AIChE WILEY InterScience® DISCOVER SOMETHING GREAT

Self-Tuning Run to Run Optimization of Fed-Batch Processes Using Unfold-PLS

José Camacho and Jesús Picó
Departamento de Ingeniería de Sistemas y Automática, Universidad Politécnica de Valencia, Valencia 46022, Spain

Alberto Ferrer
Departamento de Estadística e Investigación Operativa Aplicadas y Calidad, Universidad Politécnica de Valencia, Valencia 46022, Spain

DOI 10.1002/aic.11205
Published online May 29, 2007 in Wiley InterScience (www.interscience.wiley.com).

Unfold Partial-Least Squares (u-PLS) is a modeling method successfully applied to batch-process monitoring and end quality prediction. This method is integrated in a self-tuning optimization algorithm, based on extremum-seeking control. The optimization is driven by the gradient obtained by means of an adaptive u-PLS model. Since this is an empirical model, no first-principles based knowledge of the process is necessary. Heuristic rules are used to constrain the gradient taking into account nonlinearity and unknown causes of variability. Extensions to model the variability in initial conditions, to optimize several performance indices, and to handle inequality constraints are presented. The optimization algorithm is tested on a complex comprehensive simulated-process model for the fed-batch cultivation of Sacharomyces cerevisiae. *Results show the performance and versatility of the proposed approach, as well as its robustness to process changes.* © 2007 American Institute of Chemical Engineers *AIChE J,* 53: 1789–1804, 2007

Keywords: partial-least squares, batch processes, self-tuning systems, extremum-seeking control, run-to-run optimization

Introduction

Batch and fed-batch processing is the general trend to manufacture low-quantities of high-value products.[1] The biotechnological industry is an example of this kind of production. A large portion of bioreactors operates in batch and fed-batch mode because of their advantages over continuous processes, such as adjustability of batch time to meet specifications and repetitive and slow nature of batch processes.[2] Additionally, fed-batch is believed to outperform batch processing in fermentation processes.[3] In fed-batch fermentations, the feeding strategy needs to be somehow optimized, since both underfeeding and overfeeding lead to a decrease of product formation.[4]

The optimization and advanced control of this kind of processes is a challenging task due to the lack of measurements of principal variables.[5] The uncertainties in the process behavior and their nonlinear nature and slow response complicates this task.[6] The design of first principles models is an important research line,[7] and the fundamental knowledge is commonly used in the control system design. Jobé et al.[8] regulate the substrate feeding according to an estimation of the metabolic state of the process. Picó-Marco et al.[9,10] propose a sliding mode scheme based on a reference model. Several feeding strategies for baker's yeast production are based on regulating the substrate concentration indirectly, based on relationships observed in the process.[11]

The reliance on the process knowledge has the drawback that some of the assumed dynamic relationships may not be

Correspondence concerning this article should be addressed to J. Camacho at jcamacho@isa.upv.es.

© 2007 American Institute of Chemical Engineers

valid in all cases, that is, for all process states. Additionally, unknown side reactions may exist. The control approaches based on first principles models may be endowed with some degree of adaptation capability and robustness with respect to changes in the dynamics. Nonetheless, if their assumptions (for example, assuming Monod or Haldane kinetics) are not fulfilled the control performance may degrade beyond acceptable limits. One strategy to overcome this problem is the combination of fundamental knowledge with data-based models.[12,13] A more radical approach is to rely completely on input-output data, like in iterative learning control methods,[14,15,16,17] which take advantage of the repetitive nature of batch processes. Also, according to Lee et al.,[4] the application of artificial intelligence methods, like artificial neural networks or fuzzy logic, is achieving promising results.

In the optimization of batch processes, the same distinction among approaches can be made according to the type of information used. Approaches based on numerical computations using a fundamental model[18,19] are open-loop in nature, and may not lead to optimal performance due to process-model mismatch. Contrarily, the run-to-run optimization approach is based on adjusting the control law with the data collected from the previous batches. This can be performed in different ways, according to Srinivasan et al.[20]:

- *Optimization via model refinement*. In this approach, the process model is adjusted with the data from last batch and a numerical optimization is run to compute the following control law. The model used can be completely data-driven,[21] or combined with fundamental knowledge.[22] The optimization finishes when the predictions of the model are accurate enough.
- *Evolutionary optimization with model-based gradient*. In this approach, the next control law is updated from the current one according a the gradient law computed with a model.[23]
- *Model-free evolutionary optimization*. In this approach, the gradient is computed from input-output data solely. The number of trials required to compute a gradient is equal to the number of decision variables. A hybrid between this and the preceding approach is found in the optimization approach based on generalized constraints.[20]

Model-based optimizations tend to be faster to converge than the model-free approach. The formers are driven by the model, whereas the latter has to perform the whole optimization on a trial-error basis. On the other hand, the model-free approach does not suffer from process-model mismatch problems and is more robust to changes in the process.

In this context, projection to latent structures-based methods (PLS-based), like principal component analysis (PCA) or partial-least squares (PLS), are powerful empirical modeling tools. These kind of techniques are very useful when the variables are highly correlated. When this occurs, the modeling information (the variability of the data) is contained in a number of combinations of variables—the latent variables—much lower than the total number of original variables.

In batch and fed-batch processes, data has three-dimensional (3-D) nature: a set of variables is collected at several sampling times for each of the batches processed. Although, three-way PLS-based methods exist,[24,25,26] a commonly used approach consists of unfolding the data matrix in two dimensions, and to apply bilinear models as PCA or PLS afterwards.[27,28] These latter methods are called unfold-PCA (u-PCA), and unfold-PLS (u-PLS). The utility of u-PLS for the optimization of control trajectories in batch processes has already been reported.[29]

In this article, a self-tuning extremum-seeking controller[30] based on u-PLS is proposed to optimize the feeding profile of a fed-batch process. Extremum-seeking controllers can be roughly classified as model-based,[31,32] which rely on assumptions on the dynamics of the process and the objective function, and model free.[33,34] The approach here presented belongs to the second class, being completely data driven, so that no assumptions on the process dynamics model have to be done. Moreover, only measurements of the objective function at the end of the batch are assumed. The optimization is performed on a gradient basis using data from previously processed batches. The u-PLS is used to extract the gradient information. This is a model-free evolutionary optimization approach[a] with the advantage that the gradient is computed in the latent variable subspace. A reduction in the number of trials to compute a reliable gradient ensues.

This article is organized as follows: First, the PLS and u-PLS modeling techniques are introduced. Then, the proposed optimization procedure is described in detail. The first principles model used for simulation is explained next, and the simulation results evaluated. Finally, some conclusions and remarks are withdrawn.

Partial-Least Squares

The linear-regression problem is defined by the following expression

$$\mathbf{Y} = \mathbf{X} \cdot \mathbf{B} + \mathbf{F} \tag{1}$$

where $\mathbf{Y}$ is the $I \times M$ matrix of output variables that are to be predicted, $\mathbf{X}$ is the $I \times J$ matrix of variables available to predict $\mathbf{Y}$, $\mathbf{B}$ is the $J \times M$ matrix of regressors, and $\mathbf{F}$ is the $I \times M$ matrix of residuals. $\mathbf{B}$ constitutes a model of $\mathbf{Y}$, being $\mathbf{X}$ the inputs to the model.

Ordinary-Least Squares (OLS) performs a solution to the linear regression problem, which minimizes the quadratic error. The least-squares solution for Eq. 1 is

$$\hat{\mathbf{B}}_{LS} = (\mathbf{X}^T \cdot \mathbf{X})^{-1} \cdot \mathbf{X}^T \cdot \mathbf{Y} \tag{2}$$

The inversion of the matrix $\mathbf{X}^T \cdot \mathbf{X}$ requires it to be nonsingular. Moreover, the OLS solution is highly unstable, leading to a poor prediction performance when this matrix is ill-conditioned. Its conditioning depends on the degree of independence among variables in $\mathbf{X}$. That is to say, if variables of $\mathbf{X}$ are highly correlated, the prediction of $\mathbf{Y}$ should not be performed directly from $\mathbf{X}$.

An alternative is to predict $\mathbf{Y}$ from the latent variables in $\mathbf{X}$, that is, the subspace of $\mathbf{X}$ which contains almost all its information related to $\mathbf{Y}$. The aim of the PLS regression is to predict $\mathbf{Y}$ from the subspace of $\mathbf{X}$, which maximizes its covariance with $\mathbf{Y}$. The partial linear regression problem between normalized matrices $\mathbf{X}$ and $\mathbf{Y}$ can be stated as

[a] Notice that, although u-PLS is used to model the data, no structure is imposed to the process dynamics and so the approach is essentially model-free.

$$\mathbf{X} = \mathbf{T} \cdot \mathbf{P}^T + \mathbf{E}$$
$$\mathbf{Y} = \mathbf{T} \cdot \mathbf{Q}^T + \mathbf{F} \tag{3}$$

where **T** is an $I \times A$ matrix, which contains the projections of **X** to an A-dimensional subspace, **P** and **Q** are the $J \times A$ and $M \times A$ regressor matrices, also called loadings matrices, and **E** and **F** are the $I \times J$ and $I \times M$ matrices of residuals of **X** and **Y**, respectively. By assuring that the A dimensions in the subspace are mutually independent, matrix $\mathbf{T}^T \cdot \mathbf{T}$ is invertible and a least squares solution is adequate. Equation 3 can be rearranged in the following form

$$\mathbf{Y} = \mathbf{X} \cdot \hat{\mathbf{B}}_{PLS} + \mathbf{F} \tag{4}$$

$$\hat{\mathbf{B}}_{PLS} = \mathbf{W} \cdot (\mathbf{P}^T \cdot \mathbf{W})^{-1} \cdot \mathbf{Q}^T \tag{5}$$

where **W** is a $J \times A$ matrix of weights. Thus, a PLS model is represented by matrices **P**, **W** and **Q**.

A PLS model can be obtained using the NIPALS algorithm, included in the Appendix. The number of dimensions of the subspace of **X** can be estimated by cross-validation, choosing the one with least-square prediction error (PRESS) associated. For a deeper overview of the PLS method refer to Martens and Næs[35] and Geladi and Kowalski.[36]

Unfold-PLS

In a set of pioneering papers, Nomikos and MacGregor[28,37,38] propose the unfolding of the data matrix in the batch direction—batch-wise unfolding Eq. (6)—for monitoring purposes. This modeling approach has influenced the research concerning batch processes modeling and monitoring during the last decade.

$$\underline{\mathbf{X}}(I \times J \times K) \Rightarrow \mathbf{X}(I \times JK) \tag{6}$$

where $\underline{\mathbf{X}}$ is the original 3-D matrix of data, and **X** the batch-wise unfolded matrix.

After batch-wise unfolding, all the information available for a single batch (all variables taken at every sampling time) is treated as separated variables. The resulting 2-D matrix contains $J \times K$ variables for each batch, being I the number of batches, J the number of original variables of the batch process, and K the number of sampling times. Notice that, after unfolding, the data matrix has a huge number of highly correlated variables. Thus, the application of OLS is not feasible in this case because matrix $\mathbf{X}^T \cdot \mathbf{X}$ is singular.

Self-Tuning Extremum u-PLS

Self-tuning systems combine a recursive estimation algorithm with a synthesis algorithm.[30] In this article, the estimation algorithm is an adaptive u-PLS regression, and the synthesis one is an extremum-seeking controller used for the optimization of a performance function.

For multiple input single output (MISO) systems, an extremum control law based on the gradient estimation can be set as follows.[30]

$$\mathbf{u}_{t+1} = \mathbf{u}_t + c \cdot \frac{dy}{d\mathbf{u}} \tag{7}$$

with

$$\frac{dy}{d\mathrm{u}} = \left[\frac{dy}{du_1}, \ldots, \frac{dy}{du_J}\right]^T \tag{8}$$

where u_t is the vector of J manipulable variables at time t, c is a weighting scalar or matrix, which may be variable in t and y stands for the output or, in general, any performance index to be optimized.

In a fed-batch process, a performance function such, for instance, as biomass production can be optimized by appropriately selecting the feeding law. In this particular case, the vector **u** represents the feeding profile. By constructing an u-PLS model between the feeding law—discretized at a constant number of intervals K—and the performance index, the estimation of the gradient can be obtained from the u-PLS model. Notice that the unfolding procedure of Eq. 6 translates the dynamic optimization problem into a static one. This is done by converting the K intervals of the discretized feeding law into K different variables. Moreover, the dynamic information is implicitly included in the u-PLS model.

Let us call $\underline{\mathbf{U}}(I \times J \times K)$ the three-way data matrix containing I samples of J variables discretized to K intervals. The samples of matrix $\underline{\mathbf{U}}$ represent I different choices of the feeding law including J manipulable variables. When processing a total of I batches following the different feeding laws, a set of I performance indices can be calculated/measured and arranged in vector y. Matrix $\underline{\mathbf{U}}$ is batch-wise unfolded and the resulting matrix **U** and vector y are autoscaled—that is, normalized to zero mean and unit variance. Afterwards, a PLS model is constructed from the unfolded normalized matrices.

To separate the data normalization operations from the input-output relationships, the gradient can be decomposed as

$$\frac{dy}{d\mathbf{u}} = \frac{dy}{dy^n} \cdot \frac{dy^n}{d\mathbf{u}^n} \odot \frac{d\mathbf{u}^n}{d\mathbf{u}} \tag{9}$$

where the superscript n means normalized (autoscaled), $\odot$ stands for the element-wise matrix product, also called Hadamard or Schur product, and

$$\frac{dy}{dy^n} = \sigma_y \tag{10}$$

$$\frac{d\mathbf{u}^n}{d\mathbf{u}} = \mathbf{1}_{JK} \oslash \boldsymbol{\sigma}_{\mathrm{u}} \tag{11}$$

$$\boldsymbol{\sigma}_{\mathbf{u}} = [\sigma_{u_{1,1}}, \ldots, \sigma_{u_{1,K}}, \ldots, \sigma_{uJ,1}, \ldots, \sigma_{uJ,K}]^T \tag{12}$$

where σ_v is the standard deviation of variable v, $\mathbf{1}_{JK}$ is a vector filled with $J{\cdot}K$ 1's, $\oslash$ is the Hadamard division, and $u_{j,k}$ stands for the j-th manipulable variable at the k-th interval. The gradient of the normalized inputs-output estimated by the u-PLS model corresponds to the matrix of regressors $\hat{\mathbf{B}}_{PLS}$ (Eq. 5). Indeed this matrix contains the directions of maximum correlation between u and y, and, hence, the directions in which the greatest rate of increase of y is obtained

$$\left(\widehat{\frac{dy^n}{d\mathbf{u}^n}}\right) = \mathbf{W} \cdot (\mathbf{P}^T \cdot \mathbf{W})^{-1} \cdot \mathbf{q}^T \tag{13}$$

Note that here $\mathbf{q}$ is a vector since only one output is considered. It should be stressed that the gradient obtained is not a gradient in the sampling time direction (K-index) but in the batch direction (I-index). Therefore, Eq. 9 conveys all the information required to modify the complete feeding law so as to improve the performance of next batch. Combining Eq. 9, 10, 11 and 13 yields

$$\left(\frac{\hat{dy}}{d\mathbf{u}}\right) = \sigma_y \cdot \left(\mathbf{W} \cdot \left(\mathbf{P}^T \cdot \mathbf{W}\right)^{-1} \cdot \mathbf{q}^T\right) \oslash \boldsymbol{\sigma}_{\mathbf{u}} \tag{14}$$

The run to run optimization algorithm proposed follows the common steps of a self-tuning algorithm.[30] being:

Step 1. Process a new batch i with feeding law $\mathbf{u}_i$.

Step 2. Rebuild the u-PLS model and other estimation parameters by incorporating the new inputs and the measured output.

Step 3. Compute the gradient and the next feeding law.

Step 4. Increment the counter $i \rightarrow i + 1$, and loop back to Step 1.

The u-PLS model captures the variability of $\mathbf{u}^n$ around its average which is related to y^n. Therefore, this estimation of the gradient is valid for the average feeding law of the set of batches used to generate the u-PLS model. Thus, the control law Eq. 7 must be modified to

$$\mathbf{u}_{i+1} = \overline{\mathbf{u}}_i + c \cdot \left(\frac{\hat{dy}}{d\mathbf{u}}\right)_i \tag{15}$$

where $\overline{\mathbf{u}}_i$ is the average feeding law and

$$\left(\frac{\hat{dy}}{d\mathbf{u}}\right)_i = \sigma_{y,i} \cdot \left(\mathbf{W}_i \cdot \left(\mathbf{P}_i^T \cdot \mathbf{W}_i\right)^{-1} \cdot \mathbf{q}_i^T\right) \oslash \boldsymbol{\sigma}_{\mathbf{u},i} \tag{16}$$

where all the elements are recomputed after batch i has been processed.

Note: Hereafter, all the elements which are updated using data from batch i will use i as subindex. Values which present the subindex $i + 1$ are related to the batch which is going to be processed next.

In every loop, the model is rebuilt using the last I processed batches. This is done to adjust the model to the current state of the process. This operation is believed to be convenient not only because of the interbatch dynamics shown by some processes, but also because the optimization algorithm is actively driving the process. Thus, since the gradient is only locally valid, only last batches should be used to calibrate a local model. An alternative to the calibration of the model using the last I batches is the use of an exponentially weighted PLS model.[39,40] The other estimation parameters, such as the standard deviations in Eq. 16, are also recomputed with the last I batches.

In order to guarantee the convergence to a (local) extremum, a dither signal is added to Eq. 15, so that the input signal is persistently exciting. The dither signal can be a random or pseudo-random binary signal (PRBS) designed in order to guarantee the identifiability of the parameters. Therefore, the control law becomes

$$\mathbf{u}_{i+1} = \overline{\mathbf{u}}_i + c \cdot \left(\frac{\hat{dy}}{d\mathbf{u}}\right)_i + \boldsymbol{\rho}_{i+1} \tag{17}$$

with $\boldsymbol{\rho}_{i+1}$ a vector of $J \times K$ excitation values associated to the inputs. At the beginning of the optimization, an initial feeding law is chosen, and the exciting signal is added in order to generate the initial set of batches. These batches are used to obtain the first u-PLS model.

Heuristic rules

In practice the strategy proposed may not work properly when the objective function y is not convex with respect to the variables u, or when it is very steep near the optimum. In these cases, the step suggested by the gradient might jeopardize convergence. In the context of statistical tools used in this work, the region of validity of the gradient obtained in Eq. 16 can be estimated and used to limit the step calculated in Eq. 17.

A subset of potential feeding laws where the gradient is supposed to be valid can be defined. Statistical limits, such as D-statistic and Q-statistic limits,[41] can be computed for the feeding law and be used to restrict the solution of Eq. 17. Nonetheless, this strategy was observed to produce problems of convergence. Statistical limits have shown to be too restrictive to give the optimization algorithm the freedom necessary to explore in search of the optimum.

An alternative strategy is to limit the magnitude of the gradient used in Eq. 17 by means of the quality predicted with the u-PLS model for the next batch. Thus, too large steps in y are forbidden. The limit may be computed from the recent batches used to fit the u-PLS model. The prediction of the quality for batch $i + 1$ with the u-PLS model follows

$$\hat{y}_{i+1} = \overline{y}_i + c \cdot \hat{y}_{\mathbf{u},i+1} + \hat{y}_{\boldsymbol{\rho},i+1} \tag{18}$$

where $\overline{y}_i$ is the average quality of the batches used to fit the u-PLS model, $\hat{y}_{\boldsymbol{\rho},i+1}$ is the contribution of $\boldsymbol{\rho}_{i+1}$, and $\hat{y}_{\mathbf{u},i+1}$ the contribution of the gradient. The aim is to restrict the magnitude of $\hat{y}_{\mathbf{u},i+1}$, giving to $\hat{y}_{\boldsymbol{\rho},i+1}$ the necessary freedom to explore. The contribution of the gradient can be computed from

$$\hat{y}_{\mathbf{u},i+1} = \sigma_{y,i} \cdot \hat{y}^n_{\mathbf{u},i+1} \tag{19}$$

with $\hat{y}^n_{\mathbf{u},i+1}$ the normalized contribution of the gradient

$$\hat{y}^n_{\mathbf{u},i+1} = \left(\left(\frac{\hat{dy}}{d\mathbf{u}}\right)_i \oslash \boldsymbol{\sigma}_{\mathbf{u},i}\right)^T \cdot \left(\frac{\widehat{dy_n}}{d\mathbf{u}_n}\right)_i \tag{20}$$

The first term in Eq. 20 performs the scaling operation. The second one performs the prediction using the PLS model.

The approach used in this article is to constrain $\hat{y}^n_{\mathbf{u},i+1} \leq 3$. This is equivalent to restrict the maximum step in the output produced by the gradient to $3 \cdot \sigma_{y,i}$. Both for the initial phase of the optimization and after convergence, this limit has a good statistical interpretation. During those phases, the contribution of the gradient is almost null. Uncontrolled variability and the effect of the dither signal are the main sources of variability. Hence, the quality of the batches during those phases can be expected to be normally distributed. Theoretically, the $\overline{y}_i \pm 3 \cdot \sigma_{y,i}$ limit contains the 99.73% of the samples of a normal distribution. Thus, movements caused by the gradient are restricted to be inside the known space.

During the intervening phase of the optimization, when the gradient is actively driving the process, the assumption of normality is not valid. Nonetheless, the application of the limit defined gives enough margin for the optimization to explore the solutions space.

The restriction of the gradient can be expressed as follow

$$\left(\frac{d\tilde{y}}{d\mathbf{u}}\right)_i = \begin{cases} \left(\frac{d\hat{y}}{d\mathbf{u}}\right)_i & \hat{y}^n_{\mathbf{u},i+1} \leq 3 \\ \frac{3}{\hat{y}^n_{\mathbf{u},i+1}} \cdot \left(\frac{d\hat{y}}{d\mathbf{u}}\right)_i & \hat{y}^n_{\mathbf{u},i+1} > 3 \end{cases} \tag{21}$$

The behavior of the optimization can be further improved by tuning the parameter c of Eq. 17 conveniently. In this article, the following adaptive rule is used

$$\tilde{c}_i = 1 - \left(\frac{PRESS_A}{PRESS_0}\right)_i \tag{22}$$

where $PRESS_A$ represents the square prediction error of the cross-validated u-PLS model with A latent variables, $PRESS_0$ is the square prediction error computed only for the average output—that is, no latent variables—and subindex i once again means the values have been recalculated after the processing of the i-th batch, thus, including its data along with that of the preceding $I - 1$ batches. Notice that value A is also varying with i. The term ($1 - PRESS_A/PRESS_0$) is an indicator of how well the u-PLS model is predicting the data.[b] If no other source of variation is influencing the process apart from the manipulable variables, this index is a good estimate of the degree of linearity of the process in the explored zone. If, as it is common, other sources of variability exist—such as variable initial conditions and measurement noise—the index is still a good indicator of the reliability of the model. The underlying idea in the adaptation of c_i is the following: the more prediction error, the less reliable the fitted model is, and, thus, a more conservative approach is convenient. This is carried out by decreasing the gradient effect by means of c_i.

Although other adaptive tuning laws for c may lead to more aggressive or conservative optimizations, the one proposed here has been tested for many different linear and severe nonlinear dynamics through simulation, and the results have been satisfactory.

Finally, the amplitude of the excitation signal $\boldsymbol{\rho}_{i+1}$ can be adapted in a similar way to c_i. The excitation is necessary when the current variability induced in the process is not enough to generate a reliable model—so that a reliable gradient is available. Thus, for the initial set of batches, the excitation signal is necessary to start the search. Nevertheless, the excitation can be deactivated or faded away when the gradient is informative enough, that is, when the gradient generates enough variability to drive the optimization process. For this purpose, the index $(PRESS_A/PRESS_0)_i$ can be used. To compute the current amplitude of the excitation signal, the index $(PRESS_A/PRESS_0)_i$ is multiplied by the maximum amplitude (g, initially set)

$$\tilde{\boldsymbol{\rho}}_{i+1} = \left(\frac{PRESS_A}{PRESS_0}\right)_i \cdot g \cdot \boldsymbol{\rho}_{i+1} \tag{23}$$

where $\boldsymbol{\rho}_{i+1}$ is the original excitation signal composed of 1's and –1's. With this readjustment of the amplitude of the excitation signal, the index $(PRESS_A/PRESS_0)_i$ controls the degree of exploration (excitation) and (gradient driven) exploitation of the optimization for batch $i + 1$.

With all the described enhancements, the control law follows the next expression

$$\mathbf{u}_{i+1} = \overline{\mathbf{u}}_i + \tilde{c}_i \cdot \left(\frac{d\tilde{y}}{d\mathbf{u}}\right)_i + \tilde{\boldsymbol{\rho}}_{i+1} \tag{24}$$

Extensions of the approach

Three extensions of the presented approach may be adequate for certain processes. The first one considers the addition of the initial state of the process at the beginning of a batch. The second extension allows to optimize several performance indices. The last one deals with inequality constraints in measured variables. The advantages and implications of the extensions are detailed here.

Addition of the initial state

Although a tight control may be already performed in a plant in order to reduce the variability in the product, many sources of variability are difficult to control. In some cases, the sources of this variability, although not eliminated, can be measured. In such cases, the u-PLS can be enhanced by adding this information along with the manipulable variables. With this enhancement, the variability in the performance due to the initial state can be identified, and the effect of the feeding law can be better discriminated. The addition of the initial measurements is straightforward. The unfolded matrix containing the choices of the feeding law is augmented with those measurements

$$\mathbf{U}_{aug,i} = [\mathbf{Z}_i, \mathbf{U}_i] \tag{25}$$

with $\mathbf{Z}_i$ representing a $I \times N$ matrix of N initial measurements performed over I samples (batches). Afterward, the PLS model is computed for normalized matrices $\mathbf{U}_{aug,i}$ and y_i. The subindex i again indicates the data include the batches from $i – I + 1$ to i.

The effect of the initial state $\mathbf{z}_{i+1}$ of the next batch can be taken into account to compute the restriction of the gradient in Eq. 21. From the normalized predicted quality $\hat{y}^n_{i+1}$ for batch $i + 1$, the contribution of the gradient $\hat{y}^n_{\mathbf{u},i+1}$ and of the initial estate $\hat{y}^n_{\mathbf{z},i+1}$ can be distinguished

$$\hat{y}^n_{\mathbf{u},i+1} = \left(\left(\frac{d\hat{y}}{d\mathbf{u}}\right)_i \oslash \boldsymbol{\sigma}_{\mathbf{u},i}\right)^T \cdot \mathbf{W}_{\mathbf{U},i} \cdot (\mathbf{P}_i^T \cdot \mathbf{W}_i)^{-1} \cdot \mathbf{q}_i^T \tag{26}$$

$$\hat{y}^n_{\mathbf{z},i+1} = ((\mathbf{z}_{i+1} - \overline{\mathbf{z}}_i) \oslash \boldsymbol{\sigma}_{\mathbf{z},i})^T \cdot \mathbf{W}_{\mathbf{Z},i} \cdot (\mathbf{P}_i^T \cdot \mathbf{W}_i)^{-1} \cdot \mathbf{q}_i^T \tag{27}$$

with $\mathbf{W}_{\mathbf{Z},i}$ the part of the matrix of weights related to $\mathbf{Z}_i$, and $\mathbf{W}_{\mathbf{U},i}$ the one related to $\mathbf{U}_i$.

[b] This index is very similar to the widely used *goodness of prediction* by cross-validation index (Q^2). The only difference being that the latter uses the sum of squares after autoscaling instead of $PRESS_0$. Actually, $PRESS_0$ is the sum of squares computed by cross-validation. $PRESS_0$ was used so that the index takes values in the interval [0, 1].

Then, the effect of the initial variability can be used to adjust the correction of the gradient in Eq. 21

$$\left(\frac{d\tilde{y}}{d\mathbf{u}}\right)_i = \begin{cases} \left(\dfrac{d\hat{y}}{d\mathbf{u}}\right)_i & \hat{y}^n_{\mathbf{u},i+1} + \hat{y}^n_{\mathbf{z},i+1} \leq 3 \\ \mathbf{0}_{JK} & \hat{y}^n_{\mathbf{z},i+1} > 3 \\ \dfrac{3 - \hat{y}^n_{\mathbf{z},i+1}}{\hat{y}^n_{\mathbf{u},i+1}} \cdot \left(\dfrac{d\hat{y}}{d\mathbf{u}}\right)_i & \textit{otherwise} \end{cases} \tag{28}$$

This extension implies that the initial state $\mathbf{z}_{i+1}$ should be measured before processing the next batch $i + 1$ to compute Eq. 27. This might be difficult in some applications. In practice, the initial state can be taken into account to fit the u-PLS model using $\mathrm{U}_{aug,i}$ but not to constrain the gradient. Thus, $\hat{y}^n_{\mathbf{z},i+1} = 0$ can be assumed, making Eq. 21 and 28 equivalent. In the simulations performed, no reduction due to this assumption was observed in the performance of the optimization. Nevertheless, the corrections performed with the measurement of $\mathbf{z}_{i+1}$ may become more advantageous in strongly nonlinear processes.

Several performance indices

If a number of M performance indices are to be optimized, each one with an associated weighting factor $\psi_1, \psi_2, \ldots, \psi_M$, the extension of the model is performed as follows. Notice that the aim is to optimize the several—weighted—performance indices and not their weighted sum. This slight difference may be important in some cases, as discussed at the end of this section. First, data has to be normalized, but incorporating the weight information

$$\mathbf{Y}_i^{n(\boldsymbol{\psi})} = (\mathbf{Y}_i - \mathbf{1}_I \cdot \bar{\mathbf{y}}_i^T) \odot (\mathbf{1}_I \cdot (\boldsymbol{\psi} \oslash \boldsymbol{\sigma}_{\mathbf{y},i})^T) \tag{29}$$

with $\bar{\mathbf{y}}_i$ and $\boldsymbol{\sigma}_{\mathbf{y},i}$ the averages and standard deviations of the M quality variables computed after batch i has been added to the data, and $\boldsymbol{\psi} = [\psi_1,\psi_2,\ldots,\psi_M]$. To use the self-tuning optimization procedure, the step or *weighted gradient* is computed as

$$\left(\frac{d\hat{y}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} = \left(\left(\frac{d\mathbf{y}^{\widehat{n(\boldsymbol{\psi})}}}{d\mathbf{u}^n}\right)_i^T \cdot (\boldsymbol{\sigma}_{\mathbf{y},i} \oslash \boldsymbol{\psi})\right) \oslash \boldsymbol{\sigma}_{\mathbf{u},i} \tag{30}$$

This *weighted gradient* is a weighted sum of the gradients of the quality variables, estimated from the normalization information along with the inner Jacobian matrix (between $\mathbf{u}^n$ and $\mathbf{y}^{n(\boldsymbol{\psi})}$) approximated by the u-PLS model

$$\left(\frac{d\mathbf{y}^{\widehat{n(\boldsymbol{\psi})}}}{d\mathbf{u}^n}\right)_i = (\mathbf{W}_i \cdot (\mathbf{P}_i^T \cdot \mathbf{W}_i)^{-1} \cdot \mathbf{Q}_i^T)^T \tag{31}$$

Thus, if the *weighted gradient* is included in an optimization law such as

$$\mathbf{u}_{i+1} = \bar{\mathbf{u}}_i + \tilde{c}_i \cdot \left(\frac{d\hat{y}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} + \tilde{\boldsymbol{\rho}}_{i+1} \tag{32}$$

its contribution to $\mathbf{y}_{i+1}$ is computed following

$$\hat{\mathbf{y}}^{n(\boldsymbol{\psi})}_{\mathbf{u},i+1} = \left(\frac{d\mathbf{y}^{\widehat{n(\boldsymbol{\psi})}}}{d\mathbf{u}^n}\right)_i \cdot \left(\left(\frac{d\hat{y}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} \oslash \boldsymbol{\sigma}_{\mathbf{u},i}\right)^T \tag{33}$$

$$\hat{\mathbf{y}}_{\mathbf{u},i+1} = \hat{\mathbf{y}}^{n(\boldsymbol{\psi})}_{\mathbf{u},i+1} \odot \boldsymbol{\sigma}_{\mathbf{y},i} \oslash \boldsymbol{\psi} \tag{34}$$

In order to keep the same idea of Eq. 21, the contribution of the *weighted gradient* to the quality of batch $i + 1$ should be at most three times the standard deviation. Nonetheless, the several performance indices may be correlated—some even negatively correlated. Thus, the correction should be performed with a distance measure which takes into account this fact, the Mahalanobis distance. The Mahalanobis length of vector $\hat{\mathbf{y}}^{n(\boldsymbol{\psi})}_{\mathbf{u},i+1}$ is computed as follows

$$\ell_{i+1} = \sqrt{\hat{\mathbf{y}}^{n(\boldsymbol{\psi})T}_{\mathbf{u},i+1} \cdot \mathbf{S}_i^{-1} \cdot \hat{\mathbf{y}}^{n(\boldsymbol{\psi})}_{\mathbf{u},i+1}} \tag{35}$$

with $\mathbf{S}_i$ the covariance matrix of the output $\mathbf{Y}_i^{n(\boldsymbol{\psi})}$. Therefore, the correction of the length of the *weighted gradient* is

$$\left(\frac{d\tilde{\mathbf{y}}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} = \begin{cases} \left(\dfrac{d\hat{\mathbf{y}}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} & \ell_{i+1} \leq 3 \\ \dfrac{3}{\ell_{i+1}} \cdot \left(\dfrac{d\hat{\mathbf{y}}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} & \ell_{i+1} > 3 \end{cases} \tag{36}$$

and the optimization law

$$\mathbf{u}_{i+1} = \bar{\mathbf{u}}_i + \tilde{c}_i \cdot \left(\frac{d\tilde{\mathbf{y}}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} + \tilde{\boldsymbol{\rho}}_{i+1} \tag{37}$$

The problem arises when inverting matrix $\mathbf{S}_i$. If the performance indices are very correlated, it can be ill-conditioned and the correction of the length inaccurate. This could be solved by performing a PCA on the output. Additionally, the addition of the initial state further complicates the correction of the gradient. An easier optimization procedure with close results can be performed by defining a single performance index from the combination, for instance, the weighted sum of the M original indices. Nonetheless, handling the different performance indices following the approach of this section assures the restriction in the optimization step is applied individually to each index (Eq. 36). If the weighted sum is optimized instead, the latter is not assured, and the restriction of the gradient may not perform as expected. Care should be especially taken when very different weighting factors are being used for example, $\psi_1 >> \psi_2$.

Constraints handling

Assume soft constraints of the form

$$\mathrm{f(x)} \leq 0 \tag{38}$$

where x is a vector of process—or quality—variables collected during the batch processing which are not manipulable.[c]

The first step is to create a u-PLS model between the feeding law and a constrained variable. This model is an approximation of the inner gradient between $\mathbf{u}^n$ and x^n.

[c] If they were manipulable, the constraint could be handled *ad hoc*.

$$\left(\frac{d\hat{x}^n}{d\mathbf{u}^n}\right)_i = \mathbf{W}_{x,i}\cdot\left(\mathbf{P}_{x,i}^T\cdot\mathbf{W}_{x,i}\right)^{-1}\cdot\mathbf{q}_{x,i}^T \tag{39}$$

From this model and the normalization information, the gradient can be estimated

$$\left(\frac{d\hat{x}}{d\mathbf{u}}\right)_i = \sigma_{x,i}\cdot\left(\frac{d\hat{x}^n}{d\mathbf{u}^n}\right)_i \oslash \boldsymbol{\sigma}_{\mathbf{u},i} \tag{40}$$

Using this, the value of x_{i+1} can be predicted for the control law $\mathbf{u}_{i+1}$ returned by the optimization algorithm in Eq. 24:

$$\hat{x}_{i+1} = \bar{x}_i + \left(\frac{d\hat{x}}{d\mathbf{u}}\right)_i^T\cdot\left(\mathbf{u}_{i+1}-\bar{\mathbf{u}}_i\right) \tag{41}$$

This prediction is integrated in the constraint f, so that its satisfaction can be checked. Note that the prediction of x_{i+1} is more accurate in a close neighborhood of $\bar{x}_i$. Therefore, the estimation of when a constraint is violated is more reliable as the constraint is approached. If the constraint is estimated to be violated, a decrement (increment) in x_{i+1} is computed for its prediction to satisfy the constraint

$$\Delta_{\hat{x}_{i+1}} \quad s.t. \quad f\left(\hat{x}_{i+1}-\Delta_{\hat{x}_{i+1}}\right) \leq \lambda \leq 0 \tag{42}$$

where λ is introduced as a security margin used to take into account uncertainty in the prediction. From $\Delta_{\hat{x}_{i+1}}$, the decrement (increment) in $\mathbf{u}_{i+1}$ can be computed as

$$\Delta_{\mathbf{u}_{i+1}} = \Delta_{\hat{x}_{i+1}}\cdot\mathbf{h}\oslash\left(\frac{d\hat{x}}{d\mathbf{u}}\right)_i \tag{43}$$

where $\mathbf{h}$ is a vector of weights of sum equal to 1. Although other approaches may perform well, setting $\mathbf{h}$ in the direction of the inner (u-PLS) gradient $\left(\frac{d\hat{x}^n}{d\mathbf{u}^n}\right)_i$ is recommended here. Following this approach Eq. 43 yields:

$$\Delta_{\mathbf{u}_{i+1}} = \frac{\Delta_{\hat{x}_{i+1}}}{\sigma_{x,i}\cdot\sum\left(\frac{d\hat{x}^n}{d\mathbf{u}^n}\right)_i\cdot\boldsymbol{\sigma}_{\mathbf{u},i}} \tag{44}$$

where $\sum\left(\frac{d\hat{x}^n}{d\mathbf{u}^n}\right)_i$ is the sum of elements of the inner gradient. The applied feeding law is

$$\tilde{\mathbf{u}}_{i+1} = \mathbf{u}_{i+1} - \Delta_{\mathbf{u}_{i+1}} \tag{45}$$

Two comments about this approach for constraints handling are in due. First, this approach should not be used for hard–safety–constraints because the violation of these cannot be guaranteed. Second, if cross-validation of the u-PLS model of Eq. 39 suggests 0 latent variables, that is, the feeding law is not correlated with the constrained variable, no gradient information is available. Therefore, no adjustment can be done —Eq. 44 cannot be computed. This makes reasonable in practice to modify Eq. 23 by

$$\tilde{\boldsymbol{\rho}}_{i+1} = \left(1-\left(1-\left(\frac{PRESS_A}{PRESS_0}\right)_i\right)\cdot\left(1-\left(\frac{PRESS_{x,A'}}{PRESS_{x,0}}\right)_i\right)\right)\cdot g\cdot\boldsymbol{\rho}_{i+1} \tag{46}$$

with $PRESS_{x,A'}/PRESS_{x,0}$ computed from cross-validation of model in Eq. 39.

The extension to handle several constraints can be performed either using a single u-PLS model or with several independent models. Nevertheless, care should be taken using this approach in order that corrections to satisfy a constraint do not lead to the violation of another one.

Cultivation of *Saccharomyces cerevisiae*

The first principles model of the cultivation of *Saccharomyces cerevisiae* presented by Lei et al.[7] is used to test the optimization procedure proposed in this article. The model includes 12 stoichiometry reactions, and nine states with complex dynamics, including: glucose, pyruvate, acetaldehyde, acetaldehyde dehydrogenase, acetate, ethanol and biomass concentrations, volume and active cell material. Biomass is produced from glucose and, with a lower performance, acetate. For more details on the model refer to the cited article.

The performance index is the biomass concentration, subject to a certain maximum volume. Thus, actually total biomass is the magnitude to be maximized. The parameters of the initial solution for the fed-batch simulations are those originally used by Pham et al.[42] These were also used by Lei et al.[7] to test the first principles model in fed-batch simulation. The initial solution has seven liters, and a 1 g/L concentration of glucose. The initial active cell material is 0.1 g/L and the proportional of acetaldehyde dehydrogenase is 0.0075 (taken from batch simulations in [7]). The other initial concentrations are set to zero. The glucose concentration in the feed solution is 100 g/L.

The aim is to maximize the final biomass concentration for a fixed processing time of 10 h per batch by properly choosing the feeding profile. To this end, the feeding law is discretized to 100 intervals wherein it remains constant. After this discretization, the optimization problem is transformed into a seeking problem in a 100-Di space. Notice that this is a much more ambitious optimization than those pursued in many other articles.

The simulations performed present a ±1% of measurement noise in the final biomass concentration, as well as in the initial conditions. Initial variable conditions were simulated by adding a ±5% of variability to the biomass concentration, the active cell material and the acetaldehyde dehydrogenase. These may not seem to be very high noise and variability percentages. They are used for the sake of clarity to show the effect of the different parameters of the presented approach. The influence of higher measurement noise and initial variability percentages is also studied.

The solutions space is constrained by two restrictions. First, the feeding flow in any interval cannot be negative. Second, the reactor has a maximum volume which cannot be exceeded. The maximum volume was set to that reached with the feeding law used in [42] and [7] (an exponential phase of three h followed by a constant phase of seven h) so that direct comparison is possible. The constraints are directly applied over the particular solutions before they are used to process a batch. If the feeding flow of a certain interval is negative, then it is set to zero. If the maximum volume is exceeded, a constant is subtracted to the whole feeding law. The maximum volume constraint is also handled using the approach presented in the previous section.

Simulation Study

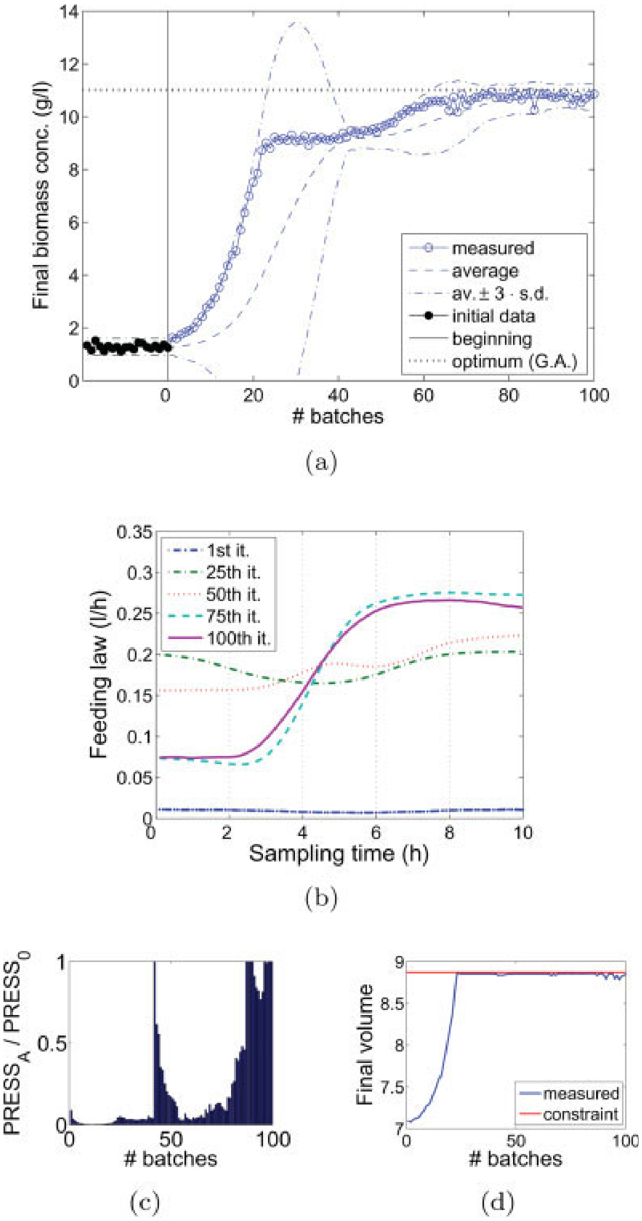
Analysis of parameters

There are several parameters of the presented optimization approach whose influence should be analyzed. First, let us have a look at the example shown in Figure 1. This optimization was performed starting from scratch that is, null feeding law. The first 20 circles in Figure 1a represent the initial set of batches. The optimization is very fast until the volume reaches the maximum established (see Figure 1d). In 20 batches, the performance index increases from 1.5 g/L to 9 g/L app. simply by incrementing the flow of substrate. The improvement stops between batches #20 and #40 (Figure 1a). This happens because the adaptive model has to be filled with batches of maximum volume to adjust to the current situation. From that point on, the optimization keeps on improving the performance. Notice that this latter improvement is obtained with a fixed volume. Therefore, at a certain point of the optimization the gradient is changed from "fill the reactor" to "optimize at a constant volume". At the end of the process, the average performance index reaches 10.74 g/L, with a standard deviation of 0.16. This result outperforms the feeding law used in [42] and [7], which obtains a performance close to 10 g/L.

The evolution of the feeding law can be seen in Figure 1b. The feeding law corresponding to the first iteration is almost null. Between iterations 25 and 50, the feeding law already fills the reactor up to its maximum volume. At iteration 75, the feeding law has already reached a steady profile.

The evolution of the index $PRESS_A/PRESS_0$ is presented in Figure 1c. The behavior of this index is very informative about the evolution of the optimization. When a model which captures a high-degree of variability of the process is identified, $PRESS_A/PRESS_0$ decreases and an improvement direction is found. The aforementioned change in the gradient can also be observed in this figure, where the interval in which each of the two gradients are used is seen as a valley in the $PRESS_A/PRESS_0$ index values.

In Figure 1a, the optimum final biomass concentration obtained by means of a genetic algorithm (GA) is also depicted. The GA is run for 100 generations with 100 individuals each—a total of 10,000 evaluations, 100 times more than the self-tuning optimizer. The GA is used to compute[d] the maximum value of the performance index, for the sake of comparison with the self-tuning controller. Therefore, the evaluations of the algorithm are performed with 0% of measurement noise and no initial variability. The GA initialized with random values yielded a poorer performance than the self-tuning optimizer, showing the suitability of the latter in high-dimensional seeking problems. Then, the set of 100 feeding laws yielded by the self-tuning controller were used as initial population for the GA. Notice this set not only includes the best solutions provided by the self-tuning controller, but also initial and intermediate ones. Thus, the region explored by the GA is large enough. As seen in Figure 1a, the optimum found by the GA is that of the extremum controller.



(a)

(b)

(c)

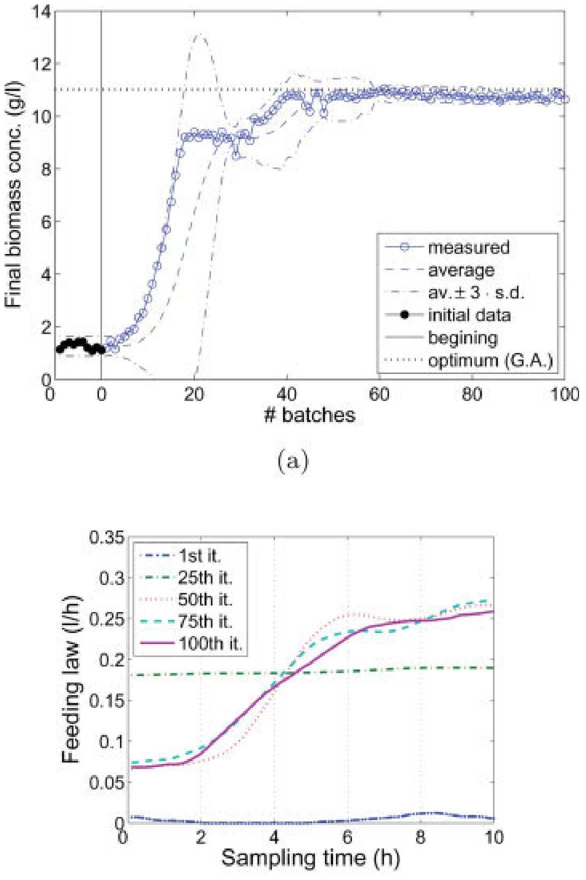
(d)

Figure 1. Simulation performed with an adaptive model generated with 20 batches.

Initial variability of ±5% (initial values included in the model) and measurement noise of ±1%. PRBS signal of adaptive amplitude ($g = 0.03$). The Figure includes: (a) performance of the batches along with the evolution of the average, and ± 3 times the standard deviation. The vertical line marks the beginning of the optimization. The horizontal dotted line marks the result obtained with a genetic algorithm; (b) feeding law for different iterations; (c) evolution of index $PRESS_A/PRESS_0$; and (d) evolution of the final volume. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

Now lets pay attention to the number of batches used to generate the model. Figure 2 shows the results of an optimization performed with the same parameters as those used in Figure 1, except for the number of batches. In this case, a window of 10 batches is used to generate the adaptive model. The main difference is clear: the optimization converges faster due to a faster movement of the average. On the other

[d] Given the complexity of the model used for the simulations, obtaining the optimum analytically is unfeasible.



(a)

(b)

Figure 2. Simulation performed with an adaptive model generated with 10 batches.

Initial variability of ±5% (initial values included in the model) and measurement noise of ±1%. PRBS signal of adaptive amplitude ($g = 0.03$). The Figure includes: (a) performance of the batches along with the evolution of the average, and ± 3 times the standard deviation. The vertical line marks the beginning of the optimization. The horizontal dotted line marks the result obtained with a genetic algorithm, and (b) feeding law for different iterations. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

hand, the advantage of a high-number of batches is that of every statistical model: less uncertainty in the estimations.

Notice that in this latter optimization, the final feeding law (Figure 2b) is not exactly equal to that of the previous one (Figure 1b). This is expected to happen because of the local optimality of the solution found and the uncertainty associated to it. This last due to the fact that the solution (the feeding law) has a dimension much bigger than the number of samples used to estimate it, the uncontrolled variability, noise, and so on. Nonetheless, the performances reached are very close together(and qualitatively equals), being also close to the solution computed with the GA.

It must be stressed that in both previous simulations, the process is started from scratch in order to highlight the capabilities of the approach. In practice, some suboptimal initial feeding law would be used, thus, shortening the lengthy transient shown in the figures.

Apart from the number of batches of the model, the other principal design parameter is the excitation signal. In this article, a PRBS is used to identify the model. The PRBS is a binary deterministic signal, which can be designed so that the identifiability of the variables in the gradient can be assured after a certain number of trials. It moves the input in such a way that the influence of each of the input variables alone in the output can be identified. Since the number of parameters is high, a 2^{12} period signal was used. On the other hand, for the effect of the PRBS to be observed at the output, the amplitude of the PRBS has to be high enough to be discriminated from the rest of the variability in the process. Nonetheless, care should be taken when increasing the amplitude in order not to take the process out of a pseudo linear area. Finally, as a practical matter, the amplitude of the PRBS should be as low as possible to avoid low-performance batches during the process.

After extensive experimentation, it was observed that filtering using a smoothing window of the PRBS, the gradient and the current feeding law has beneficial effects for the identification of the model. The filtering smoothes the signals, making the values in adjacent sampling times more close together. Since most processes have dynamic nature, it is expected that adjacent feeding intervals are somehow correlated. In general, this is true for the process variables in a batch process.[43] A filtered feeding law is somehow assuming a certain degree of dynamics in the process. If this assumption is correct, the identification of the model is improved. Care should be taken when sharp changes between phases occur during the batch processing, for instance, due to different processing steps. Also, the inclusion of the initial state can be advantageous for a better identification of the gradient.

The issues commented in the two previous paragraphs—the amplitude of the excitation signal and the signals filtering—along with the influence of other variability sources and measurement noise, have been studied for the *Sacharomyces cerevisiae* example. Twenty optimizations were performed for each of the five cases presented in Table 1, and three values of the amplitude of the excitation signal –g equal to 0.01, 0.03 and 0.05. The options of Table 1 allow to study separately the influence of g in the ideal case (Option 1), the effect of nonexplained variance on the optimization (Option 2), the improvement due to the filtering of the PRBS, the gradient and the current feeding law (Option 3), and the improvement due to the initial state incorporation to the model—without taking the initial state into account to compute the limit of the gradient (Option 4) or taking it into account (Option 5).

Each optimization trial is stopped when five consecutive batches outperform the average performance plus three times the standard deviation of the initial set of batches. This is an indication that the optimization procedure has caused a significant improvement in the performance of the process. The

Table 1. Different Options for the Simulation

Opt. 1:	No initial variability nor measurement noise
Opt. 2:	Initial variability (±5%) and measurement noise (±1%)
Opt. 3:	Opt. 2 + Filtering of the PRBS, the gradient and the final feeding law
Opt. 4:	Opt. 3 + Addition of initial biomass conc. ($\hat{y}^{n}_{z,i+1} = 0$ assumed in (28))
Opt. 5:	Opt. 3 + Addition of initial biomass conc.

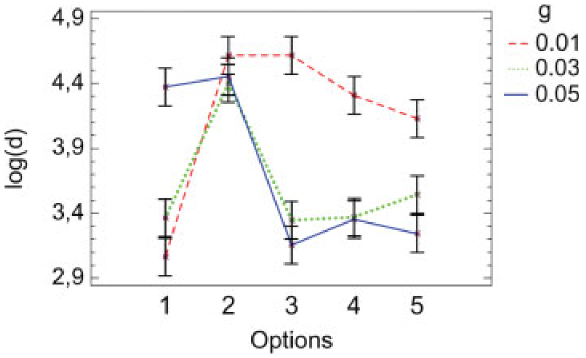


Figure 3. Analysis of variance of the influence of the options in Table 1 in the performance of the optimization process for different *g* values.

d is the number of batches to reach a significant improvement. The logarithm of *d* was used to correct for normality. Least significative difference (LSD) intervals of a 95% confidence are shown for the interaction plot between *g* and the options. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

number (d) of the batch at which the optimization is stopped is recorded. That number is a measure of the time of convergence of the optimization process. If 100 batches are processed without five consecutive outperforming the limit, it is supposed the optimization is not performing adequately and it is also stopped ($d = 100$). An analysis of variance (ANOVA) was performed on the d value, showing statistical significative differences (p-value $\leq$ 0.05) due to g, the options and their interaction. The least significative difference (LSD) intervals for a 95% confidence are shown for the interaction plot in Figure 3.

From options 1 and 2, it can be seen that the unknown sources of variability, which appear in option 2 in the form of measurement noise and initial variability, significantly reduce the performance of the optimization. This happens for both $g = 0.01$ and $g = 0.03$, but not for $g = 0.05$, for which the performance for option 1 is also poor. This means that the amplitude of the excitation signal should be carefully chosen. If the gain of the excitation signal is high enough, it takes the process out of a pseudo-linear region. When this occurs, a nonreliable gradient is being obtained reducing the performance of the optimization.

Filtering the signals made the optimization process robust to the 1% of noise and 5% of initial variability with $g = 0.03$. To see this, notice that for options 3, 4 and 5 and this g value, the performance of the optimization is similar to that of the ideal case (option 1). Now, focusing on the line representing $g = 0.05$ in Figure 3, options 3, 4 and 5 clearly outperform option 1, which is not an intuitive result since the latter is noise and variability free. This can be easily explained. As commented before, if the excitation signal is too large, the process is moving in a nonlinear area and the model is not correctly identified. Filtering the excitation signal smoothes its effect on the output. Hence, filtering produces a reduction of the variability induced in the output together with an enhancement of the identifiability. Therefore, the filtering operation is strongly recommended.

Finally, the addition of the initial biomass in the model enhances the identifiability when low values of g are used (compare options 3 and 4 for $g = 0.01$ in Figure 3). This enhancement may be more noticeable when a higher initial variability is present. Since an excitation signal of low-amplitude is desirable, it can be concluded that the inclusion of additional measurements together with the manipulable variables may be convenient in certain cases. On the other hand, taking into account the effect of the initial conditions in the heuristic limitation of the gradient did not cause any significative improvement for the cases studied. Comparing options 4 and 5 for all the g values, no significative statistical difference was found.

There is still a further analysis which can be made in order to observe the robustness of the approach to larger values of measurement noise and variability. A number of simulations using the parameters in option 5 (Table 1) is performed for different noise percentages –1%, 5% and 10%-, initial variability percentages –5% and 10%, and g values –0.03, 0.09 and 0.15. For every combination of the preceding parameters, five simulations of 100 batches each are carried out, and the final biomass concentration of the last 20 batches is recorded. An ANOVA was performed from this information (p-value $\leq$ 0.05), taking interactions into account. Both the third-order interaction and the interaction between the initial variability and g were not statistically significative. Yet, the two-order interactions between the measurement noise an the other two factors were significative. The least significative difference (LSD) intervals of a 95% confidence are shown for these interactions in Figure 4.

From Figure 4a it can be observed that the higher the percentage of measurement noise, the lower the performance. Nonetheless, the use of a sufficiently high g value attenuates the effect of the measurement noise. On the other hand, as commented before, if g is too high, the performance is reduced. For instance, using $g = 0.09$ is significative better than using $g = 0.15$ for 5% of measurement noise. Therefore g has to be carefully identified. This can be done for instance starting from a low-value and gradually incrementing it until the $PRESS_A/PRESS_0$ index starts decreasing. From Figure 4b, it can be seen that the influence of different percentages of initial variability is only evident for high-noise values. Notice that the noise also corrupts the measurements of the initial state. Therefore, it can be concluded that the measurement noise is the principal cause of reduction in the performance of the optimization. This is coherent with the results of the ANOVA, where the measurement noise alone was the most significative cause of variability.

It is clear that the optimization algorithm is not completely robust to noise nor to initial variability. Therefore, the repetition of quality measurements to reduce the noise and the tight control in the initial conditions seems justified in some critical cases. Finally, it is worth noticing that in all cases the self-tuning optimizer identified feeding profiles of similar shape (see Figure 5).

Adaptation to process changes

One of the additional advantages of the presented approach is that the feeding law is adapted when state changes or transitions in the process occur. This kind of transitions can be due to changes in the environmental conditions, degradation

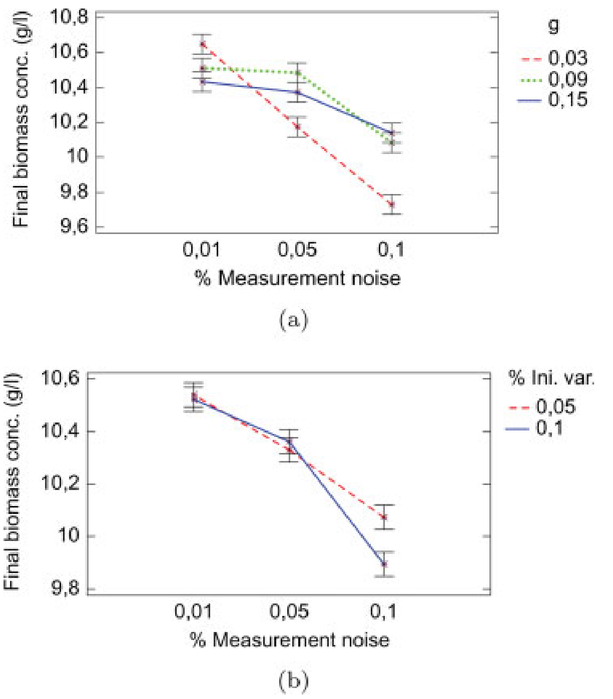


Figure 4. Analysis of variance of the influence of the *g* value in the performance of the optimization process (final biomass concentration) for different percentages of measurement noise and initial variability.

Least significative difference (LSD) intervals of a 95% confidence are shown for (a) the interaction plot between the percentage of measurement noise and *g* and (b) the interaction plot between the percentages of measurement noise and of initial variability. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

of the process and maintenance operations. Transitions can be smooth, like degradation, or sudden, like grade transitions between the processing of different products. In both cases, it is desirable to adjust the feeding law to the current situation to obtain the best performance.

In order to simulate state transitions, the yield coefficient which multiplies the stoichiometric reaction where glucose is converted in biomass is modified. This is coefficient $k7$ in.[7] In that paper, $k7$ is constant during the whole batch processing and equal to 1.203. By making $k7$ a function of the batch processing time and changing it, the optimum feeding law changes its shape. For example, if $k7$ is made bigger in the initial phase, a benefit will be obtained by increasing the feeding flow in that phase and viceversa.

In Figure 6, a simulation with several sudden transitions is presented. The simulation is started using the feeding law proposed in.[42] This contains an exponential phase of 3 h followed by a constant phase of 7 h (see Figure 6b). The first part of the optimization (up to batch number 100 including the initial set of batches) is carried out with $k7$ set to a constant value equal to 1.203. The result is that the optimization improves the performance in almost 1 g/l maintaining a constant final volume. From batch number 101 to 200, the value of $k7$ follows a increasing linear function of the batch time (see Figure 6d). After 20 batches have been processed, the procedure gradually adapts the feeding law to the new scenario, leading to a decrease of the feeding flow at the initial phase and an increase in the final phase (see Figure 6b). Finally, at batch 201, $k7$ starts following a linear decreasing function and, again, the optimization procedure adjusts the feeding law. Notice that in both the second and third state of the process, the optimization does not reach the optimum yielded by the GA. For this sort of sudden transitions, a low or adaptive number of batches in the model is desirable to accelerate the optimization. The changes in the gradient that leads the optimization can be identified through the evolution of the index $PRESS_A/PRESS_0$ in Figure 6c.

Dealing with several performance indices

This section concerns the use of several performance indices. Let us imagine the objective is to obtain 12 g/l of biomass per processed fed-batch, minimizing the volume of the final liquid. This objective can be stated as minimizing J_1 and J_2, with

$$J_1 = -(12 - \chi_f)^2 \tag{47}$$

$$J_2 = \frac{\chi_f}{\upsilon_f} \tag{48}$$

where χ_f is the final biomass concentration, and υ_f the final volume, which depends solely on the feeding law. Note that, although here the objective has been defined in this way, other possibilities exist and could be applied straightforward in the optimization procedure.

A single performance index can be obtained from Eqs. 47 and 48 as

$$J = J_1 + J_2 \tag{49}$$

No weighting factors have been added for the sake of simplicity.

The result of the optimization of J is shown in Figure 7. After convergence (approximately from batch number 60), the following batches present a final biomass concentration

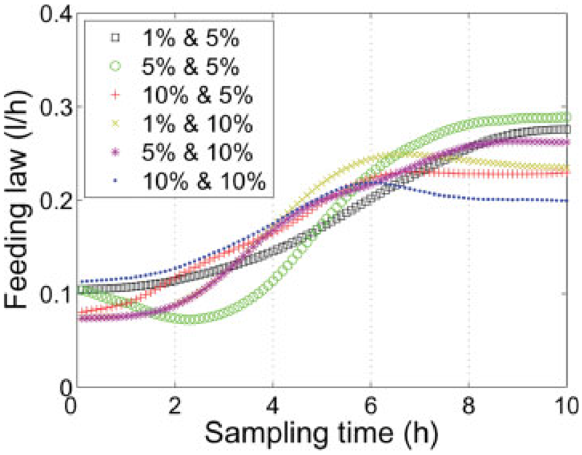


Figure 5. Examples of feeding laws optimized for different percentages of measurement noise and of initial variability.

[Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

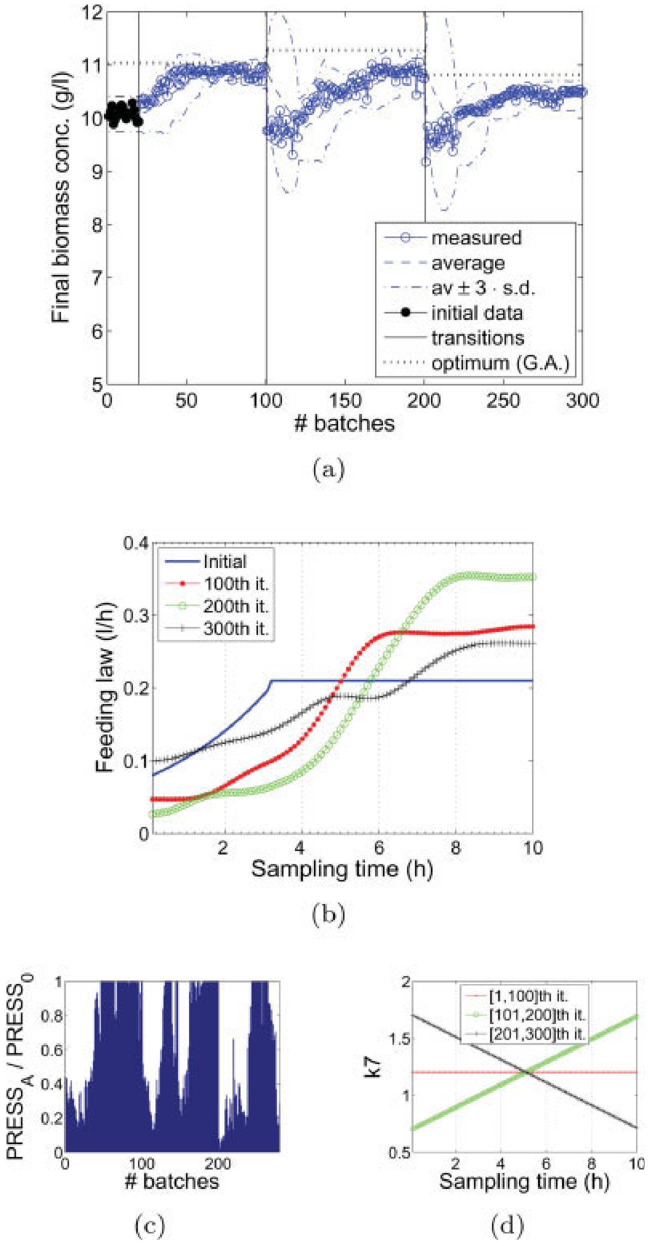


Figure 6. Simulation of sharp process changes.

Optimization performed with an adaptive model generated with 20 batches. Initial variability of ±5% (initial values included in the model) and measurement noise of ±1%. PRBS signal of adaptive amplitude ($g = 0.03$). The Figure includes: (a) performance of the batches along with the evolution of the average, and ± 3 times the standard deviation. The vertical lines mark the beginning of the optimization and the changes in the process. The horizontal dotted lines mark the result obtained with a genetic algorithm; (b) initial feeding law and average feeding law for every period; (c) evolution of index $PRESS_A/PRESS_0$; and (d) value of $k7$ during the batch for every period. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

close to 12 g/L with a volume slightly bigger than 9.21. The high variability of the last batches is caused partially by the excitation signal. To reduce this variability, the excitation signal can be switched off and switched on again if degradation or condition changes occur. This reduces part of the variability.

Another possibility is to optimize the two performance indices directly by treating them as different output variables of the u-PLS model. The result is shown in Figure 8. Although some

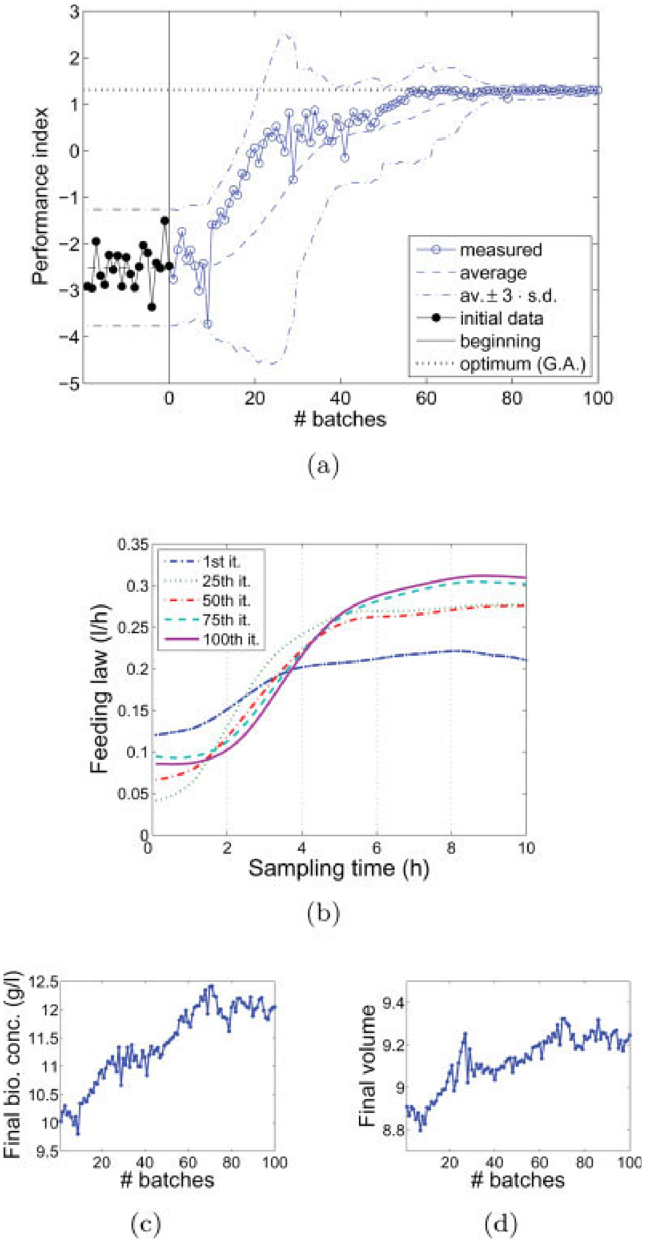
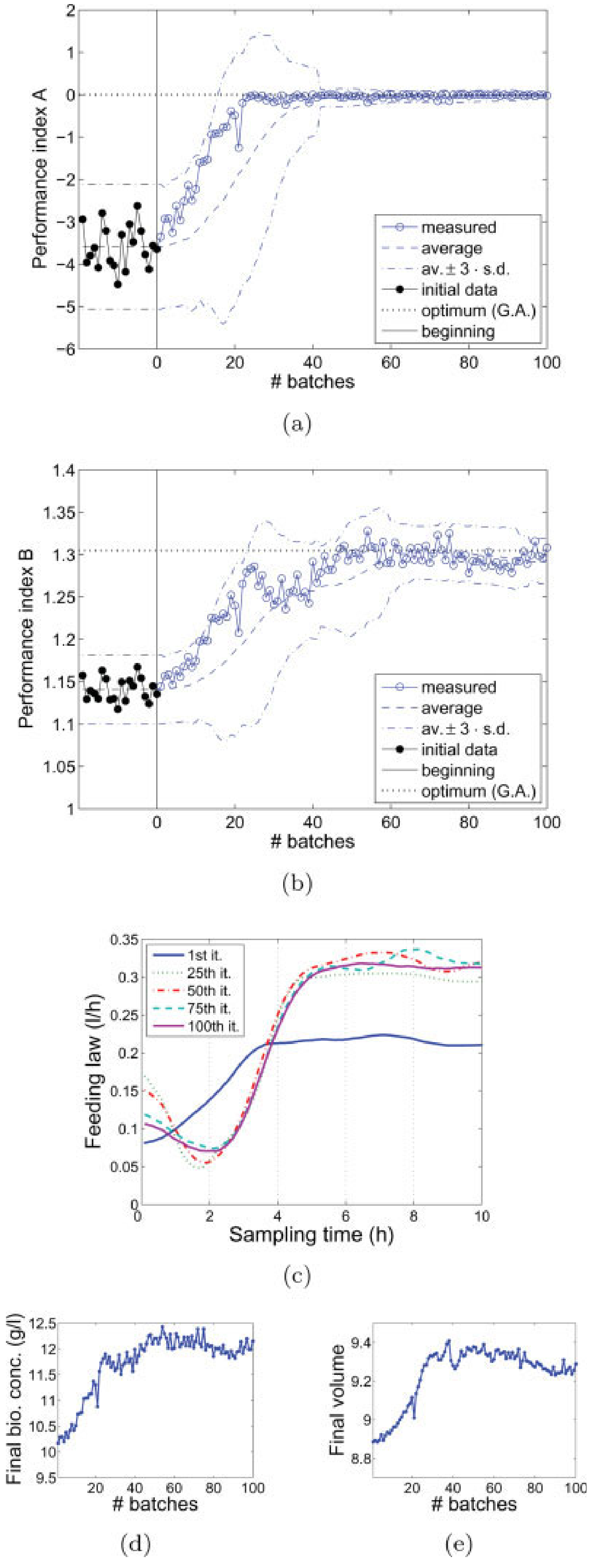


Figure 7. Simulation for the combination of two performances indices in one.

Optimization performed with an adaptive model generated with 20 batches. Initial variability of ±5% (initial values included in the model) and measurement noise of ±1%. PRBS signal of adaptive amplitude ($g = 0.03$). The Figure includes: (a) performance of the batches along with the evolution of the average, and ± 3 times the standard deviation. The vertical line marks the beginning of the optimization. The horizontal dotted line marks the result obtained with a genetic algorithm; (b) feeding law for different iterations; (c) evolution of the final biomass concentration; and (d) evolution of the final volume. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]



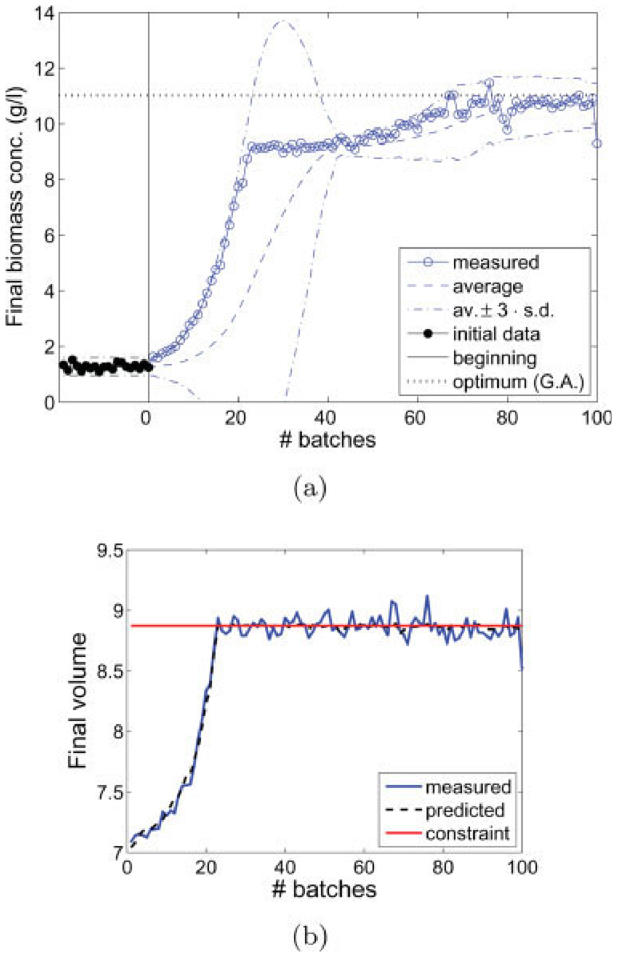


Figure 9. Simulation performed with two adaptive models generated with 20 batches for both optimization and to constrain the volume with no uncertainty margin.

Initial variability of ±5% (initial values included in the model) and measurement noise of ±1%. PRBS signal of adaptive amplitude ($g = 0.03$), and null uncertainty margin ($\lambda = 0$). The Figure includes: (a) performance of the batches along with the evolution of the average and ± 3 times the standard deviation. The vertical line marks the beginning of the optimization. The horizontal dotted line marks the result obtained with a genetic algorithm; and (b) evolution of the final volume. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

Figure 8. Simulation for two performances indices.

Optimization performed with an adaptive model generated with 20 batches. Initial variability of ±5% (initial values included in the model) and measurement noise of ±1%. PRBS signal of adaptive amplitude ($g = 0.03$). The Figure includes: (a) first performance index for the batches along with the evolution of the average and ± 3 times the standard deviation. The vertical line marks the beginning of the optimization. The horizontal dotted line marks the result obtained with a genetic algorithm; (b) second performance index; (c) feeding law for different iterations; (c) evolution of the final biomass concentration; and (d) evolution of the final volume. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

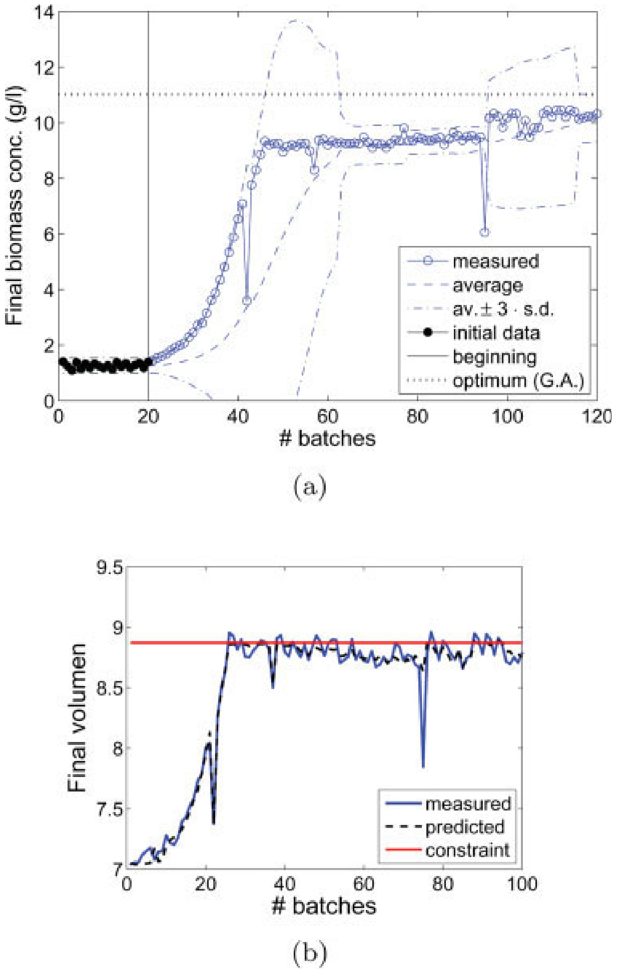


Figure 10. Simulation performed with two adaptive models generated with 20 batches for both optimization and to constrain the volume.

Initial variability of ±5% (initial values included in the model) and measurement noise of ±1%. PRBS signal of adaptive amplitude ($g = 0.03$) and uncertainty margin proportional to the model uncertainty and variability in the volume. The Figure includes: (a) performance of the batches along with the evolution of the average and ± 3 times the standard deviation. The vertical line marks the beginning of the optimization. The horizontal dotted line marks the result obtained with a genetic algorithm; and (b) evolution of the final volume. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

slight differences may be seen between both cases (Figures 7 and 8), a more detailed investigation showed no general differences in the velocity of convergence nor in the final error.

Constraints handling

In order to check the performance under presence of constraints, the experiment of Figure 1 was repeated using the information of a u-PLS model for handling the volume limitation

$$f(\upsilon) = \upsilon - \upsilon_{\max} \qquad (50)$$

Figure 9 shows an optimization trial for $\lambda = 0$ -no security margin due to uncertainty is used. The performance obtained is equivalent to that of Figure 1. The constraint is violated several times, but the process is maintained around it. Such a result is very interesting since this approach can be applied to any measured variable, even when the relationship of the constrained variables with the manipulable variables is not clear. In this latter case, how to satisfy the constraint is not trivial at all. Yet the applicability of this approach is direct.

In Figure 10, the optimization is repeated for

$$\lambda_i = -3 \cdot \left(1 - \left(\frac{PRESS_{\upsilon,A'}}{PRESS_{\upsilon,0}}\right)_i\right) \cdot \sigma_{\upsilon,i} \qquad (51)$$

where the idea is to maintain the volume at a distance to the maximum proportional to the reliability of the u-PLS model. If all the variability in the volume is explained by the u-PLS model, that is, $(PRESS_{\upsilon,A'}/PRESS_{\upsilon,0})_i \simeq 0$-, the gradient is reliable enough to bring the volume close to the maximum. If little is known about the input-constraint relationship, or if the constraint is hardly controllable from the input, a security margin is applied to maintain the batches far enough from the constraint. Figure 10 shows that the violation of the constraint is reduced but also the performance of the process. This was expected since the optimum feeding law lays at the constraint.

Conclusions

In this article, a new procedure to optimize fed-batch processes is presented. This procedure follows a self-tuning extremum-seeking algorithm performed in combination with an adaptive u-PLS model. The approach has proven to be very effective and versatile, low-sensitive to nonlinearities and uncontrolled variability, and robust to changes in the process. Extensions to handle measurements on initial conditions, several performance indices and inequality constraints were also provided.

As main advantage, no knowledge about the process is necessary. Nonetheless, if a pseudo-optimum feeding law has been computed or inferred from fundamental knowledge, it can be integrated in the optimization straightforward by using it as starting point.

The optimization procedure is specially suited for multivariate problems, but inherits the limitations of self-tuning extremum-seeking controllers, that is, the possibility of convergence to local optima and the need for exciting the process to identify the gradient. Some notions about the convenient nature of the excitation signal and the inclusion of additional information in the model—to model different sources of variability apart from the manipulable variables—have been discussed.

Although the proposal has been presented for the optimization of fed-batch fermentations, it can be used for any fed-batch process and for the optimization of manipulable variable profiles in batch processes.

Acknowledgment

Research in this area is partially supported by the Spanish government and the European Union (CICYT-FEDER DPI2005-01180 and CTM2005-06919-C03/TECNO), and by the FPU grants program, Secretaría de Estado de Educación y Universidades (Ministry of Education and Science, Spain), grant AP2003-0346. The revisors are acknowledged for their useful comments.

Notation

$\underline{A}$ = 3-D matrix
A_i = 2-D matrix
a = vector
a_i = scalar

NIPALS algorithm

The algorithm used to compute the PLS models in this article is the following:

```
normalize $U$ and $Y$
compute $PRESS_0$
start from $a = 0$
do
  $a = a + 1$
  choose start $v_a$ (one column of matrix $Y$)
  loop until convergence of $\mathbf{t}_a$
    $\mathbf{w}_a = \mathbf{X}^T \cdot \mathbf{v}_a/(\mathbf{v}_a^T \cdot \mathbf{v}_a)$
    normalize to $\|\mathbf{w}_a\| = 1$
    $\mathbf{t}_a = \mathbf{X} \cdot \mathbf{w}_a/(\mathbf{w}_a^T \cdot \mathbf{w}_a)$
    $\mathbf{q}_a = \mathbf{Y}^T \cdot \mathbf{t}_a/(\mathbf{t}_a^T \cdot \mathbf{t}_a)$
    $\mathbf{v}_a = \mathbf{Y} \cdot \mathbf{q}_a$
  end
    $\mathbf{p}_a = \mathbf{X}^T \cdot \mathbf{t}_a/(\mathbf{t}_a^T \cdot \mathbf{t}_a)$
    $\mathbf{X} = \mathbf{X} - \mathbf{t}_a \cdot \mathbf{p}_a^T$
    $\mathbf{Y} = \mathbf{Y} - \mathbf{t}_a \cdot \mathbf{q}_a^T$
    compute $PRESS_a$
while $PRESS_a < PRESS_{a-1}$
```

Literature Cited

1. Bonvin D, Srinivasan B, Hunkeler D. Control and optimization of batch processes - improvements of process operation in the production of specialty chemicals. *IEEE Trans on Cont Sys Mag*. 2006;26:34–45.
2. Edgar TF. Control and operations: when does controllability equal profitability? *Comp and Chem Eng*. 2004;29:41–49.
3. Yamanè T, Shimizu S. Fed-batch techniques in microbial processes. *Advan in Biochem Eng/Biotechnol*. 1984;30:147–194.
4. Lee J, Lee SY, Park S, Middelberg APJ. Control of fed-batch fermentations. *Biotechnol Advs*. 1999;17:29–48.
5. Zhang H, Lennox B. Integrated condition monitoring and control of fed-batch fermentation processes. *J of Process Control*. 2004;14:41–50.
6. Shimizu K. An overview on the control system design of bioreactors. *Advan in Biochem Eng/Biotechnol*. 1993;50:65–84.
7. Lei F, Rotbøll M, Jørgensen SB. A biochemically structured model for Saccharomyces cerevisiae. *J of Biotechnol*. 2001;88:205–221.
8. Jobé AM, Herwiq C, Surzyn M, Walker B, Marison I, von Stockar U. Generally applicable fed-batch culture concept based on the detection of metabolic state by on-line balancing. *Biotechnol and Bioeng*. 2003;82:627–639.
9. Picó-Marco E, Picó J, De Battista H. Sliding mode scheme for adaptive specific growth rate control in biotechnological fed-batch processes. *Int J of Control*. 2005;78:128–141.
10. De Battista H, Picó J, Picó-Marco E. Globally stabilizing control of fed-batch processes with Haldane kinetics using growth rate estimation feedback. *J of Process Control*. 2006;16:865–875.
11. Valentinotti S, Srinivasan B, Holmberg U, et al. Optimal operation of fed-batch fermentations via adaptive control of overflow metabolite. *Control Eng Practice*. 2003;11:665–674.
12. Ng CW, Hussain MA. Hybrid neural network-prior knowledge model in temperature control of a semi-batch polymerization process. *Chem Eng and Processing*. 2004;43:559–570.
13. Rotem Y, Wachs A, Lewin DR. Ethylene compressor monitoring using model-based PCA. *AlChE J*. 2000;46:1825–1836.
14. Lee KS, Lee JH. Iterative learning control-based batch process control technique for integrated control of end product properties and transient profiles of process variables. *J of Process Control*. 2003;13:607–621.
15. Gao F, Yang Y, Shao C. Robust iterative learning control with applications to injection molding process. *Chem Eng Sci*. 2001;56:7025–7034.
16. Mezghani M, Roux G, Cabassud M, Dahhou B, Le Lann MV, Casamatta G. Robust iterative learning control of an exothermic semi-batch chemical reactor. *Maths and Comps in Sim*. 2001;57:367–385.
17. Fu P. Barford JP. Simulation of an iterative learning control system for fed-batch cell culture processes. *Cytotechnology*. 1992;10:53–62.
18. Levišauskas D, Galvanauskas V, Simutis R, Žilinskas A, Žilinskas J. Optimization of biomass production in fed-batch culture by feed and dilution control actions. *Info Technol and Contr*. 2006;35:383–390.
19. Franco-Lara E, Weuster-Botz D. Estimation of optimal feeding strategies for fed-batch bioprocesses. *Bioprocess and Biosystems Eng*. 2005;27:255–262.
20. Srinivasan B, Primus CJ, Bonvin D, Ricker NL. Run-to-run optimization via control of generalized constrains. *Control Eng Practice*. 2001;9:911–919.
21. Xiong Z, Zhang J. A batch-to-batch iterative optimal control strategy based on recurrent neural network models. *J of Process Control*. 2005;15:11–21.
22. Teixeira AP, Clemente JJ, Cunha AE, Carrondo MJT, Oliveira R. Bioprocess iterative batch-to-batch optimization based on hybrid parametric/nonparametric models. *Biotechnol progress*. 2006;22:247–258.
23. Dong D, McAvoy TJ, Zafiriou E. Batch-to-batch optimization using neural networks models. *Industrial Eng and Chem Res*. 1996;35:2269–2276.
24. Tucker IR. The extension of factor analysis to three-dimensional matrices. In: Frederiksen N, Gulliksen H. *Contributions to Mathematical Psychology*. New York: Holt, Rinehart and Winston; 1964:110–162.
25. Bro R. PARAFAC. Tutorial and applications. *Chemometrics and Intelligent Lab Systems*. 1997;38:149–171.
26. Smilde AK, Bro R, Geladi P. Multi-way Analysis, Application in the Chemical Sciences. England: John Wiley & Sons; 2003.
27. Wold S, Geladi P, Esbensen K, Öhman J. Multi-way principal components and PLS analysis. *J of Chemometrics*. 1987;1:41–56.
28. Nomikos P, MacGregor JF. Monitoring batch processes using multiway principal components analysis. *AIChE J*. 1994;40:1361–1375.
29. Duchesne C, MacGregor JF. Multivariate analysis and optimization of process variable trajectories for batch processes. *Chemometrics and Intelligent Lab Systems*. 2000;51:125–137.
30. Wellstead PE, Zarrop MB. *Self Tuning Systems Control and Signal Processing*. England: John Wiley & Sons; 1991.
31. Titica M, Dochain D, Guay M. Adaptive extremum seeking control of fed-batch bioreactors. *Euro J of Control*. 2003;9:618–631.
32. DeHaan D, Guay M. Extremum-seeking control of state-constrained nonlinear systems. *Automatica*. 2005;41:1567–1574.
33. Ariyur KB, Krstić M. *Real-Time Optimization by Extremum-Seeking Control*. England: John Wiley & Sons; 2003.
34. Wang HH, Krstić M, Bastin G. Optimizing bioreactors by extremum seeking. *Int J of Adaptive Control and Signal Processing*. 1999;13:651–669.
35. Martens H, Næs T. *Multivariate Calibration*. John Wiley & Sons; 1992.
36. Geladi P, Kowalski BR. Partial least-squares regression: a tutorial. *Analytica Chimica Acta*. 1986;185:1–17.
37. Nomikos P, MacGregor JF. Multivariate SPC charts for monitoring batch processes. *Technometrics*. 1995;37:41–59.

38. Nomikos P, MacGregor JF. Multi-way partial least squares in monitoring batch processes. *Chemometrics and Intelligent Lab Systems*. 1995;30:97–108.

39. Dayal BS, MacGregor JF. Recursive exponentially weighted PLS and its applications to adaptive control and prediction. *J of Process Control*. 1997;7:169–179.

40. Qin SJ. Recursive PLS algorithms for adaptive data modeling. *Comp and Chem Eng*. 1998;22:503–514.

41. Tracy ND, Young JC, Mason RL. Multivariate control charts for individual observations. *J of Quality Technol*. 1992;24:88–95.

42. Pham HTB, Larsson G, Enfors S. Growth and energy metabolism in aerobic fed-batch cultures of Saccharomyces cerevisiae: Simulation and model verification. *Biotechnol and Bioengineering*. 1998;60: 474–482.

43. Camacho J, Picó J. Multi-phase principal component analysis for batch processes modelling. *Chemometrics and Intelligent Lab Systems*. 2006;81:127–136.

Manuscript received July 28, 2006, and revision received Mar. 24, 2007.