2011 American Control Conference
on O'Farrell Street, San Francisco, CA, USA
June 29 - July 01, 2011

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

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

***Abstract*— This work presents a data-based multi-model approach for modeling batch systems in which multiple local linear models are identified using partial least squares (PLS) regression and then combined with an appropriate weighting function that arises from fuzzy c-means clustering. The resulting data-based model is used to generate estimates of 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 batch system) using an optimization based algorithm. The empirical RTRRs are used to formulate a computationally efficient predictive controller with inherent fault-tolerant characteristics. Simulation results of a fed-batch reactor subject to noise, disturbances, and uncertain parameters demonstrate that the empirical RTRR-based MPC design consistently outperforms PI control in both a fault-free and faulty environment.**

I. Introduction

Batch and fed-batch processes play an important role in the production of expensive, low-volume products such as bio-chemicals and polymers. The primary control objective in a batch process is to reach a specified product quality by batch termination. The control problem is complicated by the presence of constraints, nonlinear, time-varying dynamics, and the absence of steady-states, which limits the performance achievable by implementing controllers designed for continuous systems. Another operational issue with batch process control that requires special attention is the occurrence of faults. Due to the emphasis on the final product quality, even benign faults can ruin an entire batch, making the ability to handle faults an intrinsic requirement of the control design. The operation and control of batch processes stand to benefit immensely from the development of dedicated batch process control tools comprising fundamental contributions in the areas of modeling and control.

Process models can be built deterministically or empirically. In the former approach, first principles are used to derive a state-space representation of the process with some parameters to be determined from experimental data. One of the limitations with deterministic modeling is the lack of sufficient measurements to uniquely determine key model parameters, and even when available, many of the simplifying assumptions taken during model development may be violated in specific situations. Empirical modeling, on the other hand, typically uses a simple model structure (often linear) between the process inputs and outputs and determines all the model parameters from plant data. However, for batch systems, plant data needed to build empirical models is essentially limited to historical databases of previous batches since identification experiments are often too expensive to justify. The model identification task is further complicated by the highly nonlinear process variable behavior in these databases. This makes conventional system identification approaches, where a single input-output linear model is identified, ill-suited for identifying an accurate model. For batch systems, the high expenses associated with every batch dictate the need for dedicated modeling tools 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.

Another drawback of conventional system identification approaches is that most approaches fail to utilize the availability of auxiliary measurements (beyond the designated outputs and inputs) in batch databases. While these measurements are often (auto and/or cross) correlated with each other and the output and input measurements, they contain valuable information regarding the underlying process states, implying an improved model (compared to input-output models) could potentially be identified if the auxiliary measurements are included in the system identification procedure. This increased availability of past process data has made latent variable regression techniques, particularly partial least squares (PLS) regression, principal component analysis (PCA), and principal component regression (PCR), popular tools used during system identification (e.g., see [1]–[3]). The limitation with latent variable methods is an inherent assumption of linearity. In the approaches to incorporate nonlinear relationships (e.g. see [4] and the references therein), the models' predictive capability depends on the choice for the nonlinear mapping.

Existing batch control approaches either specify the desired end-point quality indirectly, through first determining "optimal" measurable process variable trajectories that terminate at the desired end-point and then using advanced control tools, such as model predictive control (MPC), to track the desired trajectories, or directly in an end-point based MPC framework. In trajectory tracking approaches, the relationship between the final product quality and process variables is altered during a new batch due to unavoidable disturbances; hence, trajectories deemed optimal in off-line calculations may be significantly sub-optimal in online implementation. MPC performance, in the context of trajectory tracking or end-point based designs, is contingent on the underlying model's prediction accuracy. When available, a deterministic process model with

*Financial support by NSERC and the McMaster Advanced Control Consortium (MACC) is gratefully acknowledged.
†Corresponding author: mhaskar@mcmaster.ca

well identified parameters can be used; however, there are situations, typically in industry, when the practical application of the deterministic model-based controller is restricted due to the heavy computational demands of repeatedly solving an optimization problem embedded with the full process model. The common work-around for this issue has been to reduce the complexity of the predictive model through linearization techniques. However, the use of linearized models (deterministic or empirical) in computing the control action has limited control performance owing to the strong nonlinearities in most batch systems.

The majority of the existing work addressing faults in batch systems focuses primarily on fault detection and isolation while contributions on explicit fault tolerant control structures have essentially been robust control designs that treat faults as disturbances (e.g, see [5], [6]). The fault-tolerant characteristic in these formulations is a result of an underlying assumption of availability of sufficient control effort such that the primary control objective remains achievable even in the presence of the fault. 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 input trajectories prescribed by controllers with limited fault-tolerant properties can drive the system to a point from where the final quality is unreachable even if the fault is repaired.

Motivated by the above considerations, this work considers the problem of designing an integrated framework that merges data-based models with nonlinear control tools for fault-tolerant, predictive end-point based control of batch processes. The rest of this manuscript is organized as follows. After presenting a few preliminaries, a methodology is presented for modeling batch processes that makes use of all available measurements in an existing database and captures the nonlinear nature of the process by combining local linear models with an appropriate weighting function. The resulting model is then incorporated within the existing framework in [7], [8] for computationally efficient and fault-tolerant end-point based MPC for batch systems. Simulation results of a fed-batch reactor system (subject to sensor noise, disturbances, and time-varying parameters) are presented to demonstrate the effectiveness of the proposed modeling and control approach. Finally, we summarize our results.

II. Preliminaries

In this section, we review the key concepts, namely auto-regressive exogenous (ARX) modeling, partial least squares (PLS) regression, and fuzzy c-means clustering, that are unified in the data-based modeling approach in Section III.

A. Process Description

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

$$\begin{aligned} &\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\theta}) \\ &\boldsymbol{y} = \boldsymbol{g}(\boldsymbol{x}) + \boldsymbol{v} \\ &t \in [t_0, t_{\mathrm{f}}], \boldsymbol{u}(\cdot) \in \mathcal{U}, \boldsymbol{\theta}(\cdot) \in \boldsymbol{\Theta}, \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \end{aligned} \tag{1}$$

The vectors, $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{y} \in \mathbb{R}^p$, denote the state variables and noise corrupted output variables (respectively), $\boldsymbol{v} \in \mathbb{R}^p$ is the zero-mean, normally distributed measurement noise vector, and $\boldsymbol{u} \in \mathbb{R}^m$ is a vector of constrained manipulated inputs, taking values in a nonempty, convex set, $\mathcal{U} \triangleq \{\boldsymbol{u} \mid \boldsymbol{u}_{\min} \leq \boldsymbol{u} \leq \boldsymbol{u}_{\max}\} \subset \mathbb{R}^m$ where $\boldsymbol{u}_{\min}$ and $\boldsymbol{u}_{\max}$ define the minimum and maximum (respectively) allowable input values. The vector, $\boldsymbol{\theta} \in \mathbb{R}^q$, is constituted of bounded, possibly time-varying uncertain variables, taking values in the nonempty, compact set, $\boldsymbol{\Theta} \triangleq \{\boldsymbol{\theta} \mid \boldsymbol{\theta}_{\min} \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}_{\max}\} \subset \mathbb{R}^q$ where $\boldsymbol{\theta}_{\min}$ and $\boldsymbol{\theta}_{\max}$ denote the minimum and maximum (respectively) allowable uncertainty values. The times, t_0 and t_{f}, denote the initial time and batch termination times, respectively. Throughout the manuscript, we assume that for any input trajectory, $\boldsymbol{u}(\cdot) \in \mathcal{U}$, the solution of the batch system exists (with initial conditions, $\boldsymbol{x}_0$) and is continuous for all $t \in [t_0, t_{\mathrm{f}}]$.

B. Auto-Regression Exogenous (ARX) Models

In ARX modeling, the process outputs at a specific sampling instance depend linearly on the previous process conditions (defined by the process outputs and inputs). Mathematically, in vector form, an ARX model for a given process output takes the form shown below.

$$y(k) = \boldsymbol{\beta}\bar{\boldsymbol{x}}(k) + v(k) \tag{2}$$

where $\boldsymbol{\beta}$ is a vector of model coefficients and $\bar{\boldsymbol{x}}(k) = \begin{bmatrix}\boldsymbol{y}'(k-1) & \cdots & \boldsymbol{y}'(k-n_y) & \boldsymbol{u}'(k-1) & \cdots & \boldsymbol{u}'(k-n_u)\end{bmatrix}$ is a row vector of lagged concatenated outputs and inputs. The scalars, n_y and n_u, are the number of lags in the outputs and inputs (respectively). Note that we have assumed the same number of lags, n_y and n_u, for each output and input variable (respectively) and continue to do so for the remainder of the manuscript for notational simplicity. This assumption can be easily relaxed (i.e., n_y and n_u can be vectors). Different criteria can be used to select the "optimum" lag structure (see [9]) with the general objective being to achieve low prediction error with the minimum number of model parameters, which prevents against over-fitting and maintains model simplicity.

Estimating the model parameters, $\boldsymbol{\beta}$, is a linear regression problem. Possible co-linearities/correlations in plant data that are problematic in ordinary least squares regression can be handled by using latent variable tools such as partial least squares (PLS) regression. When using PLS regression, rather than specifying a lag structure for each individual output, one global lag structure is typically selected, and the model parameters for all the output variables are estimated simultaneously. To facilitate the regression, a response vector, $\bar{\boldsymbol{y}}$, and a regressor matrix, $\bar{\mathbf{X}}$, are constructed corresponding to $y(k)$ and $\bar{\boldsymbol{x}}(k)$ (respectively) in (2) by sorting plant data sample-wise. A response matrix, $\bar{\mathbf{Y}}$, is then generated by constructing response vectors for each output and concatenating them in a matrix.

Geometrically, in PLS modeling, the variables in $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ are projected onto orthogonal subspaces of A-pairs of latent variables. Each pair of latent variables accounts for a certain percentage of the variance in the regressor and

response matrices. Mathematically, PLS regression consists of decomposing $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ as the sum of the outer products of a score and loading vector as $\hat{\bar{\mathbf{X}}} = \sum_{j=1}^{A} \boldsymbol{t}_j \boldsymbol{p}_j'$ and $\hat{\bar{\mathbf{Y}}} = \sum_{j=1}^{A} \boldsymbol{r}_j \boldsymbol{q}_j'$ where the hats denote estimates, $\boldsymbol{t}_j$ and $\boldsymbol{r}_j$ are the input and output scores (respectively) representing the projections of the variables in $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ on their corresponding subspaces, and $\boldsymbol{p}_j$ and $\boldsymbol{q}_j$ define the orientation of j-th coordinate in the two subspaces. The noise reduction property of PLS regression stems from the idea that the lesser principal components are typically a consequence of measurement and process noise and therefore can be discarded. Because it is desired to obtain a useful relationship between the original data matrices, $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$, the two matrices are linked by an linear inner relationship between their scores of the form $\hat{\boldsymbol{r}}_j = \boldsymbol{b}_j \boldsymbol{t}_j$ where $\boldsymbol{b}_j$ is the coefficient vector defining the inner relationship. In 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 $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ and therefore obtain the optimal fit for the inner relationship. The properties and steps of the NIPALS algorithms can be found in [10]. The final result from PLS regression is a linear model between $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ where the coefficients are functions of the scores and loadings from the matrix decompositions (i.e., $\hat{\bar{\mathbf{Y}}} = \bar{\mathbf{X}}\boldsymbol{\beta}$ where $\boldsymbol{\beta} = f(\mathbf{P}, \mathbf{T}, \mathbf{Q})$).

Remark 1 With full state measurements, choosing the number of input and output lags is trivial since the process conditions can be completely described by the current states and inputs. As a result, $n_y = n_u = 1$, which then reduces the ARX model form into a discrete linear state-space system. Accordingly, the state and input transition matrices are estimated by regressing a matrix of concatenated states and inputs on a matrix of forward shifted states.

C. Fuzzy c-Means Clustering

A pre-processing step in the proposed multi-model approach is to locate the operating points around which individual local linear models are identified. One approach to find this set of operating points is to partition the batch database into a number of clusters using fuzzy c-means clustering.

Assuming full state measurements, let $\bar{\mathbf{X}}' = \begin{bmatrix} \bar{\boldsymbol{x}}_1' & \cdots & \bar{\boldsymbol{x}}_i' & \cdots & \bar{\boldsymbol{x}}_N' \end{bmatrix}$ be a matrix of N columns where each column is a different instance of $\begin{bmatrix} \boldsymbol{x}(k) & \boldsymbol{u}(k) \end{bmatrix}'$ (concatenated states and inputs at sampling instant k) in the batch database. The state-input space in $\bar{\mathbf{X}}'$ can be partitioned into L clusters using fuzzy c-means clustering, which assigns each sample, $\bar{\boldsymbol{x}}_i$, a degree of belonging to a cluster $\ell \in [1, L]$ using a continuous membership function, $M_{i,\ell}$. In fuzzy c-means clustering, the degree of $\bar{\boldsymbol{x}}_i$ belonging to cluster ℓ is taken to be inversely proportional to the squared distance between the point and cluster center, $\boldsymbol{c}_\ell$, and then normalized across all clusters [11], [12]:

$$M_{i,\ell} = \|\bar{\boldsymbol{x}}_i' - \boldsymbol{c}_\ell\|^{-2} / \sum_{\ell=1}^{L} \|\bar{\boldsymbol{x}}_i' - \boldsymbol{c}_\ell\|^{-2} \tag{3}$$

such that $\sum_{\ell=1}^{L} M_{i,\ell} = 1 \ \forall i$. The cluster center is the mean of all the points, weighted by their memberships as: $\boldsymbol{c}_\ell = \frac{\sum_{i=1}^{N} M_{i,\ell}^2 \bar{\boldsymbol{x}}_i'}{\sum_{i=1}^{N} M_{i,\ell}^2}$. For a given L, the cluster centers are computed by iteratively minimizing the objective function, $J = \sum_{i=1}^{N} \sum_{\ell=1}^{L} M_{i,\ell}^2 \|\bar{\boldsymbol{x}}_i' - \boldsymbol{c}_\ell\|^2$ [11], [12]. The algorithm is terminated when changes in the membership function values between successive iterations is smaller than a pre-defined tolerance. As this is a nonlinear optimization, this procedure can possibly terminate at a local minimum; therefore, the optimization is usually repeated numerous times starting from different initial memberships, and the results are selected for the replicate with the minimum objective function value.

III. Integrating Data-based Modeling Methods with Nonlinear Control Tools

In this section, we first propose a multi-model approach for modeling batch systems and work through the underlying details. Then, we integrate the modeling approach into a previously developed framework for fault-tolerant predictive control of batch systems.

A. Multi-model Approach

Assuming a batch database (with full state measurements) has been partitioned into L clusters using fuzzy c-means clustering, the basic idea in the proposed multi-model approach is to identify local linear models around the cluster center points and then combine them with a weighting function to describe the global nonlinear behavior. For the local models, we employ the linear discrete state-space model form. Mathematically, this idea is expressed as follows:

$$\boldsymbol{x}(k) = \sum_{\ell=1}^{L} w_\ell \hat{\boldsymbol{\beta}}_\ell \begin{bmatrix} \boldsymbol{x}'(k-1) & \boldsymbol{u}'(k-1) \end{bmatrix} \tag{4}$$

where w_ℓ is the (normalized) weight given to model ℓ of L total models and $\hat{\boldsymbol{\beta}}_\ell$ defines the ℓ-th local linear model. If the weights corresponding to the training data are known prior to estimating the individual model parameters, (4) becomes linear in $\hat{\boldsymbol{\beta}}_\ell$, and the system identification problem reduces to a regression problem solvable using PLS.

Intuitively, from the process description in (1), the weights for the local linear models should depend on the current value of the states, inputs, and uncertainty realizations since they define the system dynamics. In this work, to determine the weights given initial conditions and inputs, the normalized fuzzy clustering membership function in (3) is used. This implies that the weights for the training data are computed as a byproduct of the clustering step. Because the membership function quantifies the degree to which a state-input combination belongs to each cluster, it is indicative of which local models should be given more weight than the others. 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 (3) as the membership function value corresponding to that cluster will be close to 1 while for the remaining clusters, it will be near 0.

Remark 2 The extension of the multi-model approach for the case of limited measurements is addressed in [13]. Note also that combining weighted local linear models is a general strategy for describing global nonlinear dynamics that has been formalized in literature as piece-wise affine (PWA) (e.g. see [14]), Takagi, Sugeno, and Kang (TSK) [15], and operating-regime based [3], [16] modeling. The key delineating aspects of this work are the clustering algorithm used to partition the training data, the use of latent variable regression techniques to estimate the model parameters, and the use of a generalized continuous weighting function that is entirely data dependent and does not require precise process knowledge (a detailed discussion is available in [17]).

B. Reverse-time Reachability Region Generation using the Data-based Model

Reverse-time reachability regions (RTRRs) have been used in [7], [8] to design predictive controllers for batch systems with useful reachability and fault-tolerant characteristics. The currently available algorithm for generating RTRRs, however, requires a deterministic model, which, in many cases, may be unavailable. In this section, we develop an algorithm to generate data-based or empirical RTRRs.

Due to 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, $\mathcal{B}(\boldsymbol{x}_{\text{des}})$. 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 formal definition of an empirical RTRR is given below.

Definition 1 For the batch process described by (1) with sampling period δ for which a data-based model of the form in (4) has been identified, the empirical RTRR at time $t = t_{\text{f}} - q\delta$, indexed by q, is the following set:

$$\begin{aligned}\hat{\mathcal{R}}_q = \{\hat{\boldsymbol{x}}_0 \mid \hat{\boldsymbol{x}}(0) = \hat{\boldsymbol{x}}_0,\ &\text{Equation 4 for } k = 1, \ldots, q\ \exists \\ &\boldsymbol{u}(k) \in \mathcal{U}\ \forall k = 0, \ldots, q-1 \text{ s.t. } \boldsymbol{x}(q) \in \mathcal{B}(\boldsymbol{x}_{\text{des}})\}\end{aligned} \tag{5}$$

In formulating a RTRR-based MPC design, explicit characterizations of these sets are required. Denote $\|\cdot\|_{\mathbf{Q}}$ as a weighted norm, defined by $\|\boldsymbol{x}\|_{\mathbf{Q}} = \boldsymbol{x}^{\text{T}}\mathbf{Q}\boldsymbol{x}$. In this work, we use ellipsoids of the form $\hat{\mathcal{R}}_q = \left\{\boldsymbol{x} \mid \|\boldsymbol{x} - \boldsymbol{c}_q\|_{\mathbf{P}_q} \le 1\right\}$ where $\boldsymbol{c}_q \in \mathbb{R}^n$ denotes the ellipsoid's center and $\mathbf{P}_q \in \mathbb{R}^{n\times n}$ (positive definite and symmetric) defines its size and orientation. Note that because $q = 0$ corresponds to t_{f}, $\boldsymbol{c}_0 = \boldsymbol{x}_{\text{des}}$ and $\mathbf{P}_0$ 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. Consider the unit ball, $S(0,1) = \{\boldsymbol{x} \mid \|\boldsymbol{x}\| \le 1\}$ and the affine transformation $T(\boldsymbol{x}) = \mathbf{Q}\boldsymbol{x} + \boldsymbol{d}$ where $\mathbf{Q} \in \mathbb{R}^{n\times n}$ is a nonsingular matrix and $\boldsymbol{d} \in \mathbb{R}^n$. Applying the affine transformation to a point on the unit ball, we have $\boldsymbol{y} = \mathbf{Q}\boldsymbol{x} + \boldsymbol{d}$, which implies $\boldsymbol{x} = \mathbf{Q}^{-1}(\boldsymbol{y} - \boldsymbol{d})$. An ellipsoid can then be expressed through an affine transformation of the unit ball: $T(S(0,1)) = \left\{\boldsymbol{y} \mid \|\mathbf{Q}^{-1}(\boldsymbol{y} - \boldsymbol{d})\| \le 1\right\} = \{\boldsymbol{y} \mid \|\boldsymbol{y} - \boldsymbol{d}\|_{\mathbf{V}^{-1}} \le 1\}$ where $\mathbf{V} = \mathbf{Q}\mathbf{Q}' \in \mathbb{R}^{n\times n}$ is a positive definite symmetric matrix. Thus, defining $\mathbf{Q}_q$ and $\boldsymbol{d}_q$ is equivalent to defining $\mathbf{P}_q$ and $\boldsymbol{c}_q$.

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 a RTRR is identified for every sampling instant in the batch. Given the RTRR ellipsoid parameters at $q-1$ and I (pre-determined) points on the surface of a unit ball denoted by $\left\{\boldsymbol{x}_{\text{ub}}^{(1)}, \ldots, \boldsymbol{x}_{\text{ub}}^{(i)}, \ldots, \boldsymbol{x}_{\text{ub}}^{(I)}\right\}$, the following nonlinear program (NLP) is solved to determine $\mathbf{Q}_q$ and $\boldsymbol{d}_q$:

$$\max_{\mathbf{Q}_q, \boldsymbol{d}_q, \boldsymbol{u}^{(i)} \in \mathcal{U}} \det \mathbf{Q}_q \tag{6}$$

$$\text{subject to: } \boldsymbol{x}^{(i)} = \mathbf{Q}_q \boldsymbol{x}_{\text{ub}}^{(i)} + \boldsymbol{d}_q\ \forall i = 1, \ldots, I \tag{7}$$

$$\boldsymbol{x}_{\text{nxt}} = \sum_{\ell=1}^{L} w_\ell \hat{\boldsymbol{\beta}}_\ell \begin{bmatrix} \boldsymbol{x}'^{(i)} & \boldsymbol{u}'^{(i)} \end{bmatrix} \tag{8}$$

$$\|\boldsymbol{x}_{\text{nxt}} - \boldsymbol{c}_{q-1}\|_{\mathbf{P}_{q-1}} \le 1 \tag{9}$$

$$\mathbf{Q}_q = \mathbf{L}_q \mathbf{L}_q' \tag{10}$$

The independent decision variables in this NLP are the ellipsoid parameters ($\mathbf{Q}_q$ and $\boldsymbol{d}_q$) and I control moves. The NLP is formulated to maximize the volume of the current RTRR ellipsoid while ensuring for I 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. Equation (7) represents the affine transformation of the I unit ball points into ellipsoid surface points. Equation (10) represents the Cholesky decomposition of $\mathbf{Q}_q$, where $\mathbf{L}_q \in \mathbb{R}^{n\times n}$ is a lower triangular matrix, and ensures $\mathbf{Q}_q$ is positive definite and symmetric. In theory, I should be set to ∞ such that the optimization problem is solved over the entire surface of the ellipsoid. However, the same conclusions can be reached by choosing a sufficiently large I. Accordingly, I can be increased sequentially until changes in the solution are below some pre-defined tolerance. Note that to verify that a control action exists to drive the states inside the next RTRR for the *internal* points of the ellipsoid, the NLP formulated in [8] was used with substitution of the data-based model for the deterministic model.

C. Empirical Reverse-time Reachability Region Based Model Predictive Control (MPC) Formulation

In this section, we present a MPC design with the control objective of maintaining the states inside empirical RTRRs throughout the batch. To this end, consider a batch system described by (1) for which empirical RTRR estimates have been characterized for a given δ and $\mathcal{B}(\boldsymbol{x}_{\text{des}})$. The control action at sampling instance $q = (t_{\text{f}} - t)/\delta$ is computed by solving the NLP:

$$\min_{\boldsymbol{u}(k) \in \mathcal{U}} J_{\mathcal{R}} = \sum_{k=1}^{P} \alpha \|\hat{\boldsymbol{x}}(k) - \boldsymbol{c}_{q-1}\|_{\mathbf{P}_{q-1}} + \gamma \|\Delta \boldsymbol{u}(k)\|_{\mathbf{R}} \tag{11}$$

$$\text{subject to: } \hat{\boldsymbol{x}}(0) = \boldsymbol{x}(t) \tag{12}$$

$$\text{Equation 4} \tag{13}$$

where $\Delta \boldsymbol{u}(k)$ denotes a vector of differences between successive 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 RTRRs over the prediction horizon, P. The relative importance of the two terms in $J_{\mathcal{R}}$ can be traded off using α and γ.

1) Fault-tolerant Characteristics: A fault during a batch can invalidate the desirable properties of a control design and make the desired end-point unreachable. In cases where a fault is rectified before the batch is complete, there may be a chance to recover the batch, provided the control design has inherent fault tolerant characteristics. For the system described by (1), we consider finite duration faults in the control actuator under the assumption that upon failure, the available control effort is reduced. We define the safe-steering problem as the problem of identifying functioning input trajectories during the failure period (without requiring the value of t^{repair} or an estimate thereof to be known *a-priori*) such that the process can be driven inside the desired end-point neighborhood upon recovery of full control effort.

The key idea in the safe-steering problem is to preserve the states inside RTRRs during the failure period by employing the proposed MPC design. The empirical RTRR MPC design offers a suitable control objective during the fault repair period. In the event of a failure, standard end-point based predictive controllers may no longer be able to meet the desired control objectives and prescribe inputs which drive the process states to a point from where the process is unrecoverable even after fault repair. In contrast, by trying to maintain the process states inside empirical RTRRs, we improve the chances of maintaining the process states in a region from where it is recoverable following fault repair.

IV. Simulation Results

Simulation results of a fed-batch reactor system (subject to varying initial conditions, noisy measurements, and time-varying uncertainty) illustrating the efficacy of the proposed modeling and control design are presented in this section.

A. 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 to generate RTRRs, which are subsequently used to design the RTRR-based predictive controller.

Consider the fed-batch reactor system where an irreversible series reaction of the form $2\mathsf{A} \xrightarrow{k_1} \mathsf{B} \xrightarrow{k_2} 3\mathsf{C}$ take place. The state-space model for this system is derived in [18]. The state vector is comprised of the species concentrations and reactor temperature and volume: $\boldsymbol{x} = \begin{bmatrix} C_{\mathsf{A}} & C_{\mathsf{B}} & C_{\mathsf{C}} & T & V \end{bmatrix}'$ 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 (respectively). The manipulated inputs are the inlet feed rate, F (L/h), and heating coil temperature, T_{hx} (K), $\boldsymbol{u} = \begin{bmatrix} F & T_{\text{hx}} \end{bmatrix}'$, with constraints $\boldsymbol{u}_{\min} = \begin{bmatrix} 0 & 288 \end{bmatrix}'$ and $\boldsymbol{u}_{\max} = \begin{bmatrix} 20 & 360 \end{bmatrix}'$. The primary control objective considered is to drive $\boldsymbol{x}(t_{\text{f}})$ inside a (arbitrarily chosen) desired ellipsoidal neighborhood around $\mathbf{c}_0 = \boldsymbol{x}_{\text{des}} = \begin{bmatrix} 2.752 & 1.601 & 0.8422 & 365.756 & 112.425 \end{bmatrix}'$ characterized by $\mathbf{P}_0 = \text{diag}\{25, 400, 100, 0.04, 1\}$. The batch termination and sampling times are: $t_{\text{f}} = 1$ h and $\delta = 0.025$ h.

1) Data-based Model Development: A database consisting of 40 batches was generated for the fed-batch reactor system using the state-space model. Ten of the batches were set aside as the validation data set. In industry, historical batches are mostly "successful", which entails tracking a set of reference trajectories that terminate at the desired end-point. With two available inputs, reference trajectories of C_{B} and T were chosen to be tracked 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 reasonably smooth input trajectories. For a more realistic representation of plant data, sensor noise, disturbances, and a time-varying parameter were also considered. To simulate disturbances, the temperature of the feed flow, T^{in}, was stochastically varied throughout the duration of each batch around its nominal value (300 K) in the range $295 - 305$ K. The time-varying parameter was chosen to be the heat exchanger coefficient, UA, which had a nominal value of $30,000$ cal/(h $\cdot$ K). At the start of each batch, UA was assigned a value in the range $28,620 - 30,349$ cal/(h $\cdot$ K) and decreased exponentially to simulate fouling.

Given the database, the model identification procedure was as follows. The number of clusters, L, was varied from $10 - 100$. For each L, PLS models were fit (using the PRESS statistic to determine the number of latent variables to retain - see [19]), the state trajectories of the 10 validation batches were predicted back using the models, and the root mean squared error (RMSE) of the predictions for each model was tabulated. The PLS model and number of clusters yielding the lowest RMSE value defined the final model's parameters. For the given database, the lowest RMSE value was obtained with $L = 20$ clusters. Figures demonstrating the results of the model fitting procedure are omitted due to space considerations. The multi-model approach was able to capture the major nonlinearities in the database. Using this model, empirical RTRRs were generated for every sampling instant in the batch.

2) 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 PI control. The performance was measured according to the level set of the desired end-point neighborhood, $\mathcal{B}(\boldsymbol{x}_{\text{des}})$, corresponding to $\boldsymbol{x}(t_{\text{f}})$ or $\|\boldsymbol{x}(t_{\text{f}}) - \boldsymbol{c}_0\|_{\mathbf{P}_0}$. A value of less than unity indicated that $\boldsymbol{x}(t_{\text{f}}) \in \mathcal{B}(\boldsymbol{x}_{\text{des}})$ and the control objective was satisfied.

First, fault-free closed-loop simulations were performed for ten different initial conditions that were not in the training or validation data set, but were verified to be within the empirical RTRR at t_0. Sensor noise, a stochastic disturbance in T^{in}, and a time-varying UA were also considered. The RTRR-based predictive controller was tuned once for a specific set of

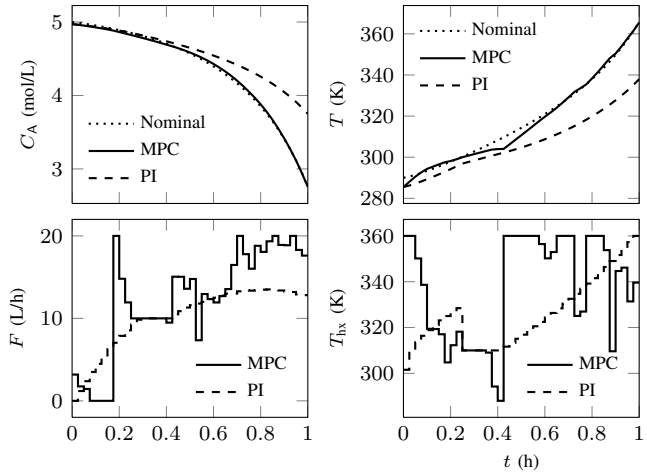

Fig. 1: State and input profiles under PI control and RTRR-based MPC with input failures between 0.25 h to 0.45 h. The nominal state trajectories that terminate at the desired end-point are also shown.

initial conditions and left unchanged for the remainder of the simulation to avoid confounding the results with tuning. For the RTRR-based MPC, the following tunings were used: $\mathbf{R} = \mathrm{diag}\{0.01, 0.005\}$ and $P = 18$.

Under fault-free conditions, the RTRR-based MPC design was able to drive the system inside $\mathcal{B}(\boldsymbol{x}_{\mathrm{des}})$ for all ten initial conditions while the PI controller failed in half of the cases. On average, the final MPC level set was 0.359 whereas for PI control, the final average level set was 2.153. To demonstrate that the MPC problem is efficiently solvable despite being nonlinear, we note that with $P = 18$, the longest CPU time taken time to solve the MPC problem was 0.452 seconds using GAMS with IPOPT as the solver on an Intel Quad Core machine.

Next, we consider faults in both of the control actuators and compare the performance of the RTRR-based MPC design with PI control. Specifically, we consider the scenario where at $t^{\mathrm{fault}} = 0.25$ h, the actuators associated with F and T_{hx} fail and the maximum inputs are reduced to: $\boldsymbol{u}_{\max} = \begin{bmatrix}10 & 310\end{bmatrix}'$. At $t^{\mathrm{repair}} = 0.45$ h, the fault is rectified and full control effort is recovered. The $\mathbf{R}$ matrix used during the fault-free simulations was maintained for the fault scenario. However, 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. The closed-loop profiles for this case are shown in Fig. 1 (only two of the states are shown for space considerations). For the PI controller, the batch is ultimately driven to a level set of 104.393, which is well outside the desired end-point neighborhood. Meanwhile, the RTRR-based MPC drives the process to a final level set of 0.173. During the failure period, the flow rate prescribed by both controllers essentially remain saturated at the new maximum flow rate. For the PI controller, the heat exchanger temperature also remains saturated during the failure period whereas the RTRR prescribes more meaningful temperature towards the latter stages of the fault in trying to maintain the process states within empirical RTRRs.

V. Conclusions

In this work, we addressed the problem of empirically modeling nonlinear batch systems using a multi-model approach. In the proposed multi-model approach, we exploited the availability of historical batch data, the simplicity of local linear models, the data extraction capabilities of PLS, and the use of appropriate clustering and weighting techniques to capture the nonlinear nature of a batch process. The resulting model from this approach was then employed to generate empirical RTRRs, which were subsequently incorporated in an inherently fault-tolerant predictive control design. The efficacy of the RTRR-based MPC design in a faulty and fault-free environment was demonstrated through a fed-batch simulation example.

References

[1] J. Flores-Cerrillo and J. F. MacGregor, "Latent variable MPC for trajectory tracking in batch processes." *J. Process Control*, vol. 15, no. 6, pp. 651–663, 2005.

[2] ——, "Control of batch product quality by trajectory manipulation using latent variable models," *J. Process Control*, vol. 14, no. 5, pp. 539 – 553, 2004.

[3] N. Fletcher, A. Morris, G. Montague, and E. Martin, "Local dynamic partial least squares approaches for the modelling of batch processes," *Can. J. of Chem. Eng.*, vol. 86, pp. 960–970, 2008.

[4] G. Baffi, E. Martin, and A. Morris, "Non-linear projection to latent structures revisited: the quadratic PLS algorithm," *Comp. & Chem. Eng.*, vol. 23, no. 3, pp. 395 – 411, 1999.

[5] M. Alamir and I. Balloul, "Robust constrained control algorithm for general batch processes." *Int. J. Contr.*, vol. 72, pp. 1271 – 87, 1999.

[6] P. Terwiesch, M. Agarwal, and D. W. T. Rippin, "Batch unit optimization with imperfect modelling: a survey." *J. Process Control*, vol. 4, pp. 238 – 58, 1994.

[7] S. Aumi and P. Mhaskar, "Safe-steering of batch processes," *AIChE J.*, vol. 55, pp. 2861–2872, 2009.

[8] ——, "Robust model predictive control and fault-handling of batch processes," *AIChE J.*, 2010, DOI: http://dx.doi.org/10.1002/aic.12398.

[9] L. Ljung, *System Identification: Theory for the User (2nd Edition)*. Prentice Hall PTR, 1998.

[10] P. Geladi and B. Kowalski, "Partial least-squares regression: A tutorial," *Anal. Chim. Acta*, vol. 185, pp. 1 – 17, 1986.

[11] G. A. F. Seber, *Multivariate Observations*. New York, NY, USA: John Wiley & Sons, 1984.

[12] R. Hathaway and J. Bezdek, "Recent convergence results for the fuzzy c-means clustering algorithms," *J. Classif.*, vol. 5, no. 2, pp. 237–247, 1988.

[13] S. Aumi, B. Corbett, and P. Mhaskar, "Data-based modeling and control of nylon-6,6 batch polymerization (accepted)," in *Proc. of the American Control Conference (ACC)*, 2011.

[14] G. Ferrari-trecate, M. Muselli, D. Liberati, and M. Morari, "A Clustering Technique for the Identification of Piecewise Affine Systems," *Automatica*, vol. 39, pp. 205–217, 2003.

[15] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. Syst. Man Cybern.*, vol. 15, no. 1, pp. 116–132, 1985.

[16] T. H. Gottsche, K. J. Hunt, and T. A. Johansen, "Nonlinear dynamics modelling via operating regime decomposition," *Math. Comput. Simul.*, vol. 46, no. 5-6, pp. 543 – 550, 1998.

[17] S. Aumi and P. Mhaskar, "Integrating data-based modeling and nonlinear control tools for batch process control (submitted)," *AIChE J.*, 2011.

[18] H. S. Fogler, *Elements of Chemical Reaction Engineering*, 4th ed. Prentice Hall, 2006, pp. 625–627.

[19] R. Bro, K. Kjeldahl, A. Smilde, and H. Kiers, "Cross-validation of component models: A critical look at current methods," *Anal. Bioanal. Chem.*, vol. 390, pp. 1241–1251, 2008.