Model Learning Predictive Control for Batch Processes: A Reactive Batch Distillation Column Case Study

Alejandro Marquez-Ruiz, Marco Loonen, M. Bahadir Saltik, and Leyla Özkan

Department of Electrical Engineering, Eindhoven University of Technology, De Zaale, Eindhoven 5612 AJ, The Netherlands

ABSTRACT: In this paper, we present the control of batch processes using model predictive control (MPC) and iterative learning control (ILC). Existing combinations of MPC and ILC are based on learning of the inputs of the process from previous batches for a fixed linear time-invariant model (LTI). However, batch processes are inherently time varying; therefore, LTI models are limited in capturing the relevant dynamic behavior. An attractive alternative is to use linear parameter varying (LPV) models because of their ability to capture nonlinearities in the control of batch processes. Therefore, in this work we propose a novel method combining MPC and ILC based on LPV models, and we call this method model learning predictive control (ML-MPC). Basically, the idea behind the method is to update the LPV model of the MPC iteratively, by using the repetitive behavior of the batch process. To this end, three different application-dependent options to estimate the parameters and disturbances of the model are proposed and are compared in simulation on a nonlinear batch reactor. Finally, the ML-MPC with one of the estimation methods is applied to an industrial reactive batch distillation column (RBD) in simulation.

INTRODUCTION

Growing quality demands of consumers, environmental constraints, limited raw materials, and increasing energy demands have pushed the process industry to continuously improve their processes and operations. Continuous processes are often developed for bulk production of a specific product and are therefore less dynamic in their capacity and product variations. Meanwhile, batch processes, which are defined by their start-up and well-defined end of the operation, are generally used for smaller quantities of specialized products, but often have the ability to change their recipes-setup between previous batches for a fixed linear time-invariant model (LTI).

For the control of batch processes, methods such as proportional–integral–derivative (PID), run-to-run, iterative learning, or model based control can be used as described in the work of Bonvin.1 Model predictive control (MPC) is a control method that includes the input and output constraints explicitly in its formulation and, therefore, has become an accepted standard within the process industry for constrained multivariate control problems.2 In a typical implementation, MPC solves an optimization problem subject to system dynamics and constraints. The solution of this optimization problem is a sequence of inputs that can steer the performance output to desired levels. Naturally, the performance of such a control strategy depends on the quality of the model that is used in predicting the process behavior.

There are two basic methods for modeling of systems: modeling based on system identification which uses only input–output data or modeling based on first principles. In the latter, process models with a wide operating window can be obtained. These kinds of models are generally nonlinear and hence require nonlinear optimization methods and can be computationally inefficient in a model based control strategy. An attractive modeling approach to deal with nonlinear systems is linear parameter varying (LPV) models. These models describe the nonlinearity by a set of linear equations by using a scheduling parameter. The use of LPV models in control synthesis has several challenges of which the identification of the model is the most important.3–6 In this direction, we have shown that a well-structured LPV model for reacting systems can be obtained by considering the reaction contribution as a disturbance term and using the extent transformation.7,8 The resulting model has a parameter dependent state space form with a diagonal state matrix. An important advantage of such representation is that the model parameters have physical meaning.

Another important characteristic of batch processes is the repetitive nature of the operation. This could be an advantage for control purposes. While humans have the ability to improve after repetitive mistakes, classical feedback controllers applied in the process industry do not have this capability. If a controller can store information from the control actions calculated in the previous batches and use this information to improve the future and the current performance, then this is called a learning action.

Special Issue: Dominique Bonvin Festschrift

Received: December 30, 2018
Revised: March 29, 2019
Accepted: April 24, 2019
Published: April 24, 2019
In the 1980s, the first learning applications were introduced in the field of robotics and are now widely known in the field of systems and control as iterative learning control (ILC) methods; see the work of Wang et al. for a survey. ILC has the ability to obtain asymptotic convergence over iterations of the same repetitive task while being subject to model uncertainties and iteration-invariant disturbances. The conventional applications of ILC are operated in an open loop manner, where the error of the previous iteration is used to update the trajectory of the next iteration; hence, they are unable to reject real time disturbances. Therefore, some methods have been proposed that have implemented ILC as a feed-forward action in the closed loop control problem; see the work of Wang et al. for more details.

Certainly, the combination of MPC and ILC (IL-MPC) is a good approach in the control of batch processes. This combination has a lot of advantages such as double integration mechanism (in the batch and between batches), constraints satisfaction among others. The general learning procedure with MPC is achieved by correcting the MPC input with past information. The mathematical formulations for IL-MPC can differ in the incorporation of integrators and constraints; see, for example, refs 2 and 13. Despite all of these advantages, the model used for prediction is still a limitation, also, the most of the processes in industry frequently show nonlinear behavior.

For nonlinear processes, such as the batch reactors, several studies have been conducted where a type of IL-MPC is implemented with nonlinear model. For example, refs 16 and 17 propose an iterative nonlinear model predictive control (INMPC) strategy. INMPC has been applied to two different batch reaction ads convergency, stability, and robustness of the closed-loop system have been analyzed and proven. Another nonlinear method, called as the nonlinear model predictive iterative learning control (NMPILC), is described in ref 19 which uses a fuzzy model that contains local linear models. In general terms, to utilize a nonlinear model in IL-MPC is the ideal case, nevertheless, as it is usual in nonlinear MPC, there are a lot of open issues when it comes to implementation such as computational load. Therefore, to close the gap between accuracy in the prediction and implementation in real time, the combination of IL-MPC with LPV models arises as a good alternative.

In this work, we propose a method not only to improve the performance of an MPC controller by utilizing the repetitive behavior of batch processes in an iterative learning fashion but also to overcome the limitations of MPC approaches using LPV models by updating the MPC and disturbance models. Similar strategies have been proposed for classic ILC; however in this work, all these ideas are combined in just one method called model learning predictive control (ML-MPC). This method inherits the advantages of ILC and MPC while a suitable model in structure (linear) and accuracy is used for prediction in every batch. For the estimation of the parameters and disturbance, we evaluate three different methods that cover a broad range of scenarios. The performance of each method is evaluated implementing them on a simple nonlinear batch reactor. Furthermore, one of the proposed learning methods is used for the control of an industrial reactive batch distillation (RBD) column.

This paper is organized as follows: In Preliminaries, we provide the basic principles of MPC, MPC for LPV models, ILC, and the current methods of iterative learning model predictive control (IL-MPC). Problem Statement presents the main problem we will be solving. In Model Learning Predictive Control, we describe our newly proposed method, including implementations on a simple example of a batch reactor. In Reactive Batch Distillation Processes, the method is applied to a more advanced process. Finally, in Conclusions, we summarize our findings.

**Preliminaries**

**MPC for LPV Systems.** MPC is a well-known control technique that uses a model to predict the future states and outputs. It solves an optimization problem, in which, usually, a quadratic cost function is minimized. In this way, the controller can steer the process outputs toward the reference profiles. The main advantage of MPC is that it can consider explicitly constraints on inputs, states, and outputs in the optimization problem. The optimization problem is solved at every time sample to determine the best sequence of inputs \( u_{k+j}^{N} \) for a given prediction horizon \( N \in Z_{+} \). The first input from this sequence is applied, and at the next time sample, the same optimization problem is solved again. This is known as the receding horizon strategy.

Despite these advantages of MPC, the use of this technology for the control of batch processes is limited if it exists at all. This is mainly due to the nonlinearities and absence of steady states. Therefore, typical linear MPC techniques could not be implemented. For batch processes, MPC based on a first-principles nonlinear model is a suitable approach. Certainly, there are different approaches to deal with these issues such as multiple models, successive linearization, or empirical nonlinear models such as artificial neural networks; however, these approaches are often subject to performance limitations and computational issues. An attractive approach to deal with batch processes and nonlinear systems is to use LPV models in the MPC formulation. With such a representation we have the potential to capture nonlinear and time varying characteristics of systems but also use linear control theory which is well developed.

In the available MPC methods for LPV systems, the future behavior of the parameter (scheduling variable) is assumed unknown over the prediction horizon. One of the most used method to solve the MPC problems for this kind of uncertainty is the worst case scenario which can be described by eq. 1.

\[
\min_{\delta u_{k+j}} \max_{\theta} \sum_{j=0}^{N-1} \|x_{k+j}\|^2_{Q} + \|u_{k+j}\|^2_{R} + \|\delta u_{k+j}\|^2_{D}
\]

subject to:

\[
x_{k+j+1} = A(\theta_{k+j})x_{k+j} + B(\theta_{k+j})u_{k+j} + B_{d}d_{k+j}
\]

\[
y_{k+j} = Cx_{k+j} + Du_{k+j}
\]

\[
e_{k+j} = r_{k+j} - y_{k+j}
\]

\[
u_{k+j} = u_{k+j-1} + \delta u_{k+j}
\]

\[
x_{k+j} \in X, u_{k+j} \in U, x_{k+N} \in \Omega
\]

\[
d_{k+j} \in D, \theta_{k+j} \in \Theta
\]

(1)

where \( x_{k+j} \in R^{n} \) denotes the system state, \( y_{k+j} \in R^{m} \) is the system output, \( u_{k+j} \in R^{p} \) is the control vector, \( d_{k+j} \in R^{r} \) are disturbances, and \( \|x\|^2_{Q} = x^TQx \) with \( Q > 0 \). The system is subject to hard constraints on state \( x_{k+j} \in X \) and input \( u_{k+j} \in U \), where
\( \mathbb{X} \subset \mathbb{R}^n \), and \( \mathbb{U} \subset \mathbb{R}^m \) are assumed convex and compact. Finally, \( \Omega \subset \mathbb{X} \) is an invariant set of the system. In this work, the sets \( \mathbb{U} \), \( \mathbb{X} \), and \( \Omega \) are assumed polyhedrons, i.e., described by linear inequalities of the form

\[
\begin{align*}
\mathbb{U} &= \{ u_{k+1} | A_uu_{k+1} \leq b_1 \} \\
\mathbb{X} &= \{ x_{k+1} | A_xx_{k+1} \leq b_2 \} \\
\Omega &= \{ x_{k+N} | A_x x_{k+N} \leq b_N \}
\end{align*}
\]

(2)

The main issue with the formulation given by optimization problem 1 is the computational demand. Therefore, it has led to approaches in which the problem is approximated (see refs 23 and 28). We also find multiparametric approaches described as explicit MPC for LPV systems\(^{26}\) and tube based MPC (TMPC)\(^{29}\). In ref 26, the high computational issues are reduced by precomputing the optimal inputs as a piecewise affine function of the state. The precomputed solutions are stored in a look-up table, such that only this look-up table can be used online.\(^{26}\) However, this method only works well if the size of the system is low.\(^{30}\) On the other hand, TMPC is based on the assumption that, while the current value of the scheduling parameter is measured exactly, the future variations in the scheduling variable belong to a sequence of sets \( \Theta_{jk} \) (tube) that describe the expected deviations from the nominal trajectory.\(^{25}\)

With this, the controller is able to derive the worst-case cost for expected future changes only, resulting in a single linear program. The sequence \( \Theta_{jk} \) can be constructed in several ways, namely classical, anticipative, and oracle, each corresponding to a different MPC scenario.\(^{25}\) An additional interesting approach called MPC for Quasi-LPV systems (QMPC) has been proposed by refs 31 and 32. In this formulation, it is assumed that the scheduling parameters \( \theta_{jk} \) and disturbances \( d_{uv} \) can be calculated by means of explicit functions of the states and inputs as \( d_{uv} = f(x_{uv}, u_{uv}) \) and \( \theta_{jk} = f_\theta(x_{uv}, u_{uv}) \). With this assumption, the min–max optimization problem given in 1 can be avoided. The main drawback of this method is to find the functions \( f(x) \) and \( f_\theta(x) \). However, for chemical process, most of the time such functions are given by constitutive equations of the physical parameter of the process, which is advantageous if a first principle model is utilized in the controller.

In general, the MPC methods for LPV systems proposed in the literature are well-defined mathematically, nevertheless, real time implementations are difficult due to their complexity and high computational load. In spite of all these issues, TMPC and QMPC combined with iterative learning control (ILC) have inspired some of the methods proposed in this paper.

Iterative Learning Control. ILC is the control method that uses data from previously performed iterations. With this data, the controller is able to learn from past “mistakes” and correct for this. Figure 1 illustrates the ILC procedure. In this figure \( P \) is the plant, and \( L \) and \( Q \) are filters. Notice in Figure 1 how the control signal induces an error signal through the plant with disturbance and how this error signal of iteration \( j \) is used for the determination of the control signal in iteration \( j + 1 \).

In classical ILC, the goal is to generate the optimal feed-forward input. The first solution for this has been applied by Arimoto on robotics,\(^{10}\) which is the also called as P-Type ILC,\(^{33}\) first-order learning algorithm,\(^{12}\) or Arimoto algorithm\(^{34}\) and is represented by

\[
u_j^i = u_j^{i-1} + L e_j^{i-1}
\]

\( \forall \ i = 1, \ldots, \infty, \forall \ j = 1, \ldots, N \)

(3)

where \( u_j^i \in \mathbb{R}^n \) is the input, \( e_j^{i-1} = y_j - y_j^{i-1} \in \mathbb{R}^n \) is the error between the output and the reference, \( N \) is the number of samples in batch, and \( L \) is a linear time-invariant (LTI) filter called a learning filter.\(^{12}\) The indices \( i \in \mathbb{Z}_{\geq 0} \) and \( j \in \mathbb{Z}_{\geq 0} \) indicate the batch index and the time sample within a batch, respectively.

To ensure that the ILC algorithm is learning the system dynamics and not the disturbances, the filter \( L \) is designed to be the inverse of the process sensitivity.\(^{35}\) The drawback of this function is that high-frequency terms of the input can increase continuously causing instability across batches. To overcome this problem and ensure convergence to a steady input profile, a robustness filter \( Q \) is added to this algorithm as can be seen in Figure 1 and eq 8. This robustness filter usually is a type of low-pass filter.\(^{35}\)

\[
u_j^i = Q [u_j^{i-1} + L e_j^{i-1}]
\]

(4)

The ILC algorithm can now be executed as shown in Algorithm 1. Feedback control can also be included in the closed loop operation with ILC in the so-called current-iteration iterative learning control.\(^{36,37}\) Then, the control update algorithm is given by

\[
u_{k+j} = \frac{Q[u_{k+j}^{i-1} + L e_{k+j}^{i-1}]}{\text{learning control}} + \frac{C_{k+j}}{\text{feedback control}}
\]

Here, the feedback control has the ability to react to batch specific disturbances. Usually, in the corresponding control scheme, as shown in Figure 1, only PD controllers are considered,\(^{36}\) as ILC has a natural integration component from one batch to the next.

ILC strategy requires repetitive processes like robotics and batch processing and if this is the case it can reject repetitive errors using past data. It adds a batch to batch integrator to the control loop and the noncausal behavior can overcome the delay in the error suppression. Additionally, model information can be used to improve the convergence speed. However, it requires error and input filtering to prevent instabilities.\(^{36}\)

Iterative Learning Model Predictive Control (IL-MPC). Different methods have already been developed to combine the benefits of the MPC control with iterative learning capabilities, as seen in the work of Lee and Lee\(^{12}\) and Adam and González.\(^{13}\)

---

**Figure 1.** Illustration of an ILC control algorithm.
In general, the IL-MPC problem considers additive disturbances and mathematically can be written as

$$\min_{\Delta \delta u_{k+j}} \| x_{k+N} \|_p^2 + \sum_{j=0}^{N-1} \| u_{k+j} \|_Q^2 + \| \Delta \delta u_{k+j} \|_R^2$$

subject to:

$$x_{k+j+1} = A x_{k+j} + B u_{k+j} + B_2 d_{k+j}$$
$$y_{k+j} = C x_{k+j} + D u_{k+j}$$
$$e_{k+j} = r_{k+j} - y_{k+j}$$
$$u_{k+j} = u_{k+j-1} + \delta u_{k+j}$$
$$\Delta \delta u_{k+j} = \delta u_{k+j} - \delta u_{k+j-1}$$
$$x_{k+j} \in \mathcal{X}, u_{k+j} \in \mathcal{U}$$
$$x_{k+N} \in \Omega, d_{k+j} \in \mathcal{D}$$  \hspace{1cm} (5)

In this formulation $\Delta$ describes the input change rate from one batch to the next. Similar to how $\delta$ includes an integrator in the in-batch time direction, $\Delta$ introduces an extra integrator in the batch-to-batch direction. Another feature of this $\Delta$ is that the speed of changes over batches can be constrained. Now, in order to calculate the optimal control solution $\Delta \delta u_{k+j}$, problem 5 can be transformed into a quadratic programming (QP) problem.

### PROBLEM STATEMENT

In the previous sections, we have described MPC and ILC strategies separately and also presented information on control approaches which combine both technologies. In IL-MPC approaches, the optimal input is calculated using the predefined model as shown in eq 5. Changes in the plant from batch to batch causing model degradation have not been considered. Besides, MPC-LPV methods are difficult for online implementation. Motivated by these observations, we present the following problem:

$$\min_{\Delta \delta u_{k+j}} \| x_{k+N} \|_p^2 + \sum_{j=0}^{N-1} \| u_{k+j} \|_Q^2 + \| \Delta \delta u_{k+j} \|_R^2$$

subject to:

$$x_{k+j+1} = A(\theta_{k+j}) x_{k+j} + B(\theta_{k+j}) u_{k+j} + B_2 d_{k+j}$$
$$y_{k+j} = C x_{k+j} + D u_{k+j}$$
$$e_{k+j} = r_{k+j} - y_{k+j}$$
$$u_{k+j} = u_{k+j-1} + \delta u_{k+j}$$
$$\Delta \delta u_{k+j} = \delta u_{k+j} - \delta u_{k+j-1}$$
$$x_{k+j} \in \mathcal{X}, u_{k+j} \in \mathcal{U}, x_{k+N} \in \Omega$$
$$d_{k+j} \in \mathcal{D}, \theta_{k+j} \in \Theta$$  \hspace{1cm} (6)

Our principal objective, therefore, is to design an MPC controller to achieve constrained tracking of the LPV representation of the batch process for a given reference trajectory. As this is a complicated problem to solve due to the need of knowledge about $\theta_{k+j}$ and $d_{k+j}$ we propose an iterative method to derive these parameters from previous batches. We call this strategy model learning predictive control.

### MODEL LEARNING PREDICTIVE CONTROL

In this paper, we propose a method where we update the model for control, such that the MPC controller performance can be improved over batches. To this end, we use the available data from the previous batch to (re)estimate the model parameters, as illustrated in Figure 2.

In this illustration, we can see two time directions, the first one being the in batch time $k$, and the second being the sequential batches denoted by $i$. After the initial batch, data is used to get estimations for $\theta^i_{k+j}, d^i_{k+j}$ for the next batch. The MPC controller then computes the optimal inputs for the next batch using these updated parameters. And this sequence is further repeated as described step by step in Algorithm 2.

#### Algorithm 2 Model Learning Predictive Control Algorithm

1. procedure Model Update Process
2. Define the LPV model structure;
3. Set the initial parameter $\theta^1$;
4. Set the initial disturbance vector $d^1$;
5. Set $i = 1$;
6. Run a single batch, with an MPC controller using the LPV model;
7. Save all the available data, such as the trajectories of inputs $u^0$, outputs $y^0$, and error $e^0$;
8. Set $i = i + 1$;
9. Perform the estimation to obtain the new $\theta^i$ and $d^i$;
10. Update the MPC model with these new parameters and disturbances;
11. Return to step 6;
12. end procedure.

This method is shown in the control block diagram in Figure 3.

### Controller Formulation

As explained before we utilize an MPC controller with an LPV model in this ML-MPC method. Additionally, we include two integrators by the use of both $\delta$ and $\Delta$ in the MPC formulation as described by eq 7.
In this problem, we include the estimation of the trajectory of the parameter \( \theta_{k+j} \) and the time-varying disturbance \( d_{k+j} \), which are determined from previous batches. In the learning method proposed in this paper, the parameters and disturbances can be estimated using multiple methods. Particularly, we propose three different methods inspired in the MPC for LPV systems approaches that interpret the available data in different ways, such methods are presented in the next section.

**Remark 1** Note that optimization problem 7 is equivalent to the traditional IL-MPC when the parameter \( \theta_{k+j} \) is constant during the batch. This fact shows that IL-MPC is a particular case of the method proposed in this paper (ML-MPC); also, the performance of the controller given by 7 is at least the same as that of 5.

**Parameter and Disturbance Estimation Methods.** We have considered three different methods for estimating the model parameters and disturbances. The first one is inspired by the Quasi-LPV approach. The second one is formulating the estimation problem as an identification problem, while the third method is inspired by adaptive control in the batch direction. All these methods are explained in the following.

**Method 1.** In the case of chemical processes it is common to have constitutive relations. Therefore, it makes sense to approach the estimation problems in a similar fashion as that seen in ref 31. In this method, we use the prior knowledge of the system (constitutive relations) to define a function of the parameters and disturbances. We can describe this method by defining the following set of equations for \( \theta_{k+j} \) and \( d_{k+j} \):

\[
\begin{align*}
\hat{\theta}_{k+j} &= f_\theta(x_{k+j}, u_{k+j}) \\
\hat{d}_{k+j} &= f_d(x_{k+j}, u_{k+j})
\end{align*}
\]

To illustrate this method, let us consider a simple batch reactor given by the following set of nonlinear equations:

\[
\begin{align*}
\frac{dT}{dt} &= \frac{UA}{MCP} (T - T_f) - \frac{\Delta HV}{MCP} k_0 e^{-E/RT} C_A^2 \\
\frac{dC_A}{dt} &= -k_0 e^{-E/RT} C_A^2
\end{align*}
\]

Be defining \( x_1 = T \), \( x_2 = C_A \), \( u = T_f \), \( \theta = \frac{UA}{MCP} \), and \( d = -\frac{\Delta HV(T)}{MCP(T)} k_0 e^{-E/RT} C_A^2 \), the model can be written as

\[
\dot{x} = -\theta(x - u) + d
\]

where \( \theta \) and \( d \) can be calculated by means of the following functions

\[
\begin{align*}
\theta &= f_\theta(x, u) = \frac{UA}{MCP(x)} \\
d &= f_d(x, u) = -\frac{\Delta HV(x)}{MCP(x)} k_0 e^{-E/RT} x_2^2
\end{align*}
\]

Note that, the parameters and disturbances can be updated in the \( n \)th batch by means of \( f_\theta(\cdot) \) and \( f_d(\cdot) \). More detail about this example is given in the next section (Illustrative Example: Nonlinear Batch Reactor). Naturally, in some cases, it is not possible to have such function for both the parameters and the disturbance; however, the method can still be used if we have a function only for either the parameters or the disturbance. An advantage of Method 1 is the ability to estimate a complete trajectory for time-varying parameters. However, it requires additional measurements besides the available input–output data, namely state information. Finally, the original optimization problem can be written as

\[
\begin{align*}
\min_{\Delta \theta_{k+j}, \Delta d_{k+j}} & \| x_{k+j+N} \|^2_p + \sum_{j=0}^{N-1} \| \delta x_{k+j} \|^2_Q + \| \Delta \delta u_{k+j} \|^2_R \\
\text{subject to:}& \\
x_{k+j+1} &= A(\theta_{k+j})x_{k+j} + B(\theta_{k+j})u_{k+j} + B_d d_{k+j} \\
y_{k+j} &= C x_{k+j} + D u_{k+j} \\
\delta x_{k+j} &= u_{k+j} - \theta_{k+j} - \theta_{k+j}^{-1} \\
\Delta \delta u_{k+j} &= \delta u_{k+j} - \delta u_{k+j}^{-1} \\
\Delta \delta u_{k+j} &= \delta u_{k+j} - \delta u_{k+j}^{-1} \\
x_{k+j} \in \mathbb{X}, u_{k+j} \in \mathbb{U}, x_{k+j+N} \in \Omega
\end{align*}
\]

\[ (7) \]

**Method 2.** In this method we use an approach similar to the data driven identification as seen in Forgione et al.\(^{38}\) We use an optimization problem to estimate the parameters and disturbance from the input and output data. The cost function of this problem is user defined and depends on the a priori knowledge the user has about the process. This problem in general can be described by

\[
[\hat{\theta}_{k+j}, \hat{d}_{k+j}] = \arg \min_{\theta_{k+j}, \theta_{k+j}, d_{k+j}} J(x_{k+j}^{-1}, u_{k+j}^{-1})
\]

subject to:

\[
\begin{align*}
\Delta \theta_{k+j} &= A(\theta_{k+j}) x_{k+j}^{-1} + B(\theta_{k+j}) u_{k+j}^{-1} + B_d d_{k+j} \\
y_{k+j} &= C x_{k+j}^{-1} + d u_{k+j}^{-1} \\
x_{k+j} &\in \mathbb{X}, u_{k+j} \in \mathbb{U}, x_{k+j+N} \in \Omega \\
\theta_{k+j} &\in \Theta, d_{k+j} \in \mathbb{D}
\end{align*}
\]

\[ (11) \]

where \( J(\cdot) \) is usually the typical quadratic cost function used in least squares problems. Note that problem 12 can be converted...
into a quadratic programming problem. This is a significant advantage compared with the proposal of ref 38 since both the parameter and disturbance can be estimated by solving a single optimization problem. Despite its advantages, this method is suitable for real time applications only if the parameter \( \theta_0 \) does not vary in time \( \theta_0 = \theta \). Otherwise, problem 12 is a nonlinear optimization problem, which is usually more difficult to solve. Another problem with this method is that by using input-output data only, the optimal solution could be affected by the noise in the system resulting in undesirable behavior by the controller. On the other hand, it can happen that the number of parameters and disturbances are larger than the size of the input-output data. This results in an ill posed problem. These issues are analyzed in the example presented in the next section (Illustrative Example: Nonlinear Batch Reactor). Finally, with Method 2, the original optimization problem does not change its formulation.

**Method 3.** This third method contains some similarities to the original ILC method, where the error trajectory of the batch is utilized to estimate the parameters and disturbance. Method 3 is inspired in adaptive control strategies in the presence of time-varying parameters. Although \( \theta \) is considered constant during the batch, this parameter could change over batches due to plant-mismatch, as the actual value of \( \theta \) is not satisfactory. This is not surprising due to the plant-uncertainty. In Table 1, we consider the same MPC settings. We set a desired temperature profile as shown in Figure 4.2

\[
\text{Proj}(\theta, y) = \begin{cases} 
  y, & \text{if } f(\theta) < 0 \\
  \frac{\nabla f}{\| \nabla f \|} y, & \text{if } f(\theta) \geq 0 \text{ and } \nabla f^T y \leq 0 \\
  \frac{\nabla f}{\| \nabla f \|} \left( f(\theta) - \frac{\nabla f}{\| \nabla f \|} y \right), & \text{if } f(\theta) \geq 0 \text{ and } \nabla f^T y > 0 
\end{cases}
\]

(14)

As we stated in a previous sections, the nonlinear reactor model eqs 15 can be written as

\[
\dot{x} = -\theta(x - u) + d
\]

(16)

In this example, \( \theta \) is assumed as a constant parameter and \( d \) a time-varying disturbance. Although \( \theta \) is considered constant during the batch, this parameter could change over batches due to fouling, as it includes the heat transfer coefficient of the heating jacket. In comparing the three methods, we consider the same MPC settings. We set a desired temperature profile as shown in Figure 4.2

![Figure 4. Reactor temperature: reference profile.](image)

The MPC is tuned with the control horizon \( N = 30 \), the cost matrices \( Q = 10^3, R = 10^4 \), and the inputs are in the sets \( u \in [-10, 0], \Delta u \in [-10, 10], \) and \( \theta_0 = 3 \) and initial disturbance vector \( d^0 = [0 \cdots 0]^T \). With these parameter and disturbance values, the performance of the MPC in tracking of the reference trajectory is not satisfactory. This is not surprising due to the plant-model mismatch, as the actual value of \( \theta \) is 0.09 and the disturbance trajectory is shown in Figure 5.

In the following, we study the performance of these methods in simulation on this batch reactor.

**Method 1.** For this specific example we do not have a function to estimate \( \theta \) as in eq 8. Therefore, we assume that we are able to
estimate the parameter perfectly such that \( \hat{\theta} = \theta \). Applying this parameter learning with the ML-MPC algorithm results in the reference tracking improvement as shown in Figure 6.

To improve the reference tracking further, Method 1 can be implemented on the batch reactor to estimate the disturbance vector based on the measurable temperature \( T \) and concentration \( C \) in the reactor. From the differential equation of the reactor, we know that the reaction rate can be described as the disturbance as shown in eq 17.

\[
\hat{d}_k = f_d(T_k, C_k) = -\frac{\Delta H V}{M C_p} k_0 e^{-E/RT} \beta_k (t-T) - 12
\]

which results in the improvement of reference tracking, from \( t = 0 \) min until \( t = 6 \) min, as shown in Figure 7. In this figure, the reference tracking improvement is clearly seen at the beginning of the batch. This is expected because the reaction rate has the highest amplitude in the start of the batch.

Method 2. In this method, we use Euler approximation to represent the differential equation as a difference equation. This derivation is explained as follows

\[
\dot{x} = \frac{x_{k+1} - x_k}{\Delta t} = \theta \left( \frac{T_j - T}{a_k} \right) + d_k, \forall k = 0, \ldots, N
\]

\[
\begin{bmatrix}
\beta_1 \\
\vdots \\
\beta_N
\end{bmatrix}
= 
\begin{bmatrix}
\alpha_1 & 0 & 0 \\
\vdots & \ddots & \vdots \\
\alpha_N & \cdots & 1
\end{bmatrix}
\begin{bmatrix}
\hat{\theta} \\
\vdots \\
\hat{d}_N
\end{bmatrix}
\]

which we can calculate \( y \) using the following least-squares problem after substituting the input and output data in \( a \) and \( b \)

\[
\min_y \|ay - b\| \quad (19)
\]

This optimization problem, however, has two issues. First of all, it is ill-posed as there are more parameters than the rank of the matrix \( a \). The second problem is that any noise in the input or output affects the estimation of the disturbance as shown in Figure 8.

If we evaluate the result of the \( \hat{d} \) for this estimation, we see that the noise in the estimated \( \hat{d} \) is much larger than actual \( d \) as shown in Figure 9.

A solution to overcome such problems is utilizing a regularization method. In the case of the batch reactor, we know that the reaction rate converges to zero and therefore the \( L_1 \) or "lasso" regularization is a good option. The least-squares problem then becomes

\[
\min_y \|ay - b\| + \lambda \|y\|_1 \quad (20)
\]

where \( \lambda \) is a tuning parameter with \( \lambda = 100 \). The solution of this problem for the first batch is shown in Figure 10. We can notice how the regularization term filters the solution from the system noise.
The result of the estimation for $d$ is shown in Figure 11. In this figure, we see that there is still some fluctuations in the least-squares estimation of $d$. These fluctuations can, however, be filtered using a lowpass filter to obtain a good estimation of the trajectory $\hat{d}$. This is shown by the “filtered $L_1$ LS” result, compared with $d$.

Now we can implement the estimated $\hat{\theta}$ in the controller model for the next batch. The result for the updated parameter is shown in Figure 12. In this result, we see that the estimation of $\hat{\theta}$ results in a direct improvement step toward the reference tracking performance. By implementing the estimated $\hat{d}$ for the next batch, we see an improvement in the reference tracking performance of the controller for the start of the batch as shown by Figure 13.

**Method 3.** Method 3 is implemented using the learning function in eq 21 based on the error.

$$\hat{\theta}_{k+1}^{i} = \hat{\theta}_{k+1}^{i-1} + \Gamma_{p} \text{Proj}(\hat{\theta}_{k+1}^{i-1} - \Delta_{k+1}^{i-1} \epsilon_{k+1}^{i-1})$$

$$d_{k}^{i} = d_{k}^{i-1} + \lambda \Delta_{k}^{i-1}$$

(21)

With $\lambda = 0.25$ we obtain the result presented in Figures 14 and 15.

An interesting observation of these results is related to the stability of this learning method. In the original ILC algorithm using eq 3 it is known that high frequent input oscillations can reduce the controller performance over iterations, whereas in the case of this implementation in the MPC controller the $\delta$ formulation prevents these high frequent fluctuations by the penalty on the input change $\Delta u$, and therefore helps to stabilize this learning algorithm.

It is important to highlight that some methods are better in the disturbance learning, and others better in parameter learning, and therefore we will compare both aspects separately for the specific batch reactor. We compare the reference tracking performance over batches by taking the $\|e\|$ as the performance measure.

In Figure 16, we present the performance of each method for the learning of the parameter. In this figure, 10 batches are displayed where the parameter of the plant is changed from $\theta = \ldots$. 

Figure 10. $L_1$ least-squares estimation: $T - T(0)$.

Figure 11. $L_1$ least-squares estimation: disturbance $d$.

Figure 12. Input and outputs with Method 2.

Figure 13. Input and outputs with Method 2: From $t = 0$ min until $t = 6$ min.

Figure 14. Input and outputs with Method 1.

Figure 15. Input and outputs with Method 3: From $t = 0$ min until $t = 6$ min.
are the liquid, gas, and reaction molar enthalpies, and the external heat flux. From the results we can conclude that the proposed methods offer good controller improvement over batches, thus being able to take into account changes in the system. The results, however, also show that each estimation method have its own strengths and limitations.

To implement the ML-MPC, it remains important to have good knowledge of the process in order to select the right estimation method for each application. We have also observed that for the batch reactor Method 1 would be the best option for implementation, under the assumption that both temperature and concentration can be measured. If the concentration estimation in the plant would not be possible, Method 2 is a good alternative.

In the next section, we apply the ML-MPC algorithm to a more advanced process in a reactive batch distillation column using only one of the learning methods.

### REACTIVE BATCH DISTILLATION PROCESSES

A reactive batch distillation (RBD) column is a unit operation utilized generally for chemical processes involving equilibrium reactions. With the use of a distillation column, light products are removed so that the reaction proceeds in the products direction. A general form of RBD is illustrated in Figure 18. The reactor of such RBD is enclosed by a jacket which can heat or cool the reactor. On top of the reactor is the distillation column with multiple trays. The vapor rising from the reactor and the liquid flow (reflux) coming from the top of the column get into contact at the tray with liquid holdup.

**Rigorous Dynamic Model of RBD.** A general rigorous dynamic model for the $i$th stage of the RBD is formulated and given by the following set of equations,

\[
\begin{align*}
\frac{dn_{i,j}}{dt} &= L_{i-1}x_{i-1,j} - L_{i}x_{i,j} - \xi_{ij} + n_{L}r_{ij} \\
\frac{dn_{G,i}}{dt} &= V_{i} + V_{i+1} - V_{i,j} + \xi_{ij} \\
\frac{dn_{T,i}h_{i}}{dt} &= L_{i-1}h_{i-1} - L_{i}h_{i} - V_{i}H_{i} + V_{i+1}H_{i+1} + n_{L} + \sum_{j=1}^{N_{T}} (-\Delta H_{i,j})n_{L} + Q_{ext,i} \\
\forall i &= 1, \ldots, N_{T}, \forall j = 1, \ldots, N_{G} \quad (22)
\end{align*}
\]

where $n_{L,j}$, $n_{G,i}$, $r_{ij}$, $\xi_{ij} \in \mathbb{R}$ are the numbers of moles in the liquid gas phases and the reaction rate component and mass transfer rate of the $j$th component, respectively. $V_{i} \in \mathbb{R}$ and $L_{i} \in \mathbb{R}$ are the outlet gas and liquid flow, and $V_{i+1} \in \mathbb{R}$ and $L_{i-1} \in \mathbb{R}$ are the inlet gas and liquid flow from the $(i+1)$th and $(i-1)$th trays. Furthermore, $h_{i}H_{i} - \Delta H_{i,j}Q_{ext,i} \in \mathbb{R}$ are the liquid, gas, and reaction molar enthalpies, and the external heat flow, respectively. Besides, $n_{T,i}$ is the total number of moles and is given by $n_{T,i} = n_{L} + n_{G}$, where $n_{L}$ and $n_{G}$ are the total liquid...
The RBD consists of 6 stages \((N_t = 6)\) with only 3 internal trays. Stage equilibrium is assumed in the process with NRTL activity coefficient model for the liquid phase; molar vapor holdup is negligible with respect to the molar liquid holdup, an isothermal total condenser is assumed, and the reaction is limited to the reboiler (reactor). Only four species are assumed to evaporate \((p = 4)\). In total, the chemical process has 8 species \((N_C = 8)\) with 9 reactions, but 5 independent \((N_R = 5)\), i.e., the stoichiometric matrix \(N^T \in \mathbb{R}^{8 \times 9}\) but has rank 5. The following are mole and temperature initial conditions in the reactor: 
\[
n_{\text{init,RBD}}(0) = 20 \text{ kmol}, \quad n_{\text{init,R } \text{OH}}(0) = 20 \text{ kmol}, \quad \text{a n d} \quad T_{N_t}(0) = 373 \text{ K}.
\]

**Control Objective.** In the RBD, the objective is to steer the temperature of the reactor to a reference temperature. This reference is a predefined trajectory to obtain optimal product qualities from a batch in the RBD. This reference profile is shown in Figure 19. With this temperature profile, the water in...

---

Table 2. Conditions of the Rigorous Dynamic Modeling of RBD Processes

<table>
<thead>
<tr>
<th>(i)</th>
<th>(L_{i-1})</th>
<th>(L_i)</th>
<th>(V_{ext,1})</th>
<th>(V_i)</th>
<th>(\xi_{ij})</th>
<th>(x_{i-1,j})</th>
<th>(\tau_{ij})</th>
<th>(Q_{int,ij})</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(L_0)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>(x_{e,1})</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>(L_1 + L_0)</td>
<td>(V_{ext,1})</td>
<td>0</td>
<td>(\xi_{ij})</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(N_t)</td>
<td>(L_{N_t-1})</td>
<td>0</td>
<td>0</td>
<td>(V_i)</td>
<td>(\xi_{ij})</td>
<td>(x_{i-1,j})</td>
<td>(T_{N_t,ij})</td>
<td>(Q_{int,ij})</td>
</tr>
</tbody>
</table>

and gas molar holdup. The liquid mole fraction \(x_{ij}\) and the vapor mole fraction \(y_{ij}\) are given by

\[
n_t = \sum_{j=1}^{N_\text{C}} n_{i,j}, \quad n_G = \sum_{j=1}^{N_\text{C}} n_{i,j}\n_{G,j} = x_{ij} = \frac{n_{i,j}}{n_t}, \quad y_{ij} = \frac{n_{i,j}}{n_G},
\]

\(\forall \quad i = 1, \ldots, N_C, \quad j = 1, \ldots, N_t\).

The model given by the set of eqs 22 is subject to the following conditions presented in Table 2.

**LPV Representation of an RBD Column.** The extent decomposition for RBD processes can be achieved finding a linear transformation \(T_{i}^{*}\) and \(T_{i}^{**}\) similar to the one computed in Amrhein et al.\(^{40}\) such that system 22 can be represented in terms of new independent states for the gas and the liquid phase as follows:

**Liquid phase**

\[
\begin{bmatrix}
    x_{i,j} \\
    x_{\text{in},j} \\
    \lambda_{i,j}
\end{bmatrix} =

\begin{bmatrix}
    T_{i}^{**} \\
    T_{i}^{*} \\
    T_{i,\text{ext}}
\end{bmatrix}
\begin{bmatrix}
    n_t \\
    n_G \\
    n_{\text{ext}}
\end{bmatrix}
\]

\(\forall \quad i = 1, \ldots, N_C, \quad j = 1, \ldots, N_t\).

**Gas phase**

\[
\begin{bmatrix}
    x_{\text{in},j} \\
    \lambda_{i,j}
\end{bmatrix} =

\begin{bmatrix}
    T_{i}^{**} \\
    T_{i,\text{ext}}
\end{bmatrix}
\begin{bmatrix}
    n_t \\
    n_{\text{ext}}
\end{bmatrix}
\]

where \(x_{i,j} \in \mathbb{R}^{N_t}\) is the extent of reaction, \(x_{\text{in},j} \in \mathbb{R}^{N_t+1}\) are the extent of inlet flow, \(x_{\text{inj},j} \in \mathbb{R}^{N_t-N_t-p-1}\), and \(x_{\text{inj},j} \in \mathbb{R}^{N_t-N_t-p-1}\) are the extent of reaction and inlet flow invariants. Finally, we obtain the following LPV representation for the 4th tray

\[
\begin{bmatrix}
    x_{i,j} \\
    \xi_{ij} \\
    \xi_{ij}
\end{bmatrix} =

\begin{bmatrix}
    A_{i,j}(\theta_{i}) \\
    0 \\
    0
\end{bmatrix}
\begin{bmatrix}
    x_{i,j} \\
    \xi_{ij} \\
    \xi_{ij}
\end{bmatrix} +

\begin{bmatrix}
    B_{i,j} \\
    0 \\
    0
\end{bmatrix}
\begin{bmatrix}
    \theta_{i} \\
    \theta_{i} \\
    \theta_{i}
\end{bmatrix} +

\begin{bmatrix}
    u_{i,j} \\
    u_{i,j} \\
    Q_{\text{ext},i}
\end{bmatrix} +

\begin{bmatrix}
    B_{i,j} \\
    0 \\
    0
\end{bmatrix}
\begin{bmatrix}
    \theta_{i} \\
    \theta_{i} \\
    \theta_{i}
\end{bmatrix} +

\begin{bmatrix}
    B_{i,j} \\
    B_{i,j}
\end{bmatrix}
\begin{bmatrix}
    d_{i,j} \\
    d_{i,j}
\end{bmatrix}
\begin{bmatrix}
    C_{i,j} \\
    0
\end{bmatrix} \begin{bmatrix}
    x_{i,j} \\
    \xi_{ij}
\end{bmatrix} +

\begin{bmatrix}
    0 \\
    1
\end{bmatrix}
\begin{bmatrix}
    T_{i}
\end{bmatrix}
\]

\(\forall \quad i = 1, \ldots, N_C, \quad j = 1, \ldots, N_t\).

Details about the extent transformation for the RBD can be found in the work of Marquez-Ruiz et al.\(^9\)

**Case Study: Synthesis of Unsaturated Polyester in an RBD.** Consider an industrial reactive batch distillation process for the synthesis of unsaturated polyester from maleic anhydride and propylene glycol. This process involves four types of reactions.\(^{41}\) First maleic anhydride reacts with propylene glycol and produces a maleic acid end group and a propylene glycol end group with an ester bridge. This is a fast and exothermic reaction \((\Delta H = -40 \text{ kJ/mol})\). Esterification proceeds by the reaction of different acid and alcohol end groups to form new ester bridges and water, or by reaction of a glycol hydroxyl group with an acid end group to form an ester bridge and water. The double bond in maleic anhydride can be isomerized or saturated. Saturation of the double bond causes cross-linking in the polymer, and approximately 10—20% of the double bonds are saturated in the preparation of the polyester. The three reactions, esterification, isomerization and saturation form a network of nine reactions. These reactions are summarized in Table 3.

Table 3. Basic Reactions in the Polyesterification of Unsaturated Carboxylic Acids with Diols\(^{41}\)

<table>
<thead>
<tr>
<th>reaction</th>
<th>stoichiometry</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(\text{RCOOH}<em>{1D} + \text{R}’\text{OH} = \text{RCOOR’}</em>{1D} + \text{H}_2\text{O})</td>
</tr>
<tr>
<td>2</td>
<td>(\text{RCOOH}<em>{2D} + \text{R}’\text{OH} = \text{RCOOR’}</em>{2D} + \text{H}_2\text{O})</td>
</tr>
<tr>
<td>3</td>
<td>(\text{RCOOH}_3 + \text{R}’\text{OH} = \text{RCOOR’}_3 + \text{H}_2\text{O})</td>
</tr>
<tr>
<td>4</td>
<td>(\text{RCOOH}<em>{1D} = \text{RCOOH}</em>{2D})</td>
</tr>
<tr>
<td>5</td>
<td>(\text{RCOOR’}<em>{1D} = \text{RCOOR’}</em>{2D})</td>
</tr>
<tr>
<td>6</td>
<td>(\text{RCOOH}_{1D} + 0.5\text{R’OH} = \text{RCOOH}_3)</td>
</tr>
<tr>
<td>7</td>
<td>(\text{RCOOH}_{2D} + 0.5\text{R’OH} = \text{RCOOH}_3)</td>
</tr>
<tr>
<td>8</td>
<td>(\text{RCOOR’}_{1D} + 0.5\text{R’OH} = \text{RCOOR’}_3)</td>
</tr>
<tr>
<td>9</td>
<td>(\text{RCOOR’}_{2D} + 0.5\text{R’OH} = \text{RCOOR’}_3)</td>
</tr>
</tbody>
</table>

The polyesterification reactions are summarized in Table 3.
the reactor vessel is eliminated as shown by the bottom plot in Figure 19.

We have two inputs, the heat-flux to the reactor and the reflux of distilled liquid in the top of the reactor for the control of the reactor temperature. In a recent paper by ref 42, several conventional control strategies have been studied for this system. In this work, we consider a model based control strategy. The tracking the temperature reference is difficult since when the boiling point of the products in the reactor is reached, the temperature cannot be increased anymore by the heat flow input. This behavior can be observed by the optimal profile in Figure 19 where the temperature only increases when the temperature is below the boiling point. The reflux input of the RBD has also interesting dynamic behavior that has to be taken into account. The reflux input only attenuates the temperature in the reactor when there is already some product distilled. When this is the case, the reflux cools the reactor when the distillation tray above the reactor has a lower temperature than the reactor, while it would heat the reactor when the tray above the reactor has a higher temperature than the reactor.

**Simulation Results.** In this work, we will consider the control of the reference temperature by manipulating the heat-flow only. We assume that the optimal input trajectory of the reflux is used for the input, which can be seen in Figure 20.

By considering the manipulation of the heat flow in the reactor jacket only, the LPV model can be reduced to:

\[
\frac{dT}{dt} = -\beta T + \alpha_0 Q_{\text{ext}} + \alpha_1 L + \beta_1 r + \kappa V
\]

First, we apply the ML-MPC with a fixed parameter \( \theta \) for the batch time. In this case we learn the disturbance by using Method 1, where we assume the liquid and vapor flows and the mass in the reactor measurable. With these measurements the disturbance trajectory can be calculated. The results of learning the parameter and disturbance trajectories for the controller are shown in Figure 21.

In these results, we see that, due to a small cooling action around 5000 s in the first batch, the temperature in the reactor drops and it is not able to reach the reference anymore. At the same time, the heat flow input is high, because the boiling temperature of the reactor is decreased. This boiling point decrease has been caused by the reflux that returns water back to the reactor. In the second batch, we learn the disturbance caused by the reflux; therefore, the controller cools the reactor slightly less around 5000 s, such that the temperature in the reactor does not decrease, and the reference tracking performance is improved for the next 10 000 s.

While this method improves the performance of RBD operation, it does not consider changes in the plant as the parameters are not re-estimated. We, therefore, implement another controller with the time-varying parameter. We start in the initial batch with the poorly chosen parameter constant for the full batch. In the second batch the varying trajectory is estimated. The results for this application are shown in Figure 22.

While in the first batch the temperature is only increased by the exothermic reaction in the reactor as the heat flow input is zero, in the second batch the controller is able to track the reference better. Similar to the previous controller implementation, cooling of the reactor around 5000 s causes a temperature...
drop in the reactor in the second batch. And by learning this undesired behavior, the controller for the following batches then increases the temperature around 5000 s to be able to remain at the reference temperature.

We have also tested the ML-MPC algorithm for 15 batches, with changes in the initial conditions of the plant at the 11th and 14th batches. We obtain the convergence of the Euclidean norm of the error over batches as seen in Figure 23.

Figure 23. Convergence of the Euclidean norm over batches, with changes in the initial conditions at batches 11 and 14.

CONCLUSIONS

In this paper we have proposed a model learning predictive control (ML-MPC) method, based on the repetitive behavior of the batch processes. To this end, the LPV model used in the controller is updated using information from the previous batch. Inspired by the MPC for LPV systems in literature, three different application-dependent options to estimate the parameters and disturbance of the model have been proposed and compared in simulation on a nonlinear batch reactor. The best controllers are able to converge to their best performance within approximately two batches, which is better than the IL-MPC method in the work of Oh and Lee. Finally, by applying the ML-MPC on a reactive batch distillation column, we have shown the ability to adjust the controller to complex nonlinear behaviors. The parameter and disturbance convergence to their actual values in limited number of batches. In the case of changes in the initial conditions the controller is also able to adapt quickly.

AUTHOR INFORMATION

Corresponding Author
*E-mail: lozkan@tue.nl.

ORCID
Leyla Özkan: 0000-0001-8442-772X

Notes
The authors declare no competing financial interest.

REFERENCES

(4) Bruzélus, F. Linear parameter-varying systems-an approach to gain scheduling; Chalmers University of Technology, 2004.
(20) Shen, D.; Han, J.; Wang, Y. Convergence analysis of ILC input sequence for underdetermined linear systems. Science China Information Sciences 2017, 60, 099201.