disturbances. However, upon fault occurrence in a batch system, the final product quality may become unreachable if the fault is not repaired sufficiently fast. Additionally, implementing an input trajectory prescribed by controllers with limited fault-tolerant properties can drive the system states to a point from where the final quality is not reachable even if the fault is repaired. In response to these issues, we developed a control and safe-steering framework[11] that utilized a first-principles model and presented a computationally efficient MPC design that addressed the problem of determining how to utilize functioning inputs during fault rectification to enable desired product properties reachability following fault repair. The proposed design represents a computationally efficient framework that is amenable for integration with appropriately derived data-based models for control of batch processes.

Motivated by the above considerations, this work considers the problem of designing an integrated framework seamlessly merging data-based models with nonlinear control tools for control of batch processes. The rest of this manuscript is organized as follows: First, the class of processes considered is presented followed by reviews of the auto-regressive exogenous (ARX) type modeling approach, PLS regression, and the clustering algorithm used in the proposed multi-model approach. Next, a framework is presented for developing a data-based model for batch processes that makes use of all existing measurement and captures the nonlinear nature of the batch. In this modeling approach, a database of previous batch trajectories is initially clustered into a number of operating regions, a weighting scheme is then devised for the training data that can also be utilized to appropriately weight the local models given an initial condition and input, and finally, local linear models are estimated simultaneously using partial least squares (PLS) regression. The resulting model is then incorporated within the RTRR-based MPC and safe-steering framework. Specifically, a methodology and algorithm are presented to generate RTRRs using the data-based model for their subsequent use within a RTRR-based MPC design for batch processes. Then, simulation results of a fed-batch reactor system and a nylon-6,6 batch polymerization system, subject to sensor noise, disturbances, and time-varying parameters, are presented to demonstrate the effectiveness and applicability of the proposed modeling and control approach in the presence of limited measurements. Finally, we summarize our results.

Preliminaries

In this section, we first describe the class of batch processes considered. Then, we give an overview of Auto-Regressive eXogenous (ARX) type modeling, a popular technique for developing linear input-output models of dynamic systems, and then illustrate how multivariate regression tools, such as partial least squares (PLS) regression, can be used within the ARX type modeling framework to utilize all available measurement data (beyond just the input/output data). Then, we review fuzzy *c*-means clustering and the concept of reverse-time reachability regions (RTRRs).

Process description

We consider batch process systems subject to input constraints and failures described by:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}_\sigma) \\ \boldsymbol{y} &= \boldsymbol{g}(\boldsymbol{x}) \\ t \in [t_0, t_f], &\ \boldsymbol{u}_\sigma(\cdot) \in \mathcal{U}_\sigma, \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \end{aligned} \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ denotes the vector of state variables, $\boldsymbol{y} \in \mathbb{R}^p$ denotes the vector of measurable output variables, and $\boldsymbol{u}_\sigma \in \mathbb{R}^m$ denotes the vector of constrained manipulated inputs, taking values in a non-empty convex subset $\mathcal{U}_\sigma$ of $\mathbb{R}^m$, where $\mathcal{U}_\sigma = \{\boldsymbol{u} \in \mathbb{R}^m | \boldsymbol{u}_{\min,\sigma} \leq \boldsymbol{u} \leq \boldsymbol{u}_{\max,\sigma}\}$, where $\boldsymbol{u}_{\min,\sigma}, \boldsymbol{u}_{\max,\sigma} \in \mathbb{R}^m$ denote the constraints on the manipulated inputs. The times t_0 and t_f denote the initial time and batch termination times, respectively. The variable, $\sigma \in \{1, 2\}$, is a discrete variable that indexes fault-free and faulty operation with $\sigma = 1$

signifying fault-free operation and $\sigma = 2$ signifying faulty operation. The fault scenarios considered in this work involve actuator failure resulting in reduced available control effort for a finite duration of time lasting from the time of fault occurrence, t^{fault}, to the time of fault repair, t^{repair}. Throughout the manuscript, we assume that for any $\boldsymbol{u}(\cdot) \in \mathcal{U}_{\sigma}$, the solution of the batch system of Eq. 1 exists and is continuous for all $t \in [t_0, t_f]$.

ARX models

In the ARX type modeling approach, the process outputs at a specific sampling instance are assumed to depend linearly on the previous process conditions (defined by the process outputs and inputs). Mathematically, ARX models are defined as:

$$\boldsymbol{y}(k) = \sum_{i=1}^{n_y} \mathbf{A}_i \boldsymbol{y}^{\mathrm{T}}(k-i) + \sum_{j=1}^{n_u} \mathbf{B}_j \boldsymbol{u}^{\mathrm{T}}(k-j) + \boldsymbol{v}(k) \tag{2}$$

where $\boldsymbol{y}(k) \in \mathbb{R}^p$ and $\boldsymbol{u}(k) \in \mathbb{R}^m$ are the process output and input vectors at sampling instant k (respectively), $\mathbf{A}_i \in \mathbb{R}^{p \times p}$ and $\mathbf{B}_i \in \mathbb{R}^{p \times m}$ denote the model coefficient matrices, $1\boldsymbol{v}(k) \in \mathbb{R}^p$ is the noise vector, and n_y and n_u denote the (maximum) number of time lags in the outputs and inputs (respectively) and define the order of the ARX model. For outputs which do not require the maximum number of lags, the appropriate elements in $\mathbf{A}_i$ and $\mathbf{B}_j$ can be set to zero. In the case of full state measurements, $n_y = n_u = 1$ is a natural choice, which then results in the state-space system described by:

$$\boldsymbol{x}(k) = \mathbf{A}\boldsymbol{x}(k-1) + \mathbf{B}\boldsymbol{u}(k-1) \tag{3}$$

To facilitate the estimation of the model coefficient matrices, Eq. 2 can be rewritten in matrix form as follows (for a given n_y and n_u):

$$\boldsymbol{y}(k) = \beta \bar{\boldsymbol{x}}(k) + \boldsymbol{v}(k) \tag{4}$$

where $\beta = [\,\mathbf{A}_1 \ \cdots \ \mathbf{A}_{n_y} \ \ \mathbf{B}_1 \ \cdots \ \mathbf{B}_{n_u}\,]$ (appropriately zero padded) collects the model coefficient matrices and $\bar{\boldsymbol{x}}^{\mathrm{T}}(k) = [\boldsymbol{y}^{\mathrm{T}}(k-1) \cdots \boldsymbol{y}^{\mathrm{T}}(k-n_y) \boldsymbol{u}^{\mathrm{T}}(k-1) \cdots \boldsymbol{u}^{\mathrm{T}}(k-n_u)]$ is a vector of lagged concatenated outputs and inputs. Given plant data, a response matrix, $\mathbf{Y}$, and regressor matrix, $\overline{\mathbf{X}}$, can be constructed corresponding to $\boldsymbol{y}(k)$ and $\bar{x}(k)$ (respectively) in Eq. 4 by sorting the data sample-wise, and the model parameters can be estimated using partial least squares (PLS) regression to account for any co-linearity and correlation in the data due to the variables describing the same underlying phenomena in the process or because of the data being collected under closed-loop conditions.[12–14]

Mathematically, PLS regression consists of decomposing $\overline{\mathbf{X}}$ and $\mathbf{Y}$ as the sum of the outer products of a score and loading vector:

$$\mathbf{X} = \sum_{j=1}^{A} \boldsymbol{t}_j \boldsymbol{p}_j^{\mathrm{T}} + \mathbf{E} = \mathbf{T}\mathbf{P}^{\mathrm{T}} + \mathbf{E} \tag{5}$$

$$\mathbf{Y} = \sum_{j=1}^{A} \boldsymbol{r}_j \boldsymbol{q}_j^{\mathrm{T}} + \mathbf{F} = \mathbf{R}\mathbf{Q}^{\mathrm{T}} + \mathbf{F} \tag{6}$$

where $\boldsymbol{t}_j$ and $\boldsymbol{r}_j$ are the input and output scores representing the projections of the variables in $\overline{\mathbf{X}}$ and $\mathbf{Y}$ on the subspaces, $\boldsymbol{p}_j$ and $\boldsymbol{q}_j$ define the orientation of the corresponding subspaces, the matrices, $\mathbf{T}$, $\mathbf{P}$, $\mathbf{R}$, and $\mathbf{Q}$, simply contain their corresponding vectors, and $\mathbf{E}$ and $\mathbf{F}$ denote residual matrices. Because it is desired to obtain a useful relationship between the original data matrices, $\overline{\mathbf{X}}$ and $\mathbf{Y}$, the two matrices are linked by an inner relation between their scores of the form:

$$\boldsymbol{r}_j = \boldsymbol{b}_j \boldsymbol{t}_j + \boldsymbol{e}_j$$

where $\boldsymbol{b}_j$ are the coefficients of the inner relationship and $\boldsymbol{e}_j$ are the residuals. In common PLS algorithms, such as nonlinear iterative partial least squares (NIPALS), the subspace orientation and scores for both matrices are determined simultaneously so as to maximize the correlation between $\overline{\mathbf{X}}$ and $\mathbf{Y}$ and therefore obtain the optimal fit for the inner relationship.[15,16] The final result from PLS regression is a linear model between $\overline{\mathbf{X}}$ and $\mathbf{Y}$ where the coefficients are functions of the scores and loadings from the matrix decompositions:

$$\mathbf{Y} = \mathbf{X}\beta_{\text{PLS}} + \mathbf{G}$$

where $\mathbf{G}$ denotes the residuals and $\beta_{\text{PLS}} = f(\mathbf{P},\mathbf{T},\mathbf{Q})$.

Fuzzy c*-means clustering*

Prior to developing multiple dynamic PLS models, an important step in the proposed multi-model approach is to first locate the operating points around which the individual local linear models will be identified. One approach to find this set of operating points is to partition the historical batch database into a number of clusters (i.e., a group of points in the database that are mathematically similar). Subsequently, a corresponding linear model can be identified for each cluster. For the current work, we employ fuzzy c-means clustering[17] to partition the database (the results are not limited to this particular type of clustering; any other meaningful clustering algorithm can be used instead). Assuming full state measurements in a given batch database (see the simulation example for the case of limited availability of measurements), let $\overline{\mathbf{X}}^{\mathrm{T}} = [\,\bar{\boldsymbol{x}}_1 \ \cdots \ \bar{\boldsymbol{x}}_i \ \cdots \ \bar{\boldsymbol{x}}_N\,]$ be a matrix of $n+m$ rows and N columns where each column is a different instance of $[\boldsymbol{x}^{\mathrm{T}}(k)\ \boldsymbol{u}^{\mathrm{T}}(k)]$ (vertically concatenated states and inputs at sampling instant k.) The state-input space in $\overline{\mathbf{X}}^{\mathrm{T}}$ can be partitioned into L different clusters using fuzzy clustering, which assigns each sample, $\bar{x}_i$, a degree of belonging to a cluster $\ell \in [1, L]$. The partition information can be represented by an L by N membership matrix, $\mathbf{U} = \{\mu_{\ell,i}\}$, where each row contains the membership information for ℓ-th cluster for all N points. In fuzzy clustering, the elements in $\mathbf{U}$ must satisfy the following conditions:[18]

$$\mu_{\ell,i} \in [0,1], \quad \text{for } 1 \le \ell \le L \quad 1 \le i \le N \tag{7}$$

$$\sum_{\ell=1}^{N} \mu_{\ell,i} = 1, \quad \text{for } 1 \le i \le N \tag{8}$$

$$0 < \sum_{i=1}^{L} \mu_{\ell,i} < N, \quad \text{for } 1 \le \ell \le L \tag{9}$$

Equation 8 requires that the total membership of each observation (which ranges from 0 to 1) equals one. The cluster centers (based on minimizing the total variance in the data from cluster centers) are obtained by minimizing the following objective function (the so-called c-means functional)[17,19]:

$$J = \sum_{i=1}^{N} \sum_{\ell=1}^{L} (\mu_{\ell,i})^f ||\bar{\boldsymbol{x}}_i^{\mathrm{T}} - \boldsymbol{c}_\ell||^2 \tag{10}$$

where $\boldsymbol{c}_\ell \in \mathbb{R}^{n+m}$ denote the cluster center vectors, which have to be determined. The weighting exponent parameter, f, determines the fuzziness of the resulting clusters with $f = 1$ implying hard, non-overlapping partitions. For this work, (as is typically the case) we choose $f = 2$. The partition matrix elements, $\mu_{\ell,i}$, and cluster centers, $\boldsymbol{c}_\ell$, that minimize the objective function and satisfy the constraints in Eqs. 7–9 have been shown to be (for $f = 2$)[17,19]:

$$\mu_{\ell,i} = \frac{\|\bar{\boldsymbol{x}}_i^{\mathrm{T}} - \boldsymbol{c}_\ell\|^{-2}}{\sum_{\ell=1}^{L} \|\bar{\boldsymbol{x}}_i^{\mathrm{T}} - \boldsymbol{c}_\ell\|^{-2}} \tag{11}$$

and

$$\boldsymbol{c}_\ell = \frac{\sum_{i=1}^{N} \mu_{i,\ell}^2 \bar{\boldsymbol{x}}_i^{\mathrm{T}}}{\sum_{i=1}^{N} \mu_{i,\ell}^2} \tag{12}$$

where $\left\|\bar{x}_i^{\mathrm{T}} - c_\ell\right\|$ denotes the Euclidean distance between point i and the ℓ-th cluster center. From Eq. 12, it can be seen that the center point of each cluster is also the mean of all the points, weighted by their membership degrees. In view of this, in fuzzy clustering, the degree of $\bar{x}_i$ belonging to cluster ℓ is taken to be inversely proportional to the squared distance between the point and cluster center, $\boldsymbol{c}_\ell$ (i.e., $\mu_{\ell,i} \propto \left\|\bar{\boldsymbol{x}}_i^{\mathrm{T}} - c_\ell\right\|^{-2}$), which is then normalized across all clusters.

Remark 1. In the fuzzy clustering done for this work, points that are mathematically "similar" according to the Euclidean 2-norm are clustered, resulting in (overlapping) $n + m$-dimensional spherical clusters. To account for the different variances in the directions of the coordinate axes of $\overline{\mathbf{X}}$, each variable can be scaled to unit variance through dividing by its standard deviation. This is equivalent to weighting the norm used in the clustering algorithm by an appropriate diagonal matrix (i.e., a diagonal matrix comprising the inverse variances of each variable). To accommodate non-spherical clusters, extensions of fuzzy c-means clustering that consider different norms or different shapes can be utilized as well.[20,21]

Remark 2. The number of clusters in fuzzy c-means clustering is the most important parameter in the algorithm. Well-defined criteria (based on the cluster geometry) to iteratively refine the number of clusters have been presented,[22–24] and to evaluate the goodness of the final fuzzy partitions, many validation measures have also been introduced.[23,25] Among these, the Xie-Beni index,[25] which is a ratio of the total within-cluster variance to the separation of the cluster centers (and therefore should be minimal for the best partition), has been found to perform well in practice.

Remark 3. In the absence of full state measurements, the columns in $\overline{\mathbf{X}}^{\mathrm{T}}$ are replaced by concatenated lagged outputs and inputs: $\left[\boldsymbol{y}^{\mathrm{T}}(k-1) \quad \cdots \quad \boldsymbol{y}^{\mathrm{T}}(k-n_y)\boldsymbol{u}^{\mathrm{T}}(k-1)\cdots\boldsymbol{u}^{\mathrm{T}}(k-n_u)\right]^{\mathrm{T}}$. In this case, the dimension of the space required to be clustered is $(n_y \times p) + (n_u + m)$, which may be prohibitively large for computation. In this work, this high dimensionality problem was addressed by first decomposing the $\overline{\mathbf{X}}$ matrix using principal component analysis (PCA) and subsequently clustering the resulting latent variable (or score) space. The resulting loading matrix from PCA, $\mathbf{P}$, can be used to relate the original cluster space variables (i.e., the lagged inputs and outputs) to the latent variables according to $\mathbf{T} = \overline{\mathbf{X}}\mathbf{P}$ where $\mathbf{T}$ denotes the projections of each row in $\overline{\mathbf{X}}$ onto the subspace (i.e., the scores). Note that this result follows from Eq. 5 and the fact that $\mathbf{P}$ is orthonormal. Typically, a much lower number of principal components (compared to $(n_y \times p) + (n_u + m)$ is required to be retained to completely characterize $\overline{\mathbf{X}}$ since $\overline{\mathbf{X}}$ can include many lagged variables and therefore correlations among the columns.

Reverse-time reachability regions

The reverse-time reachability region (RTRR) of a batch system at time, t, $\mathcal{R}(t)$, was defined as the set of process states from where a batch process could be steered to a desired end-point, $\boldsymbol{x}_{\mathrm{des}}$, by the end of the batch while satisfying the input constraints.[11] The definition of the discrete time version of this set is reproduced below:

Definition 1. For the batch process described by Eq. 1 with sampling period δ, the RTRR at time $t = t_{\mathrm{f}} - q\delta$, indexed by q, is the set[11]:

$$\begin{aligned}\mathcal{R}_q = \{\boldsymbol{x}_0 \mid \boldsymbol{x}(t) = \boldsymbol{x}_0,\ \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})\ \exists\ \boldsymbol{u} = \{\boldsymbol{u}[i]\} \in \mathcal{U} \\ \forall i = 1,\ldots,k \text{ where } \boldsymbol{u}[i] = \boldsymbol{u}(i\delta) \text{ and satisfies } \boldsymbol{u}(t) = \boldsymbol{u}[i] \\ \forall t \in [i\delta,(i+1)\delta),\ \text{such that } \boldsymbol{x}(t_{\mathrm{f}}) = \boldsymbol{x}_{\mathrm{des}}\}\end{aligned} \tag{13}$$

An algorithm to generate estimates of RTRRs as point sets at each sampling instance consists of successive off-line integrations of the reverse-time model of the system (i.e., $\dot{\boldsymbol{x}} = -\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})$).[11] However, this algorithm requires a first-principles model of the process, which may not always be available. Another contribution of the present work is the modification of the RTRR generation algorithm to use a data-based model of the process.

Integrating Data-Based Modeling Methods with Nonlinear Control Tools

In this section, we first propose a general multimodel approach for modeling batch systems that is applicable using full state measurements or limited output measurements and work through the underlying details. Then, for the case of full state measurements, we present an optimization based algorithm to sequentially characterize RTRR estimates using the resultant data-based model and then formulate a predictive controller design using these sets.

Multimodel approach

In presenting the multi-model approach, we first assume full state measurements and address the case of limited measurements through a remark (see Remark 7). In the proposed multi-model approach, a necessary preprocessing step is to cluster the batch database into L clusters as described earlier. Assuming this step has been completed, the basic idea in the proposed approach is to identify several local linear models around the cluster center points and then combine them with appropriate weights to describe the global nonlinear behavior. For the individual linear models, we employ the state-space models in Eq. 3, which are identified using PLS regression. Mathematically, this idea is expressed by the following model:

$$\boldsymbol{x}(k) = \sum_{\ell=1}^{L} w_\ell(k)\hat{\beta}_\ell\left[\boldsymbol{x}^{\mathrm{T}}(k-1) \quad \boldsymbol{u}^{\mathrm{T}}(k-1)\right] \tag{14}$$

where w_ℓ is the (normalized) weight given to model ℓ of L total models and $\hat{\boldsymbol{\beta}}_\ell$ is a matrix of model coefficients (with appropriate dimensions). Note that if the weights are determined independently (from estimating the individual model parameters), Eq. 14 becomes linear in $\hat{\boldsymbol{\beta}}_\ell$ and the system identification problem reduces to a PLS regression problem.

Intuitively, from the process description in Eq. 1, the weight placed on a local linear model should depend on the proximity of the states and input to the center point of the set of points for which the local linear model is identified, captured in the normalized fuzzy clustering membership function (Eq. 11). For instance, if a state-input combination nearly coincides with a specific cluster center point, the local linear model corresponding to that cluster should be given most of the weight. This is consistent with Eq. 11 as the membership function value corresponding to that cluster will be close to 1 while for the remaining clusters, the membership function will be near 0.

Since all the local models can potentially contribute during prediction, it is important to identify the local linear PLS models simultaneously. To facilitate this regression, the process data can be arranged in the following linear model form[26]:

$$\mathbf{Y} = \begin{bmatrix} \mathbf{U}_1 \otimes \bar{\mathbf{X}} & \cdots & \mathbf{U}_\ell \otimes \bar{\mathbf{X}} & \cdots & \mathbf{U}_L \otimes \bar{\mathbf{X}} \end{bmatrix} \begin{bmatrix} \hat{\beta}_1 & \cdots & \hat{\beta}_\ell & \cdots & \hat{\beta}_L \end{bmatrix}^{\mathrm{T}} \tag{15}$$

where $\otimes$ denotes element by element multiplication and $\mathbf{U}_\ell$ is defined as:

$$\mathbf{U}_\ell = \begin{bmatrix} \mu_{\ell,1} & \cdots & \mu_{\ell,1} \\ \vdots & \cdots & \vdots \\ \mu_{\ell,N} & \cdots & \mu_{\ell,N} \end{bmatrix}$$

Note that $\mathbf{U}_\ell$ has the same dimensions as $\bar{\mathbf{X}}$. Since Eq. 15 is linear in the parameters, $\hat{\beta}_\ell$, PLS regression can be performed to simultaneously identify the local linear models. In summary, the proposed multi-model approach unifies the concepts of ARX modeling, PCA/PLS techniques, fuzzy c-means clustering, and multiple linear models inspired from a "state-space" representation, where the clustering is done based on states/outputs and inputs, the model parameters are identified using the strength of PCA/PLS techniques, and the nonlinearity of the process is captured via the use of clustering techniques in combination with local linear models.

Remark 4. The proposed modeling method addresses some of the key issues with existing modeling approaches for batch systems. In contrast to PWA modeling in which a crisp clustering algorithm such as *k*-means is typically used and the model selection is a discrete decision, the current multi-model approach employs fuzzy clustering and allows weighting the different models appropriately using a continuous weighting function. This becomes particularly important during periods of transition in the process when it is evolving from one operating region to another. In this case, as multiple models will be weighted appropriately, the information from several different models can be used, resulting in more accurate predictions. In multi-model approaches using crisp clustering algorithms such as PWA modeling, only samples belonging to a specific cluster can contribute in determining the model and no surrounding information is used. That is, artificial boundaries are established for the clusters and their corresponding models, which can lead to abrupt changes in predictions during transitional periods. Furthermore, the presence of discrete model selection in the PWA framework also negatively impacts its use in optimization-based control designs by requiring solutions of optimization problems that include continuous (the control action) as well as discrete (the choice of the model) variables.

Remark 5. Note that the key difference between the present approach and the latent variable modeling methods[8] is the parameter used in the clustering. The latent variable modeling approaches rely on the batch trajectories being 'similar' such that the effect of state and input evolution on the process dynamics can be captured through 'time' or 'phase' markers. In contrast, the present work explicitly accounts for the dependence of the dynamics on the process states and inputs, thereby allowing more variation in the batch trajectories, essentially leading to a richer model. The latent variable modeling[1] also leads to an inherently linear PCA model whereas the weighting in the proposed modeling approach allows capturing nonlinearity in the data/process. Furthermore, when using the PCA model to predict the outputs given a set of control moves, a missing data algorithm is used and the outputs are computed implicitly (not explicitly as in the proposed approach) such that they maintain the average correlation structure as in the batch database.[1] However, in many cases, the correlation structure determined by the PCA model may no longer hold for a new initial condition due to the strong nonlinearities in the system and/or there are significant changes in the correlation structure as the batch proceeds. In contrast, if states/inputs close to the initial condition were encountered (albeit say at a different time point in historical batch trajectories), the present approach will be able to use that information to predict meaningful process evolution for the new batch.

Remark 6. While the present approach has the ability to capture the different phases that a batch goes through due to the inherent nonlinearity of the dynamics, it does not explicitly account for time-varying dynamics or where the dynamics exhibit a switched nature (i.e., phases where particular reagents are added only during the phase). While explicit incorporation of these conditions remains outside the scope of the present work, the efficacy of the proposed method to handle time-varying parameters is demonstrated in the simulation example.

Remark 7. For the case of limited output measurements, the matrix $\bar{\mathbf{X}}$ (or its PCA decomposition as described in Remark 3) is required to be clustered. Thus, to compute the model weights (w_ℓ) using Eq. 11, the $\bar{x}$ vector in Eq. 11 is constituted of lagged outputs and inputs (or their scores as described in Remark 3). This implies the lagged outputs and inputs are used to infer the current operating conditions as opposed to the current states and inputs. The proposed model in the case of limited measurements takes the following form:

$$\hat{\boldsymbol{y}}(k) = \sum_{\ell=1}^{L} w_\ell(k)\hat{\beta}_\ell \times \begin{bmatrix} \boldsymbol{y}^{\mathrm{T}}(k-1) & \cdots & \boldsymbol{y}^{\mathrm{T}}(k-n_y) & \boldsymbol{u}^{\mathrm{T}}(k-1) & \cdots & \boldsymbol{u}^{\mathrm{T}}(k-n_u) \end{bmatrix} \tag{16}$$

This model can be readily incorporated in conventional output-based predictive controllers for batch systems. In this work, we utilize the output model form in designing a trajectory tracking predictive controller for a nylon-6,6 batch polymerization process.

Reverse-Time Reachability Region Generation Using the Data-Based Model

In this section, assuming full state measurements, we present a methodology to generate RTRRs using the data-based modeling approach developed earlier. Due to unavoidable discrepancies between a process and its empirical model, instead of considering exact reachability to a desired end-point, we consider reachability to a desired end-point neighborhood, $\mathcal{B}(\boldsymbol{x}_{des})$. We define a data-based version of a RTRR as the set of states from where the data-based model of the process can be driven inside the desired end-point neighborhood by the end of the batch. Denoting this set at sampling instant q as $\hat{\mathcal{R}}_q$, the definition of an empirical RTRR is stated as:

Definition 2. For the data-based model of the form in Eq. 14, the empirical RTRR at time $t = t_{\mathrm{f}} - q\delta$, indexed by q, is the set:

$$\hat{\mathcal{R}}_q = \{\hat{x}_0 \mid \boldsymbol{x}(0) = \hat{x}_0,\ \boldsymbol{x}(k) = \sum_{\ell=1}^{L} w_\ell(k) \hat{\beta}_\ell\left[\boldsymbol{x}^{\mathrm{T}}(k-1) \quad \boldsymbol{u}^{\mathrm{T}}(k-1)\right] \text{ for } k = 1,\ldots,q \ \exists\, \boldsymbol{u}(k) \in \mathcal{U} \ \forall\, k = 0,\ldots,q-1 \text{ such that } \boldsymbol{x}(q) \in \mathcal{B}(\boldsymbol{x}_{des})\} \quad (17)$$

In formulating an empirical RTRR-based predictive control design, explicit characterizations of these sets are required. In this work, we use ellipsoids to mathematically express empirical RTRRs at each sampling instance. The general form of the ellipsoid expression is given below:

$$\hat{\mathcal{R}}_q \approx \left\{\boldsymbol{x} \mid (\boldsymbol{x} - \boldsymbol{c}_q)^{\mathrm{T}} \mathbf{W}_q (\boldsymbol{x} - \boldsymbol{c}_q) \leq 1\right\} \quad (18)$$

where the vector $\boldsymbol{c}_q \in \mathbb{R}^n$ denotes the center of the ellipsoid, the positive-definite symmetric matrix $\mathbf{W}_q \in \mathbb{R}^{n\times n}$ defines its size and orientation, and q indexes the batch sampling instances as before. We note that because $q = 0$ corresponds to t_{f}, $\boldsymbol{c}_0 = \boldsymbol{x}_{\mathrm{des}}$ and $\mathbf{W}_0$ is a user defined matrix based on the acceptable variance level of the final product quality. An equivalent representation of the ellipsoid was used in this work in which the ellipsoid is expressed as the image of a unit ball under an affine transformation. That is, consider the unit ball, $S(0,1) = \{\boldsymbol{x}|\boldsymbol{x}^{\mathrm{T}}\, \boldsymbol{x} \leq 1\}$ in $\mathbb{R}^n$ and the affine transformation $T(\boldsymbol{x}) = \mathbf{H}\boldsymbol{x} + \boldsymbol{d}$ where $\mathbf{H} \in \mathbb{R}^{n\times n}$ is a non-singular matrix and $\boldsymbol{d} \in \mathbb{R}^n$. Applying the affine transformation to a point on the unit ball, we have $z = \mathbf{H}\boldsymbol{x} + \boldsymbol{d}$, which implies $\boldsymbol{x} = \mathbf{H}^{-1}(z - \boldsymbol{d})$. An ellipsoid can then be expressed through an affine transformation of the unit ball:

$$\begin{aligned} T(S(0,1)) &= \{z|(\mathbf{H}^{-1}(z-d))^{\mathrm{T}}(\mathbf{H}^{-1}(z-d)) \leq 1\} \\ &= \{z|(z-d)^{\mathrm{T}}\mathbf{V}^{-1}(z-d) \leq 1\} \end{aligned} \quad (19)$$

where $\mathbf{V} = \mathbf{H}\mathbf{H}^{\mathrm{T}} \in \mathbb{R}^{n\times n}$ is a positive-definite symmetric matrix. Thus, from Eq. 19, defining $\mathbf{H}_q$ and $\boldsymbol{d}_q$ is equivalent to defining the ellipsoid parameters $\mathbf{W}_q$ and $\boldsymbol{c}_q$ in Eq. 18.

Starting at $q = 1$, an explicitly characterized estimate of $\hat{\mathcal{R}}_q$ is identified from where the model states can be driven inside $\hat{\mathcal{R}}_{q-1}$. This procedure is repeated until an empirical RTRR is identified for every sampling instant in the batch. Given the RTRR ellipsoid parameters at $q - 1$ and I (predetermined) points (generated from a uniform distribution) on the surface of a unit ball denoted by $\{\boldsymbol{x}_{\mathrm{ub}}^{(1)},\ldots,\boldsymbol{x}_{\mathrm{ub}}^{(i)},\ldots,\boldsymbol{x}_{\mathrm{ub}}^{(I)}\}$, the following NLP is solved to determine the ellipsoid parameters, $\mathbf{H}_q$ and $\boldsymbol{d}_q$ (and therefore $\mathbf{W}_q$ and $\boldsymbol{c}_q$):

$$\max_{\mathbf{H}_q,\boldsymbol{d}_q,\boldsymbol{u}^{(i)}\in\mathcal{U}} \det \mathbf{H}_q \quad (20)$$

$$\text{subject to}: \boldsymbol{x}^{(i)} = \mathbf{H}_q\boldsymbol{x}_{\mathrm{ub}}^{(i)} + \boldsymbol{d}_q \,\forall\, i = 1,\ldots,I \quad (21)$$

$$\boldsymbol{x}_{\mathrm{nxt}} = \sum_{\ell=1}^{L} w_\ell \hat{\beta}_\ell [\boldsymbol{x}^{(i)} \quad \boldsymbol{u}^{(i)}] \quad (22)$$

$$w_\ell = \frac{||[\boldsymbol{x}^{(i)} \quad \boldsymbol{u}^{(i)}]^{\mathrm{T}} - \boldsymbol{c}_\ell||^{-2}}{\sum_{\ell=1}^{L} ||[\boldsymbol{x}^{(i)} \quad \boldsymbol{u}^{(i)}]^{\mathrm{T}} - \boldsymbol{c}_\ell||^{-2}} \quad (23)$$

$$(\boldsymbol{x}_{\mathrm{nxt}} - \boldsymbol{c}_{q-1})^{\mathrm{T}}\mathbf{W}_{q-1}(\boldsymbol{x}_{\mathrm{nxt}} - \boldsymbol{c}_{q-1}) \leq 1 \quad (24)$$

$$\mathbf{H}_q = \mathbf{L}_q\mathbf{L}_q^{\mathrm{T}} \quad (25)$$

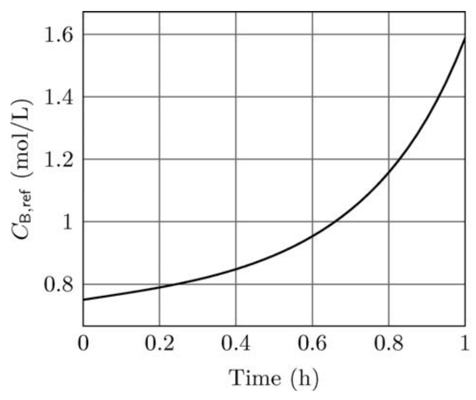
The independent decision variables in this NLP are the ellipsoid parameters ($\mathbf{H}_q$ and $\boldsymbol{d}_q$) and I control moves (corresponding to the I initial conditions on the surface of the ellipsoid). The NLP is formulated to maximize the volume of the current RTRR ellipsoid while ensuring for I uniformly distributed points on the surface of the ellipsoid, there exists a control action (as prescribed by a predictive controller using the data-based model) that can drive the ellipsoid surface point inside the next RTRR. Eq. 21 represents the affine transformation of the I unit ball points into the ellipsoid surface points. Eq. 25 represents the Cholesky decomposition of $\mathbf{H}_q$, where $\mathbf{L}_q \in \mathbb{R}^{n\times n}$ is a lower triangular matrix, and ensures $\mathbf{H}_q$ is positive-definite and symmetric. Note that ascertaining the feasibility of the optimization problem for the I surface points does not guarantee the feasibility of all points on the surface, or for that matter, for the internal points. While the nonlinear (and non-convex) nature of the optimization problem prevents such guarantees, in practice this conclusion can be reached by choosing a sufficiently large I. To ensure that the I chosen is "sufficiently large", in this work, I was increased until changes in the solution were below a predefined tolerance. To further verify that a control action exists to drive the states inside the next RTRR for the internal points of the ellipsoid, an appropriately defined NLP[27] is solved.

Empirical Reverse-Time Reachability Region Based MPC Formulation

Consider a batch system described by Eq. 1 for fault-free conditions and for which empirical RTRR estimates have been characterized for a given δ and $\mathcal{B}(\boldsymbol{x}_{\mathrm{des}})$. The control action at sampling instance $q = (t_{\mathrm{f}} - t)/\delta$ is computed by solving the following NLP:

$$\min_{\boldsymbol{u}(k)\in\mathcal{U}} J_{\mathcal{R}} = \sum_{k=1}^{P} \alpha(\hat{\boldsymbol{x}}(k) - \boldsymbol{c}_{q-k})^{\mathrm{T}}\mathbf{W}_{q-k}(\hat{\boldsymbol{x}}(k) - \boldsymbol{c}_{q-k}) + \gamma(\Delta\boldsymbol{u}(k)^{\mathrm{T}}\mathbf{R}\Delta\boldsymbol{u}(k)) \quad (26)$$

$$\text{subject to}: \hat{\boldsymbol{x}}(0) = \boldsymbol{x}(t) \quad (27)$$



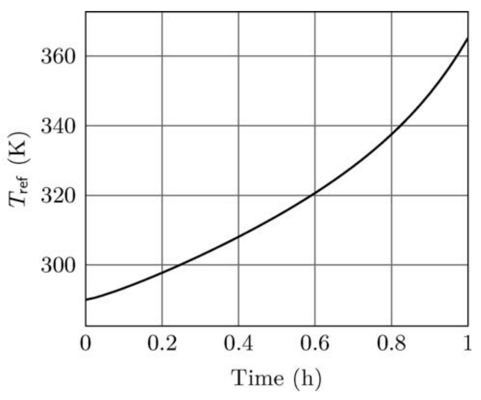


Figure 1. Nominal trajectories of C_B and T tracked using two PI controllers to generate the batch database for the fed-batch system.

$$\hat{x}(k+1) = \sum_{\ell=1}^{L} w_\ell(k)\hat{\beta}_\ell [\, x^T(k) \quad u^T(k)\,] \tag{28}$$

$$w_\ell = \frac{||[\, x^T(k) \quad u^T(k)\,]^T - c_\ell||^{-2}}{\sum_{i=1}^{L} ||[\, x^T(k) \quad u^T(k)\,]^T - c_i||^{-2}} \tag{29}$$

where $\Delta u(k) \in \mathbb{R}^m$ denotes a vector of differences between the current and previous input values across the horizon, $\mathbf{R}$ is a positive-definite weighting matrix, and the objective function, $J_\mathcal{R}$, is formulated to minimize variations in the control moves and maintain the process states inside the empirical RTRRs over the prediction horizon. The relative importance of the two terms in $J_\mathcal{R}$ can be traded off using appropriate values for the weights, α and γ.

Remark 8. The predictive model in the MPC formulation, specifically the nonlinear weighting function, makes this optimization problem a NLP, which can potentially be too computationally expensive for real-time application. However, this nonlinearity, while capturing the process dynamics much better than a single linear model, is likely to be much less severe compared to non-linearities typically found in first-principles deterministic models. Consequently, the optimization problem should remain efficiently solvable even for moderate values of P.

Remark 9. Note that due to the unavoidable plant-model mismatch, the proposed MPC formulation does not offer any guarantees regarding the reachability of the process inside the desired end-point neighborhood. In particular, even if one were to impose a constraint in the MPC requiring the states to go to the next RTRR, the feasibility of the constraint (guaranteed if the current states are in the corresponding RTRR) would not guarantee that the process state would be inside the RTRR at the next step. Yet, the determination of the RTRRs, specifically the ellipsoid matrices, provide useful weighting matrices to penalize state deviations to enforce the states to never significantly diverge from conditions where the desired end-point neighborhood can be reached. This also results in important fault-tolerant characteristics as discussed next.

Remark 10. To generate empirical RTRRs using the data-based model, the database must include state measurements. If the number of states are known and a deterministic model of the system is available but overly complex for use in any online applications (such as state estimation or model-based control), the states of the system can be back calculated off-line from the database measurements using a variety of state estimation tools such as a moving horizon estimator, extended Kalman filter, or the unscented Kalman filter. The resulting states can be used to populate the database, and a state-space model of the form shown in Eq. 14 can be developed. This model will capture the nonlinearity in the batch and is more amenable to online applications and usable for generating empirical RTRR estimates (and a corresponding RTRR based MPC design).

Remark 11. Note that if a fault occurs during the batch operation, in the absence of any knowledge of the fault repair time, the only meaningful control objective is to take control action at the current time such that if full control effort were to be restored at the next time step, reachability to the desired end-point neighborhood can be achieved. The RTRR-based MPC, by preserving the states in the reachability regions during the fault repair period, implements exactly this control objective. In contrast, end-point based predictive controllers try to achieve a (potentially) inherently unachievable objective- that of driving the process to the desired end-point neighborhood subject to the reduced control effort, and in doing so could drive the process to a point from where the desired end-point neighborhood is not reachable even after fault repair.

Simulation Results

In this section two simulation examples are presented. The first one illustrates the details of the proposed modeling and control design subject to varying initial conditions, noisy measurements, and time-varying uncertainty and the second demonstrates an application subject to lack of full state information.

Fed-batch reactor

In this section, a data based model of a fed-batch system is extracted from an artificially generated historical database using the proposed modeling methodology. Then, the resulting model is utilized in the RTRR-based predictive controller. To this end, consider a fed-batch reactor system where an irreversible series of reactions of the form $2A \xrightarrow{k_1} B \xrightarrow{k_2} 3C$ take place. The state-space model has been derived using standard modeling assumptions and using typical parameters values for this system.[28] The state vector is $x = [C_A \; C_B \; C_C \; T \; V]^T$ where C_A, C_B, and C_C denote the concentrations of species A, B, and C (respectively) and T and V denote the reactor temperature and volume

Table 1. Simulation Parameters Used for Database Generation for the Fed-Batch System

	Initial Condition Range	Sensor Noise
C_A	4.861 – 5.106 mol/L	±2.2% of original signal
C_B	0.692 – 0.801 mol/L	±2.8% of original signal
C_C	0.446 – 0.539 mol/L	±2.1% of original signal
T	280.931 – 300.057 K	0.10 standard deviation
V	97.952 – 101.959 L	0.10 standard deviation

PI controller parameters	Loop 1	Loop 2
Controlled variable	C_B	T
Manipulated variable	F	T_{hx}
Proportional gain	6.10	−0.035
Integral time (h)	0.80	0.01

(respectively). The manipulated inputs are the inlet feed rate, F (L/h), and heating coil temperature, T_{hx} (K), $\boldsymbol{u} = [F\ T_{hx}]^T$, with constraints $\boldsymbol{u}_{min} = [0\ 288]^T$ and $\boldsymbol{u}_{max} = [20\ 360]^T$. The primary control objective considered is to drive the process states inside a (arbitrarily chosen) desired end-point around $\boldsymbol{x}_{des} = [2.752\ 1.601\ 0.8422\ 365.756\ 112.425]^T$. The desired neighborhood is taken to be an ellipsoid with $\mathbf{W}_0 = \text{diag}\{25,400,100,0.04,1\}$ and $\boldsymbol{c}_0 = \boldsymbol{x}_{des}$. The batch termination time, t_f, is taken to be 1 hour with a sampling period, δ, of 0.025 h.

Data-based model development

A database consisting of forty batches was generated for the fed-batch reactor system using the state-space model. Ten batches were set aside as the validation data set. With two available inputs, reference trajectories of C_B and T (see Figure 1) were chosen to be tracked by manipulating F and T_{hx} (respectively) using two PI controllers. Both PI controllers were tightly tuned for one set of initial conditions and fixed for the remaining 39 batches. The criteria used to tune the PI controllers was the integral of time-weighted absolute error (ITAE) and a reasonably smooth input trajectory.

For a more realistic representation of plant data, sensor noise, disturbances, and a time-varying parameter were also considered. The range of initial conditions and sensor noise levels are summarized in Table 1 along with the PI tuning parameters. To simulate disturbances, T^{in} was stochastically varied throughout the duration of each batch around its nominal value in the range 295 – 305 K. The time-varying parameter was chosen to be the heat exchanger coefficient, UA. At the start of each batch, UA was assigned a value in the range 28,620 – 30,349 cal/(h · K) and then decreased exponentially to simulate fouling.

With forty time steps in each batch and thirty batches, the operating region space required to be partitioned consisted of

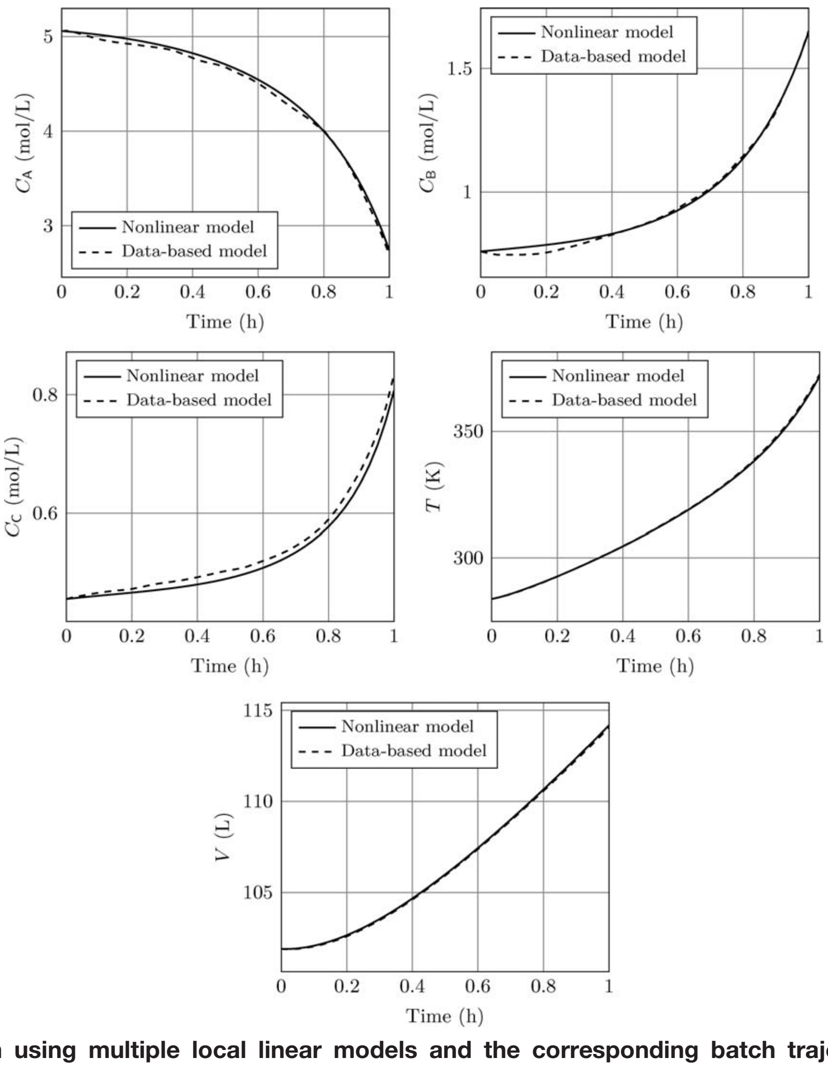


Figure 2. Output from using multiple local linear models and the corresponding batch trajectory in the training data for the fed-batch system.

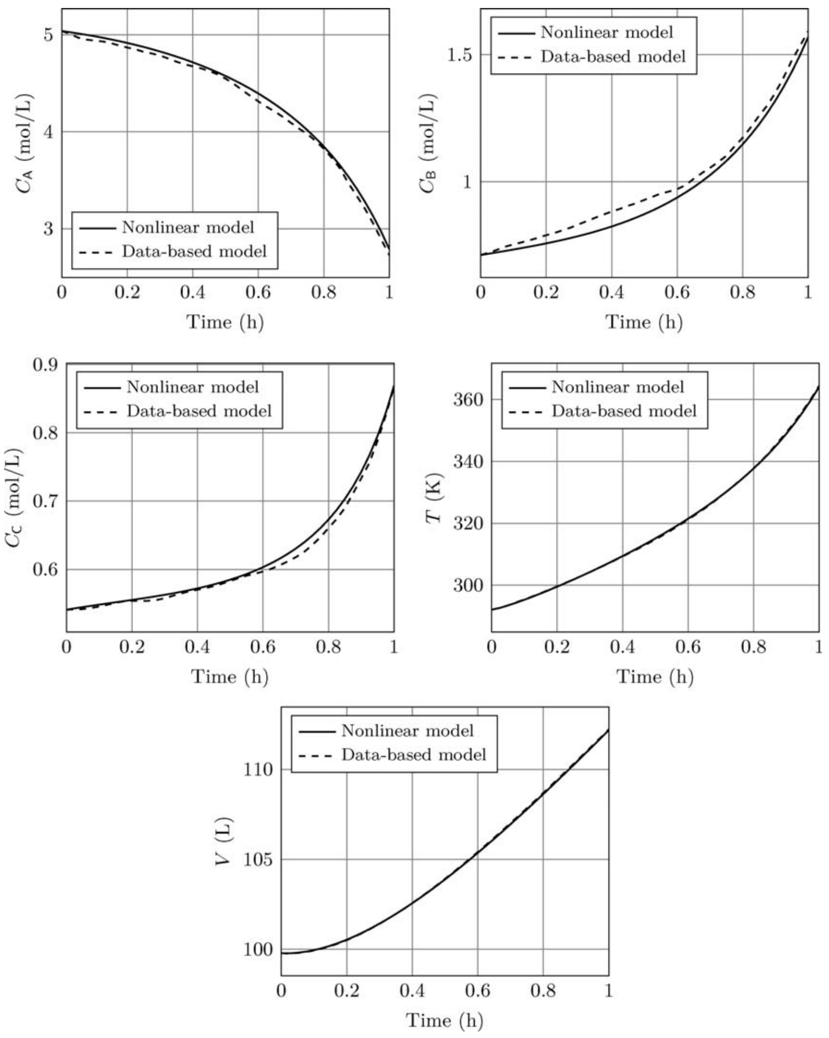


Figure 3. Output from using multiple local linear models and the corresponding batch trajectory in the validation data set, demonstrating the good prediction capability of the proposed modeling method for the fed-batch system.

1200 observations. To find the optimum number of clusters, L was varied from $L = 10$ to $L = 100$. For each clustering choice, PLS models were fit for a range of principal components, the trajectories of the validation batches were predicted using the PLS models, and the root mean squared error (RMSE) in the predictions was tabulated. The lowest RMSE in the predictions was obtained with $L = 20$ clusters and 142 principal components. Note that from the data arrangement used to compute the PLS models (Eq. 15), the maximum number of principal components is 152. Generally, retaining a high number of principal components produced very low residuals, but the predictive capabilities of the model are reduced due to over-fitting. This is because with an excessive number of principal components, the model begins to fit the random noise element in the data. A trade-off between low residuals and high predictive power was found using 142 principal components.

For a specific initial condition and a set of input trajectories in the database, Figure 2 shows the model output from the local linear models together with the corresponding database trajectories. Similar fits were observed for the remaining 29 different initial conditions and input trajectories. Figure 3 illustrates the predictive capabilities of the model for an initial condition and input trajectories which are not in the training data, but in the validation data set. The temperature and volume ranges in the figures are significantly larger compared to the ranges for the concentrations because their initial values (and values for the duration of the batch) are an order of magnitude greater than all the concentrations. As a result, the prediction error for the concentrations are more noticeable compared to the temperature and volume. Overall, the multi-model approach was able to capture the major nonlinearities in the database.

Table 2. Final Level Sets of $\mathcal{B}(\boldsymbol{x}_{\text{des}})$ Achieved Using a PI Controller and the RTRR-Based Predictive Controller for the Fed-Batch System

Initial Condition	1	2	3	4	5	6	7	8	9	10
Final PI level set	2.479	7.220	0.436	1.972	2.233	0.808	0.682	4.045	0.463	1.190
Final MPC level set	0.700	0.939	0.237	0.932	0.160	0.164	0.0263	0.0883	0.152	0.190

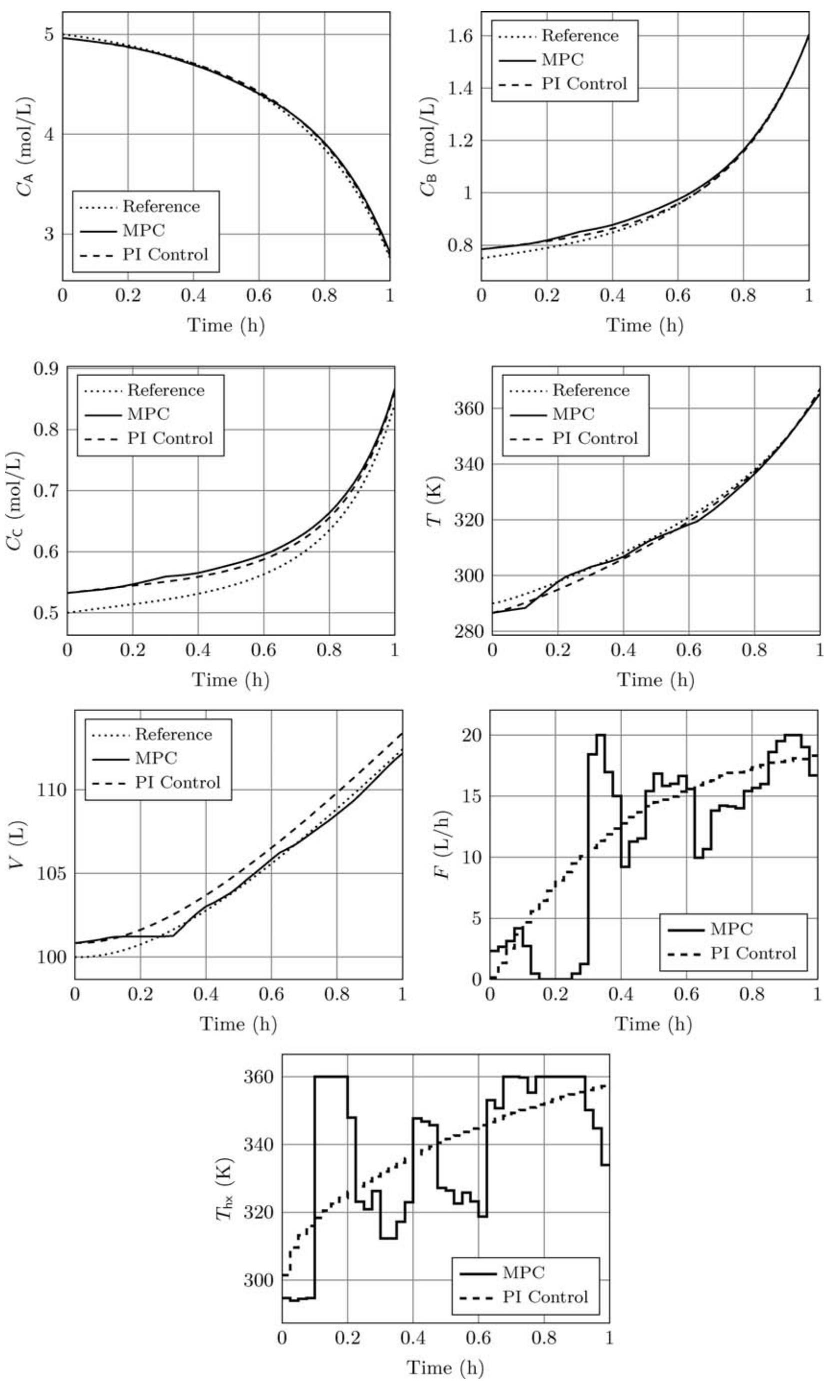


Figure 4. Representative state and input profiles of the fed-batch reactor system under PI control and RTRR-based MPC with no input failures.

The nominal set of state trajectories that terminate at the desired end-point is also shown.

MPC implementation using the data-based model

In this section, the proposed RTRR-based tracking MPC design is implemented on the fed-batch reactor system and the control performance is compared with that under PI control. The performance was measured by computing the level set of the desired end-point neighborhood for the final values of the states. A value of less than one indicated the final states were within the desired neighborhood (see Eq. 18). Closed-loop simulations were performed for ten different initial conditions that were not in the training or validation data set. All initial conditions were also verified to be within the empirical RTRR at the initial time. Sensor noise, a stochastic disturbance in T^{in}, and a time-varying UA parameter were also considered. Similar to the PI tuning, the RTRR-based predictive controller was tuned once for a specific set of initial conditions and left unchanged for the remainder of the simulations to avoid confounding the results with tuning. For RTRR-based MPC, the following values were picked for the tuning parameters: $\mathbf{R} = \text{diag}\{0.01, 0.005\}$ and $P = 18$. The final level sets obtained from the simulations are shown in Table 2. The RTRR-based MPC design was able to drive the system inside $\mathcal{B}(\boldsymbol{x}_{des})$ for all ten initial conditions whereas the PI controller failed in more than half of the cases. On

Table 3. Tracking Performance of the PI Controller and the Proposed MPC Formulation for the Nylon-6,6 Batch Polymerization System

Initial Condition	1	2	3	4	5	6	7	8	9	10
PI T ITAE	9.10	10.18	3.33	4.95	12.99	6.99	13.23	3.14	14.63	4.91
MPC T ITAE	2.18	1.51	1.33	1.76	2.98	1.21	2.72	1.19	1.90	1.89
PI P ITAE	3.08	3.46	5.02	2.71	1.93	10.29	5.18	1.20	1.43	4.61
MPC P ITAE	1.01	1.22	1.47	2.03	1.43	1.04	1.57	0.860	1.31	1.58

average, the final MPC and PI level sets were 0.359 and 2.153 (respectively). In Figure 4, a representative set of closed-loop profiles is presented (initial condition #10). To demonstrate that the MPC problem remains efficiently solvable despite being nonlinear, we note that with $P = 18$, the longest CPU time required time to solve the MPC problem was 0.452 seconds (using GAMS with IPOPT as the solver on an Intel Quad Core machine).

To demonstrate the fault-tolerance of the RTRR-based controller, we consider faults in both of the control actuators and compare the performance of the RTRR-based MPC design with PI control. Specifically, starting from $\boldsymbol{x}(0) = [4.977\ 0.763\ 0.539\ 289.485\ 100.456]^{\mathrm{T}}$, we consider the scenario where at $t^{\mathrm{fault}} = 0.25$ hours, the actuators associated with the flow rate and heating coil fail and the maximum flow rate and heating coil temperature are reduced to $\boldsymbol{u}_{\max} = [10\ 310]^{T}$ (from $[20\ 360]^{T}$). At $t^{\mathrm{repair}} = 0.45$ hours, the fault is rectified and full control effort is recovered. Note that during the failure period, the prediction horizon of the RTRR-based MPC was reduced from $P = 18$ to $P = 1$, to avoid having to assume the failure situation any longer than necessary (in computing the control action). For the PI controller, the batch is driven to the end-point of $\boldsymbol{x}(t_{\mathrm{f}}) = [3.647\ 1.237\ 0.712\ 339.084\ 110.763]^{T}$, which is well outside of $\mathcal{B}(\boldsymbol{x}_{\mathrm{des}})$ resulting in a final level set value of 104.393. On the other hand, the RTRR-based MPC drives the process to $\boldsymbol{x}(t_{\mathrm{f}}) = [2.810\ 1.588\ 0.867\ 365.605\ 112.148]^{T}$ which corresponds to a final level set of 0.173. The closed-loop profiles for this case are shown in Figure 5. The PI controller prescribes the heat exchanger temperature to remain saturated during the failure period whereas the RTRR prescribes more meaningful temperature towards the latter stages of the fault. As a result, the RTRR-based controller is able to recover after the fault and essentially begins to track the nominal state trajectories which terminate at the desired end-point.

Nylon-6,6 batch polymerization

In this section, we apply the data-based modeling methodology on a complex nonlinear batch polymerization process to extract models for the key measurable process variables. Subsequently, we employ the model in a trajectory tracking predictive controller and compare the tracking performance with a PI controller. For our studies, we utilize the mathematical model of nylon-6,6 polymerization wherein the polymer is produced by the amidation of adipic acid and hexamethylenediamine (HMD) in a batch autoclave reactor with a steam jacket for providing the heat needed for vaporization (and reaction) and a valve for venting vaporized water.[29] The reaction model, modeling assumptions (and their explanations), parameter values, and kinetic relationships are available in existing literature[29,30] and omitted here for brevity. The final state-space model of the process consists of nine coupled ordinary differential equations (ODEs) with the state vector comprised of the molar amounts of each functional group and evaporated HMD, the reaction medium mass, temperature, and volume, and reactor pressure. The final model takes the general form shown in Eq. 1 with the outputs, $\boldsymbol{y}$, being the reaction mixture temperature, T (K), and reactor pressure, P (psi). The manipulated inputs are the steam jacket pressure, P_j (psi), and vent rate, v (kg/h), $\boldsymbol{u} = [P_j\ v]^{\mathrm{T}}$, constrained between $\boldsymbol{u}_{\min} = [700\ 0]^{\mathrm{T}}$ and $\boldsymbol{u}_{\max} = [1800\ 2000]^{\mathrm{T}}$. The duration of the batch is $t_{\mathrm{f}} = 3$ hours with a sampling period of 60 s.

For this work, we focus specifically on tracking trajectories of the reaction medium temperature, T, and reactor pressure, P, by manipulating the steam jacket pressure, P_j and vent rate, v. We assume reference trajectories for T and P, denoted by T_{ref} and P_{ref} (respectively), have been identified appropriately in some fashion, and Figure 6 presents these specific trajectories.

Data-based model development

For the nylon-6,6 system, the pressure dynamics were significantly faster than the temperature dynamics, leading to a weak correlation between the two outputs. Consequently, individual linear models were computed for the outputs as opposed to a PLS model which predicts both outputs simultaneously. To develop the data-based models for the two outputs, a database of previous batches was first generated. To this end, the deterministic nylon-6,6 polymerization model[29] was simulated fifteen times from different initial conditions (five batches were reserved as the validation data set). As in the fed-batch case, the set of reference temperature and pressure profiles presented in Figure 6 were tracked reasonably well using two PI controllers to generate the database. For the PI loop-pairing, the vent rate was used to track the reactor pressure while the steam jacket pressure was used to track the temperature. Both PI controllers were tightly tuned (using ITAE and smooth input trajectories as the criteria) for one set of initial conditions and fixed for the remaining batches.

The following parameters were required to be specified for the modeling approach: the lag structure (n_y and n_u) and the number of clusters, L. To understand the identification procedure, consider just the temperature model. The temperature model was identified by first generating a series of different models as follows: first, n_y and n_u were specified, thus setting the dimensions of $\bar{\mathbf{X}}$. Next, $\bar{\mathbf{X}}$ was decomposed using PCA, the resulting score space was clustered for $L = 2$ to $L = 20$, and local linear models were identified for each choice of L via principal component regression (PCR).[15] The quality of each model was evaluated by its RMSE in predicting the five validation batches. This was repeated for several choices for the lag structure. For the temperature model, $L = 5$ and lags of 1, 0, 1, and 1 in the temperature, reactor pressure, steam jacket pressure, and vent (respectively) yielded the lowest RMSE. Note that zero lags for the reactor pressure indicates that the reactor pressure was not used in predicting the reaction mixture temperature. A similar procedure

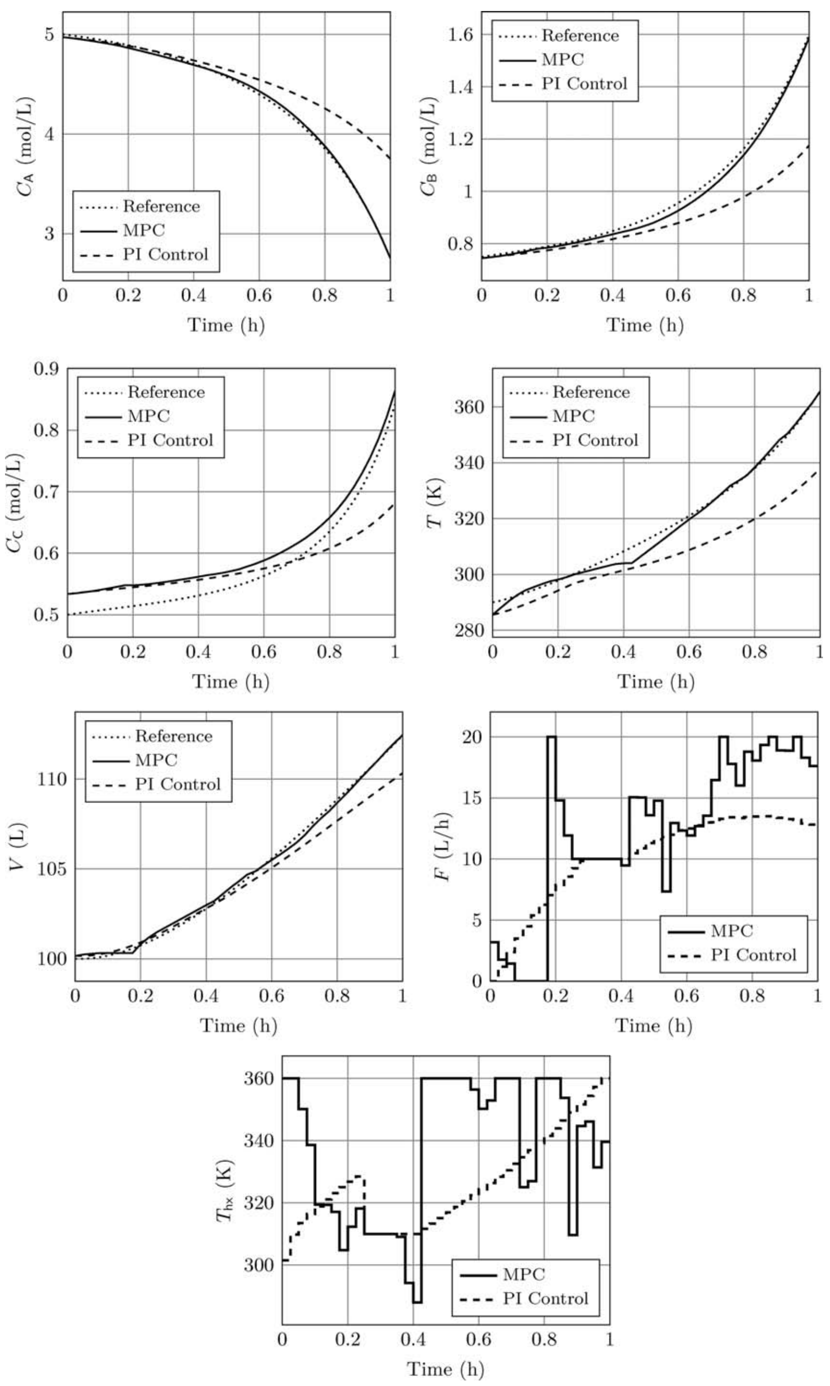


Figure 5. Representative state and input profiles of the fed-batch reactor system under PI control and RTRR-based MPC with input failures between 0.25 and 0.45 h.

The nominal set of state trajectories that terminates at the desired end-point is also shown. The profiles demonstrate the fault-tolerant characteristic of the RTRR-based MPC design.

was repeated for the pressure model, and the final lag structure was found to be 0, 1, 0, and 1 for the temperature, reactor pressure, steam jacket pressure, and vent rate (respectively) for which $L = 1$. These results were expected for the pressure model since they correspond to a linear dynamic model between the pressure and vent rate, which is consistent with the state-space model. In Figure 7, we compare the output of the nonlinear model with the model output from the data-based model for a set of initial conditions that was in the original training data. In Figure 8, we consider initial conditions which were in the validation data set. Overall, the multi-model approach captured the nonlinear nature of the batch and provided relatively reliable predictions.

MPC implementation using the data-based model

A predictive controller for tracking reference temperature and pressure profiles for the nylon-6,6 polymerization process is presented in this section. The control action at each sampling instance can be computed by solving the optimization problem below.

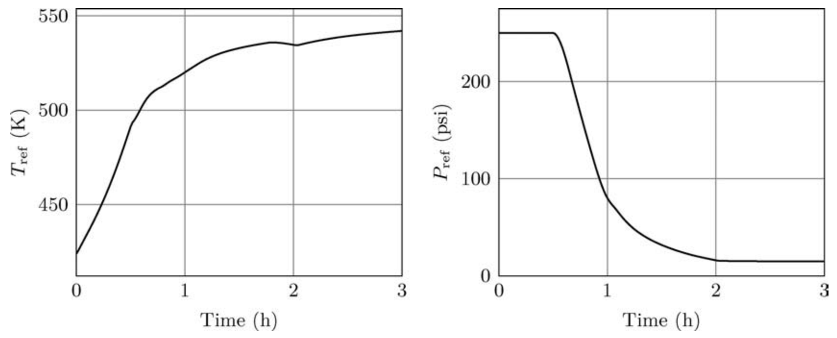


Figure 6. Nominal trajectories of *T* and *P* tracked using two PI controllers to generate the batch database for the nylon-6,6 batch polymerization system.

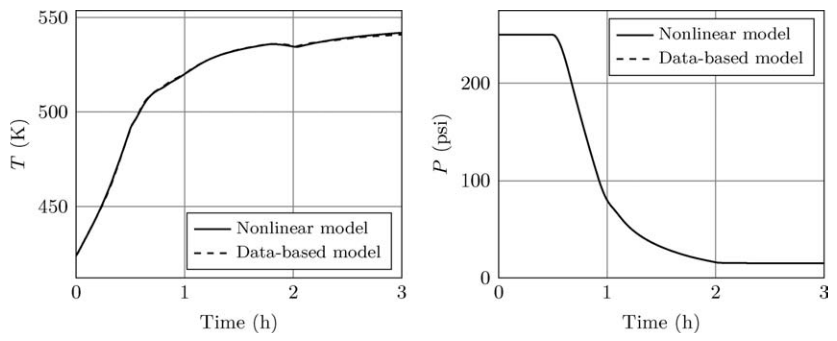


Figure 7. Output from using multiple local linear models and the corresponding batch trajectory in the training data for the nylon-6,6 batch polymerization system.

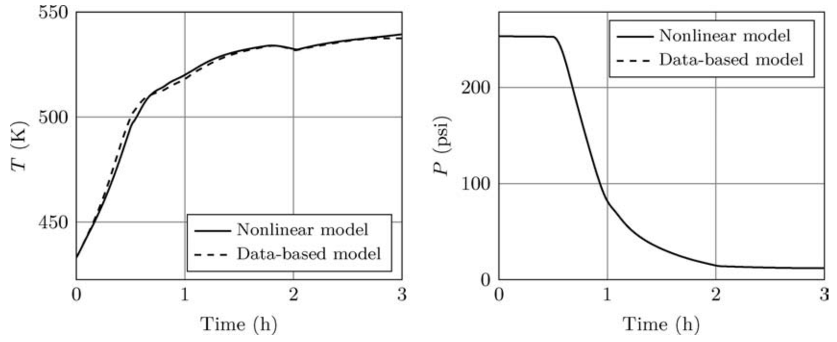


Figure 8. Output from using multiple local linear models and the corresponding batch trajectory in the validation data set for the nylon-6,6 batch polymerization system, demonstrating the good prediction capability of the proposed modeling method.

$$\min_{\boldsymbol{u}(k)\in\mathcal{U}} J = \sum_{k=1}^{P} ||\hat{\boldsymbol{y}}(k) - \boldsymbol{y}_{\mathrm{ref}}(k)||_{\mathbf{Q}} + ||\Delta \boldsymbol{u}(k)||_{\mathbf{R}} \tag{30}$$

$$\text{subject to: Eq. } (16) \tag{31}$$

$$\hat{\boldsymbol{y}}(0) = \boldsymbol{y}(t) \tag{32}$$

where $\mathbf{Q}$ and $\mathbf{R}$ are positive-definite matrices, and P is the prediction horizon.

Closed-loop simulations for ten new initial conditions (all from within the range of initial conditions in the training data) were performed using the proposed trajectory tracking MPC design, and the performance was compared against a PI controller. The tuning parameters used for the proposed MPC were: $\mathbf{Q} = \mathrm{diag}\{2.75, 27.5\}$, $\mathbf{R} = \mathrm{diag}\{0.02, 0.02\}$, and $P = 12$.

On average, as shown in Table 3, the proposed MPC controller offered a significant improvement of approximately 78% and 65% for temperature and pressure tracking (respectively). In all simulations, the proposed predictive controller outperformed the PI controller. A representative set of closed-loop simulation results is presented in Figure 9. In this case, the ITAEs for the proposed predictive controller improved on the PI controller by 77% and 26%. Overall, the simulation results clearly demonstrate the advantages of implementing the proposed trajectory tracking predictive controller over PI control.

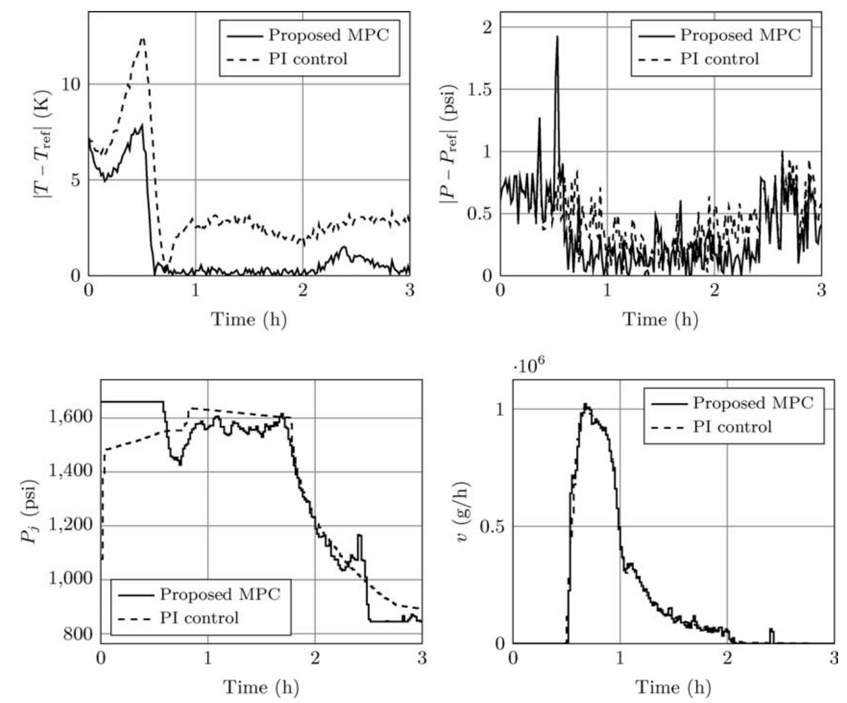


Figure 9. Representative output tracking error and input profiles of the nylon-6,6 batch polymerization system under PI control and the proposed trajectory tracking MPC design.

Conclusions

In this work, we addressed the problem of uniting empirical modeling approaches with nonlinear control tools for control of batch systems. In the proposed approach, we exploited the availability of historical batch data, the simplicity of local linear models, the data extraction capabilities of PLS/PCR, and the use of appropriate clustering and weighting techniques in conjunction with multiple models to capture the nonlinear nature of a batch process. The resulting model from this approach was employed to generate empirical RTRRs, which were subsequently incorporated in a predictive control design. The efficacy of the RTRR-based MPC design and superior performance, as well as fault-handling ability, with respect to PI control was demonstrated through a fed-batch simulation example. The data-based modeling approach was also applied to derive the models for use in a trajectory tracking MPC design for a nylon-6,6 batch polymerization process with limited measurements. The tracking performance of the proposed trajectory tracking controller was demonstrated to be significantly superior to PI control.

Acknowledgments

Financial support from the McMaster Advanced Control Consortium and the Natural Sciences and Engineering Research Council of Canada (NSERC) through the CGS(D) award for Siam Aumi is gratefully acknowledged.

Literature Cited

1. Flores-Cerrillo J, MacGregor JF. Latent variable MPC for trajectory tracking in batch processes. *J Process Control.* 2005;15:651–663.
2. Bhat SA, Huang B. Preferential crystallization: multi-objective optimization framework. *AIChE J.* 2009;55:383–395.
3. Corriou JP, Rohani S. A new look at optimal control of a batch crystallizer. *AIChE J.* 2008;54:3188–3206.
4. Cruickshank SM, Daugulis AJ, McLellan PJ. Dynamic modeling and optimal fed-batch feeding strategies for a two-phase partitioning bioreactor. *Biotech Bioeng.* 2000;67:224–233.
5. Soroush M, Kravaris C. Optimal-design and operation of batch reactors. 1. Theoretical framework. *Ind Eng Chem Res.* 1993;32:866–881.
6. Soroush M, Kravaris C. Optimal-design and operation of batch reactors. 2. A case-study. *Ind Eng Chem Res.* 1993;32:882–893.
7. Trifkovic M, Sheikhzadeh M, Rohani S. Multivariable real-time optimal control of a cooling and antisolvent semibatch crystallization process. *AIChE J.* 2009;55:2591–2602.
8. Zhang GP, Rohani S. On-line optimal control of a seeded batch cooling crystallizer. *Chem Eng Sci.* 2003;58:1887–1896.
9. Palanki S, Vemuri J. Optimal operation of semi-batch processes with a single reaction. *Int J Chem Reac Eng.* 2005;3:A17.
10. Pistikopoulos EN, Dua V, Bozinis NA, Bemporad A, Morari M. On-line optimization via off-line parametric optimization tools. *Comp Chem Eng.* 2002;26:175–185.
11. Aumi S, Mhaskar P. Safe-steering of Batch Processes. *AIChE J.* 2009;55:2861–2872.
12. Huang B, Ding XS, Qin SJ. Closed-loop subspace identification: an orthogonal projection approach. *J Process Control.* 2005;15: 53–66.
13. Lin WL, Qin SJ, Ljung L. On consistency of closed-loop subspace identification with innovation estimation. In: *Proceedings of the IEEE Conference On Decision And Control (CDC).* Atlantis, Bahamas 2004:2195–2200.
14. Wang J, Qin SJ. Closed-loop subspace identification using the parity space. *Automatica.* 2006;42:315–320.
15. Geladi P, Kowalski BR. Partial Least-Squares Regression: A tutorial. *Anal Chim Acta.* 1986;185:1–17.
16. Höskuldsson A. PLS Regression Methods. *J Chem.* 1988;2:211–228.
17. Seber GAF. *Multivariate Observations.* New York, NY: Wiley, 1984.
18. Ruspini EH. Numerical methods for fuzzy clustering. *Inf Sci.* 1970;2:319–350.
19. Bezdek J. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Norwell, MA: Kluwer, 1981.
20. Bezdek J, Coray C, Gunderson R, Watson J. Detection and characterization of cluster substructure I. Linear structure: fuzzy *c*-lines. *SIAM J Appl Math.* 1981;40:339–357.
21. Gustafson DE, Kessel WC. Fuzzy clustering with a fuzzy covariance matrix. In: *Proceedings of the IEEE Conference on Decision and Control (CDC).* San Diego, CA, USA 1979;761–766.
22. Frigui H, Krishnapuram R. Clustering by competitive agglomeration. *Pattern Recognit.* 1997;30:1109–1119.
23. Gath I, Geva AB. Unsupervised optimal fuzzy clustering. *IEEE Trans Pattern Anal Mach Intell.* 1989;11:773–780.

24. Krishnapuram R, Freg C. Fitting an unknown number of lines and planes to image data through compatible cluster merging. *Pattern Recognit.* 1992;25:385–400.
25. Xie XL, Beni G. A Validity Measure for Fuzzy Clustering. *IEEE Trans Pattern Anal Mach Intell.* 1991;13:841–847.
26. Yen J, Wang L, Gillespie Ch.W. Improving the interpretability of TSK fuzzy models by combining global learning and local learning. *IEEE Trans Fuzzy Syst.* 1998;6:530–537.
27. Aumi S, Mhaskar P. Robust model predictive control and fault-handling of batch processes. *AIChE J.* 2010;57:1796–1808.
28. Fogler HS. *Elements of Chemical Reaction Engineering*, 4th ed. Upper Saddle River, NJ: Prentice Hall 2006:625–627.
29. Russell SA, Robertson DG, Lee JH, Ogunnaike BA. Control of product quality for batch nylon-6,6 autoclaves. *Chem Eng Sci.* 1998;53:3685–3702.
30. Joly M, Pinto JM. Optimal control of product quality for batch nylon-6,6 autoclaves. *Chem Eng J.* 2004:97;87–101.

Manuscript received Feb. 17, 2011, and revision received Jun. 13, 2011.