PhD DISSERTATION

Author: José Camacho Páez
Supervisors: Jesús Picó i Marco
Alberto J. Ferrer Riquelme

New Methods Based on the Projection to Latent Structures for Monitoring, Prediction and Optimization of Batch Processes.

November 15, 2007

Group of Control of Complex Systems
Department of Systems Engineering and Control
Technical University of Valencia

A Rosa

This work 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.

This Thesis has been awarded with the second Rosina Ribalta Prize, edition IX, to the best doctoral projects in the area of Information and Communication Technologies, from the EPSON Foundation.

Abstract

The present Thesis is devoted to the study and application of new methods based on the projection to latent structures to batch processes. The potential economic profit and safety and ecological benefit in the improvement of the modelling, monitoring and control of these processes is widely accepted in both the academic and industrial fields.

The document has been structured in three blocks. Firstly, the related literature is revised and summarized in an introductory part. The principal projection to latent structures methods are presented using a comprehensive common nomenclature. Also, the main contributions in the literature for the modelling, monitoring and control of batch processes are discussed.

In the second block of the document, the general features of batch process data and the basis of the methods based on the projection to latent structures are studied. The study, as the rest of the Thesis, is limited to bilinear modelling approaches. This allows to observe the limitations of these modelling methods when applied to batch processes, some of which are stated here for the first time. In this block, a first theoretical analysis is carried out to investigate the dynamic covariance structures of different modelling approaches. The conclusion of this analysis is that the multi-phase modelling structure has a nice feature which makes it specially attractive for batch process modelling: the flexibility to adjust to the process nature. Multi-phase and single-phase processes, with short or long term dynamics -or even combination of both- can be effectively modelled with the multi-phase structure. The second step in this block of the document is to investigate one of the most widely used strategies to set the parameters of the projection to latent structures methods: cross-validation. New algorithms for cross-validation are proposed and extended to the batch process data case. This step is of vital importance for the development of an automatic modelling algorithm for batch data. This algorithm is presented in the last part of this second block of the Thesis. The algorithm, named Multi-phase algorithm, has been designed together with a set of tools to analyze and handle multi-phase models: the Multi-phase Framework. This framework is a modelling approach which combines the automatic recognition capabilities of the algorithms from the field of Computer Science with the well-founded data analysis from the field of Statistics.

After the study of the modelling of batch processes, several applications where this modelling is necessary are investigated in the last block of the document. The problems addressed are: the off-line and on-line monitoring, the on-line end-quality prediction, the on-line estimation of batch trajectories and the process optimization. All these problems share the common feature that batch processes are difficult to understand and to model. Nonetheless, each of the problems studied presents its particularities, which need to be addressed independently. The contributions in each of the applications go beyond the use of the new modelling approach. In both the off-line and on-line monitoring of batch processes, several improvements in the development of the monitoring system are suggested. In the on-line end-quality prediction and estimation of trajectories, a broad comparison among several modelling approaches is supplied. Finally, a new optimization algorithm, based on the combination of self-tuning extremum seeking and projection to latent structures methods, is proposed.

Resumen

La presente Tesis doctoral versa sobre el estudio y aplicación de nuevos métodos basados en técnicas estadísticas de proyección sobre estructuras latentes en procesos por lotes. La mejora del modelado, la monitorización y el control de este tipo de procesos tiene una elevada importancia estratégica tanto desde el punto de vista del potencial beneficio económico como del de la seguridad y protección medioambiental.

Este documento está estructurado en tres bloques. En el bloque introductorio se revisa y resume la literatura relacionada con el tema de la Tesis. En este bloque se introducen los principales métodos basados en técnicas de proyección sobre estructuras latentes utilizando una nomenclatura común. Adicionalmente se discuten las principales aportaciones al modelado, la monitorización y el control de procesos por lotes halladas en la literatura.

El segundo bloque del documento está dedicado al estudio de las características generales de los datos provenientes de procesos por lotes, así como de los fundamentos estadísticos de los métodos basados en la proyección sobre estructuras latentes. Este estudio, como el resto de la Tesis, está limitado a métodos bilineales y analiza las limitaciones de los métodos para el modelado de procesos por lotes, algunas de las cuales constituyen contribuciones originales de la presente tesis. El bloque comienza con el estudio de las estructuras dinámicas de varianza-covarianza capturadas por distintas propuestas de modelado. La principal conclusión es que la estructura denominada multi-fase posee una cualidad que la hace especialmente atractiva para el modelado de procesos por lotes: la flexibilidad para ajustarse a la naturaleza del proceso. Tanto procesos multi-fase como de una sola fase, con dinámicas de alto o reducido orden, pueden ser modelados eficazmente con la estructura multi-fase. Tras este estudio, el bloque continúa con el análisis de una de las estrategias más usadas para ajustar el valor de los parámetros en los métodos basados en la proyección sobre estructuras latentes: la validación cruzada. Se han propuesto dos nuevos algoritmos de validación cruzada, desarrollándose posteriormente su extensión al caso particular del modelado de procesos por lotes. Este paso es de vital importancia a la hora de definir un algoritmo automático para el modelado de procesos por lotes, algoritmo que es presentado en la última parte del bloque con el nombre de algoritmo Multi-fase. El algoritmo Multi-fase ha sido incluido, conjuntamente con una

serie de herramientas para el análisis y manejo de modelos multi-fase, en el llamado Entorno Multi-fase. Este entorno para el modelado de procesos por lotes combina el componente automático de los algoritmos de la Ingeniería Informática con el análisis de datos, propio del campo de la Estadística.

Tras el estudio del modelado en el segundo bloque de la Tesis, en un tercer bloque se investigan distintas aplicaciones en el ámbito de los procesos por lotes donde el modelado es necesario. En concreto: la monitorización fuera de línea y en línea, la predicción en línea de la calidad final, la estimación de trayectorias y la optimización de procesos. Todos estos problemas comparten la limitación común de que los procesos por lotes son difíciles de entender y modelar. Sin embargo, cada problema presenta también sus peculiaridades, que deben ser abordadas de forma diferente. Las contribuciones del presente trabajo van más allá de la aplicación de la nueva propuesta de modelado a los distintos problemas. En la monitorización en línea y fuera de línea se proponen varias mejoras del sistema de monitorización. En el caso de la predicción de la calidad final y la estimación de trajectorias, se realiza una vasta comparativa de distintas técnicas de modelado. Finalmente, se propone un nuevo algoritmo de optimización de procesos, basado en la combinación de la búsqueda de extremos auto-ajustada y los métodos basados en la proyección sobre estructuras latentes.

Resum

La present tesi doctoral tracta l'estudi i aplicació de nous mètodes basats en tècniques estadístiques de projecció sobre estructures latents en processos per lots. La millora del modelatge, monitorització i control d'aquest tipus de processos té una elevada importància estratègica tant des del punt de vista del potencial benefici econòmic com des del de la seguretat i protecciò mediambientals.

Aquest document està estructurat en tres blocs. En el bloc introductori es revisa i resumeix la literatura relacionada amb el tema de la tesi. En aquest bloc s'introdueixen els principals mètodes basats en tècniques de projecció sobre estructures latents emprant una nomenclatura comuna. Addicionalment, es discuteixen les principals aportacions al modelatge, la monitorització i el control de processos per lots trobades en la literatura.

El segon bloc del document està dedicat a l'estudi de les característiques generals de les dades provinents de processos per lots, així com dels fonaments estadístics dels mètodes basats en la projecció sobre estructures latents. Aquest estudi, com la resta de la tesi, està limitat a mètodes bilineals i analitza les limitacions dels mètodes per al modelatge de processos per lots, algunes de les quals constitueixen contribucions originals de la present tesi. El bloc comiença amb l'estudi de les estructures dinàmiques de variança-covariança capturades per distintes propostes de modelatge. La principal conclusió és que l'estructura anomenada multi-fase té una qualitat que la fa especialment atractiva per al modelatge de processos per lots: la flexibilitat per ajustar-se a la natura del procés. Tant processos multi-fase com d'una sola fase, amb dinàmiques d'alt o reduït ordre, poden ser modelitzats eficaçment amb l'estructura multi-fase. Després d'aquest estudi, el bloc continua amb l'anàlisi d'una de les estratègies més emprades per ajustar el valor dels paràmetres en els mètodes basats en la projecció sobre estructures latents: la validació creuada. S'han proposat dos nous algoritmes de validació creuada, desenvolupant-se posteriorment la seua extensió al cas particular del modelatge de processos per lots. Aquest pas és de vital importància a l'hora de definir un algoritme automàtic per al modelatge de processos per lots, algoritme que és presentat en la darrera part del bloc amb el nom d'algoritme Multi-fase. L'algoritme Multi-fase ha sigut inclòs, juntament amb una sèrie de ferramentes per a l'anàlisi i maneig de models multi-fase, en l'anomenat

Entorn Multi-fase. Aquest entorn per al modelatge de processos per lots combina la component automàtica dels algoritmes de l'Enginyeria Informàtica amb l'anàlisi de dades propi del camp de l'Estadística.

Després de l'estudi del modelatge en el segon bloc de la tesi, en un tercer bloc s'investiguen distintes aplicacions en l'àmbit dels processos per lots en què el modelatge és necessari. En particular: la monitorització fora de línia i en línia, la predicció en línia de la qualitat final, l'estimació de trajectòries i l'optmització de processos. Tots aquestos problemes comparteixen la limitació comuna de la dificultat per entendre i modelitzar els processos per lots. Tanmateix, cada problema presenta també les seues peculiaritats, que deuen ser abordades de manera diferent. Les contribucions del present treball van més enllà de l'aplicaió de la nova proposta de modelatge als distints problemes. En la monitorització en línia i fora de línia es proposen diverses millores al sistema de monitorització. En el cas de la predicció de la qualitat final i l'estimació de trajectòries, es porta a terme una vasta comparativa de distintes tècniques de modelatge. Finalment, es proposa un nou algoritme d'optimizació de processos, basat en la combinació de la cerca d'extrems auto-ajustada i els mètodes basats en la projecció sobre estructures latents.

Agradecimientos

Me gustaría mucho agradecer el apoyo, tanto a nivel científico y técnico como (y preferentemente) a nivel humano, de mis dos directores de tesis. Ha habido momentos especialmente duros durante este trayecto, en los que siempre me han brindado su comprensión y apoyo incondicional.

Quiero agradecer a Kenneth S. Dahl (DuPont), Neal B. Gallagher (Eigenvector) y Daniel Aguado (Calagua, UPV) su amabilidad al suministrar los conjuntos de datos utilizados en la experimentación y a Enric Picó la traducción del resumen a valenciano.

Quiero recordar también a todos aquellos que han sido o son mis compañeros, que tanto han influido en mi trabajo y me lo han hecho una pizca más ameno: Rosa, Sam, Pedro, Jose, Evamparo, Roberto, Sergio, Kiko, Nacho, Nuria, Jose Manuel (Agriculture Man) y muchos más.

Es un hecho que el tiempo libre influye de forma directa en el trabajo de una persona. Mi agradecimiento va especialmente dirigido a las personas responsables de tantos buenos momentos. En primer lugar a Rosa. Gran parte de la confianza y constancia en este proyecto surge de ella, que es mi fuente de maná. En segundo lugar a toda mi familia: mis padres y mis suegros, a los que tanto echo de menos; mis hermanos Ana y Salvi, Golo y Paloma, Belem y Nacho; mis sobrinos Tom, Nico, Andrea y Pablo; Mary Nieves y todo un enorme etc. Por último, a todos mis amigos de Granada (Félix, Jorge, Álvaro, Julio, Fran, Ruth y Paco, Patxy, Javi), Valencia (Sam y Maria, Javi y Laia, Jose y Sara, Toni, Javi, Patxy, Alex y Michael) y Adra (Cristina y Fran, Lourdes y Antonio, Diana y Juan).

Valencia, 2007

Table of Contents

Justification, Objectives and Contributions

Batch processing[a] consists of the repetition of a number of actions to produce batches of a given product. A specific combination of raw materials is initially charged in an appropriate recipient. Then, materials are processed for a finite amount of time, following a predefined recipe. Finally, the product of a batch is discharged. This production strategy is used in industries of very different nature -eg. in pharmaceutical, chemical, biochemical, metal, etch or food industries- to manufacture high value, specialty products. Two principal features make batch processing specially attractive for this: the repetitive nature and the flexibility. The repetitive nature allows to learn from the process on a batch-to-batch basis, so that corrections can be applied to the recipe in order to optimize production. Flexibility allows to manufacture different products using the same processing line, simply by changing the recipe of raw materials and control actions.

Batch processes are fairly more complex to understand and to model than continuous processes. Most industrial processes are non-linear. This non-linearity has been traditionally overcome in continuous processes by maintaining the process close to an operation point, commonly a stable state, where the behavior of the process is approximately linear. Nonetheless, batch processes present operation trajectories instead of operation points, which require a big effort for the proper understanding and modelling. This inherent complexity is inherited by all the applications for which a process model is necessary, e.g. monitoring, prediction and optimization.

In typical applications, industrial batch processes produce an enormous amount of three-way data (batches $\times$ variables $\times$ sampling times) which is recorded for on-line treatment or posterior analysis. These are, nonetheless, difficult tasks due to the volume of the data and a low signal-to-noise ratio. One of the most widely used methods to extract the information from that kind of data are the methods based on the projection to latent structures (PLS-based methods), such as Principal Component Analysis (PCA) and Partial Least Squares (PLS). The main idea of these methods is to reduce

[a] Throughout this Thesis, the term batch process and processing will be used in a wide sense, including both batch and fed-batch -or semi-batch- processes. In fed-batch processes, some materials are supplied during the processing of a batch.

the dimension of the data by finding the linear combinations of variables that maximize some given criterion.

The PLS-based methods can be used for the development of data-driven models of a batch process, which may serve to different purposes: monitoring, quality prediction and optimization. Principally in the first two cases, PLS-based models have been reported to yield excellent performance in real problem applications. Moreover, in the context of the Spanish industry, very little has been done in this matter. Therefore, the development and application of traditional and advanced PLS-based methods has an important potential market.

The principal objectives pursued in this work are the following:

a) To identify the limitations in the modelling of batch processes with PLS-based methods.

There are several problems or limitations which may arise when applying PLS-based methods to model batch process data. When a specific process is to be modelled, one should know the tools or procedures available in the literature which may be more adequate for the current case. In order to properly understand the differences among the modelling approaches proposed to date, some effort in differentiation and classification needs to be done. This includes the following:

- Theoretical study of the features of PLS-based methods.
- Analysis of the principal features of batch process data.
- Study of the state-of-the-art in PLS-based modelling of batch processes.
- Interrelation of the three preceding points, in order to establish when the procedures suggested in a certain approach are appropriate for the modelling of a specific process.

b) To propose new methods for the overcoming of the limitations found

Once the principal limitations of PLS-based models are found, the next objective is to propose solutions for those limitations, having the practical applicability in mind. These solutions should be justified theoretically and proved experimentally using the data available. The data-driven recognition of certain features in the data, so as to automatically identify the convenient modelling procedure, is a principal concern in this work.

c) To apply these methods in different real problems.

There are several applications in which the PLS-based model of a batch process may be used: monitoring, estimation, prediction, regulation, tracking, optimization, etc. The third objective of this Thesis is to test the modelling solutions proposed in the preceding step to some of these problems. Comparisons with the state-of-the-art approaches should be supplied.

With the previous goals in mind, the Thesis is structured as follows. The first chapter of this document is devoted to introduce the state of the art of the modelling, monitoring and control of batch processes. This introduction is mainly focus on PLS-based methods. Next, Chapter 2 presents the hardware and software tools used in the experiments of the Thesis, along with the data sets treated. Data from four processes of different nature have been used: a polymerization process, a fermentation process, a waste-water treatment process and an etch process.

The second part of the document is devoted to the study of the modelling of batch processes with PLS-based methods. As commented before, this modelling presents several problems. PLS-based models in their traditional form are linear models, whereas batch processes are commonly non-linear. Also, the duration of the processing of a batch may be variable in some processes. This makes necessary the alignment of the process data in order to apply most PLS-based methods. A high number of approaches for the overcoming of some of these problems have been proposed in the literature. In many cases, new approaches are presented just because they work properly for a particular application, e.g. on-line monitoring, and a given number of processes. Nonetheless, a clear understanding of why these approaches perform successfully and which are the advantages and disadvantages in front of the others is seldom supplied. The most important contribution of this Thesis from a theoretical point of view is the unifying analysis of the principal approaches for batch process modelling in Chapter 3. Single model and multi-model approaches are studied from the covariance matrices used for PCA and PLS. From this analysis, conclusions already stated by other authors are confirmed and some new conclusions are withdrawn. Moreover, it leads to conclude that the Multi-phase (MP) structure proposed in this work, in which a number of local PCA or PLS sub-models is used for different parts of the process, is specially appropriate for batch process modelling. The advantage of the MP approach is that the model structure, including the number of sub-models, is flexible. Thus, this structure can be adjusted to the specific dynamic features of the process, instead of using a rigid structure like most of the approaches proposed to date.

The use of the MP structure brings us to the necessity of defining both a performance (or loss) function and a calibration algorithm. The calibration algorithm is in charge of fitting the parameters of the MP model of a specific

process, so that the performance function is optimized. The definition of the performance function is straightforward for a predictive model, such as a PLS model. In that case, the most logical performance function may be defined using the -quadratic- prediction error of the output -quality- variable. Nonetheless, when instead of prediction one is doing compression with PCA, the nature of the performance function is not so clear. Several cross-validatory methods have been proposed to estimate the prediction error in a PCA model, so as to identify the optimum number of Principal Components (PCs). None of these methods are perfect nor generally accepted. It is significative that the most important contributions to that matter date back to the last 70's and the early 80's. In Chapter 4, the limitations of those methods are investigated and a new pair of cross-validation algorithms for PCA are proposed. Moreover, the algorithms are extended to evaluate MP models of a batch process, for which additional parameters apart from the number of PCs have to be set.

Once the performance function is defined, a calibration algorithm has to be designed. This calibration is a problem within the optimization area. Notice some batch processes may present inter-batch dynamics, so that the model may need to be updated after a certain number of batches. Therefore, the calibration algorithm should be as fast as possible. The approach defended in the Thesis is based on a Greedy search, which is a fast optimization strategy at the expense of yielding sub-optimal solutions. This is implemented by the definition of a number of new algorithms, which conform the Multi-phase Analysis Framework proposed in Chapter 5. To calibrate MP models, the MP Analysis Framework provides the flexibility to adjust the model structure to the dynamic nature of the process under study. The apparition of several phases, with dynamics of different order and changes in the correlation structure among variables[b], is effectively identified. This adjustment of the model structure yields performance improvements in several applications. Also, the MP approach provides a set of tools very valued for process understanding and data handling.

The third part of this work is devoted to the study of a number of practical applications of PLS-based models in batch processes. Several problems have been addressed: the off-line and on-line monitoring, the on-line end-quality prediction, the on-line estimation of batch trajectories and the process optimization. All these problems share the common feature that batch processes are difficult to understand and to model. Nonetheless, each of the problems studied presents its particularities, which need to be addressed independently.

In batch processes, a number of quality measurements are usually collected so that quality constraints imposed in the product can be assessed. Some of these measurements have to be obtained from laboratory analysis. This may take a number of hours after a batch has been processed. Moreover, laboratory analysis are commonly invasive and selective, so that only some

[b] See Appendix A for an explanation of what is understood as changing correlation structure.

samples of the batch are assessed and their product have to be discarded. The off-line or end-of-batch monitoring system of a batch process is used to decide whether a batch was produced in-control or out-of-control once its processing has finished. If the monitoring system performs adequately, it can be used to assess the quality of a batch avoiding any delay and destructive analysis. In Chapter 6, the off-line monitoring is studied. A theoretical explanation of the limitations of traditional approaches based on a single model of the process is carried out, supported by some experimental results. In a second part of the chapter, the use of the MP Analysis Framework for the detection of outliers, the calibration of a MP model of the process and the development of the off-line monitoring system is exemplified.

The monitoring system is especially useful if it is able to monitor a batch not only at the end of the processing, but during its processing. This is referred as on-line monitoring. Chapter 7 deals with the on-line monitoring of batch processes, one of the most studied applications in the literature. As in the previous chapter, this one starts introducing the limitations of the most established approaches. Then, the application of the MP Analysis Framework for on-line monitoring is presented and compared with those approaches.

Apart from monitoring, PLS-based models may be used to predict the quality a batch of product is expected to achieve. Thus, a batch of poor quality can be discarded without finishing its processing or some readjustments can be performed on the manipulable variables so as to improve its quality. Additionally, in some processes like waste-water treatment processes or bio-reactors, the trajectory of a variable which is expensive or difficult to measure may be estimated for control and monitoring purposes. Chapter 8 is devoted to the study of estimation and prediction issues. In it, a comparative of several modelling approaches based on PLS is carried out, including the MP Analysis Framework. The principal aim of this study is to understand how dynamics are built in the models.

Since batch processes are widely used for the production of specialty products, the optimization of those processes has an important economic component. Nonetheless, this optimization is a challenging task due to the lack of measurements of principal variables, the uncertainties in the process behavior and its non-linear nature. The application of PLS-based methods to the optimization of batch processes is only in its beginnings. A new optimization algorithm is proposed in Chapter 9 which relies in the theory of extremum seeking in combination with PLS and some statistical based heuristics. The result is a very fast algorithm, specially suited for mega-variate[c] problems. Nonetheless, because an extremum seeking approach has been adopted, the algorithm finds local maxima. Note that even a local maxima solution is a good solution for mega-variate problems, as in the case of batch processes.

Finally, the last part of the Thesis is devoted to draw some general conclusions.

[c] Optimization problems where the search space has thousands of variables.

Summarizing, the main contributions of this work are:

- The theoretical analysis of the principal approaches for modelling batch three-way data with bilinear models. This analysis is performed from the study of the covariance matrices used in the calibration of PCA.
- The proposal of two new cross-validation algorithms for PCA models. The algorithms were compared with some well-known approaches using both simulated and real data, yielding promising results.
- The design of a number of algorithms which conform a general framework based on PCA and PLS: the Multi-Phase framework. This framework is devoted to the analysis, modelling and handling of batch process data. The appropriate use of the tools within the framework allows to investigate the process under study in a computationally efficient way and helps to improve the process understanding.
- The comparison and study of several on-line monitoring approaches based on PCA and Multivariate Statistical Process Monitoring. The effect of the modelling method in the performance of the monitoring charts is analyzed. Also, some suggestions for the development of the charts are offered.
- The comparison and study of several on-line prediction approaches based on PLS. This study allows to improve the understanding of how dynamics are built in the PCA/PLS models of batch data.
- The development of a new algorithm for batch process optimization based on PLS. The performance of this algorithm has been tested in simulation, showing very promising results. The approach is effective and versatile, low sensitive to non-linearities 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.

Other contributions are:

- The design of a cross-validatory algorithm for PCA models from batch process data. This is done in order to identify the convenient arrangement in two-way form of the three-way matrix of data of a batch process. The algorithm proposed can be successfully used to evaluate and compare different arrangements.
- The theoretical demonstration of the distortion of faults in on-line monitoring when the batch-wise unfolded models are used.
- The extension of the formulation of Trimmed Scores Regression to PLS.
- The extension of the adaptive hierarchical approach for batch processes monitoring to PLS.

The following written contributions have arisen as a result of the work performed:

Refereed Journal Papers

[1] J. Camacho and J. Picó. *Monitorización de Procesos por Lotes mediante PCA Multifase*, Revista Iberoamericana de Automática e Informática Industrial, 3(3):78-91 (2006).

[2] J. Camacho and J. Picó. *Multi-Phase Principal Component Analysis for Batch Processes Modelling*, Chemometrics and Intelligent Laboratory Systems, 81(2):127-136 (2006).

[3] J. Camacho and J. Picó. *Online Monitoring of Batch Processes using Multi-Phase Principal Component Analysis*, Journal of Process Control, 10(16):1021-1035 (2006).

[4] J. Camacho, J. Picó and A. Ferrer. *Self-tuning run to run optimization of fed-batch processes using unfold-PLS*, AIChE Journal, 53(7):1789-1804 (2007).

[5] J. Camacho, J. Picó and A. Ferrer. *Bilinear modelling of batch processes. Part I: Theoretical discussion*, Submitted to Journal of Chemometrics (2007).

[6] J. Camacho, J. Picó and A. Ferrer. *Bilinear modelling of batch processes. Part II: PLS comparative*, Submitted to Journal of Chemometrics (2007).

[7] J. Camacho, J. Picó and A. Ferrer. *Multi-Phase Analysis Framework for Handling Batch Process Data*, Submitted to Journal of Chemometrics (2007).

[8] J. Camacho, J. Picó and A. Ferrer. *Leave-n-Samples-Out Cross-validation in PCA for Missing Data Imputation and Measurement Noise Reduction*, In elaboration for Chemometrics and Intelligent Laboratory Systems (2007).

Conference presentations and posters

[9] J. Camacho, J. Picó and A. Ferrer. *A new look at the dynamic covariance structure of various approaches for batch process modelling*, 10th Scandinavian Symposium on Chemometrics (2007).

[10] J. Camacho, J. Picó and A. Ferrer. *New Cross-Validation Methods in Principal Component Analysis*, 10th Scandinavian Symposium on Chemometrics (2007).

[11] J. Camacho, J. Picó and A. Ferrer. *A new algorithm for selecting the unfolding method and the number of sub-models in batch process modelling with PCA*, 10th Scandinavian Symposium on Chemometrics (2007).

[12] J. Camacho, J. Picó and A. Ferrer. *Multi-Phase Analysis Framework for Handling Batch Process Data*, 10th Scandinavian Symposium on Chemometrics (2007).

This Thesis has been distinguished with the second Rosina Ribalta Award to the best Doctoral Thesis Project in the field of the Information and Communication Technologies, in its IX edition, awarded by Institute of Technoethics of the EPSON Foundation.

Part I

Introduction.

1 State of the art

1.1 Introduction to batch processing

Batch processes play an important role in the pharmaceutical, biochemical, food, etch and metal industry, among others. They are processes which mainly consist of the continuous repetition of three steps: charge of the vessel, processing during a finite period of time and discharge [13]. Raw materials and process variables follow a specific recipe in order to obtain within specification product, maximizing the use of the machinery [14]. For that purpose, initial and processing conditions have to be controlled.

To assess the quality of the product, a number of quality measurements are usually collected. Depending on the availability and economical cost of sensors for those measurements, either their evolution is available during the batch processing, they are only available right after a batch has been finished or they have to be obtained subsequently from laboratory analysis. In the last case, the quality measurements may not be available before starting the processing of the following batch.

The literature has paid less attention to batch processes than to continuous processes. The techniques used for monitoring and controlling continuous processes are not directly applicable to batch processes [15]. In continuous processes, one tries to maintain an operation point or stable state. In batch processes, the good performance is yielded under Normal Operation Conditions (NOC), so that process variables describe a specific trajectory during the processing. The modelling of the NOC is complex due to the non-linear and varying dynamics present and commonly the uneven duration of the batches, among other problems [16]. This challenging modelling, in combination with the no availability and delay of quality measurements, makes monitoring, controlling and optimizing very hard. In Figure 1.1, the trajectories of 3 process variables corresponding to 50 batches are presented together with the final quality measurements.

In spite of the inherent complexity in batch processing, it presents a number of advantages over continuous processing [14]:

– The processing time of a batch can be adjusted to meet the quality specifications.

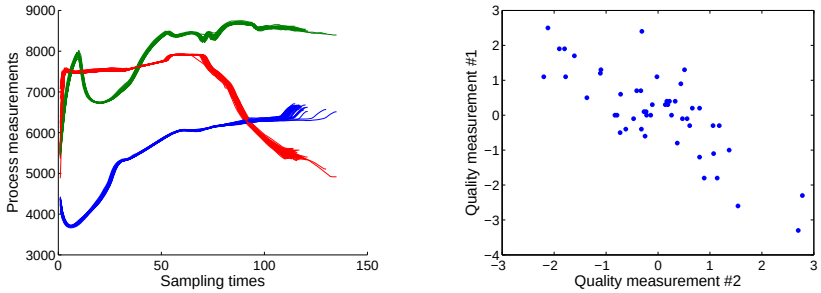


Fig. 1.1. Process variables trajectories and final quality values of 50 batches of a Nylon 6'6 polymerization process.

- The repetitive nature of batch processes allows to improve the performance on a batch to batch basis.
- Many batch processes tend to be slow processes, which makes possible the optimization in real time.
- The relationship between controllability and economical profit is more directly established in batch processing than in continuous processing, since part of the optimization goals in the former -e.g. reduce the batch duration, reduce the out-of-specification products, etc.- have an explicit economical component.

This chapter describes the state of the art of the modelling, monitoring and control of batch processes. Section 1.2 presents a classification of the modelling approaches for batch processes. Section 1.3 is focused on the multivariate statistical modelling based on the projection to latent structures. Section 1.4 briefly describes the monitoring of batch processes. Section 1.5 introduces the different approaches for control and optimization.

1.2 Process models

The two modelling paradigms which divide the control theory [17] can be also found in the field of the batch processes. On the one hand, the models inherited from the field of the classic mechanics or white-box models are developed from the knowledge of the process. After an -quite often hard- investigation period, a number of equations or rules which capture the behavior of the process are extracted. On the other hand, the models inherited from the electronic engineering or black-box models are calibrated solely from input-out data.

With a similar idea, in [18], the following classification is established:

- Knowledge-based white-box models.

– Data-based black-box models.
– Hybrid grey-box models.

where the third category is a hybrid of the preceding two. The organization of the present section is based on this classification.

1.2.1 Knowledge-based white-box models

In this category both the models obtained from physical-chemical laws -first principles or fundamental models- and those obtained from the expert knowledge of a process -expert systems- are included. The development of these models is a task which, depending on the process, may be challenging and time consuming. Nonetheless, their design is an important research line [19] and there are available models for many industrial processes.

Although a model of this sort may cover a wide range of the process operation [18] it only models the known part of it [20]. The reliance on the process knowledge has the drawback that some of the assumed dynamic laws may not be valid in all cases, i.e. for all process states. Additionally, unknown side reactions may exist. The approaches based on knowledge-based models may be endowed with some degree of adaptation capability and robustness with respect to changes in the dynamics. Nonetheless, if their assumptions (e.g., assuming Monod or Haldane kinetics) are not fulfilled their performance may degrade beyond acceptable limits. In the particular case of batch processes, the development of models solely from fundamental or expert knowledge is highly challenging and the model uncertainties are specially dramatic. This causes the apparition of several hybrid approaches, as discussed later.

Fundamental models are also commonly used in the literature for the development of process simulators, where control, monitoring or optimization approaches can be validated without the necessity of using the real process.

1.2.2 Data-based black-box models

The automation of industry together with the reduction of prices of the sensor and data acquisition technology have caused the apparition of huge historical data-bases. Black-box models make the most of these data-bases. The input-output data are used to calibrate the parameters of the model which can be used afterwards for different purposes. Nonetheless, black-box models not only requires a large quantity of data, but also that the data are rich in information regarding the process. This makes that, in many cases, historical data-bases alone are not enough for the calibration of valid models [18] and a design of experiments is necessary.

The historical data-bases are composed of large quantities of highly correlated data, where the signal-noise ratio is low [20]. This makes the information cannot be explicitly found in the data, and methods for extracting this information are necessary [21]. Nonetheless, batch data present a number

of features which make this extraction hard [16] and which may be relaxed using preprocessing methods:

– Nonlinear dynamics and time-varying dynamics.
– Uneven batch length.
– Presence of noise, collinear data and outliers.
– Variables of different magnitude and variance.

In this context, Multivariate Statistical Analysis Methods based on the Projection to Latent Structures (PLS-Based methods) are frequently used for the generation of empirical models. One of the benefits of using these methods is that the process understanding is improved [22]. Therefore, although data-based, these methods may not be catalogued as black-box, because useful information is gained from the modelling. Other empirical methods are the Artificial Neural Networks (ANN) [15, 23] or Hybrids of different data-based techniques [24, 25, 26].

1.2.3 Hybrid Models

As commented before, the use of knowledge-based models has the drawback that the unknown part of the process is not modelled. Hybrid models are based on the idea of using knowledge-based models together with data-driven models. The latter are used to capture the unknown process-model mismatch.

The Hybrid Neural Network presented by Ng and Hussain [27] uses ANNs to model some internal parameters, which are difficult to measure. The output of the ANNs are used as the input of the first principles models where the rest of the process behavior is modelled. Authors also propose an adaption mechanism for their model. Van Sprang et *al.* [28] investigate the monitoring of batch processes with grey Tucker-1 and Tucker-3 models. A similar idea can be found in [29], where the aim is the control of the process. In this case, unfold-Partial Least Squares (uPLS) is used to model the residuals of a first principles model. In all this approaches, the shortcomings of using knowledge-based models alone are highlighted and the use of hybrid models carefully designed is defended.

1.3 Multivariate statistical modelling of batch processes

In a data set formed by a high quantity of very correlated variables, the real dimension of the space in which the data vary is much lower [20]. The aim of PLS-based methods is to find the underlying structures which are hidden in the data. These structures can be used as a model, reducing the volume of data and, in theory, separating information from noise. This reduction makes possible the effective use of the information contained in large historical data bases for the monitoring and control of processes.

The data set collected from a batch process is three-way: a set of variables are measured at different sampling times during the processing of a batch, and this is repeated for a number of batches. The knowledge about the process may determine which variables should be measured and the sampling frequency.

The duration of the processing of a batch may be variable in some processes. Take for instance the case when the end time depends on the amount of product produced or a quality constraint which has to be met. Commonly in these cases, data have to be properly aligned -equalized- to apply analysis and modelling methods based on establishing patterns of the batch evolution. Several alignment methods can be found in the literature [13, 30, 31, 32, 33, 34]. After the alignment, data matrix $\underline{\mathbf{X}}(I \times J \times K)$ contains the values of J process variables at K sampling times in I batches.

Optionally, a number of initial conditions and quality measurements may be collected in a process. The L initial conditions measured for the I batches are arranged in matrix $\mathbf{Z}(I \times L)$. The M quality variables, collected at K_y sampling times, are stored in matrix $\underline{\mathbf{Y}}(I \times M \times K_y)$. Commonly, the quality variables are only measured at the end of the batch. Therefore, they may be arranged in a two-way matrix $\mathbf{Y}(I \times M)$.

In these definitions some simplifications are assumed. First, the same process and quality variables are assumed to be measured throughout the batch duration. Also, the sampling frequency of different variables in $\underline{\mathbf{X}}$ or $\underline{\mathbf{Y}}$ is supposed to be equal. Nonetheless, if these assumptions are not met, data can always be interpolated, divided or rearranged so that the data matrices can be defined as it has been done here.

To analyze the three-way matrix of data of a batch process with PLS-based methods, two choices are available. First, the data can be rearranged in two-way form for the application of bilinear models, such as Principal Component Analysis (PCA) and Partial Least Squares (PLS). The second choice is to apply directly three-way methods [35], like Parallel Factor Analysis (PARAFAC) [36], Tucker-3 [37] or N-PLS [38].

1.3.1 Two-way Models

First, both PCA and PLS are introduced. Then, their application to batch data is addressed.

Principal Components Analysis

PCA is defined for the analysis of two-way matrices of the form $\mathbf{X}(I \times J)$, composed of I objects with the observations of J variables. It finds the subspace of a certain dimension A (parameter of the calibration algorithm) which contains most variability of the data. For that, PCA extracts the orthogonal

directions of maximum variance in the variables space, the so-called Principal Components (PCs), which are linear combinations of the variables. PCA follows the equation (1.1)

$$\mathbf{X} = \mathbf{T} \cdot \mathbf{P}^T + \mathbf{E} \tag{1.1}$$

where $\mathbf{X}$ is the $I \times J$ data matrix, $\mathbf{P}$ is the $J \times A$ loadings matrix, with A the number of PCs of the model, $\mathbf{T}$ is the $I \times A$ scores matrix and $\mathbf{E}$ is the $I \times J$ residuals matrix.

The direction of the PCs are obtained from the eigenvectors of $\mathbf{X}^T \cdot \mathbf{X}$. The eigenvectors are arranged as the columns of $\mathbf{P}$, ordered by explained variance. The rows of $\mathbf{T}$ are the projections of the data to the latent subspace of the PCs. PCA reduces the original dimension of the data J to the number of PCs A. When no dimension reduction is done, PCA is just a rotation of the axes. If only the first PCs are included in the model, the variance explained by the residuals is treated as noise. The Non-linear Iterative Partial Least Squares (NIPALS) algorithm [39, 40] allows to compute the PCs in order, so that the discarded directions does not have to be computed. In (1.2), the projection (score) of a new observation in the PCA subspace is computed:

$$\mathbf{t}_{new} = \mathbf{x}_{new} \cdot \mathbf{P} \tag{1.2}$$

where $\mathbf{x}_{new}$ is a $1 \times J$ vector representing a new object and $\mathbf{t}_{new}$ is a $1 \times A$ vector representing its projection to the latent subspace.

For the estimation of the convenient A, a common practise is to use cross-validation (see Chapter 4). For a deeper explanation of the PCA method refer to [41].

Partial Least Squares

The linear regression problem is defined by the following expression:

$$\mathbf{Y} = \mathbf{X} \cdot \mathbf{B} + \mathbf{F} \tag{1.3}$$

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 measurements 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 of the model.

Ordinary Least Squares (OLS) performs a solution to the linear regression problem which minimizes the quadratic error. The least squares solution for (1.3) is:

$$\hat{\mathbf{B}}_{LS} = (\mathbf{X}^T \cdot \mathbf{X})^{-1} \cdot \mathbf{X}^T \cdot \mathbf{Y} \tag{1.4}$$

The inversion of the matrix $\mathbf{X}^T \cdot \mathbf{X}$ requires it to be non-singular. Moreover, the OLS solution is highly unstable, leading to a poor prediction performance when this matrix is ill-conditioned. The good conditioning of such

a matrix is dependent 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 $\mathbf{X}$. In this case, the subspace of interest is that 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:

$$\begin{aligned} \mathbf{X} &= \mathbf{T} \cdot \mathbf{P}^T + \mathbf{E} \\ \mathbf{Y} &= \mathbf{T} \cdot \mathbf{Q}^T + \mathbf{F} \end{aligned} \tag{1.5}$$

where $\mathbf{T}$ is an $I \times A$ matrix which contains the projections of $\mathbf{X}$ to the latent A-dimensional subspace, $\mathbf{P}$ and $\mathbf{Q}$ are the $J \times A$ and $M \times A$ regressor matrices, also called loadings matrices, and $\mathbf{E}$ and $\mathbf{F}$ are the $I \times J$ and $I \times M$ matrices of residuals of $\mathbf{X}$ and $\mathbf{Y}$, 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 (1.5) can be rearranged in the following form:

$$\mathbf{Y} = \mathbf{X} \cdot \hat{\mathbf{B}}_{PLS} + \mathbf{F} \tag{1.6}$$

with:

$$\hat{\mathbf{B}}_{PLS} = \mathbf{W} \cdot (\mathbf{P}^T \cdot \mathbf{W})^{-1} \cdot \mathbf{Q}^T \tag{1.7}$$

where $\mathbf{W}$ is a $J \times A$ matrix of weights. Thus, a PLS model is represented by matrices $\mathbf{P}$, $\mathbf{W}$ and $\mathbf{Q}$. In (1.8), the estimation of a new observation with a PLS model is computed:

$$\hat{\mathbf{y}}_{new} = \mathbf{x}_{new} \cdot \hat{\mathbf{B}}_{PLS} \tag{1.8}$$

where $\mathbf{x}_{new}$ is a $1 \times J$ vector representing a new object and $\hat{\mathbf{y}}_{new}$ is a $1 \times M$ vector representing the estimation of the output variables.

A common way of obtaining a PLS model is a variant of the NIPALS algorithm. The number of dimensions of the latent subspace can be estimated by cross-validation. For a deeper explanation of the PLS method refer to [40] and [42].

There are several regression methods, apart from OLS and PLS, that are not presented in this document. Among them, Principal Components Regression (PCR) is based on regressing $\mathbf{Y}$ onto the scores of a PCA model of $\mathbf{X}$ (1.1), and Ridge Regression [43] is based on adding a multiple of the identity matrix to the matrix $\mathbf{X}^T \cdot \mathbf{X}$ before inverting it. Both PCR and Ridge Regression try to overcome the inversion problem in ill-conditioned matrices.

Application of bilinear models to batch data

Since batch data are three-way, the data matrix $\underline{\mathbf{X}}$ has to be conveniently rearranged in a number of two-way matrices to apply bilinear models as PCA and PLS. There are at least three methods to do this. i) The traditional option is to unfold the three-way matrix of data in a single two-way matrix [44, 45]. It has been stated that only the unfolding in the batches direction -batch-wise unfolding, see Figure 1.2(a)- and in the variables direction -variable-wise unfolding, see Figure 1.2(b)- have interest [46]. Nonetheless, the batch dynamic unfolding [47] may be an alternative choice. This method is equivalent to the variable-wise unfolding with the inclusion of lagged measurement-vectors (LMVs) as additional variables [3] and can be seen as a generalization of the other unfolding procedures: if no LMV is added, the resulting matrix is the same as the one after variable-wise unfolding; if all possible LMVs are added, the resulting matrix is the same as the one after batch-wise unfolding. Also intermediate cases are possible, for instance in Figure 1.2(c) the batch dynamic unfolding with 1 LMV is shown[a]. ii) The second option is to use an adaptive approach where current and past information are combined, e.g. by using hierarchical models [48] (not shown). iii) The third option is the use of K local models [49], each one modelling the information corresponding to a sampling time alone -Figure 1.2(d).

Both the unfolding and splitting in K models -options i) and iii)- can be combined in approaches such as the evolving modelling [50, 49]. In this approach, each model corresponding to a sampling time is fitted with the data collected from the beginning of the processing to that sampling time. For that, data are batch-wise unfolded. Also the Moving Window approach [51] can be understood as a combination of unfolding and splitting. In this case, for each of the K models, only part of the immediate past information is combined with the current sampling time data.

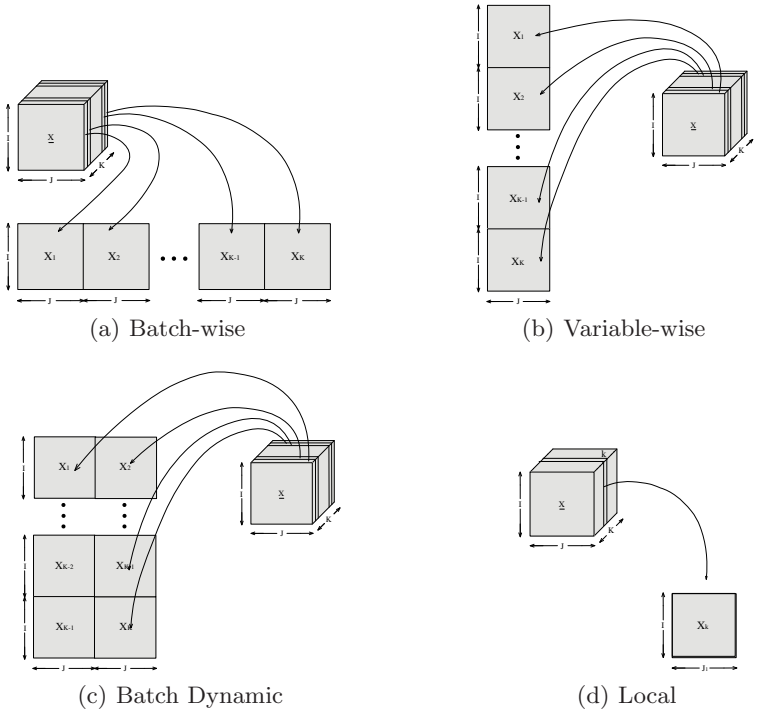
Another approach is the multi-stage, multi-phase modelling, based on the calibration of independent models for different intervals of a batch process. The data set collected from a process is split in intervals according to the sampling time mode. In each interval, data are unfolded to fit a bilinear model. Because of the use of intervals instead of single sampling times to generate the sub-models[b], the number of sub-models is lower than K. This makes easier the interpretation and handling of the models.

To fit multi-stage, multi-phase models, the intervals have to be somehow determined. This has been done from different points of view:

– Expert knowledge. Ündey et *al.* [16, 52] suggested to use several models to represent the multi-stage nature of the processes for end-of-batch and

[a] Both the notions of unfolding methods -or directions- and number of LMVs will be used indistinctly throughout the document.

[b] From here on, the term "sub-models" will be used to design the model of a single or several intervals of a process.



(a) Batch-wise (b) Variable-wise

(c) Batch Dynamic (d) Local

Fig. 1.2. Different arrangements of three-way data in two-way form.

on-line monitoring. They divide the process according to the different processing units and the distinguishable operations inside a unit.

– Process analysis. In [22], the authors propose to divide the model in two based on changes in the variance explained by the principal components during the batch. Lennox et *al.* [53] suggest the use of multiple models to account for changes in the correlation structure when using Squared Prediction Error (SPE) charts.
– Automatic recognition. Sub-PCA [54, 33] was proposed for the on-line monitoring of batch processes and uses a clustering algorithm to detect segments of the batch which can be properly modelled by a single model.

The intervals of the first category are knowledge-based, whereas the other two -i.e., both using process analysis and automatic recognition- are data-based. Knowledge-based and data-based intervals do not necessary have to coincide [54]. Following the definition in [2], from here on knowledge-based intervals are named "stages" whereas data-based ones are called "phases".

1.3.2 Three-way Models

Parallel Factor Analysis

The modelling method named Parallel Factor Analysis (PARAFAC) [55, 56] follows next expression:

$$x_{ijk} = \sum_{a=1}^{A} t_{ia} p_{ja}^{v} p_{ka}^{t} + e_{ijk} \tag{1.9}$$

where x_{ijk} is the value in the matrix of data $\underline{\mathbf{X}}(I \times J \times K)$ for batch i, variable j and sampling time k, A is the number of components, i.e. the dimension of the sub-space captured by the model, $t_{ia} \epsilon \mathbf{T}(I \times A)$ is the score for batch i in the component a, $p_{ja}^{v} \epsilon \mathbf{P^v}(J \times A)$ is the loading of variable j in component a, $p_{ka}^{t} \epsilon \mathbf{P^t}(K \times A)$ is the loading of sampling time k in component a and $e_{ijk} \epsilon \underline{\mathbf{E}}(I \times J \times K)$ is the residual corresponding to x_{ijk}.

Imagine $\mathbf{x}_{new}(1 \times JK)$ contains the batch-wise unfolded data collected for a new batch $\mathbf{X}_{new}(J \times K)$. Then, the projection (score) of this new batch in the PARAFAC subspace is computed following:

$$\mathbf{t}_{new} = (\mathbf{P^t} * \mathbf{P^v})^{+} \cdot \mathbf{x}_{new} \tag{1.10}$$

where $*$ stands for the Khatri-Rao product and superscript '+' indicates the Moore-Penrose inverse [35].

The PARAFAC modelling method is based on the principle of Parallel proportional profiles developed by Cattell [57]. Under certain conditions, the PARAFAC model is rotational unique, contrarily to what happens with PCA which is rotational free. If data are trilinear, the unique solution in PARAFAC may be valuable for data interpretation. The principal drawback of PARAFAC is the appearance of degenerate solutions, when the model is inappropriate.

To fit PARAFAC models, the Alternating Least Squares (ALS) algorithm can be used. Since PARAFAC models are computational demanding, alternative methods to the cross-validation may be used for determining the number of components, such as the split-half method or the core consistency [58]. For a more detailed information regarding the PARAFAC method, please refer to [36, 35].

Tucker-3

The Tucker-3 model was originally proposed by Ledyard R. Tucker [37]. The principal justification beneath Tucker models is that factors of different components may interact. Take for instance the case of the Singular Value Decomposition (SVD) of two-way matrices:

$$\mathbf{X} = \mathbf{U} \cdot \mathbf{S} \cdot \mathbf{V}^T + \mathbf{E} \tag{1.11}$$

where $\mathbf{U} \cdot \mathbf{S} = \mathbf{T}$ and $\mathbf{V} = \mathbf{P}$ in (1.1). $\mathbf{S}$ is a diagonal matrix containing the singular values of X in decreasing order.

Now, instead of matrix $\mathbf{S}$, let us define the core matrix $\mathbf{G}$ which does not have to be diagonal, allowing interactions between factors in different components. Extending this idea to the three-way case yields the model:

$$x_{ijk} = \sum_{a=1}^{A} \sum_{a_2=1}^{A_2} \sum_{a_3=1}^{A_3} t_{ia} p^v_{ja_2} p^t_{ka_3} g_{aa_2a_3} + e_{ijk} \tag{1.12}$$

where A, A_2 and A_3 are the number of components for the three modes and $g_{aa_2a_3} \epsilon \underline{\mathbf{G}}(A \times A_2 \times A_3)$ is the loading associated to the interaction between components a, a_2 and a_3.

The principal differences of the Tucker-3 model in comparison to the PARAFAC model are [59]:

- The number of components can vary in the three modes.
- $\underline{\mathbf{G}}$ is not super-diagonal.
- Tucker models present rotational freedom.

The Tucker-3 model performs a compression of the three modes of the data matrix. If only two of the modes are to be compressed, the model is named Tucker-2 [35]. Following the same idea, Tucker-1 models only compress one of the modes. Batch-wise unfolded PCA models can be seen as special cases of Tucker1 models.

Let imagine $\underline{\mathbf{X}}$ contains data from a batch process. The scores of a new batch are then computed as follows ([28]):

$$\mathbf{V} = (\mathbf{G} \cdot (\mathbf{P^t} \otimes \mathbf{P^v})^T)^T \tag{1.13}$$

$$\mathbf{t}_{new} = \mathbf{x}_{new} \cdot \mathbf{V} \cdot (\mathbf{V}^T \cdot \mathbf{V})^{-1} \tag{1.14}$$

where $\otimes$ stands for the Kronecker product [35] and G is the matricized core matrix.

To fit Tucker models, the Alternating Least Squares (ALS) algorithm can be used. Cross-validation is a valid way for determining the number of components. For a more detailed information regarding Tucker models, please refer to [35].

Three-way Regression Models

Several extensions of the two-way regression methods have been proposed for the case when the matrix of data $\underline{\mathbf{X}}$ available to predict Y is three-way. Similar than in PCR, a PARAFAC, Tucker1, Tucker2 or Tucker3 model can be fitted from matrix $\underline{\mathbf{X}}$ and then Y can be regressed on the scores matrix T. Alternatively, Rasmus Bro [38] proposed the N-PLS algorithm, extending the PLS algorithm to the N-way case.

1.3.3 Preprocessing

In most of the cases, at least in the applications studied in this Thesis, the PLS-based methods are applied to centered and scaled data. The data are centered so that the analysis is focused on the variability around the average. Data are scaled to homogenize the importance of the variables in the models. For the sake of easy understanding of the strategies and algorithms, these preprocessing operations are not included in most of the equations presented in this work. Unless otherwise stated, it will be assumed that the data in the equations have been properly preprocessed. Nonetheless, for some parts of the document, the explicit treatment of the preprocessing information is mandatory. That is the case of the cross-validation study in Chapter 4 and of the optimization approach presented in Chapter 9. Therefore, it is useful to understand how the preprocessing information is included in the equations of the models.

The PCA model equation (1.1) with the preprocessing information yields:

$$(\mathbf{X} - \mathbf{1} \cdot \mu^T) \odot (\mathbf{1} \cdot \varsigma^T) = \mathbf{T} \cdot \mathbf{P}^T + \mathbf{E} \tag{1.15}$$

where $\mathbf{1}$ is a $I \times 1$ vector of ones, μ is the $J \times 1$ vector containing the averages of the variables, ς is the $J \times 1$ vector containing the weight applied to the variables and $\odot$ stands for the Hadamard (element to element) product. By rearranging the equation, so that the complete model composed of preprocessing and PCA information lies at the right side of the equation, it yields:

$$\mathbf{X} = \mathbf{1} \cdot \mu^T + (\mathbf{T} \cdot \mathbf{P}^T + \mathbf{E}) \oslash (\mathbf{1} \cdot \varsigma^T) \tag{1.16}$$

where $\oslash$ is understood as the Hadamard division. This expression can be divided in modelled and residual parts:

$$\mathbf{X} = (\mathbf{1} \cdot \mu^T + (\mathbf{T} \cdot \mathbf{P}^T) \oslash (\mathbf{1} \cdot \varsigma^T)) + \mathbf{E}_\varsigma \tag{1.17}$$

The score of a new observation is obtained from:

$$\mathbf{t}_{new} = (\varsigma^T \odot (\mathbf{x}_{new} - \mu^T)) \cdot \mathbf{P} \tag{1.18}$$

Similarly, the preprocessing information can be added to the expressions of the PARAFAC and Tucker3 models. E.g. in PARAFAC:

$$x_{ijk} = (\mu_{jk} + \frac{1}{\varsigma_{jk}} \cdot \sum_{a=1}^{A} t_{ia} p^v_{ja} p^t_{ka}) + e_{\varsigma,ijk} \tag{1.19}$$

An interesting discussion regarding proper centering and scaling of two-way and multi-way data is presented in [60].

The inclusion of the preprocessing information in PLS (1.6) yields:

$$(\mathbf{Y} - \mathbf{1} \cdot \mu_{\mathbf{Y}}^T) \odot (\mathbf{1} \cdot \varsigma_{\mathbf{Y}}^T) = ((\mathbf{X} - \mathbf{1} \cdot \mu_{\mathbf{X}}^T) \odot (\mathbf{1} \cdot \varsigma_{\mathbf{X}}^T)) \cdot \hat{\mathbf{B}}_{PLS} + \mathbf{F} \quad (1.20)$$

with $\mu_{\mathbf{Y}}$ and $\varsigma_{\mathbf{Y}}$ the $M \times 1$ vectors containing the averages and weights of the variables in $\mathbf{Y}$ and $\mu_{\mathbf{X}}$ and $\varsigma_{\mathbf{X}}$ the $J \times 1$ vectors containing the averages and weights of the variables in $\mathbf{X}$. Rearranging the equation:

$$\mathbf{Y} = (\mathbf{1} \cdot \mu_{\mathbf{Y}}^T + (((\mathbf{X} - \mathbf{1} \cdot \mu_{\mathbf{X}}^T) \odot (\mathbf{1} \cdot \varsigma_{\mathbf{X}}^T)) \cdot \hat{\mathbf{B}}_{PLS}) \oslash (\mathbf{1} \cdot \varsigma_{\mathbf{Y}}^T)) + \mathbf{F}_{\varsigma_{\mathbf{Y}}} \quad (1.21)$$

1.3.4 Extensions

Several extensions of the modelling methods introduced have been proposed, motivated by certain features of the industrial processes or the data sets. Among them, the nonlinear, multi-block and adaptive approaches are explained here:

The PLS-based methods, in their traditional form, are based on the identification of components which are linear combination of variables. Nonetheless, in the real world most relationships are -to a greater or lesser extent- nonlinear. The aim of nonlinear PLS-based methods is to identify nonlinear combinations of variables for compression [61] or regression [25].

In some data sets, the variables intuitively arrange in different blocks. This conceptual structure in the data may be inherited by the model, in order to enhance its interpretability. This is the aim of the multi-block approaches revised in [62, 63]. Multi-block models follow a hierarchical architecture, with a number of low-level models, modelling intra-block relationships, and a super model capturing inter-block relationships. Both the low-level models and the super model are fitted jointly in the same procedure. Because of that, it should be noted that the low-level models are not local to the block, in the sense that they are not fitted independently from the rest of data out of the block.

The dynamics inherent in continuous processes and the inter-batch dynamics of batch processes degrade the model performance. After a certain period of use, a model has to be re-calibrated or updated using the incoming data. The aim of adaptive approaches is to avoid the re-calibration or simply to give more importance to the data collected recently. In the literature, the updating of the models is commonly based on a recursive algorithm [61, 64, 65] or on a multi-block structure [48, 66].

1.4 Monitoring

1.4.1 Off-line Monitoring (Post Analysis)

The performance of the monitoring system in an industrial process is very important from an economical point of view. An accurate monitoring system

saves time in the detection of production problems [20, 67] and so, it saves money.

For the monitoring of industrial processes, one of the most used methodologies and the one studied in this work is the Multivariate Statistical Process Monitoring (MSPM) based on PLS-based models. A PLS-based model -commonly a PCA model- is calibrated in order to develop a number of monitoring charts. It is customary to monitor both the scores and the residuals in a complementary pair of charts: the Q-statistic, which compress the residuals; and the D-statistic or Hotelling's T2 statistic [68], computed from the scores. With the statistics computed from the calibration data, control limits at a certain confidence level can be established in the charts [69, 70, 71, 13]. Afterwards, new data are monitored using these limits. Thus, abnormalities are detected when the limits are exceeded. Also, making the most of the nature of PLS-based models, the contribution of the variables to an abnormality signaled can be investigated with the contribution plots [72, 73].

In batch processes, a number of quality measurements are usually collected so that quality constraints imposed in the product can be assessed. Some of these measurements have to be obtained from laboratory analysis. Therefore, the quality measurements of a batch may not be available before starting the processing of the following batch. In such cases, an abnormality may affect the product of several batches before being detected. Notice abnormal product sometimes has to be discarded. Moreover, laboratory analysis are commonly invasive and selective, so that only some samples of the batch are assessed and then discarded. The off-line or end-of-batch monitoring of a batch process is used to decide whether a batch was produced in-control or out-of-control once its processing has finished. If the monitoring system performs adequately, it can be used to assess the quality of a batch avoiding any delay and destructive analysis.

Commonly, for the off-line monitoring of batch processes, the monitoring charts are computed from a PCA model fitted from batch-wise unfolded data [31, 45]. Both the D-statistic and the Q-statistic for batch i can be computed from the following equations:

$$D_i = \sum_{a=1}^{A} \left(\frac{t_{ai} - \mu_{\mathbf{t}_a}}{\sigma_{\mathbf{t}_a}}\right)^2 \tag{1.22}$$

$$Q_i = \sum_{j=1}^{J} \sum_{k=1}^{K} (e_{ijk})^2 \tag{1.23}$$

where t_{ai} represents the score of the batch in the a-th component, $\mu_{\mathbf{t}_a}$ and $\sigma_{\mathbf{t}_a}$ stand for the mean and the standard deviation of the scores of that component in the calibration data, respectively, and e_{ijk} represents the residual value corresponding to the variable j and sampling time k.

The scores are linear combinations of the original variables and so, according to the Central Limit Theorem, they are supposed to be approxi-

mately Normal distributed [13]. Also the residuals can be assumed to follow a multi-normal distribution. Under the assumption that the scores follow a multivariate normal distribution, it holds [15] that for new incoming data, the D-statistic (times a constant) follows an F distribution of A and $I - A$ degrees of freedom:

$$D \sim \frac{A \cdot (I^2 - 1)}{I \cdot (I - A)} F_{A,(I-A)} \tag{1.24}$$

Therefore, the corresponding Upper Control Limit (UCL) for the D-statistic at significance level (type I risk) α is given by:

$$UCL(D)_\alpha = \frac{A \cdot (I^2 - 1)}{I \cdot (I - A)} F_{(A,(I-A)),\alpha} \tag{1.25}$$

Regarding the UCL for the Q-statistic, several procedures can be used. Jackson and Mudholkar [70] showed that an approximate critical value at significance level α is given by:

$$UCL(Q)_\alpha = \theta_1 \cdot \left[\frac{z_\alpha \sqrt{2\theta_2 h_0^2}}{\theta_1} + 1 + \frac{\theta_2 h_0 (h_0 - 1)}{\theta_1^2} \right]^{\frac{1}{h_0}} \tag{1.26}$$

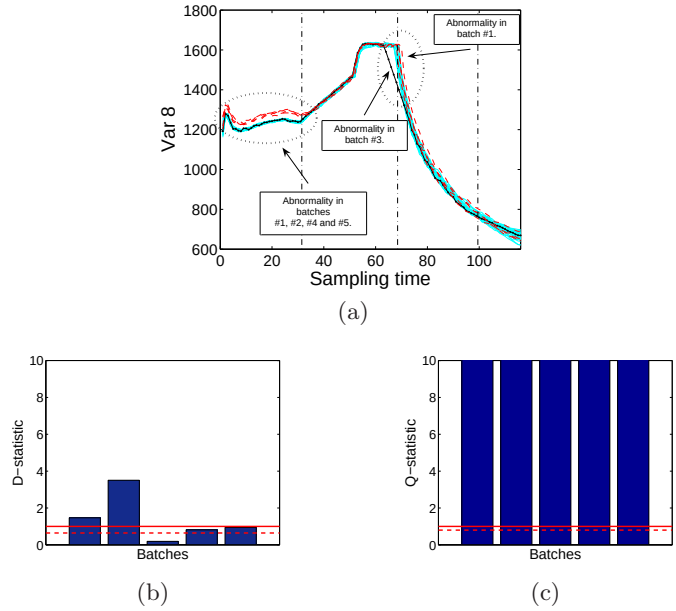
where $\theta_n = \sum_{a=A+1}^{rank(\mathbf{X})} (\lambda_a)^n$, with $rank(\mathbf{X})$ the rank of the batch-wise unfolded matrix of process data $\mathbf{X}$, $h_0 = 1 - \frac{2\theta_1 \theta_3}{3\theta_2^2}$, λ_a are the eigenvalues of matrix $\frac{1}{I-1} \cdot \mathbf{E}^T \cdot \mathbf{E}$, where $\mathbf{E}$ is the matrix of residuals, and z_α is the $100 \cdot (1 - \alpha)\%$ standardized normal percentile. Alternatively, one can use an approximation based on the weighted chi-squared distribution proposed by Box [69]. For more details on the development of control limits for the monitoring charts, refer to [74].

An example of D-statistic and Q-statistic monitoring charts, where five abnormal batches are shown, is presented in Figure 1.3. The abnormalities of the batches are highlighted in the trajectory of one of the variables shown in Figure 1.3(a).

1.4.2 On-line Monitoring

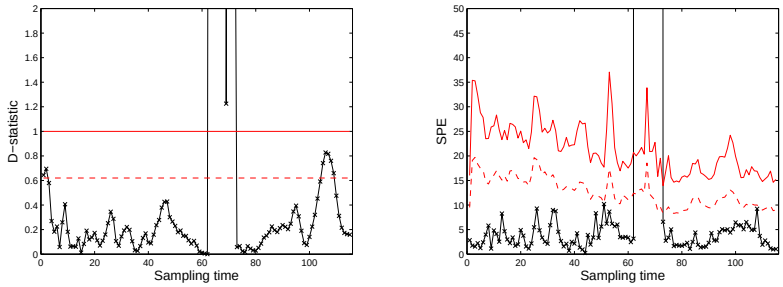
The monitoring system is especially useful if it is able to monitor the batch not only at the end of the processing, but during the processing. The on-line monitoring of batch processes using PLS-based models commonly relies in two control charts built from the D-statistic and the SPE [13]:

$$D_{i,k} = \sum_{a=1}^{A} \left(\frac{t_{aik} - \mu_{\mathbf{t}_{ak}}}{\sigma_{\mathbf{t}_{ak}}} \right)^2 \tag{1.27}$$



(a) (b) (c)

Fig. 1.3. Nylon 6'6 Polymerization process. Batch trajectories of variable 8 (a) D-statistic chart (b) and Q-statistic chart (c). Off-line monitoring.



(a) (b)

Fig. 1.4. D-statistic chart (a) and SPE chart (b) examples computed from the data of the Nylon 6'6 polymerization process. The batch plotted is the third abnormal batch of Figure 1.3. On-line monitoring.

$$SPE_{i,k} = \sum_{j=1}^{J} (e_{ijk})^2 \tag{1.28}$$

Once again, the D-statistic is assumed to be F-distributed [71] and the limits of the SPE are computed following the approach of Jackson and Mud-

holkar [70]. Thus, control limits can be computed according to (1.25) and (1.26), respectively. To achieve an adequate performance of the monitoring charts, it is highly recommended to readjust the control limits using the calibration data in a leave-one-out basis [3, 75]. Limits are raised or lowered so that the Overall Type I (OTI) risk equals the imposed significance level (ISL). Following the definition in [13], the OTI is the percentage of faults in the NOC calibration batches:

$$OTI = 100 \cdot \frac{nf}{I \cdot K}\% \tag{1.29}$$

where nf is the total number of faults -i.e., single sampling times where the statistic computed for a batch crosses the limit- in the NOC calibration data.

An example of D-statistic and SPE monitoring charts is presented in Figure 1.4. The batch being monitored presented an abnormal behavior between sampling times 63 and 73.

Several proposals for the on-line monitoring of batch processes can be found in the literature. The online monitoring procedure of Nomikos and MacGregor [13] is based on the batch-wise unfolding and thus the data of a complete batch are needed to compute t_{aik} in (7.1). Then, some kind of predictions or assumptions have to be done. The quality of these approximations is not supposed to affect the performance of the monitoring system because both the control limits and the statistics for a new batch are calculated in the same way. The model obtained after batch-wise unfolding has the nice feature of capturing the dynamics of the process. Nonetheless, it has been shown that these dynamics can be captured by including only a few lagged variables [47, 2].

A different strategy is based on generating a model for every sampling time of the batch duration. If each model includes only the data of a sampling time, then it is called a local model. If each model incorporates the measurements from the beginning of the batch to a sampling time, then it is called an evolving model [50, 49]. Hierarchical PCA [48] follows the latter method but giving different weight to the current measurements of the variables. This is done by calculating the scores in a hierarchical fashion. Finally, if only the immediate part of the past measurements is included in every model together with the current measurements, the procedure is called Moving Window PCA (MWPCA) [51]. Time-Varying State Space Modelling (TVSS) [76] applies the same latter philosophy in the generation of a state space model, instead of a PCA model. The problem of these methods is the number of models needed, which sometimes is not justified by an improved performance. An alternative method, named Batch Dynamic PCA (BDPCA) [47], includes the information of the immediate past measurements, but generating only one model. This restricts the modelling to processes where the correlation structure and dynamics do not change during the batch processing.

From another point of view, Wold et *al.* [31] developed a very complete monitoring framework based on variable-wise unfolding and then rearranging in batch-wise. Nevertheless, some authors have stated that the proposed online monitoring system has lower detection ability than the batch-wise approach of Nomikos and MacGregor [13] when a multi-stage process is modelled with a single model [77]. In [52] a two model framework is developed in order to take advantage from both the batch-wise and the variable-wise unfolding methods in parallel. The latter paper incorporates the idea of multistage models [16], where independent models are used for different intervals of the batch processing time. Lu et *al.* [54] base their modelling procedure in the automatic division in independent models using a clustering method. This division is improved in terms of prediction power of the resulting model when taking into account the time ordering of the measurements as in Multi-Phase PCA (MPPCA) presented in this document [1, 2].

1.5 Batch Process Control

Trelea et *al.* [15] propose the following classification for the control strategies of batch processes:

- Use pre-defined operating conditions and terminate at a fixed time.
- Use pre-defined operating conditions and determine on-line the termination time.
- Follow a pre-defined trajectory using tracking techniques.
- Systematically re-compute the trajectory to reach the desired final state -or product quality- based on predictions provided by a process model.

The two first choices are too rigid to deal with unexpected disturbances, since they are open-loop approaches. The second one is only able to compensate disturbances which postpone the termination time. In the third approach, the process is regulated to follow a specific trajectory in the process variables. Nonetheless, the control is still performed in an open-loop fashion and therefore it is not robust in front of unexpected perturbations. The fourth choice is the most appropriate control approach for a batch process. This can be performed at two levels [78]:

- Within-batch (on-line) control, in which on-line measurements of the current batch are used to perform adjustments in the manipulable variables.
- Batch-to-batch (off-line) control, in which the measurements collected from previous batches are used in the control of the current batch.

In both on-line and off-line levels, approaches based on the regulation -or tracking control- of process variables should be distinguished from those aimed at optimizing the final quality of a batch. If only process variables are controlled, there is the risk that the effect of those variables in the final

quality varies or behaves unexpectedly, worsening the control performance. Thus, the introduction of the end-quality explicitly in the control system is, in principle, a more powerful approach.

1.5.1 Within-Batch Control

The within-batch control of batch processes relies on on-line measurements to perform control adjustments. The lack of sensors for the measurements of principal variables makes this control a challenging task [79]. The uncertainties in the process behavior and their non-linear nature and slow response complicate this task [80]. The design of first principles models is an important research line [19] and the fundamental knowledge is commonly used in the control system design. Jobé et *al.* [81] regulate the substrate feeding of a fed-batch culture according to an estimation of the metabolic state of the process. Picó-Marco et *al.* [82, 83] propose a sliding mode scheme based on a reference model, also for fed-batch cultivation processes. Several feeding strategies for baker's yeast production are based on regulating the substrate concentration indirectly, based on relationships observed in the process [84].

The reliance on the process knowledge has the drawback that some of the assumed dynamic relationships may not be valid in all cases, i.e. 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 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 [27]. A more extreme approach is to rely completely on input-output data [85].

In this context, control systems must be distinguished according to the objectives of the control. If some process variables are wanted to follow a certain trajectory, Proportional-Integral-Derivative (PID) controllers or Model Predictive Controllers (MPC) can be used. Contrarily, the Model-Based Optimization (MBO) approach is based on a model which relates process measurements with the final quality or a performance function, which are to be optimized. MBO can be also implemented using the MPC scheme.

When implementing on-line batch control based on MPC, the objective is to find an 'optimum' control adjustment for each time interval. From a general point of view, a batch process can be represented by the Multiple-Input Multiple-Output (MIMO) state space model in (1.30):

$$\begin{aligned} \dot{\mathbf{p}} &= f(\mathbf{p}, \mathbf{u}) \\ \mathbf{v} &= g(\mathbf{p}, \mathbf{u}) \\ \mathbf{q} &= h(\mathbf{p}_f) \end{aligned} \tag{1.30}$$

where $\mathbf{p}$ stands for the state, $\mathbf{u}$ are the inputs or manipulable variables, $\mathbf{v}$ the outputs, control or process variables, $\mathbf{q}$ the final quality and $\mathbf{p}_f$ the state at the end of the batch.

Depending on the control system strategy, the adjustments are performed for the control of the process variables ($\mathbf{v}$), the process states ($\mathbf{p}$ or $\mathbf{p}_f$) or the final quality ($\mathbf{q}$). If the state at a certain interval k is not measurable, it may be estimated using an observer:

$$\hat{\mathbf{p}}_k = f_{obs}(\mathbf{v}_1, ..., \mathbf{v}_k, \mathbf{u}_0, ..., \mathbf{u}_{k-1}) \tag{1.31}$$

where $\hat{\mathbf{p}}_k$ is the estimated state and the observer takes as inputs the process variables up to interval k and the manipulable variables up to interval $k-1$. Using the observer, the model equations (1.30) can be integrated forward in time to predict future values of process variables, states and quality variables.

In each control interval, an optimization run, possibility restricted by a set of constraints, is performed to obtain the convenient future values of the manipulable variables. The performance function J to optimize, subject to the constraints in (1.33), is defined as:

$$min_{\mathbf{u}_k,...,\mathbf{u}_{k+N-1}} J \tag{1.32}$$

$$S(\mathbf{p}, \mathbf{u}) \leq 0, T(\mathbf{p}_f) \leq 0 \tag{1.33}$$

where $\mathbf{u}_k, ..., \mathbf{u}_{k+N-1}$ are the set of control adjustments in the manipulable variables up to a N intervals horizon -which may include the rest of the batch duration-, J can be defined in terms of the final quality or alternatively of the adjustment of the state or process variables to a predefined trajectory, and $S(\mathbf{p}, \mathbf{u})$ and $T(\mathbf{p}_f)$ are path and terminal constraints, respectively. After each optimization run, one single control action is performed ($\mathbf{u}_k$) and the procedure is repeated for the next interval.

Lakshmanan and Arkun [86] utilize a process model made up by the interpolation of multiple local linear models. The observer is a Kalman filter for each local linear model and the interpolation is based on Bayesian theory. Lee et *al.* [87] define a state space model of the tracking error of reference trajectories and also use a Kalman filter to predict future states. This approach is extended in [88] to incorporate the final quality in the error definition. Dorsey and Lee [89] define a state space model and use PCA to reduce its dimensionality. Russell et *al.* [85] use PLS for MBO. The Kalman filter once again is used in the two previous references.

In some cases, the optimization algorithm may be too much time consuming for its execution in every time interval. An alternative is the use of mid-course corrections [90], where control adjustments are only performed in specific points of the batch. Flores-Cerrillo and MacGregor [91][78] use mid-course corrections and PLS for implementing MBO. In this approach,

corrections are only performed when statistical controls limits defined are exceeded.

1.5.2 Run-to-Run Batch Control

The run-to-run (R2R) control is aimed at adjusting elements of the control system -e.g. the tracking reference, the process model, the control law, etc- from the information collected from previous batches.

From equations (1.30) and (1.31), the performance function J^i, constrained by (1.35) and given the initial state $\mathbf{p}_0^i$, can be specified as:

$$min_{\mathbf{u}_0,\ldots,\mathbf{u}_f} J^i \tag{1.34}$$

$$S(\mathbf{p},\mathbf{u}) \leq 0, T(\mathbf{p}_f) \leq 0 \tag{1.35}$$

where subscript i stands for the number of batch.

Once again, in the literature there are approaches for controlling process variables, like Iterative Learning Control (ILC) methods [92, 93, 94], and approaches for process optimization.

Optimization approaches based on numerical computations using a fundamental model [95, 96] are open-loop in nature, and may not lead to optimal performance due to process-model mismatch. Contrarily, The R2R optimization overcomes this problem. The R2R optimization can be performed in different ways, according to Srinivasan et *al.* [97]:

- *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 [23] or combined with fundamental knowledge [98]. 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 to the gradient law computed with a model [25].
- *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 [97].

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.

1.5.3 Combined Control

On one hand, within-batch control does not take advantage of the repetitive nature of batch processes. On the other hand, the use of R2R adjustments alone is not robust in front of unexpected disturbances inside a batch. Both control levels can be combined to overcome those problems.

Lee et *al.* [87] combine MPC with ILC in what they call batch-MPC (BMPC). This approach is presented for tracking nominal trajectories of process variables. In [88], this approach is extended to perform quality control and renamed Quality control-combined BMPC (QBMPC). Dorsey and Lee [89] propose to apply MPC from a model identified from the data of previous batches. Flores-Cerrillo and MacGregor [78] use an adaptive unfold-PLS (uPLS) for adjusting the relationships between process variables and quality variables and use this model to perform mid-course corrections.

2 Materials and Methods

2.1 Hardware

All the computations have been performed with a notebook Intel Pentium 4 CPU 2.66 GHz, 1 GB of RAM.

2.2 Software

The software packages used are:

– Microsoft Windows XP Professional (Spanish), 2002 Version with Service Pack 2.

– Matlab 6.5 and R2006a (The MathWorks).

– PLSToolbox 3.5 (Eigenvector Research, Inc.).

– StatGraphics Plus 5.1 (Statistical Graphics Corp.).

Only the PCA and PLS algorithms and the control limits of certain statistical charts are computed using the PLSToolbox. The rest is implemented using self-coded software in Matlab. A complete toolbox containing most of the proposals in the document has been developed. StatGraphics is used to perform the Analysis of Variance (ANOVA) utilized in several comparisons through the present document.

2.3 Processes

Six data sets from four different processes are used in the experiences of this document. The structure of the data is listed in Table 2.1.

Etch Process This data set was used in [99]. Data from this process belong to different operation points. Only the first 30 batches, which belong to the same operation point, are used. The total data set includes 30 batches, 12 variables and 80 sampling times.

Table 2.1. Structure of the data sets under study: (number of batches × number of variables × number of sampling times) for process data and (number of batches × number of sampling times) for quality data.

Name (in short)	Complete Data-set	NOC Calibration	NOC Test	Abnormalities
Etch	$\underline{\mathbf{X}} = (30 \times 12 \times 80)$	-	-	-
Nylon	$\underline{\mathbf{X}} = (50 \times 9 \times 116)$	$\underline{\mathbf{X}} = (31 \times 9 \times 116)$	$\underline{\mathbf{X}} = (5 \times 9 \times 116)$	$\underline{\mathbf{X}} = (5 \times 9 \times 116)$
Saccha. A	$\underline{\mathbf{X}} = (64 \times 10 \times 100)$	$\underline{\mathbf{X}} = (30 \times 10 \times 100)$	$\underline{\mathbf{X}} = (14 \times 10 \times 100)$	$\underline{\mathbf{X}} = (20 \times 10 \times 100)$
Saccha. B	$\underline{\mathbf{X}} = (78 \times 2 \times 100)$ $\mathbf{Y} = (78 \times 100)$	$\underline{\mathbf{X}} = (52 \times 2 \times 100)$ $\mathbf{y} = (52 \times 100)$	$\underline{\mathbf{X}} = (26 \times 2 \times 100)$ $\mathbf{y} = (26 \times 100)$	- -
Water A	$\underline{\mathbf{X}} = (145 \times 5 \times 340)$	$\underline{\mathbf{X}} = (69 \times 5 \times 340)$	$\underline{\mathbf{X}} = (35 \times 5 \times 340)$	$\underline{\mathbf{X}} = (40 \times 5 \times 340)$
Water B	$\underline{\mathbf{X}} = (22 \times 5 \times 340)$ $\mathbf{y} = (22 \times 18)$	$\underline{\mathbf{X}} = (18 \times 5 \times 255)$ $\mathbf{y} = (18 \times 18)$	- -	- -

Nylon 6'6 Polymerization This is a well-known real data set of 50 batches obtained from Reactor A of the polymerization of nylon 6'6 provided by Dupont Co. For more details of the process, see [22]. The data have been used in several papers [13, 47, 2, 3]. Data from this set are not equalized, but each measurement vector includes information about the stage it belongs to. Equalizing is done by linear interpolation from this information. Nine variables which take values at 116 sampling times conform the data of an equalized batch. From the 50 batches, 36 were catalogued as normal in a preliminary analysis. These 36 batches are split in calibration (31 batches) and test (5 batches) sets. Five abnormal batches complete the test set.

Saccharomyces Cerevisiae The model of the *Saccharomyces cerevisiae* cultivation process presented in [19] is used to generate data by simulation. Data for batches under NOC are generated introducing gaussian noise of low magnitude in the initial conditions (5%) and measurements (1%). Some batches processed with slightly modified values of the internal constants are added, in order to augment the variability among batches. After the fixed interval of 100 sampling times, the product of a batch is discharged from the reactor.

For the monitoring experiments (Saccha. A in Table 2.1), the data matrix representing NOC contains 30 batches for calibration; and 14 normal and 20 abnormal batches conform the test set. Ten variables are measured every sampling time.

For prediction experiments (Saccha. B in Table 2.1), a set of 52 batches is generated for calibration and 26 for test. The purpose is to predict the biomass concentration from 2 process variables: the specific oxygen uptake rate and the specific carbon dioxide evolution rate.

The model of [19] is also used to assess the performance of the optimization approach presented in this document in Chapter 9.

Waste-Water Treatment These data were collected in laboratory from a sequential batch reactor (SBR) [100] and used to compare different approaches for monitoring [77] and prediction [101]. Each batch consists of several stages with constant duration. Five process variables are measured every 1.06 minutes yielding 340 sampling time points.

For the monitoring experiments (Water A in Table 2.1), the data matrix representing NOC contains 69 batches for calibration; 35 normal batches and 40 abnormal batches conform the test set.

For prediction experiments (Water B in Table 2.1), only two of the stages of the process were studied: the anaerobic and the aerobic. A total of 255 sampling times conform the data in this case. Following the recommendations of [100], the trajectories of all the process variables but the oxygen concentration are substituted by their increment from the beginning of the batch. Twenty-two batches belonging to three different seedings of the system conform the complete data set. Four batches are discarded because of problems with the sensor and numerous missing values. The five process variables are used to predict the phosphorus content.

Part II

Modelling of Batch Processes with PCA and PLS.

3 Modelling of Batch Process data

Part of the contents of this chapter have been included in the following publications:

[2] J. Camacho and J. Picó. *Multi-Phase Principal Component Analysis for Batch Processes Modelling*, Chemometrics and Intelligent Laboratory Systems, 81(2):127-136 (2006).

[3] J. Camacho and J. Picó. *Online Monitoring of Batch Processes using Multi-Phase Principal Component Analysis*, Journal of Process Control, 10(16):1021-1035 (2006).

[5] J. Camacho, J. Picó and A. Ferrer. *Bilinear modelling of batch processes. Part I: Theoretical discussion*, Submitted to Journal of Chemometrics (2007).

[7] J. Camacho, J. Picó and A. Ferrer. *Multi-Phase Analysis Framework for Handling Batch Process Data*, Submitted to Journal of Chemometrics (2007).

[9] J. Camacho, J. Picó and A. Ferrer. *A new look at the dynamic covariance structure of various approaches for batch process modelling*, 10th Scandinavian Symposium on Chemometrics (2007).

3.1 Introduction

The approaches for modelling batch processes with PLS-based methods can be roughly classified into two categories: The single model approach and the multi-model approach.

In the single model approach, a single PLS-based model is generated for the whole process. Although three-way modelling methods exist [37, 38, 36, 35], the most commonly used procedure is to unfold the three-way matrix of data in two dimensions [45, 44]. In the resulting matrix, data are rearranged so that two of the original dimensions are merged in one. Afterwards, a bilinear model such as PCA or PLS can be fitted.

When using a single bilinear model, the direction of unfolding depends on the variability which is wanted to be modelled. According to [46], for the treatment of batch process data, only batch-wise unfolding (3.1) and variable-wise unfolding (3.2) have interest:

$$\underline{\mathbf{X}}(I \times J \times K) \Rightarrow \mathbf{X}(I \times JK) \tag{3.1}$$

$$\underline{\mathbf{X}}(I \times J \times K) \Rightarrow \mathbf{X}(KI \times J) \tag{3.2}$$

with I the number of batches, J the number of variables and K the number of sampling times. Batch-wise unfolding treats the data of a complete batch as an object. Variable-wise unfolding treats data collected each sampling time of a batch as an object. Moreover, since data are usually centered and scaled to unit variance after the unfolding operation, additional differences between both unfolding methods are found because of the preprocessing.

The direction of unfolding also influences the design of an on-line application. If batch-wise unfolding is used [13], the measurements which are not available have to be imputed at every sampling time. The methods designed to impute missing data from a PCA and PLS model [102, 103] can be used for this purpose. If variable-wise unfolding is used no imputation is necessary, but the dynamics of the process in the form of auto-covariances and lagged cross-covariances among variables are not captured [46]. In [31], three monitoring levels are generated, where only the first two are used for on-line monitoring. The level 1 consists of a variable-wise model. Thus, it does not take into account the auto-covariances and lagged cross-covariances. The level 2 uses the scores obtained from the previous model to generate the monitoring charts. This is carried out by rearranging the data as in batch-wise unfolding. The authors also conceive the possibility of generating a PCA model from these rearranged data. By doing so, auto-covariances and lagged cross-covariances of the scores of level 1 are captured in the level 2. [52] proposes to take advantage of both batch-wise and variable-wise unfolding procedures by generating two models from the data, one after each unfolding. BDPCA or BDPLS [47] is another single-model approach equivalent to the variable-wise unfolding

with the addition of LMVs as variables [3]. This addition incorporates the dynamic information into the model.

The multi-model approach is traditionally based on the generation of a bilinear (PCA or PLS) model for every sampling time of the batch duration [48, 49, 51]. These models do not need to impute measurements and are able to capture the dynamics by including LMVs as variables, but the price to pay is the generation of such a number of models. Again, several proposals can be found in the literature, which mainly differ in the data used to generate the sub-models. If each sub-model includes only the data of a sampling time, then it is called a local model [49]. If each sub-model incorporates the measurements from the beginning of the batch to a sampling time, then it is called an evolving model [50, 49]. Hierarchical models [48] follow the latter method but giving different weight to the current measurements of the variables. This is done by calculating the scores in a hierarchical fashion. Finally, if only the immediate part of the past measurements is included in every sub-model together with the current measurements, the procedure is called MWPCA [51] or MWPLS.

One of the most appreciated benefits of using PLS-based technics is that, by conveniently studying the calibrated model, the process understanding can be improved [22]. Although the calibration of a sub-model for each sampling time can be advantageous for the on-line application, such a number of sub-models can be complex to handle and difficult to interpret. Recently, the merging of some of the sub-models in one representing certain periods of the process has been studied. In [16] and [52], authors propose the use of a sub-model for every stage of the process. Also, several sub-models can be defined for a single stage if necessary. [54] proposes a clustering algorithm for the automatic identification of a number of sub-models for on-line monitoring. The same objective is pursued in this document, but here the sub-models are identified using the Multi-phase algorithm defined in Chapter 5.

In this current chapter, a theoretical discussion regarding the differences among the principal modelling approaches is carried out. This discussion is limited to bilinear PLS-based models. In Section 3.2, some useful notation is introduced. In Section 3.3, the single model approach is studied. In Section 3.4, the multi-model approach based in the design of a single PLS-based model for every sampling time, i.e. the K-models approach, is analyzed. In Section 3.5 the multi-phase modelling approach is presented. Finally, Section 3.6 gives some conclusions. Readers more familiar with the way of modelling the dynamics with autoregressive models than with the PCA or PLS modelling of batch processes are suggested to read Appendix A before continuing with this chapter.

3.2 Some Notation

Let us define again $\underline{\mathbf{X}}(I \times J \times K)$ as the process data matrix collected from a batch process and aligned, which contains the values of J variables at K sampling times in I batches. To apply a bilinear modelling method, these data have to be rearranged in two dimensions. This can be achieved by unfolding $\underline{\mathbf{X}}$, by dividing $\underline{\mathbf{X}}$ in a number of two-way matrices or by a combination of both.

3.2.1 Unfolding

The batch dynamic unfolding can be expressed as:

$$\mathbf{X} = unfold(\underline{\mathbf{X}}, n) \equiv \underline{\mathbf{X}}^{(n)} \tag{3.3}$$

where n stands for the number of LMVs included as variables and:

$$n = \{k-1 \quad : \quad k \,\epsilon\, \{1, 2, .., K\}\} \tag{3.4}$$

Therefore, the batch-wise unfolding is:

$$\mathbf{X} = \underline{\mathbf{X}}^{(K-1)} \tag{3.5}$$

and the variable-wise unfolding:

$$\mathbf{X} = \underline{\mathbf{X}}^{(0)} \tag{3.6}$$

3.2.2 Number of sub-matrices

Let $\underline{\mathbf{X}}_{k_i:k_e}$ contain the data of $\underline{\mathbf{X}}$ from sampling time k_i to k_e. One way to arrange $\underline{\mathbf{X}}$ in two-way is to divide the data in K local sub-matrices:

$$\mathbf{X} = \{\mathbf{X}_k : k = 1, ..., K\} \tag{3.7}$$

where $\mathbf{X}_k \equiv \underline{\mathbf{X}}_k^{(0)}$.

Combining the unfolding with the division in several sub-matrices, many other approaches can be specified. For instance, the evolving approach:

$$\mathbf{X} = \{\underline{\mathbf{X}}_{1:k}^{(k-1)} : k = 1, ..., K\} \tag{3.8}$$

3.2.3 Generalized PCA and PLS models of batch data

A generalized arrangement in two dimensions can be defined as:

$$\mathbf{X} = \{\underline{\mathbf{X}}_{\phi_l}^{(n_l)} : l = 1, ..., L\} \tag{3.9}$$

with:

$$\phi_l = k_{il} : k_{el}$$

$$n_l = \{k - 1 \quad : \quad k \,\epsilon\, \{1, 2, .., k_{el}\}\} \tag{3.10}$$

$$s.t.\ k_{il} \leq k_{el},\ k_{i(l+1)} \leq k_{el} + 1,\ k_{i1} = 1,\ k_{eL} = K$$

According to (3.9) and (3.10), the arrangement in two-way will be complete in the sense that all the measurement vectors in the original three-way matrix of data are included. It also contemplates the possibility that some of these measurement vectors may be repeated several times after the arrangement, due to the inclusion of LMVs.

Finally, a generalized PCA and PLS model of a batch process is completely specified by:

$$M_{PCA} = \{PCA(\underline{\mathbf{X}}_{\phi_l}^{(n_l)}, a_l) : l = 1, ..., L\} \tag{3.11}$$

$$M_{PLS} = \{PLS(\underline{\mathbf{X}}_{\phi_l}^{(n_l)}, \underline{\mathbf{Y}}_{\phi'_l}^{(n'_l)}, a_l) : l = 1, ..., L\} \tag{3.12}$$

where a_l is the number of LVs of sub-model l.

3.3 Single Model Approach

Two traditional methods have been proposed for the end-of-batch and on-line monitoring of batch processes: the approach of Nomikos and MacGregor [45, 13] and the approach of Wold et *al.* [31]. The principal difference between these methods is found in the modelling for on-line monitoring. In [13], the batch-wise unfolding (Figure 3.1(a)) procedure is used. In [31], the variable-wise unfolding (Figure 3.2(a)) is used as first step to create the monitoring system. In both cases, the unfolded data are auto-scaled -i.e. each variable of the unfolded matrix is centered and scaled to unit variance. This implies that the -e.g. PCA- models in both approaches are modelling different data, since for the first approach the average trajectory of the process variables is subtracted whereas for the second one it is not. Therefore, the influence of the unfolding direction and of the preprocessing method in the differences found between both proposals is not clearly distinguishable.

Although the implications of the unfolding direction in a on-line system have been pointed out by several authors [46, 77] and introduced in the

first section of this chapter, here this issue is more deeply revisited with an independent point of view from the preprocessing mechanism. Furthermore, these implications do not seem to be very well understood by part of the research community yet. For instance, some authors believe that by batch-wise unfolding, the resulting model does not include the dynamics of the variation around the average trajectory [47, 76]. It will be shown that this conclusion is wrong.

The effect of the unfolding direction in the information contained by a model can be observed from the resulting covariance matrix. Let imagine the matrix $\underline{X}$ corresponds to the data collected from a batch process after the subtraction of the average trajectory[a].

3.3.1 Batch-wise unfolding

In Figure 3.1(b) the resulting $JK \times JK$ covariance matrix after the batch-wise unfolding of the data, i.e. $\underline{\mathbf{X}}^{(K-1)}$ (Figure 3.1(a)), is shown. It can be seen that the relationships among all the variables at different sampling times are modelled by the PCA (or PLS) performed over this matrix. These relationships include the variances and instantaneous cross-covariances of the variables at every sampling time -represented by the square sub-matrices $\{VC_1 ... VC_K\}$ in the diagonal- along with the auto-covariances and lagged cross-covariances of the variables -relationships outside these sub-matrices. Sub-matrices $V_{k,k+d}$ for $k = 1, ..., K - d$ contain the auto-covariances and cross-covariances of order d. In Figure 3.1(b), only sub-matrices $V_{k,k+1}$ are drawn for the shake of simplicity.

The dynamics of the variation around the average trajectory are represented by the relationships in sub-matrices $V_{k,k+d}$. It can be concluded that, unlike some authors have claimed, batch-wise unfolded models do incorporate the linear dynamics (around the average trajectory) of the process. Furthermore, these models are able to capture changing relationships among variables, since sub-matrices VC_k and $V_{k,k+d}$ for two different values of k -say 1 and 10- are placed in different parts of the covariance matrix, and so treated independently.

3.3.2 Variable-wise unfolding

The $J \times J$ covariance matrix obtained after variable-wise unfolding, i.e. $\underline{\mathbf{X}}^{(0)}$ (Figure 3.2(a)), is equivalent, but a constant factor equal to $\frac{1}{K}$, to the sum of the matrices $\{VC_1 ... VC_K\}$ (see Figure 3.2(b)). This has two consequences: first, variable-wise models only incorporates the variances and instantaneous cross-covariances of the variables and not the dynamics; and second, each

[a] The average trajectory is subtracted so that the resulting two-way matrix after both batch-wise and variable-wise unfolding is centered and the same data are modelled.

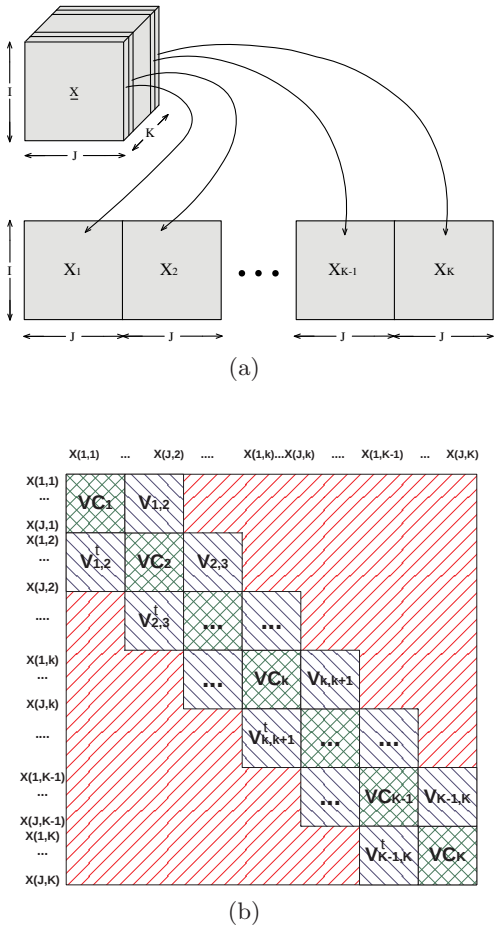


Fig. 3.1. Batch-wise unfolding (a) and resulting covariance matrix (b), with I the number of batches, J the number of variables and K the number of sampling times.

relationship between a pair of variables in the covariance matrix is an average of their relationship throughout the batch duration. Thus, this modelling strategy is only valid when the correlation structure[b] of a process is more or less constant. On the other hand, the number of batches required to generate a model in variable-wise unfolding is lower than in batch-wise unfolding. After the former unfolding, for every batch, a number of objects equal to the length of the batch is available. In batch-wise unfolding, a batch is a single object.

[b] See Appendix A for an explanation of what is understood as changing correlation structure.

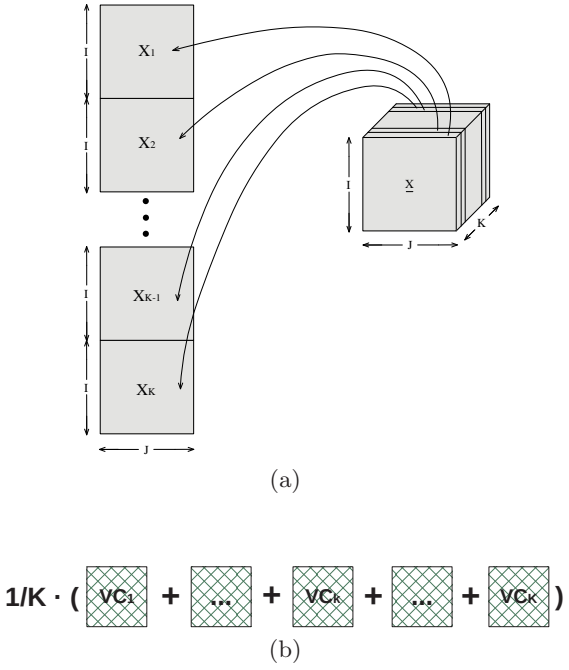


Fig. 3.2. Variable-wise unfolding (a) and resulting covariance matrix (b), with I the number of batches, J the number of variables and K the number of sampling times.

In [31], data are variable-wise unfolded, auto-scaled and then regressed to time with PLS. Care should be taken when following this strategy so that data are split in intervals where the correlation structure is almost constant [52, 77].

3.3.3 Batch dynamic unfolding

In [47], authors proposed a different way of arranging the three-way matrix of data in two dimensions for on-line monitoring. This unfolding procedure, followed by PCA or PLS modelling, is called BDPCA or BDPLS, respectively. In [3], the equivalence of this method to the variable-wise unfolding with LMVs addition was demonstrated. In the proposal of [47], an augmented matrix $\mathbf{X}_{i,n}((K-n) \times J \cdot (n+1)))$ is generated for every batch, including the n immediate LMVs, with J the number of variables and K the number of sampling times (3.13):

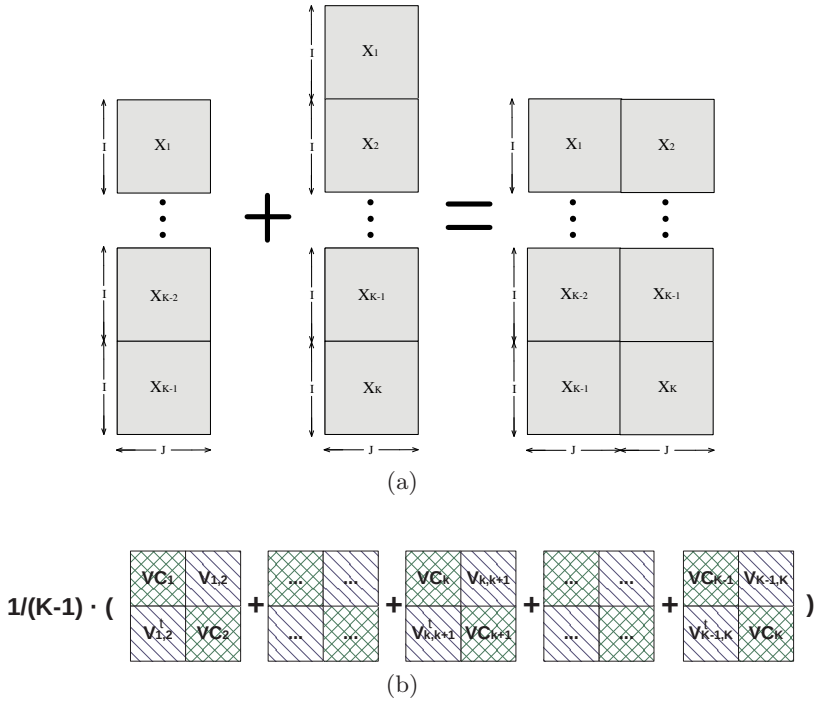


Fig. 3.3. Batch dynamic unfolded data matrix with 1 LMV (a) and resulting covariance matrix (b), with I the number of batches, J the number of variables and K the number of sampling times.

$$\mathbf{X}_{i,n} = \begin{bmatrix} \mathbf{x}_{i,1} \ \ldots \ \mathbf{x}_{i,n} \ \mathbf{x}_{i,n+1} \\ \mathbf{x}_{i,2} \ \ldots \ \mathbf{x}_{i,n+1} \ \mathbf{x}_{i,n+2} \\ \ldots \\ \mathbf{x}_{i,K-n} \ \ldots \ \mathbf{x}_{i,K-1} \ \mathbf{x}_{i,K} \end{bmatrix} \tag{3.13}$$

where $\mathbf{x}_{i,k}(1 \times J)$ is the measurement vector of the J variables for batch i and sampling time k.

Then, a *pseudo-covariance* matrix $\mathbf{pS}_{i,n}$ is calculated for every $\mathbf{X}_{i,n}$ (3.14). The term 'pseudo' is used here because the authors of [47] subtract the average trajectory from the data, so that the mean vector of $\mathbf{X}_{i,n}$ does not have to be zero.

$$\mathbf{pS}_{i,n} = \frac{(\mathbf{X}_{i,n})^T(\mathbf{X}_{i,n})}{K-n-1} \tag{3.14}$$

Finally, the *pseudo-covariance* matrices are averaged obtaining matrix $\mathbf{S}_{avg,n}$ (3.15), which is the same matrix but a constant factor (the denom-

inator) as the covariance matrix of the variable-wise unfolded matrix when including n immediate LMVs as variables (the proof is straightforward).

$$\mathbf{S}_{avg,n} = \frac{(K-n-1)\sum_{i=1}^{I} \mathbf{pS}_{i,n}}{J(K-n)} \tag{3.15}$$

Figure 3.3 shows an example of the batch dynamic unfolding method with the addition of one LMV, $\underline{\mathbf{X}}^{(1)}$, along with the associated covariance matrix. From this figure it is easy to understand the effect of the addition of LMVs. Dynamics of order d are built in the model by matrices $V_{k,k+d}$. The more LMVs added, the more dynamic information included in the model. This unfolding method can be seen as a generalization of the traditional unfolding procedures: if no LMV is added, the resulting matrix is the same as the one after variable-wise unfolding; if all possible LMVs are added, the resulting matrix is the same as the one after batch-wise unfolding. As stated in [2], depending on the nature of the process, part of the dynamic information may be negligible and it could be advantageous not to incorporate it to the model. Thus, identifying the optimum number of LMVs from the data of a process is, a-priori, a more powerful modelling approach than using an extreme case, like in batch-wise or variable-wise unfolding.

It should be pointed out that when the number of LMVs is low, the resulting batch dynamic model is very close to the traditional way of modelling the dynamics with autoregressive models. Auto-correlations and cross-correlations are obtained independently of the precise sampling time. This model structure is very useful for control purposes and can be used to relax the necessity of alignment of the batches [33]. Nonetheless, as it happens for variable-wise unfolded models, modelling with a reduced number of LMVs assumes a constant correlation structure during the batch. The more LMVs included in the model, the less the number of objects -rows- in the unfolded matrix - $\underline{\mathbf{X}}^{(n)}$ has $I \cdot (K-n)$ rows. Therefore, by adding LMVs, the interval where the correlation structure is imposed to be constant is reduced and the model is more capable to capture changes in the dynamics. At the same time, the dynamics built in the model are more dependent on the precise sampling time. A batch dynamic model of the form $\underline{\mathbf{X}}^{(n)}$ assumes the dynamics of the process can be considered time-invariant for a $K-n$ sampling times period.

Although batch dynamic unfolding provides the possibility to capture time-varying correlation structures by adding a sufficient number of LMVs, this may lead to over-parameterized models. Take the extreme example of a process in which time-varying but only static relationships among variables exist. This process can be modelled by batch-wise unfolding the data and applying a bilinear PLS-based method. Then, time-varying static relationships will be effectively modelled by matrices VC_k in Figure 3.1. Nonetheless, the rest of the covariance matrix will be just noise, since there are not significant dynamic relationships.

3.4 K-Models Approach

The K-models approach is based on generating a single sub-model for every sampling time. Local models [49] are the simplest example of this approach, where the sub-model associated to a sampling time is computed from the data collected at that sampling time alone. Data are split according to the equation:

$$\mathbf{X} = \{\mathbf{X}_k : k = 1, ..., K\} \tag{3.16}$$

Nonetheless, the inclusion of the dynamics of the process can be crucial for a good modelling performance. A straightforward alternative to local modelling is evolving modelling [50, 49], where all the possible LMVs are included in a sub-model as additional variables:

$$\mathbf{X} = \{\underline{\mathbf{X}}_{1:k}^{(k-1)} : k = 1, ..., K\} \tag{3.17}$$

More sophisticated approaches can be used. Not all the lagged information may be of interest or at least be given the same importance in every sub-model. Uniformly Weighted Moving Window (UWMW) or Exponentially Weighted Evolving Window (EWEW) models may be defined in order to better adjust the multi-model to the nature of the process. UWMW modelling uses the current data along with those of the immediate n_k LMVs to generate the current sub-model:

$$\mathbf{X} = \{\underline{\mathbf{X}}_{k-n_k:k} : k = 1, ..., K\} \tag{3.18}$$

n_k is *the size of the window*, a calibration parameter of the UWMW model.

An EWEW sub-model includes all the lagged measurements to the current sampling time:

$$\mathbf{X} = \{\underline{\mathbf{X}}_{1:k} \odot \underline{\mathbf{W}}_{1:k} \ : k = 1, ..., K\} \tag{3.19}$$

where $\underline{\mathbf{W}}_{1:k}$ is a weighting matrix and $\odot$ stands for the Hadamard (element to element) product. $\underline{\mathbf{W}}_{1:k}$ is constructed according to an exponentially decreasing value, the *forgetting factor* $\lambda_k \epsilon [0, 1]$, so that each measurement losses importance as the process advances. The weight of the measurement-vector collected at time $k - d$, for the generation of the sub-model at time k, is $(\lambda_k)^d$, being the weight of the current measurements always $(\lambda_k)^0 = 1$.

The parameters n_k and λ_k in UWMW and EWEW models are calibrated independently for every single sub-model -that is why the subscript k is used-, for instance by using cross-validation.

Additionally, there are two ways to carry out the inclusion of lagged measurements: as additional variables -columns- or as additional objects -rows- of the unfolded matrix. In Figure 3.4, these two ways are shown for the addition of 1 LMV to the current measurement vector at sampling time k.

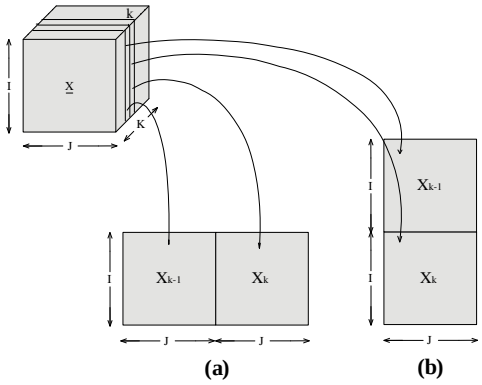


Fig. 3.4. Addition of 1 LMV to the local matrix of data at sampling time k: as additional variables -columns- (a) and as additional objects -rows- (b).

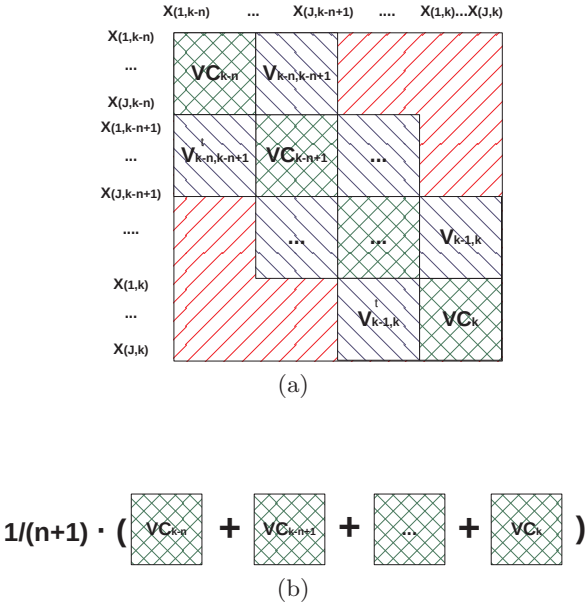


Fig. 3.5. Covariance matrices of UWMW models: LMVs added as variables -columns- (a) or as objects -rows- (b) in the unfolded matrices. n is the size of the window.

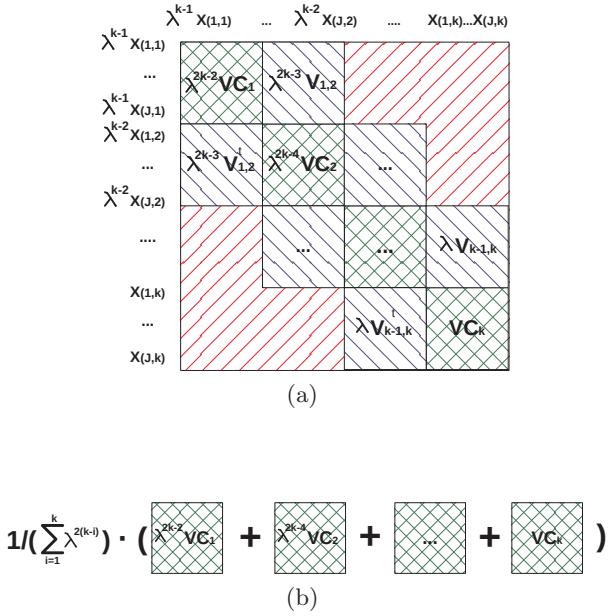


Fig. 3.6. Covariance matrices of EWEW models: LMVs added as variables -columns- (a) or as objects -rows- (b). λ is the forgetting factor.

In Figure 3.5, the covariance matrices of an UWMW model for the inclusion of the lagged measurements as variables or objects, are shown. Both arrangements can be respectively noted as:

$$\mathbf{X} = \{\underline{\mathbf{X}}_{k-n_k:k}^{(n_k)} : k = 1, ..., K\} \tag{3.20}$$

and

$$\mathbf{X} = \{\underline{\mathbf{X}}_{k-n_k:k}^{(0)} : k = 1, ..., K\} \tag{3.21}$$

In Figure 3.6, the same is shown for an EWEW model. Also, both arrangement approaches can be noted as:

$$\mathbf{X} = \{\underline{\mathbf{X}}_{1:k}^{(k-1)} \odot \underline{\mathbf{W}}_{1:k}^{(k-1)} \ : k = 1, ..., K\} \tag{3.22}$$

and

$$\mathbf{X} = \{\underline{\mathbf{X}}_{1:k}^{(0)} \odot \underline{\mathbf{W}}_{1:k}^{(0)} \ : k = 1, ..., K\} \tag{3.23}$$

The two modelling approaches have different properties. The EWEW models take into account all the past information. Thus, they handle a bigger amount of data than UWMW models and so they are more computationally-demanding. Nonetheless, a recursive PLS algorithm [104, 64] may be applied to reduce that complexity. Both UWMW and EWEW coincides for $\lambda_k = 0$ and $n_k = 0$, values for which they become local models, and for $\lambda_k = 1$ and $n_k = k - 1$, for which they become evolving models when the LMVs are included as variables. Therefore, EWEW and UWMW strategies can be seen as a general parametrization where evolving and local models are extreme cases.

Comparing the covariance matrices of Figures 3.6(a) and 3.5(a) with those of Figures 3.6(b) and 3.5(b), it should be stressed that the inclusion of LMVs as variables is more similar to the batch-wise unfolding, whereas the effect of this inclusion as additional objects resembles the case of variable-wise unfolding. A direct conclusion might be extracted: the dynamic information contained in the auto-covariances and lagged cross-covariances is not captured with the inclusion of lagged objects. Notice that this information is contained in the sub-matrices of the general form $V_{t,t+d}$, which are only present in the covariance matrices of Figures 3.5(a) and 3.6(a). Since the dynamics are of crucial importance in the modelling, a poor modelling performance of the approaches which do not include lagged variables is expected.

Finally, in [48] a hierarchical K-models approach is presented. The basis of this approach is to combine the past and local information with an adaptive hierarchical PCA model. The model for every sampling time is obtained from the high level loadings of the immediate past sampling time model and the measurements collected at the current sampling time. The analysis of this approach from the covariance matrix is not trivial and it is not performed here. Nonetheless, in Chapter 8, two extensions of this approach to PLS -see Appendix E- are compared with the other models studied here and differences are highlighted.

3.5 Multi-phase Approach

The multi-phase approach is proposed to overcome the shortcomings of the single model approach with a reduced number of sub-models. Firstly, batch-wise models may not perform adequately when several periods of the batch process are non-linearly related or independent. This is discussed in Section 3.5.1. Secondly, as commented before, variable-wise models are poor models facing changing correlation structures. This is further studied in Section 3.5.2. Finally, batch dynamic models can be affected by both nonlinearity or independence and changes in the correlation structure, as it will be explained in Section 3.5.3. In the three cases, the model can be improved by modelling independently periods of the batch: the phases.

Let us define a phase as a segment of the batch duration which is well approximated by a single linear model. From this definition it should be stressed that the optimum division in phases of a process depends on the structure of the linear models used. Using batch-wise models, the phases are periods where the variables are linearly related. Using variable-wise models, the phases are periods with a constant correlation structure. Finally, for batch dynamic models, the phases are periods with constant dynamics and where the current and n LMVs are linearly related.

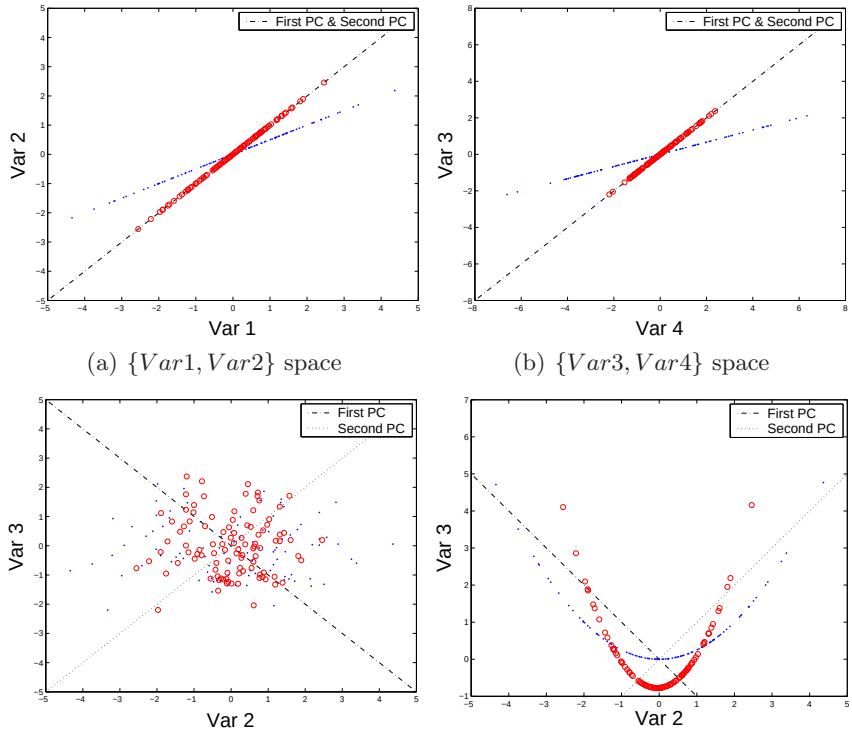
The multi-phase modelling approach provides more flexible models than the single model and K-models approaches. Nonetheless, we are aware that it has also its limitations. First, as it is well known, the use of local linear models for modelling non-linearity does not take into account part of the multivariate nature of the data -i.e., when approximating non-linearity with local models, part of the information is lost. Second, the changes in the correlation structure of a batch may be smooth. In that case, smooth transitions between sub-models may be more adequate than a crisp division in phases. Although smooth transitions can be achieved with the currently available modelling techniques, for instance by using the Fuzzy theory or the Mixtures of Probabilistic PCA [105], it should be noted that these methodologies may complicate the use and easy interpretation of the models. The use of fuzzy logic for multi-phase modelling was originally proposed in [2] and has recently been investigated in [106]. This possibility is not treated in this document and it may be a very interesting future line of study.

3.5.1 Phases in batch-wise data

When data include independent or non-linearly related groups of variables, a different model should be designed for each of these groups. In Figure 3.7 a pair of examples are shown. Imagine a data set composed of a hundred of observations of four variables, where the relationship between the first two (Figure 3.7(a)) and between the last two (Figure 3.7(b)) is perfectly linear. Nonetheless, the first pair of variables is either independent of the last pair (example in Figure 3.7(c)) or not-linearly related to it (example in Figure 3.7(d)).

When the pairs of variables are independent, the analysis can be performed separately for each pair without loss of prediction power and with less number of LVs (lower size of model). In the example, a PCA model of the four variables had to include two PCs whereas the models of the pairs of variables needed only one. Moreover, imagine the second PC is not added to the complete model because its explained variance is low. This can occur with auto-scaled data when one of the independent groups of variables has many more variables than the other. In this case, the little group of variables could be poorly modelled. Modelling the groups separately solves the problem.

When dealing with non-linear relationships, a linear PLS-based model presents a low prediction power. Figure 3.7(d) shows a quadratic relationship



(a) $\{Var1, Var2\}$ space

(b) $\{Var3, Var4\}$ space

(c) $\{Var2, Var3\}$ space, Independent variables

(d) $\{Var2, Var3\}$ space, Non-linear relationship

Fig. 3.7. Independent or non-linearly related groups of variables. Non-preprocessed observations are represented by points. Preprocessed (mean-centered and auto-scaled) observations are represented by circles.

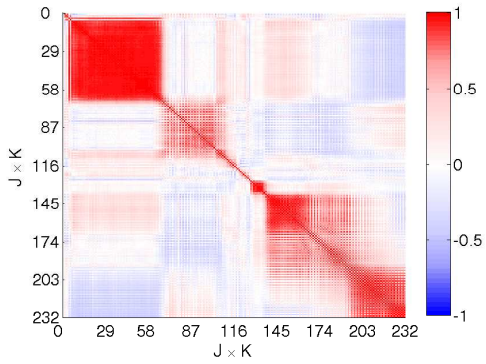


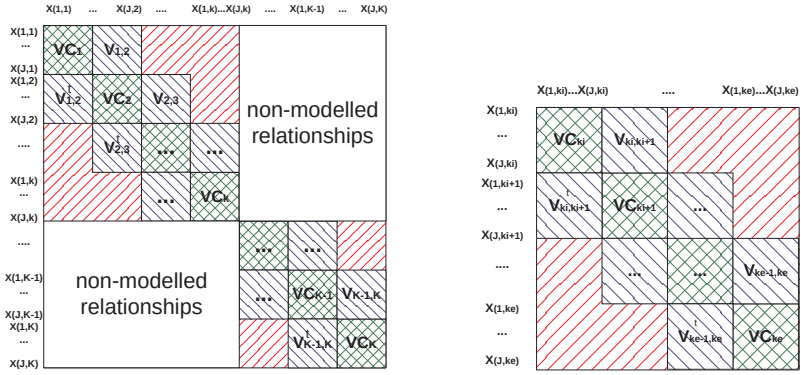
Fig. 3.8. Correlation of two variables measured 116 times (batch duration) after batch-wise unfolding and preprocessing (auto-scaling). Nylon 6'6 cultivation process.

between variable 2 and 3. The PCA model which approximates the non-linear relationship with one PC produces large residuals and so, high prediction error. If the model is fitted for monitoring, some normal observations can be determined to be abnormal and the other way round, augmenting Type I (normal observations detected as faults) and II (non-detected faults) risks. An alternative is to include another PC in the model, but the problem in the definition of useful control limits is still present. For instance, in Figure 3.7(d), any preprocessed observation which falls between the mean (crossing point of the PCs) and the quadratic curve of circles is abnormal, but the D-statistic and Q-statistic commonly used in combination with the PLS-based model [13] would catalogue it as nearer to the average behavior than any of the calibration samples. This will happen unless the information obtained from the rest of the relationships among process variables detect the abnormality. Additionally, the model is supposed to be able to predict the values of the first pair of variables from the second pair and the other way round from a linear relationship that does not truly exist.

Non-linear dynamics approximated with a linear combination of variables causes a high prediction and monitoring error in a model of a process. Approximating the non-linearity with a non-linear model can be the solution, but this is a difficult task and a big and rich data set is needed. When dealing with a huge amount of variables, this is almost impossible. In batch processes, a compromising solution is to model only the dynamics which are well approximated by a linear model. Nowadays, as the information of a process is repeated because a big amount of variables is collected, a good model of the process can be obtained by modelling the linear or pseudo-linear relationships alone. Note that, to improve the prediction power, the multiple models have to be independent. Thus, a multi-block approach [62, 63] should not be applied to model a multi-phase process because this approach forces a linear relationship among blocks at the highest level of the model. With a multi-block model not only the prediction power is not improved but it can be reduced.

Non-linear relationships and independence between two variables are examples of relationships poorly approximated with a linear model. The variables so related show low correlation values. In batch processes, two observations of the vector of process variables commonly show less correlation as the time (grade of completion of the batch) distance between them grows. In Figure 3.8, the correlation map of the batch-wise unfolded data of two variables of the polymerization of nylon 6'6 process is shown. The batch length is 116 and data include the measurements of 50 batches. The result of the batch-wise unfolding is a bi-dimensional matrix with 50 observations of 232 variables.

The figure shows a diagonal of high correlation. The dark rectangles of the graphic represent the segments of the batch where the variables are highly correlated with their preceding values. The correlation gets negligible as the



(a) (b)

Fig. 3.9. (a) Modelled and non modelled relationships after a division of a batch-wise model. (b) Covariance matrix of a sub-model in the Multi-phase approach for batch-wise data.

time distance grows. The formation of rectangles of different sizes in the correlation map means that the auto-correlation and lagged cross-correlation is dependant on the phase of the batch and that the modelling with linear models should be done independently for different phases.

In Figure 3.9(a), the effect of the division in phases of batch-wise data in the resulting covariance matrix is shown. Part of the relationships are simply not modelled, but notice that this part may show low correlation values (as in the light areas of Figure 3.8). The resulting covariance matrix of the sub-model of a phase from sampling time k_i to k_e is presented in Figure 3.9(b).

Following the notation presented in this chapter, a general multi-phase partition from batch-wise unfolded data is represented by:

$$\mathbf{X} = \{\underline{\mathbf{X}}_{k_{il}:k_{el}}^{(k_{el}-k_{il})} : l = 1, ..., L\}$$
$$s.t.\ k_{il} \leq k_{el},\ k_{i(l+1)} = k_{el} + 1,\ k_{i1} = 1,\ k_{eL} = K \tag{3.24}$$

It should be stressed that this approach is a generalization of the batch-wise unfolding. When properly calibrated, multi-phase models following (3.24) outperform batch-wise models in terms of prediction error [2] and are specially suited for off-line batch process monitoring [1, 7].

3.5.2 Phases in variable-wise data

The variable-wise models have two principal drawbacks: they do not model dynamic information and impose a constant correlation structure [20]. The

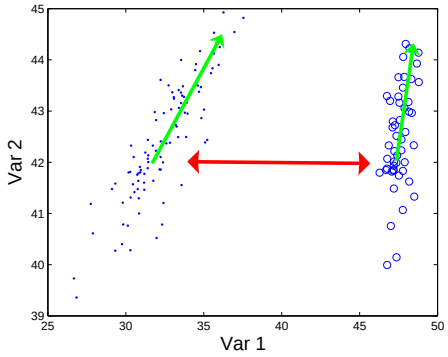


Fig. 3.10. Example of a data set with two clusters. The horizontal arrow represents the first PC fitted for the whole data set. The other two arrows represent the first PC fitted for each of the two clusters.

first drawback can be overcome by adding LMVs, yielding batch dynamic models. The second can be overcome by using several sub-models[c].

The problem of modelling changing correlation structures is the same problem when the objects form clusters in the variables-space. The presence of clusters means that the objects of a cluster represent a different reality than the objects of another cluster. If clusters are not modelled separately, there is a high risk of modelling the inter-cluster variability instead of the intra-cluster variability. This has negative consequences when the model is used for monitoring or prediction. In Figure 3.10, an example of a data set with two clusters is shown. The direction of the first PC of the PCA model fitted with the complete data set is represented by the horizontal arrow. The residuals of this PC are very high and so it is the predictive error. Additionally, this model would have a poor monitoring performance, even if additional PCs were added to it. An improved modelling performance is obtained if the clusters are identified and modelled separately.

In Figure 3.11, the scores of a batch from the *Saccharomyces cerevisiae* cultivation process in a 2 PCs variable-wise model are shown. It can be seen that the distribution of the scores of the first 50 sampling times (approximately) is very different to that of the last 50 sampling times. These two groups of sampling times should be modelled separately.

The resulting covariance matrix of the sub-model of a phase from sampling time k_i to k_e is presented in Figure 3.12. The local models of the single sampling times are averaged as in variable-wise unfolding.

A general multi-phase partition from variable-wise unfolded data is represented by:

[c] Also, by adding a sufficient number of LMVs, changing correlation structures are effectively modelled at the expense of over-parametrization.

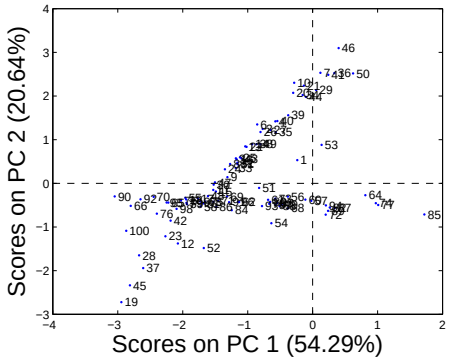


Fig. 3.11. Scores of a batch for the 2 first PCs of a variable-wise model from data of the *Saccharomyces cerevisiae* cultivation process.

$$\mathbf{X} = \{\underline{\mathbf{X}}^{(0)}_{k_{il}:k_{el}} : l = 1, ..., L\}$$
$$s.t.\ k_{il} \leq k_{el},\ k_{i(l+1)} = k_{el} + 1,\ k_{i1} = 1,\ k_{eL} = K \tag{3.25}$$

3.5.3 Phases in batch dynamic data

The resulting covariance matrix of the sub-model of a phase from sampling time k_i to k_e, for batch dynamic data, is presented in Figure 3.13. The covariance matrices of the UWMW models of the single sampling times -from k_i to k_e- are averaged.

The multi-phase models based on batch dynamic data use several sub-models for capturing changing correlation structures. The auto-covariances and lagged cross-covariances are modelled by including LMVs as variables. The size of the window n depends on the amount of past information needed. Each sub-model can have a different optimum n value. By properly calibrating this parameter, problems with non-linear relationships or independence are also avoided.

A general multi-phase partition from batch dynamic unfolded data is represented by:

$$\mathbf{X} = \{\underline{\mathbf{X}}^{(n_l)}_{max(k_{il}-n_l,1):k_{el}} : l = 1, ..., L\} \tag{3.26}$$

with:

$$n_l = \{k - 1 \quad : \quad k\ \epsilon\ \{1, 2, .., k_{el}\}\}$$
$$s.t.\ k_{il} \leq k_{el},\ k_{i(l+1)} = k_{el} + 1,\ k_{i1} = 1,\ k_{eL} = K \tag{3.27}$$

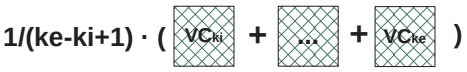


Fig. 3.12. Covariance matrix of a sub-model in the multi-phase approach for variable-wise data.

Notice that the sub-models are not completely disjoint, since n_l measurement vectors are used to fit both sub-models $l-1$ and l.

The matrix of Figure 3.12 is a particular case (for $n = 0$) of that of Figure 3.13. Nonetheless, a multi-phase partition of batch-wise data (Figure 3.9) cannot be seen as a particular case of the multi-phase partition of batch dynamic data. This is a consequence of the definition in (3.26) and (3.27). In this definition, the only possibility of obtaining disjoint sub-models is for variable-wise unfolding, i.e. $n_l = 0$ for $l \in \{1, 2, .., L\}$. Therefore, the multi-phase partition of batch-wise data in (3.24), which also yields disjoint sub-models, is not contemplated in (3.26) and (3.27), except for the case of a single phase batch-wise model. An alternative definition of a general multi-phase partition from batch dynamic unfolded data is presented in Appendix B. This definition has the nice property that it is a generalization of both the multi-phase partitions from variable-wise and batch-wise data and may be studied in the future.

The definition of (3.26) and (3.27) is a general parametrization which includes batch-wise models, variable-wise models, batch dynamic models and UWMW models, being more flexible than any of them. Although here UWMW models have been used to model the phases, an alternative approach is to use EWEW models.

The multi-phase models of batch dynamic data are specially suited for their use on-line [3, 6].

3.6 Conclusions

In this chapter, a theoretical discussion regarding the differences among several modelling approaches for batch process data is performed. Approaches are based on bilinear PLS-based modelling and are classified in two main groups: single model and multi-model approaches. The latter, in turn, are classified in K-models approaches and multi-phase approaches.

Single model approaches are those in which the three-way matrix of data is unfolded into a two-way matrix and then a bilinear PLS-based method is applied. Both batch-wise unfolding and variable-wise unfolding can be seen as special cases of a general unfolding method: the batch dynamic unfolding. In the batch dynamic unfolding, a number of LMVs are added to the

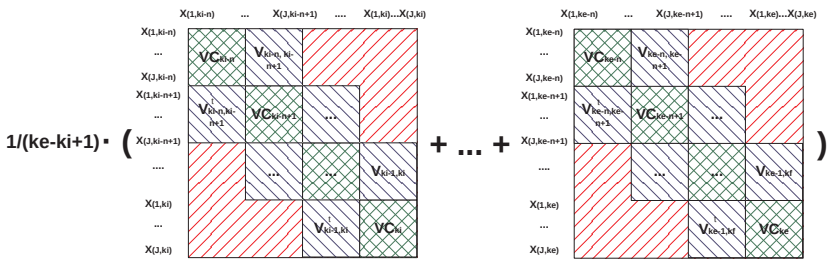


Fig. 3.13. Covariance matrix of a sub-model in the Multi-phase approach for batch dynamic data.

current measurement vector to form the object -row- of the unfolded two-way matrix. If no LMV is added, the resulting matrix is the same yielded with variable-wise unfolding. If all possible LMVs are added, then the unfolding matrix is the same yielded with batch-wise unfolding. For the sake of parsimony of the resulting model[d], the less the number of variables in the unfolded matrix the better. Also, the inclusion of independent or non-linearly related variables in the same object is not recommendable for PCA nor for PLS. Therefore, the number of LMVs in an object should be kept as low as possible, but being high enough to capture the -possibly changing- dynamics. Thus, for a constant correlation structure and when there is little influence of past information in current predictions, the unfolded matrix should have a low number of LMVs. When there are changing dynamics or most of the information collected throughout the batch influences the prediction, the appropriate models will be more similar to batch-wise models, with many LMVs included in the unfolded matrix. In general terms, batch dynamic unfolded models will be over-parameterized due this way of modelling time-varying correlation structures.

The K-models approach is based on the calibration of one bilinear PLS-based model per sampling time. This approach is able to capture changing dynamics and avoids problems with non-linearity and independence between phases of the process. Nonetheless, its principal drawback is the high number of sub-models, which makes the calibration and interpretation challenging. This is also the less parsimonious approach and so a larger data set is needed for a proper calibration.

[d] Taking into account solely the part of the model applicable to new incoming data. For instance, in PCA the number of parameters is understood as the number of elements in the loadings matrix and in PLS the number of elements in the regressors matrix.

Multi-phase models overcome the limitations of single-model approaches with a reduced number of sub-models. This is the most flexible and potent modelling approach of those studied here, as long as the number of sub-models and their parameters are properly calibrated taking into account the nature of the data. Therefore, the convenience of this approach relies on the existence of a calibration algorithm for multi-phase models.

4 Cross-validation algorithms

Part of the contents of this chapter have been included in the following publications:

[8] J. Camacho, J. Picó and A. Ferrer. *Leave-n-Samples-Out Cross-validation in PCA for Missing Data Imputation and Measurement Noise Reduction*, In elaboration for Chemometrics and Intelligent Laboratory Systems (2007).

[10] J. Camacho, J. Picó and A. Ferrer. *New Cross-Validation Methods in Principal Component Analysis*, 10th Scandinavian Symposium on Chemometrics (2007).

[11] J. Camacho, J. Picó and A. Ferrer. *A new algorithm for selecting the unfolding method and the number of sub-models in batch process modelling with PCA*, 10th Scandinavian Symposium on Chemometrics (2007).

4.1 Introduction

The principal decision when calibrating a PCA or PLS model is the optimum number of LVs. When the model is going to be used for future observations, the suggested criterion for this choice is cross-validation [107, 13]. In cross-validation, data are divided in G groups. Each time, a model is calibrated from the whole data set but a group. Afterwards, the data from that group are predicted with the model and a criterion of goodness of fit (CGF) is computed, commonly a sum-of-square-prediction-errors (PRESS) or a root-mean-of-square-prediction-errors (RMSPE). This is repeated for each of the G groups and a total CGF for a model with a LVs is obtained. From the shape of the CGF function or a modification of itself, evaluated for several numbers of LVs, the optimum number of LVs is estimated.

The PRESS and RMSPE of two-way data can be expressed as:

$$PRESS = \sum_{i=1}^{I} \sum_{j=1}^{J} e_{i,j}^2 \tag{4.1}$$

$$RMSPE = \sqrt{\frac{1}{I \cdot J} \sum_{i=1}^{I} \sum_{j=1}^{J} e_{i,j}^2} \tag{4.2}$$

where $e_{i,j}$ is the prediction error for object i and variable j.

The cross-validatory technique can be extended to the bilinear modelling of batch process data. From the theoretical discussion of the differences among the modelling approaches in [20] and in the previous chapter, it is known that different approaches are able to capture different features of a process. Moreover, a wide range of batch processes with different nature exist. Some processes may present short or long term dynamics, some others may show different phases in the batch processing, etc. Thus, the choice of the model structure -understanding model structure as the arrangement of three-way data in a set of two-way matrices and the number of LVs used to model each matrix- depends on the current process under analysis.

By using cross-validation, different model structures can be evaluated and compared for a particular case of study, so that the best one is used to model the process. This is in principle a more flexible and appropriate approach than using a fixed model structure for all the cases. This is carried out in the second part of this chapter.

This chapter is organized as follows: Section 4.2 presents the special nomenclature used in this chapter. Section 4.3 is devoted to the study of two-way cross-validation. Section 4.4 extends this study to batch three-way data. Section 4.5 offers some concluding remarks.

4.2 Special nomenclature of the chapter

For the sake of understanding, the typical nomenclature of missing data literature [102] has been inherited here with a slightly different meaning. A superscript asterisk ($*$) is used to specify the data used to fit the model in each cross-validation iteration. A superscript pound ($\#$) is used to specify the data not used in the model fitting and which are currently being predicted in each cross-validation iteration.

Subscripts have two different meanings, depending on where they are applied. For data or residual matrices, subscripts are used to specify the data from a group of rows, columns or combinations of both. For scores or loadings matrices, subscripts also stand for the number of LVs. The subscript *aug*, standing for *augmented*, means that the data matrix has been augmented with additional information. The same subscript applied to scores or loadings matrices means that they have been obtained from an augmented data matrix.

The sense in which one should understand some of the terms in this chapter is as follows:

- **object:** each single row of a two-way matrix.
- **variable:** each single column of a two-way matrix.
- **sample:** each single value of a two-way matrix.
- **latent variables:** original variables from which a number of observable variables are generated.
- **observable variables:** variables of a data matrix.

4.3 Two-way Cross-validation

4.3.1 Overview of the cross-validation in PLS

The cross-validatory computation of the prediction error in PLS models is fairly easier than in PCA models. This is because PLS models are predictive models, thus the prediction error is easily computed by comparing the actual values with those predicted with the model. PCA models are compression models and it is not completely clear how to compute this error. For PLS models, cross-validation may be performed in as follows:

- Divide the data randomly object-wise in G groups.
- For each of the G groups:
  - Use the data from all but that group to calibrate a PLS model.
  - Estimate the y-values of that group using the model.
  - Subtract predictions from the actual values to compute the prediction error.

Then, the sum of the square errors can be used as CGF. This simple mechanism cannot be used with PCA, as it will be discussed later. Formally, the PLS cross-validation follows next algorithm:

For each LV ($a = 1...A$)
  For each group of objects ($g = 1...G$)
    Form $\mathbf{X}^*$ and $\mathbf{Y}^*$ with data from all groups but g
    Form $\mathbf{X}^\#$ and $\mathbf{Y}^\#$ with data from g
    Calibrate PLS model from $\mathbf{Y}^*$ and $\mathbf{X}^*$, obtaining $\hat{\mathbf{B}}^*_{PLS,a}$
    $\hat{\mathbf{Y}}^\# = \mathbf{X}^\# \cdot \hat{\mathbf{B}}^*_{PLS,a}$
    $\mathbf{E}_g = \mathbf{Y}^\# - \hat{\mathbf{Y}}^\#$
  end
  $PRESS_a = \sum_{i=1}^{I} \sum_{m=1}^{M} e_{i,m}^2$
end

where $PRESS_a$ is the PRESS computed for a LVs.

4.3.2 Introduction to the cross-validation in PCA

Although several cross-validatory approaches have been proposed for PCA models, see [108], those by Wold [109] and Eastment and Krzanowski [107] are the most cited and influent ones. The proposal by Wold is a very fast cross-validation based on the NIPALS procedure [39]. The approach is incremental in the sense that in each iteration, after a PC has been included in the model, the rest of the computations are performed from the residuals and not from the original data matrix. Wold suggested to include PCs to the model whereas the following index R is below 1:

$$R_a = \frac{PRESS_a}{SSE_{a-1}} \tag{4.3}$$

where SSE_{a-1} is the sum of square residuals after $a-1$ PCs have been extracted. The difference between PRESS and SSE is that the former is computed by cross-validation whereas the latter is computed from the residuals of the model fitted at once from all the objects.

Eastment and Krzanowski [107] proposed an alternative scheme based on the singular value decomposition (SVD) algorithm. The prediction error is computed following a leave-one-out procedure. This makes this algorithm more computationally demanding than the approach of Wold. This approach includes in the PCA model all the PCs up to the last one for which the following index W exceeds 1:

$$W_a = \frac{(PRESS_{a-1} - PRESS_a)/DOF_a}{PRESS_a/DOF_{rem}} \tag{4.4}$$

where DOF_a is the number of degrees-of-freedom (DOFs) used to fit the a-th PC and DOF_{rem} is the remaining DOFs after the a-th PC has been added to the model.

A very different approach for PCA cross-validation was suggested by Wold [109] as a possible alternative for those for which the NIPALS procedure is not available -something very difficult nowadays. Wold did not pay very much attention to this approach. Nonetheless, it presents an attractive feature: the minimum of the PRESS computed is supposed to signal the optimum number of PCs. This is, in principle, a logical behavior for the prediction error: decrease as the addition of PCs improves the prediction performance of the model and increase when this addition is noisy. This approach is being currently used by the PLS-Toolbox software [110] and supplies the basis for the approach of this chapter.

4.3.3 Leave-n-samples-out Cross-validation

The alternative procedure suggested by [109] performs the following steps:

- Divide the data object-wise in G groups. Although in some implementations the objects are supposed to be independent [108], this is in general not true. For instance in process modelling, objects can be correlated because of the dynamics of the process. Therefore, a random selection of the objects belonging to the groups may be preferable than any pre-established criterion.
- For each of the G groups:
  - Use the data from all but that group to calibrate a PCA model.
  - Divide randomly the data of the remaining group in H groups variable-wise.
  - For each of the H groups:
    - Treat samples from that group as missing values and recover them from the PCA model and the data of the $H-1$ groups of variables.
    - Subtract predictions from the actual samples to compute the prediction error.

This cross-validation method is based on two ideas: a) if the model is going to be used for future observations, in each iteration of the cross-validation the PCA model should not be fitted from the data of the object which is going to be predicted -this is also true for PLS cross-validation; and b) since the PCA model establishes relationship structures among the variables, its prediction power should be measured by predicting the value of a variable from the rest taking into account these structures -i.e., the PCA model.

Algorithmically, the method can be specified as follows:

For each PC ($a = 1...A$)
  For each group of objects ($g = 1...G$)
    Form $\mathbf{X}^*$ with data from all groups but g
    Form $\mathbf{X}^\#$ with data from g
    Calibrate a PCA model from $\mathbf{X}^*$, obtaining $\mathbf{P}_a^*$ and $\mathbf{T}_a^*$
    For each group of variables ($h = 1...H$)
      Set $\mathbf{X}_h^\# = 0$
      $\mathbf{T}_a^\# = \mathbf{X}^\# \cdot \mathbf{P}_a^*$
      $\hat{\mathbf{X}}^\# = \mathbf{T}_a^\# \cdot \mathbf{P}_a^{*t}$
      Restore its actual value to $\mathbf{X}_h^\#$
      $\mathbf{E}_{g,h} = \mathbf{X}_h^\# - \hat{\mathbf{X}}_h^\#$
    end
  end
  $PRESS_a = \sum_{i=1}^{I} \sum_{j=1}^{J} e_{i,j}^2$
end

One controversy point in a cross-validation algorithm is to decide whether the preprocessing information, i.e. the average and weight of the variables, should be estimated either from $\mathbf{X}$ or from $\mathbf{X}^*$ and then applied to $\mathbf{X}^\#$. A discussion regarding this matter can be found in several papers [109][111]. Here, under the assumption that the model will be applied to future observations, the second option is preferred. Thus, strictly speaking, the models are calibrated following:

$$\mathbf{X}^* = \mathbf{1} \cdot \mu^{*T} + (\mathbf{T}_a^* \cdot \mathbf{P}_a^{*T}) \oslash (\mathbf{1} \cdot \varsigma^{*T}) + \mathbf{E}_{a,\varsigma}^* \tag{4.5}$$

where μ^* is the $J \times 1$ vector containing the averages of the variables and ς^* is the $J \times 1$ vector containing the weights applied to the variables in $\mathbf{X}^*$, and $\oslash$ is understood as the Hadamard (element to element) division.

Let as call the cross-validation algorithm presented the leave-n-samples-out (LnSO) algorithm. The LnSO method should be clearly distinguished from what is sometimes called leave-n-out procedure and understood as leave-n-objects-out (LnOO). In LnOO, the samples corresponding to all the variables of an objet are predicted at once:

For each PC ($a = 1...A$)
  For each group of objects ($g = 1...G$)
    Form $\mathbf{X}^*$ with data from all groups but g
    Form $\mathbf{X}^\#$ with data from g
    Calibrate a PCA model from $\mathbf{X}^*$, obtaining $\mathbf{P}_a^*$ and $\mathbf{T}_a^*$
    $\mathbf{T}_a^\# = \mathbf{X}^\# \cdot \mathbf{P}_a^*$
    $\hat{\mathbf{X}}^\# = \mathbf{T}_a^\# \cdot \mathbf{P}_a^{*t}$
    $\mathbf{E}_g = \mathbf{X}^\# - \hat{\mathbf{X}}^\#$
  end
  $PRESS_a = \sum_{i=1}^{I} \sum_{j=1}^{J} e_{i,j}^2$
end

If the PRESS curve presents a minimum, the model corresponding to that minimum is supposed to optimize the modelling performance -i.e. to minimize the prediction error- as long as the prediction error is conveniently computed. Although faster than LnSO, the principal drawback of the LnOO method is that $PRESS_a$ is monotonous decreasing with a. Therefore, since it does not present a minimum, it cannot be used directly to select the number of PCs.

On the other hand, the LnSO procedure has a principal limitation: What about the independent variables? Many times, mostly when the number of variables collected is reduced, data may present some variables which behave completely independently from the rest. This is not a problem for PCA modelling, since that sort of variables can be modelled by a single PC. Nonetheless, it is a problem for the cross-validation method LnSO explained, since the value of that independent variables cannot be predicted from the rest. This feature, in combination with the fact that the PRESS is a sum of the prediction errors for several variables, which may be predicted with very different accuracy by the same model, may lead to misleading PRESS curves as it will be shown.

It should be stressed that the improvement in the estimation of $\mathbf{T}_a^\#$ in the LnSO algorithm, by using more statistically efficient methods for the imputation of missing values such as the Projection to the Model Plane (PMP) [102] and Trimmed Score Regression (TSR) [103], does not necessary lead to an improvement in the estimation of the optimum number of PCs. As a matter of fact, it was observed that some of these methods cause the over-estimation of this number.

4.3.4 Corrected Methods

The approach of this chapter to correct the LnSO method is to replicate the information in the data. This intuitively has the consequence that independent variables are not independent any more and a LnSO cross-validatory

procedure can be used. Nonetheless, the direct duplication of the data in matrix $\mathbf{X}_{aug}$ (4.6) yields a weak cross-validation method in front of measurement noise.

$$\mathbf{X}_{aug} = [\mathbf{X}, \mathbf{X}] \tag{4.6}$$

An alternative and very promising approach is to use the information of a PCA model of the data. This information has been filtered with the PCA model, so that most of the noise has been subtracted from the data when the model has only significant PCs. For a number of PCs equal to a, the matrix of data can be augmented in two similar ways:

$$\mathbf{X}_{aug} = [\mathbf{X}, \mathbf{T}_a] \tag{4.7}$$

$$\mathbf{X}_{aug} = [\mathbf{X}, \mathbf{T}_a \cdot \mathbf{P}_a^t] \tag{4.8}$$

Although not exactly equal, these are very similar approaches, been the first one preferred because the total number of variables in $\mathbf{X}_{aug}$ is lower and so it is the cross-validation processing time. It should be taken into account that if $\mathbf{X}$ is scaled inside the cross-validation procedure and (4.7) is used, $\mathbf{T}_a$ must not be scaled. This is because the scale in $\mathbf{T}_a$ is proportional to the relevance of the PCs in terms of variance and so it must be preserved.

The idea of using an augmented matrix of data and the LnSO cross-validation procedure has been captured in the definition of two different algorithms. In absence of better names, let us call them fast-CLnSO (fast corrected-leave-n-samples-out) and CLnSO (corrected-leave-n-samples-out). The difference between both algorithms is that whereas the former uses (4.7), CLnSO generates a different augmented matrix for each variable-wise group in a similar way to (4.8), so that the negative effect of the measurement noise is reduced. The drawback of doing so is that the algorithm is slowed down.

The fast-CLnSO is represented by the following algorithm:

For each PC ($a = 1...A$)
  Calibrate a PCA model from $\mathbf{X}$, obtaining $\mathbf{P}_a$ and $\mathbf{T}_a$
  For each group of objects ($g = 1...G$)
    Form $\mathbf{X}^*$ and $\mathbf{T}_a^*$ with data from all groups but g
    Form $\mathbf{X}^\#$ and $\mathbf{T}_a^\#$ with data from g
    $\mathbf{X}_{aug}^* = [\mathbf{X}^*, \mathbf{T}_a^*]$, remember not to scale $\mathbf{T}_a^*$
    Calibrate a PCA model from $\mathbf{X}_{aug}^*$, obtaining $\mathbf{P}_{aug,a}^*$ and $\mathbf{T}_{aug,a}^*$
    For each group of variables ($h = 1...H$)
      Set $\mathbf{X}_h^\# = 0$
      $\mathbf{X}_{aug}^\# = [\mathbf{X}^\#, \mathbf{T}_a^\#]$
      $\mathbf{T}_{aug,a}^\# = \mathbf{X}_{aug}^\# \cdot \mathbf{P}_{aug,a}^*$
      $\hat{\mathbf{X}}_{aug}^\# = \mathbf{T}_{aug,a}^\# \cdot \mathbf{P}_{aug,a}^{*t}$
      Restore its actual value to $\mathbf{X}_h^\#$
      $\mathbf{E}_{g,h} = \mathbf{X}_h^\# - \hat{\mathbf{X}}_h^\#$
    end
  end
  $PRESS_a = \sum_{i=1}^{I} \sum_{j=1}^{J} e_{i,j}^2$
end

The CLnSO is represented by the following algorithm:

For each PC ($a = 1...A$)
  Calibrate a PCA model from $\mathbf{X}$, obtaining $\mathbf{P}_a$ and $\mathbf{T}_a$
  For each group of objects ($g = 1...G$)
    Form $\mathbf{X}^*$ and $\mathbf{T}_a^*$ with data from all groups but g
    Form $\mathbf{X}^\#$ and $\mathbf{T}_a^\#$ with data from g
    For each group of variables ($h = 1...H$)
      $\mathbf{X}_{aug}^* = [\mathbf{X}^*, \mathbf{T}_a^* \cdot \mathbf{P}_{a,h}^t]$
      Calibrate a PCA model from $\mathbf{X}_{aug}^*$, obtaining $\mathbf{P}_{aug,a}^*$ and $\mathbf{T}_{aug,a}^*$
      Set $\mathbf{X}_h^\# = 0$
      $\mathbf{X}_{aug}^\# = [\mathbf{X}^\#, \mathbf{T}_a^\# \cdot \mathbf{P}_{a,h}^t]$
      $\mathbf{T}_{aug,a}^\# = \mathbf{X}_{aug}^\# \cdot \mathbf{P}_{aug,a}^*$
      $\hat{\mathbf{X}}_{aug}^\# = \mathbf{T}_{aug,a}^\# \cdot \mathbf{P}_{aug,a}^{*t}$
      Restore its actual value to $\mathbf{X}_h^\#$
      $\mathbf{E}_{g,h} = \mathbf{X}_h^\# - \hat{\mathbf{X}}_h^\#$
    end
  end
  $PRESS_a = \sum_{i=1}^{I} \sum_{j=1}^{J} e_{i,j}^2$
end

Notice that CLnSO needs the calibration of a PCA model for each of the $G \times H$ different group of samples, whereas in fast-CLnSO, LnSO and LnOO a PCA model is fitted only for each of the G groups of objects. This is the reason why CLnSO is more computationally demanding. The algorithm by Eastment and Krzanowski [107] needs of $G + H$ SVD runs and the one by Wold [109] of G PCA runs.

4.3.5 Experimental Results

In LnSO, if G is set to the total of objects, I, and H is set to the total of variables, J, a leave-one-sample-out (L1SO) procedure is implemented. The drawback of L1SO is the computational demand and its advantage is that there is no risk of an inappropriate choice of the groups. This inappropriate choice may cause, for instance, the over-estimation of the prediction error when strongly correlated variables are included systematically in the same h-group. The importance of the over-estimation depends on the data nature, so that it may be negligible when a high number of correlated variables exist or may not in other cases. With the same idea, L1OO, fast-CL1SO and CL1SO procedures can be defined. The study of a convenient -in general terms- value for G and H is beyond the scope of this investigation and we will restrict ourselves to the leave-one-out case.

The approaches of Wold [109] and Eastment and Krzanowski [107], along with L1OO, L1SO, CL1SO and fast-CL1SO will be compared. Two different experiments have been performed for the study presented in this chapter. Firstly, a set of data matrices obtained by simulation are analyzed. The optimum number of PCs are known a priori and the data matrices are contaminated with measurement noise of different magnitudes. Secondly, a real data set used in both [109] and [107] is analyzed. The data set corresponds to the gas chromatography retention index matrix published by McReynolds [112].

Simulations

Three simulated data matrices are presented in this section. The three have a very different nature, with different number of latent and observable variables. In all the cases, the latent variables are generated independently at random following a normal distribution of 0 mean and standard deviation 1. Each observable variable is obtained from one latent variable alone or as a linear combination of two latent variables. Notice that the final data sets used in the comparison are composed of observable variables alone.

For the first data set, 8 latent variables were generated and from them, a total of 10 observable variables (ratio 8/10) in the following way: from each possible combination of 2 of the first 4 latent variables, 6 observable variables are obtained; from each possible combination of 2 of the following 3 latent variables, 3 observable variables are obtained; the last observable variable coincides with the last latent variable. Mathematically:

$$x_i = lv_j + lv_k, i\epsilon\{1,..,6\}, j \neq k\epsilon\{1,..,4\}$$
$$x_i = lv_j + lv_k, i\epsilon\{7,8,9\}, j \neq k\epsilon\{5,6,7\}$$
$$x_{10} = lv_8$$

where x_i stands for the i-th observable variable and lv_j stands for the j-th latent variable.

For the second data set, 12 latent variables were generated and from them, a total of 27 observable variables (ratio 12/27). The 12 latent variables are present in the data set as observable variables. The rest 15 observable variables are obtained from the possible combinations of 2 of the first 6 latent variables.

$$x_i = lv_j, i\epsilon\{1,..,12\}, j\epsilon\{1,..,12\}$$
$$x_i = lv_j + lv_k, i\epsilon\{13,..,27\}, j \neq k\epsilon\{1,..,6\}$$

For the third data set, 15 latent variables were generated and from them, a total of 50 observable variables (ratio 15/50). The first 5 latent variables are present as observable variables. The rest 45 observable variables are obtained from the possible combinations of 2 of the last 10 latent variables.

$$x_i = lv_j, i\epsilon\{1,..,5\}, j\epsilon\{1,..,5\}$$
$$x_i = lv_j + lv_k, i\epsilon\{6,..,50\}, j \neq k\epsilon\{6,..,15\}$$

Measurement noise is generated independently for each observable variable and at random, following a normal distribution of 0 mean. The standard deviation used depends on the percentage of noise chosen to be added to the data sets. A m% measurement noise is generated with standard deviation $\sqrt{m/100}$. Thus, a 1% measurement noise is generated with standard deviation 0.1, a 4% measurement noise is generated with standard deviation 0.2 and so on.

Table 4.1. Number of PCs detected by different approaches in the first simulated example (8 latent variables)

% Noise	R	min(R)	W	L1SO	fast-CL1SO	CL1SO
1%	2	**8**	2	6	**8**	**8**
4%	2	**8**	2	6	**8**	**8**
9%	2	**8**	2	6	**8**	**8**
16%	2	**8**	2	6	**8**	**8**
25%	2	9	0	6	9	9

First data set The PRESS curves of the different approaches, along with the R (4.3) and W (4.4) statistics, are presented in Figure 4.1 for measurement noise percentages from 1% to a 25%. This graphical information is completed with Table 4.1, where the optimum number of PCs, according to

each approach, is presented. Notice L1OO is not included in the table since it does not point out a specific number of PCs unequivocally. This also happens in the other data sets studied.

First, let us focus on the approach of Wold [109], Figure 4.1(a). Although the PRESS curves computed with this approach are almost monotonous decreasing, all of them yielding their minimum at 9 PCs, the R-statistic corrects this tendency. This statistic is very irregular, presenting high differences in its values for consecutive PCs. It exceeds the unity at 3 PCs for the first time. Therefore, only 2 PCs would be catalogued as significant. Nonetheless notice that, in most of the cases, the minimum of R is reached at 8 PCs (see Table 4.1).

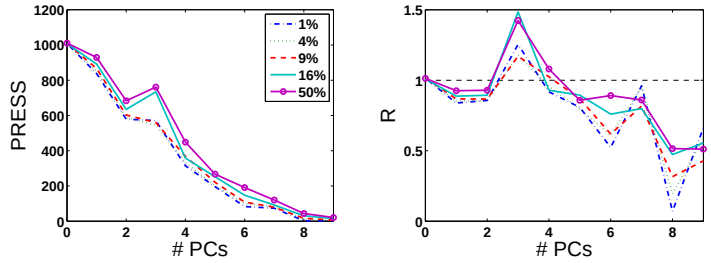
The PRESS of the approach of Eastment and Krzanowski [107] is mostly decreasing (Figure 4.1(b)). The W-statistic is also very irregular. Probably, an experienced observer would determine that 5 or 6 PCs should be added to the models. Nonetheless, strictly following the authors recommendations, only 2 PCs would be determined to be significant in most of the cases.

The L1OO approach, Figure 4.1(c), yields also a monotonous decreasing PRESS. It is sometimes suggested to find a "knee" in that kind of PRESS curves. Nevertheless, in some cases, like in this very case, it can be difficult to find. Moreover, a "knee" is a very subjective notion which cannot be clearly defined in an automatic framework for the selection of the number of PCs.

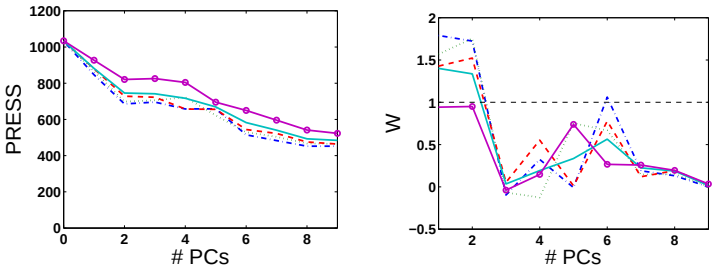
As it can be seen in Figure 4.1(d), the L1SO approach is not able to identify the optimum number of PCs. Remember in this first data set, only one latent variable out of the 8 appeared in a single observable variable. Therefore, there is only one independent variable in the data set. The best estimation of the number of PCs with L1SO is represented by the minimum PRESS value, as stated by Wold [109]. Since in all the cases, 6 PCs where determined to be significant with L1SO, the conclusion is that this approach may present problems even in the detection of significant PCs modelling non-independent variables. As a matter of fact, if the independent observable variable is eliminated from the data, the L1SO approach keeps on being confused, pointing out 5 significant PCs. On the other hand, it is worth to note that the PRESS curves obtained with L1SO, L1OO and with the approach of Eastment and Krzanowski have a very similar shape, with the principal difference that L1SO is not strictly decreasing.

Finally, the corrected approaches fast-CL1SO (Figure 4.1(e)) and CL1SO (Figure 4.1(f)) successfully pointed out 8 as the optimum number of PCs for percentages of measurement noise up to a 16%.

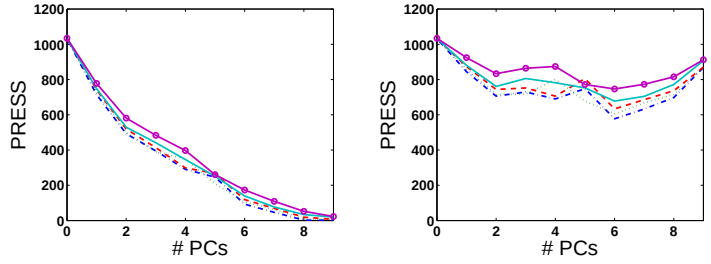
Second data set The PRESS curves of the different approaches, along with the R (4.3) and W (4.4) statistics, are presented in Figure 4.2 for measurement noise percentages from 1% to a 25%. This graphical information is completed with Table 4.2, where the optimum number of PCs, according to each approach, is presented. Notice the table shows the results up to a 36% of



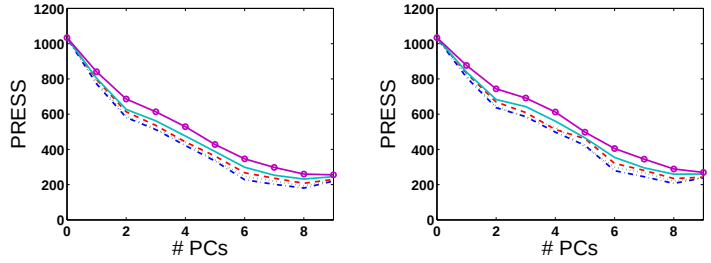
(a) Wold [109]



(b) Eastment and Krzanowski [107]

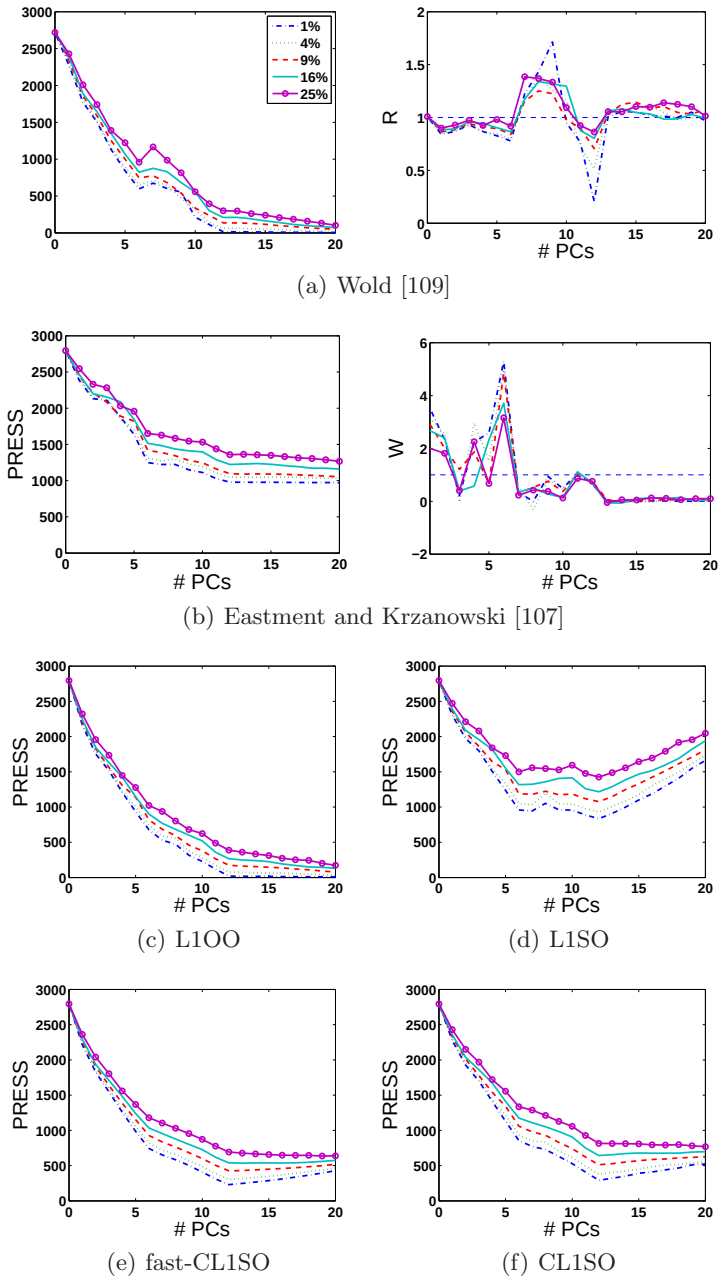


(c) L1OO (d) L1SO



(e) fast-CL1SO (f) CL1SO

Fig. 4.1. Different approaches for the selection of the number of PCs in the first simulated example (8 latent variables), corrupted with 1% (dashdot line), 4% (dotted line), 9% (dashed line), 16% (solid line) and 25% (solid line with circles) of measurement noise.



(a) Wold [109]

(b) Eastment and Krzanowski [107]

(c) L1OO

(d) L1SO

(e) fast-CL1SO

(f) CL1SO

Fig. 4.2. Different approaches for the selection of the number of PCs in the second simulated example (12 latent variables), corrupted with 1% (dashdot line), 4% (dotted line), 9% (dashed line), 16% (solid line) and 25% (solid line with circles) of measurement noise.

Table 4.2. Number of PCs detected by different approaches in the second simulated example (12 latent variables)

% Noise	R	min(R)	W	L1SO	fast-CL1SO	CL1SO
1%	6	**12**	11	**12**	**12**	**12**
4%	6	**12**	11	**12**	**12**	**12**
9%	6	**12**	6	**12**	**12**	**12**
16%	6	**12**	11	**12**	13	**12**
25%	6	**12**	6	**12**	19	17
36%	6	11	6	10	20	20

measurement noise (not included in Figure 4.2) so as to see that both min(R) and L1SO return a wrong number of PCs for that percentage.

There are several things which should be stressed from the outcomes. Again, the R-statistic (Figure 4.2(a)) exceeds the unity for a much lower number of PCs than the optimum one, but it presents its minimum at that value (12 PCs) for up to a 25% of measurement noise. With the increase of the percentage of noise, the R curve is smoothen so that the minimum stops pointing out 12 PCs. The W-statistic (Figure 4.2(b)) is very irregular and, again, needs of visual inspection for the determination of the correct number of PCs. In the shape of the PRESS curves in L1SO (Figure 4.2(d)), three intervals can be found: from 0 to 6 PCs, the PRESS falls fast; from 7 to 12 PCs, the PRESS is irregular without a clear (decreasing or increasing) tendency; from 13 PCs onwards the PRESS is monotonous increasing. This example shows the behavior of L1SO. PCs modelling latent variables which are very represented in the data (like the first 6) are clearly detected as significant. When the latent variables do not have a clear presence in the data, they are not. This is the case of the last 6 latent variables, included in the data as independent variables. Nonetheless, it should be noted that the L1SO approach identified the 12 PCs for noise percentages up to a 25%. Similar results were also observed for other data sets simulated with the same features: a number i of latent variables which produce i independent observable variables plus a number j of latent variables which homogeneously produce $k = \sum_{l=1}^{j} l$ observable variables.

Again, fast-CL1SO (Figure 4.2(e)) and CL1SO (Figure 4.2(f)) returned the correct number of PCs up to high noise levels. For this current data set, the principal benefit of correcting the L1SO approach with these methods is that the corrected PRESS curves are monotonous decreasing while the PCs added to the model are significant. This is intuitively the expected behavior of the prediction error. Also, it has the advantage that when the addition of a PC yields an increase of PRESS, there is no need to compute the PRESS values of the rest PCs since they are known to be non-significant. The price to pay when augmenting the matrix of data in the corrected approaches is a reduction of robustness in front of noise (see Table 4.2). Nonetheless, note

that they still perform well for high noise levels (9% fast-CL1SO and 16% CL1SO).

Table 4.3. Number of PCs detected by different approaches in the third simulated example (15 latent variables)

% Noise	R	min(R)	W	L1SO	fast-CL1SO	CL1SO
1%	11	**15**	13	13	**15**	**15**
4%	12	10	13	13	**15**	**15**
9%	11	10	13	13	**15**	**15**
16%	10	10	13	13	20	**15**
25%	10	10	10	13	20	16

Third data set Like in the other two sections, the results of the comparison of the approaches in the determination of the optimum number of PCs for the third data set are shown both graphically (Figure 4.3) and numerically (Table 4.3).

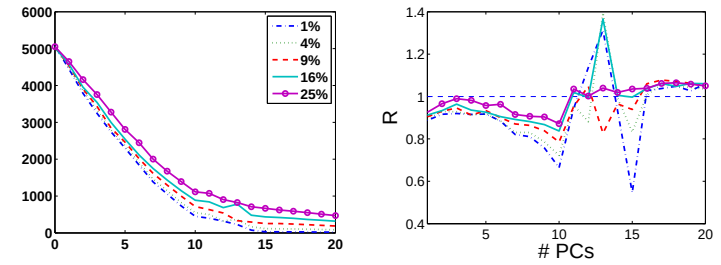
Except some exceptions, similar results to those of the preceding examples are observed in this third example. The most noticeable difference is that the minimum of the R-statistic (Figure 4.3(a)) only outputs the correct number of PCs for a 1% of measurement noise. The minimum of R has a nice interpretation in the sense that it contains the ratio between variability captured and prediction power. Therefore, the minimum R is maximizing the prediction performance of the information captured in a model. Nonetheless, since in (4.3) the SSE computed for $a-1$ PCs is compared with the PRESS for a PCs, the amount of variability captured by a certain PC is implicitly taken into account to compute R. Since the PCs are extracted ordered by the variability captured, i.e. the amount of SSE reduced in the residuals, the first PCs are more likely to yield a minimum in R. When the amount of data captured in different PCs varies greatly, i.e. when the eigenvalues corresponding to the PCs are very different, the minimum of R will be misleading.

The approach of Eastment and Krzanowski (Figure 4.3(b)), L1OO (Figure 4.3(c)) and L1SO (Figure 4.3(d)) show noisy PRESS curves, all of them very similar. Also, both W and L1SO underestimate the optimum number PCs (see Table 4.3).

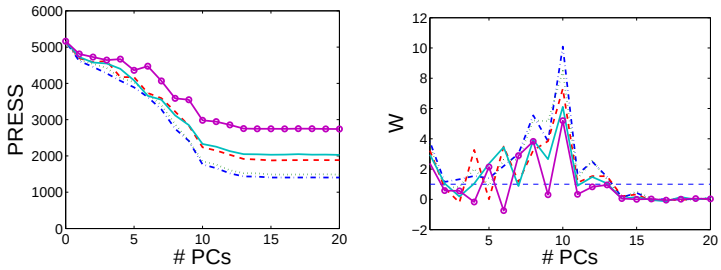
fast-CL1SO (Figure 4.3(e)) and CL1SO (Figure 4.3(f)) present smooth PRESS curves and correctly determine 15 PCs up to a 9% and 16% of measurement noise, respectively (see Table 4.3).

McReynolds Data

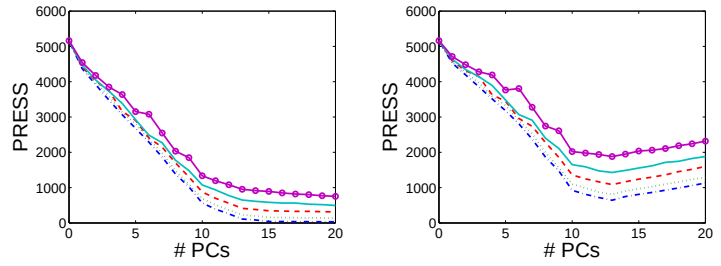
The data set used here is the same of [107]. It contains 225 objects with 10 variables each. One of the 226 original objects presented in [112] was eliminated from the data due to incompleteness. As in [109] and [107], the data



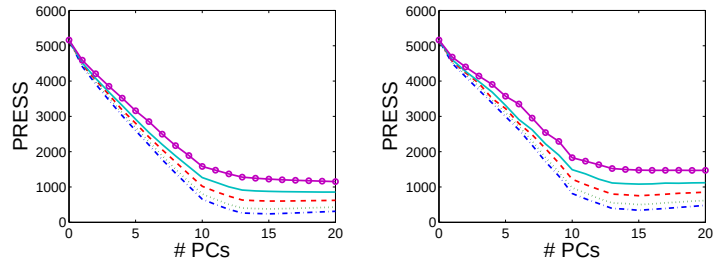
(a) Wold [109]



(b) Eastment and Krzanowski [107]

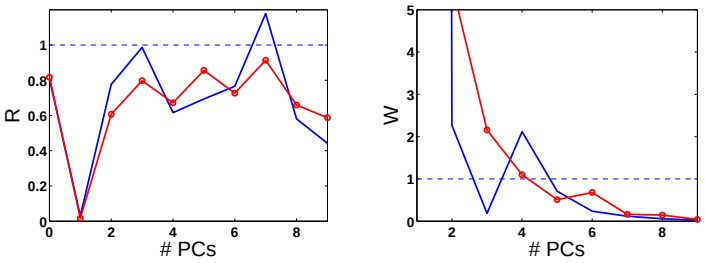


(c) L1OO (d) L1SO



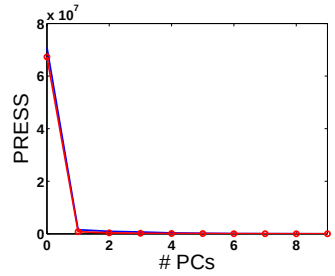
(e) fast-CL1SO (f) CL1SO

Fig. 4.3. Different approaches for the selection of the number of PCs in the third simulated example (15 latent variables), corrupted with 1% (dashdot line), 4% (dotted line), 9% (dashed line), 16% (solid line) and 25% (solid line with circles) of measurement noise.

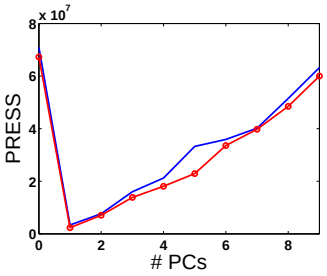


(a) Wold [109]

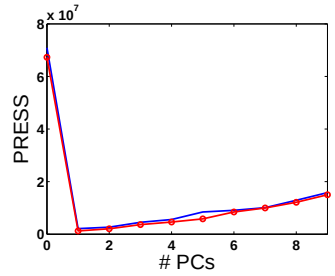
(b) Eastment and Krzanowski [107]



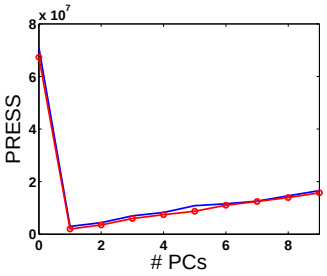
(c) L1OO



(d) L1SO



(e) fast-CL1SO



(f) CL1SO

Fig. 4.4. Different approaches for the selection of the number of PCs in the McReynolds data set (1970): complete data set (solid line) and reduced data set (solid line with circles).

Table 4.4. Number of PCs detected by different approaches in the McReynolds data set (1970).

	R	min(R)	W	L1SO	fast-CL1SO	CL1SO
Full data	2	1	4	1	1	1
Reduced data	5	1	4	1	1	1

was analyzed with and without outliers (a total of 13 outliers are found by Wold and Andersson [113]). The results are presented in Figure 4.4 and Table 4.4. Some small differences in the R-statistics presented here with those published by Wold are observed, probably caused by differences in the selection of the groups of the cross-validation and in the pre-treatment of the data used. Nonetheless, note that the differences are negligible and that the number of PCs in Table 4.4 are the published ones.

Figure 4.4 shows that the R and W statistics are affected by the presence of the 13 outliers. The shape of both statistics is different when computed with and without outliers. This was commented in the original papers. Whereas Wold stated that the optimum number of PCs was masked because of the outliers, Eastment and Krzanowski stated that in both cases the same number of PCs would be determined with their approach.

In the case of L1OO, L1SO, fast-CL1SO and CL1SO, the elimination of the outliers did not lead to a significative change in the shape of PRESS. At least not for the only one significant PC found. It can be seen that the correction of the PRESS in fast-CL1SO and CL1SO reduces the increment that the noise causes in the PRESS (Compare Figures 4.4(d), (e) and (f) from 2 to 9 PCs). This was also observed through the experimentation performed in simulation and is related with the reduction of robustness in front of noise in fast-CL1SO and CL1SO.

In the experimentation performed with simulated data, concretely in Tables 4.1, 4.2 and 4.3, it can be observed that L1SO tends to under-estimate the optimum number of PCs when there are latent variables with little influence in the data. Contrarily, fast-CL1SO and CL1SO tend to over-estimate this number for measurement noise percentages superior to a 9% or 16%. Since the three methods coincide in finding one single PC in the real data set, this should be taken as the correct number. Additionally, the minimum value of W also points out one PC. Nonetheless, care should be taken since real data do not necessary have to meet PCA modelling assumptions -i.e., that the information is hidden in the form of latent, linear combinations of variables- as simulated data were imposed to.

4.3.6 Discussion

From the results obtained in simulation, it can be stated that the R-statistic proposed by Wold, or in particular the minimum of this statistic, is interesting for the determination of the optimum number of PCs in a PCA model. Note that this good performance is yielded when the cross-validatory strategy of [109] is used. The R statistic computed with other cross-validatory strategies, such as L1SO, yielded poor results. On the other hand, it was shown that the reliability of the minimum of R depends on the similarity of the eigenvalues corresponding to the PCs. If these are of very different magnitude, the minimum of R is misleading.

The W-statistic presented not such a good performance. Imposing a hard limit in 1 for this statistic does not perform adequately in most of the cases and the visual inspection is necessary.

The L1OO approach is not appropriate for the determination of the number of PCs, at least if this decision is made directly from the PRESS curve.

The L1SO approach has problems when the latent variables are present in the data with different importance. That it to say, when the eigenvalues corresponding to the PCs are very different. The corrections performed to this approach yield very attractive outcomes, outperforming any of the other approaches.

Finally, it should be stressed that real data do not have to meet the PCA assumption, which assumes that the data can be properly modelled by a number of linear combinations of variables. The PCA from real-world data may include significant -relevant- PCs, partly significant PCs, which present significant loadings for some variables but noisy ones for others, and noisy PCs. Therefore, the partly significant PCs cannot be catalogued as significant or noisy, and so there is not a perfect solution: if one adds the PC, then is adding noise to the model; if the PC is not added, then part of the systematic variation is not included in the model. Additionally, a significant or partly significant PC may be included after a noisy PC has been already added to the PCA model, just because the variance explained by the latter is higher. In those cases, the PRESS curves obtained with the cross-validation algorithms may show strange shapes. Note that this is an intrinsic problem of PCA, and not of the cross-validation algorithms themselves.

4.4 Cross-validating batch data

Cross-validation is used to select the number of LVs of the PLS-based models in two-way data. The same technique can be extended to select the convenient arrangement in two-way form for the bilinear PLS-based modelling a specific batch process data set. This is the aim of this section.

The cross-validation strategy defined here to evaluate batch process models follow next steps:

a) Arrange the batch data in one or several two-way matrices.
b) Compute the prediction errors using cross-validation.
c) Fold the matrix/matrices of prediction errors to obtain the three-way matrix of errors.
d) Compute the PRESS.

In the first step, data are arranged according to a certain value of parameters ϕ_l and n_l in:

$$\mathbf{X} = \{\underline{\mathbf{X}}_{\phi_l}^{(n_l)} : l = 1, ..., L\} \tag{4.9}$$

with:

$$\phi_l = k_{il} : k_{el}$$

$$n_l = \{k - 1 \quad : \quad k \,\epsilon\, \{1, 2, .., k_{el}\}\} \tag{4.10}$$

$$s.t.\ k_{il} \leq k_{el},\ k_{i(l+1)} \leq k_{el} + 1,\ k_{i1} = 1,\ k_{eL} = K$$

In the second step, one of the cross-validatory algorithms presented in the first part of the chapter is run. For PLS, the algorithm of Section 4.3.1 is used. For PCA, the fast-CLnSO presented in Section 4.3.4 is preferred because of its computational efficiency. In this step, the interest is not in the PRESS computed by the algorithms, but in the matrix of errors **E** from where it is computed. This matrix will have the same structure than the data arranged in two-way in the first step. That is to say, if data were batch-wise unfolded, **E** would be a batch-wise unfolded matrix. If data were split in several sub-matrices, **E** would be split in the same sub-matrices.

The third step consists of folding back the two-way **E** matrix to the original three-way structure of the data, so that $e_{i,j,k}$ corresponds to the prediction error of $x_{i,j,k}$, the measurement of variable j at sampling time k in batch i. Notice this folding operation may not be trivial for some arrangements in two-way. The key of the cross-validation of batch data presented here is the folding operation. From this rearranged prediction error $\underline{\mathbf{E}}$, the PRESS can be computed in a fourth step.

The four steps can be included in a single algorithm. For instance, for PCA modelling using the fast-CLnSO cross-validation algorithm, the following algorithm may be defined:

For each sub-model ($l = 1...L$)
  $\mathbf{X} = \underline{\mathbf{X}}_{\phi_l}^{(n_l)}$
  Calibrate a PCA model of a_l PCs from $\mathbf{X}$,
    obtaining $\mathbf{P}_{a_l}$ and $\mathbf{T}_{a_l}$
  For each group of batches ($b = 1...B$)
    Form $\mathbf{X}^*$ and $\mathbf{T}^*_{a_l}$ with data from all groups but b
    Form $\mathbf{X}^\#$ and $\mathbf{T}^\#_{a_l}$ with data from b
    $\mathbf{X}^*_{aug} = [\mathbf{X}^*, \mathbf{T}^*_{a_l}]$, remember not to scale $\mathbf{T}^*_{a_l}$
    Calibrate a PCA model of a_l PCs from $\mathbf{X}^*_{aug}$,
      obtaining $\mathbf{P}^*_{aug,a_l}$ and $\mathbf{T}^*_{aug,a_l}$
    For each sample group of variables ($h = 1...H$)
      Set $\mathbf{X}^\#_h = 0$
      $\mathbf{X}^\#_{aug} = [\mathbf{X}^\#, \mathbf{T}^\#_{a_l}]$
      $\mathbf{T}^\#_{aug,a_l} = \mathbf{X}^\#_{aug} \cdot \mathbf{P}^*_{aug,a_l}$
      $\hat{\mathbf{X}}^\#_{aug} = \mathbf{T}^\#_{aug,a_l} \cdot \mathbf{P}^{*t}_{aug,a_l}$
      Restore its actual value to $\mathbf{X}^\#_h$
      $\mathbf{E}^l_{b,h} = \mathbf{X}^\#_h - \hat{\mathbf{X}}^\#_h$
    end
  end
end
Fold back matrices $\mathbf{E^l}$ yielding $\underline{\mathbf{E}}$
$PRESS = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} e^2_{i,j,k}$

Equivalently, for PLS:

For each sub-model ($l = 1...L$)
  $\mathbf{X} = \underline{\mathbf{X}}_{\phi_l}^{(n_l)}$
  $\mathbf{Y} = \underline{\mathbf{Y}}_{\phi'_l}^{(n'_l)}$
  For each group of batches ($b = 1...B$)
    Form $\mathbf{X}^*$ and $\mathbf{Y}^*$ with data from all groups but b
    Form $\mathbf{X}^\#$ and $\mathbf{Y}^\#$ with data from b
    Calibrate PLS model from $\mathbf{Y}^*$ and $\mathbf{X}^*$, obtaining $\hat{\mathbf{B}}^*_{PLS,a_l}$
    $\hat{\mathbf{Y}}^\# = \mathbf{X}^\# \cdot \hat{\mathbf{B}}^*_{PLS,a_l}$
    $\mathbf{E}^l_b = \mathbf{Y}^\# - \hat{\mathbf{Y}}^\#$
  end
end
Fold back matrices $\mathbf{E^l}$ yielding $\underline{\mathbf{E}}$
$PRESS = \sum_{i=1}^{I} \sum_{m=1}^{M} \sum_{k=1}^{K} e^2_{i,m,k}$

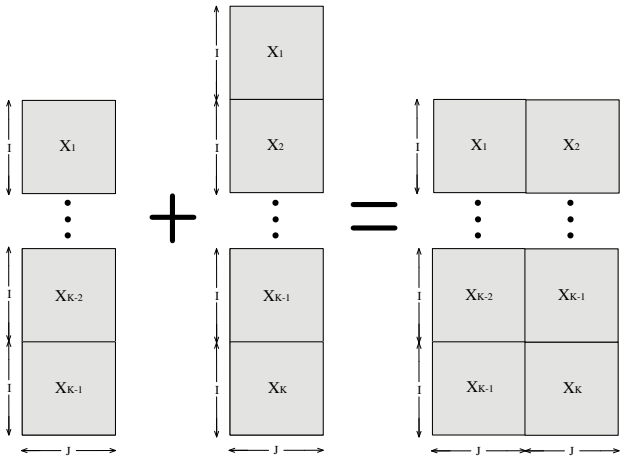


Fig. 4.5. Batch dynamic unfolded data matrix augmented with 1 LMV.

The following discussion and experimentation will be focused on the PCA cross-validation of models from batch process data, for being more complex than the PLS case. Nonetheless, the same results can be extended to PLS directly.

4.4.1 Folding unfolded data

Imagine matrix $\underline{\mathbf{C}}$, which may contain data from a certain interval of a batch process, is unfolded according to $\underline{\mathbf{C}}^{(n)}$. For $n > 0$, the same measurement-vector may appear several times in two-way. For instance, let us take the case of the batch dynamic unfolded data with 1 LMV of Figure 4.5. The resulting matrix is noted by $\underline{\mathbf{X}}^{(1)}$. Since 1 LMV is added to the data, the measurements collected from sampling time 2 to $K-1$ appear twice in $\underline{\mathbf{X}}^{(1)}$. This makes the size of the unfolded matrix bigger than that of the original one. Imagine the dimension of $\underline{\mathbf{X}}$ is $30 \times 5 \times 100$, containing a total of 15000 values. Both the variable-wise unfolded matrix $\underline{\mathbf{X}}^{(0)}$, with dimension $(30 \cdot 100) \times 5$, and the batch-wise unfolded matrix $\underline{\mathbf{X}}^{(K-1)}$, with dimension $30 \times (5 \cdot 100)$, present the same number of total values. Nonetheless, it can be seen that $\underline{\mathbf{X}}^{(1)}$ is $30 \times (10 \cdot 99)$, with a total of 29700 values.

The fact that data may grow up in quantity with the arrangement in two-way form makes necessary to establish how to compute an error of a certain measurement which is represented several times. Imagine data from $\underline{\mathbf{X}}$ are arranged in two-way according to (4.9) and (4.10). The number of times x_{ijk} appears in $\mathbf{X}$ depends on n_l and ϕ_l. Let us note that number as η_{ijk}. For instance, in the example of Figure 4.5 ($\underline{\mathbf{X}}^{(1)}$):

$$\eta_{ijk} = \begin{cases} 1 & for\ k = 1 \\ 2 & for\ k > 1\ and\ k < K \\ 1 & for\ k = K \end{cases} \tag{4.11}$$

To compute e_{ijk}, i.e. the model error to predict x_{ijk}, two choices are available: to average the several error values corresponding to x_{ijk} (4.12) or simply to take the error for the last appearance of x_{ijk} in the two-way data (4.13):

$$\hat{e}_{ijk} = \frac{1}{\eta_{ijk}} \cdot \sum_{d=1}^{\eta_{ijk}} x_{ijk} - \hat{x}_{ijk}^{d} \tag{4.12}$$

$$\hat{e}_{ijk} = x_{ijk} - \hat{x}_{ijk}^{\eta_{ijk}} \tag{4.13}$$

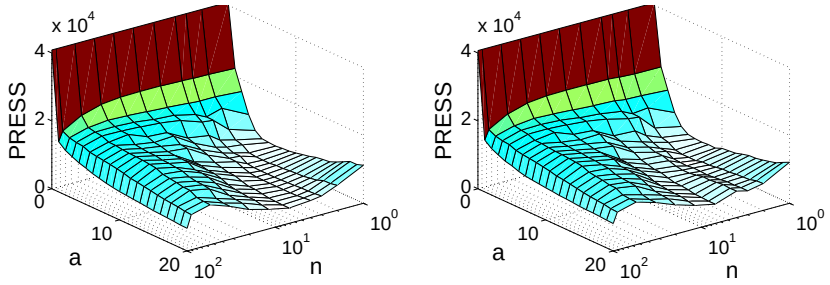
where superscript d in $\hat{x}_{ijk}^{d}$ specifies an order of appearance of x_{ijk} in $\mathbf{X}$. The order between different sub-matrices $\underline{\mathbf{X}}_{\phi_l}^{(n_l)}$ is taken according to l. Inside the same sub-matrix the order is taken according to the number of row in the unfolded matrix. Therefore, x_{ijk}^{1} is the appearance of x_{ijk} in the sub-model with lowest l-value in the lowest number of row. For instance, in $\underline{\mathbf{X}}^{(1)}$ (see Figure 4.5) values of the form x_{1j2} appear in the first row and in the $(I+1)$-th row. Thus, $\hat{x}_{1j2}^{1}$ are the cross-validated predictions of x_{1j2} performed for the data in the first row of $\underline{\mathbf{X}}^{(1)}$.

Experimentally, it was observed that option (4.12) yields smoother PRESS curves than (4.13). This was expected since the former is an average. Nonetheless, since the LMVs are used to predict the current measurement vector and not the other way round, (4.13) is preferred.

4.4.2 Improving the computational efficiency

In order to make the algorithm computationally efficient, it is necessary to reduce the number of iterations. This is accomplished by reducing the number of groups of batches (B) and of variables (H). The elements -batches or variables- belonging to a group should be chosen randomly. It was observed that if this random selection is maintained for all the models which are compared, B and H can be reduced to very low values (from 3 to 7) with almost no loss of comparison performance.

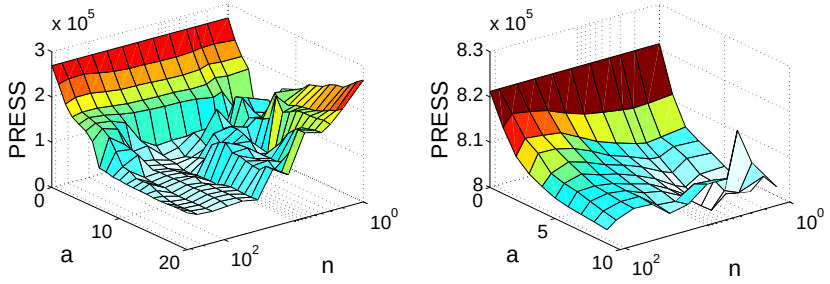
In Figure 4.6, an example with the data of the cultivation of Saccharomyces Cerevisiae, is shown. Single models of the form $PCA(\underline{\mathbf{X}}^{(n)}, a)$, i.e. with different number of LMVs (n) and PCs (a), are compared using the algorithm presented. The comparison is shown in form of surface. Figure 4.6(a) shows the result of a leave-one-out approach, where $B = 30$ and H depends on n according to $H = 10 \cdot (n + 1)$. Figure 4.6(b) is computed for $B = 3$ and $H = 3$. Both shapes are very similar, but the latter has been computed more than 10 times faster.



(a) 30000 Groups (b) 9 Groups

Fig. 4.6. Comparative of models of the form $PCA(\underline{\mathbf{X}}^{(n)}, a)$ in terms of PRESS using a leave-one-out cross-validation (a) and a cross-validation based on 3×3 groups of data (b). Cultivation of Saccharomyces Cerevisiae process.

4.4.3 Comparing models with different unfolding methods



(a) Waste-water treatment (b) Nylon 6'6 polymerization

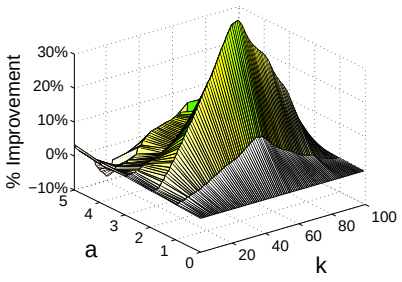
Fig. 4.7. Comparative of models of the form $PCA(\underline{\mathbf{X}}^{(n)}, a)$ in terms of PRESS using a cross-validation based on 3×3 groups of data.

In Figure 4.7, the data from the Nylon 6'6 polymerization process and the Waste-water treatment process are used. Once again, single models of the form $PCA(\underline{\mathbf{X}}^{(n)}, a)$ are compared using the cross-validatory algorithm presented and $B = 3$ and $H = 3$. Since the shape of the surface of PRESS is different for the three processes under study (see Figures 4.6 and 4.7), it can be concluded that the optimum model structure (in terms of the number of LMVs and PCs) is dependent on the nature of the process. For instance, for the waste-water treatment process (Figure 4.7(a)), a single model should

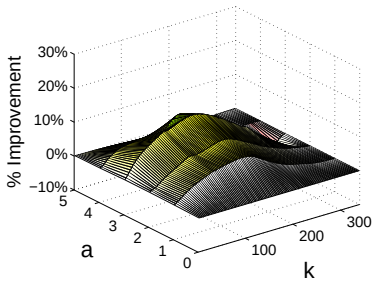
present a high n value, contrarily to what happens with the Nylon 6'6 polymerization process (Figure 4.7(b)).

4.4.4 Division in sub-models

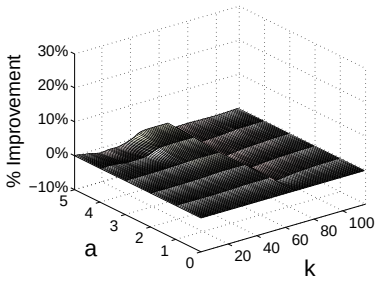
The last experiment performed is devoted to analyze how the division of a model of a batch process in two sub-models may affect the modelling performance. Models of the form $\{PCA(\underline{\mathbf{X}}^{(0)}_{1:k-1}, a), PCA(\underline{\mathbf{X}}^{(0)}_{k:K}, a)\}$ are compared in Figure 4.8.



(a) Saccha. Cerev. Cult.



(b) Waste-water treatment



(c) Nylon 6'6 polymerization

Fig. 4.8. Comparative of models of the form $\{PCA(\underline{\mathbf{X}}^{(0)}_{1:k-1}, a), PCA(\underline{\mathbf{X}}^{(0)}_{k:K}, a)\}$ in terms of PRESS using a cross-validation based on 3×3 groups of data.

In Figure 4.8(a), obtained for the Saccharomyces Cerevisiae cultivation process, the improvement (computed in terms of PRESS) exceeds the 40% when the model is conveniently divided in two. The optimal division is close to the middle of the batch -concretely, in $k = 53$. In Figure 4.8(b) the improvement slightly exceeds a 10% (for $k = 144$) and in Figure 4.8(c) it is negligible. As a matter of fact, in some cases the performance is worsened.

The location of the optimum k for the division in two sub-models is also different for the three processes. Therefore, it can be concluded that the optimum model structure (in terms of number of sub-models and location of these) is dependent on the nature of the process.

4.5 Conclusions

In this chapter, two novel cross-validation algorithms for the determination of the optimum number of PCs in a PCA model are proposed. The algorithms were compared with some well-known approaches using both simulated and real data.

The proposed algorithms, named fast corrected-leave-n-samples-out (fast-CLnSO) and corrected-leave-n-samples-out (CLnSO), have been tested in their simplest but computationally lengthiest form: fast-CL1SO and CL1SO. The fast-CL1SO approach is faster than the CL1SO approach, but also less robust to measurement noise.

From the experimentation with simulated data performed to date, part of which is included in this chapter, it was observed that both fast-CL1SO and CL1SO always determine correctly the optimum number of PCs when data are corrupted with up to a 9% and a 16% of measurement noise, respectively. Notice that these noise levels are reasonable measurement noise levels in real applications.

There is a wide range of approaches for modelling batch process data with PCA and PLS. All those approaches principally differ in how the batch three-way data are arranged in two-way form: from batch-wise unfolding to variable-wise unfolding and from a single data matrix -and thus a single PLS-based model- to multiple data matrices -multiple models. In a second part of the chapter, the cross-validation strategy is extended for its application to batch three-way data. This is done in order to identify the convenient arrangement in two-way form of the three-way matrix of data of a batch process for PCA or PLS modelling. The algorithms proposed in this chapter can be successfully used to evaluate and compare different arrangements. Moreover, the fact that the optimum modelling structure depends on the specific case of study is proven by the results obtained with the algorithm presented for PCA.

5 Multi-Phase Principal Component Analysis and Partial Least Squares

Part of the contents of this chapter have been included in the following publications:

[1] J. Camacho and J. Picó. *Monitorización de Procesos por Lotes mediante PCA Multifase*, Revista Iberoamericana de Automática e Informática Industrial, 3(3):78-91 (2006).

[2] J. Camacho and J. Picó. *Multi-Phase Principal Component Analysis for Batch Processes Modelling*, Chemometrics and Intelligent Laboratory Systems, 81(2):127-136 (2006).

[3] J. Camacho and J. Picó. *Online Monitoring of Batch Processes using Multi-Phase Principal Component Analysis*, Journal of Process Control, 10(16):1021-1035 (2006).

[5] J. Camacho, J. Picó and A. Ferrer. *Bilinear modelling of batch processes. Part II: PLS comparative*, Submitted to Journal of Chemometrics (2007).

[7] J. Camacho, J. Picó and A. Ferrer. *Multi-Phase Analysis Framework for Handling Batch Process Data*, Submitted to Journal of Chemometrics (2007).

[12] J. Camacho, J. Picó and A. Ferrer. *Multi-Phase Analysis Framework for Handling Batch Process Data*, 10th Scandinavian Symposium on Chemometrics (2007).

5.1 Introduction

The multi-stage, multi-phase modelling approach is based on the calibration of independent models for different intervals of a batch process. The data set collected from a process is split in intervals according to the sampling time mode. In each interval, data are unfolded to fit a bilinear model. In this document, the intervals determined from the process knowledge are called stages whereas those determined from the data are called phases.

Ündey et *al.* [16, 52] suggested to use several models to represent the multi-stage nature of the process for end-of-batch and on-line monitoring. They divide the process in stages, representing the different processing units and for distinguishable operations inside a unit. In [22], the authors propose to divide the model in two based on changes in the variance explained by the principal components during the batch. Lennox et *al.* [53] suggest the use of multiple models to account for changes in the correlation structure when using SPE charts.

In some cases, e.g. after batch-wise unfolding, the convenient intervals in which divide the data may not have a clear physical interpretation and so they cannot be determined by fundamental knowledge. On the other hand, the identification of the phases by process analysis can be a challenging task. The automatic identification of the phases overcomes these problems and seems to be a good alternative, as long as the recognition algorithm performs adequately.

The method named Sub-PCA [54] is, in the authors knowledge, the only attempt previous to this work to automatically determine the phases of a batch process. It was proposed for the on-line monitoring of batch processes and uses a clustering algorithm to detect segments of the batch which should be modelled by a single model. As it will be shown (see Chapter 6), a clustering algorithm is not a wise choice, since it does not take into account the sampling time ordering. Alternatively to the clustering, the Multi-Phase (MP) algorithm presented in this work uses the sampling time ordering implicitly for the identification of the phases. This approach has been used to improve the PCA modelling of batch-wise unfolded data [2] and in combination with variable-wise or batch dynamic unfolded data for the monitoring of batch processes [3].

In this chapter, the original MP algorithm [1, 2, 3] is enhanced for its general use with any unfolding procedure (batch-wise, variable-wise and batch dynamic) using either PCA or PLS. Also, a number of postprocessing algorithms has been defined to reduce the complexity of the MP model and to mixture several MP models into one. All these algorithms form the basis of the MP Analysis Framework.

This chapter is organized as follows: Section 5.2 presents the MP algorithm; Section 5.3 introduces the postprocessing methods applicable to MP models. Section 5.4 proposes the MP analysis framework and Section 5.5

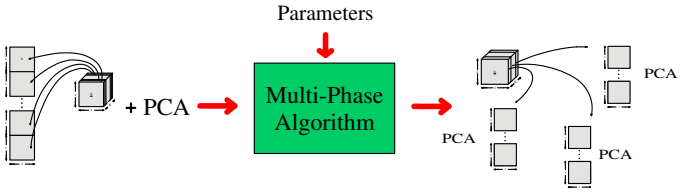


Fig. 5.1. Multi-Phase (MP) algorithm. Example of use.

presents an example of use. Finally, Section 5.6 gives some concluding remarks.

5.2 Multi-Phase Algorithm

The MP algorithm is aimed at automatically identifying the phases of a batch process. It takes as inputs the data matrix of the process, somehow unfolded, and a bilinear modelling method, and returns a partition in phases of the sampling time mode. An example with variable-wise unfolding and PCA is shown in Figure 5.1. The output of the example is a MP model formed by three variable-wise sub-models -since the input was variable-wise unfolded-, each of them modelling a different phase of the process -possibly with a different number of PCs. The algorithm has been defined for any unfolding method, namely: batch-wise, batch dynamic and variable-wise unfolding; and for PCA or PLS modelling.

The MP algorithm presented here is a slight modification of that presented in [3]. The algorithm is composed of a high-level routine and a low-level routine. The high-level routine is in charge of setting the number of phases and the number of LVs of the corresponding sub-models. This is done following a Greedy optimization approach, so that each time the best local improvement is incorporated to the global solution:

i. Select an unfolding and a modelling method.

ii. Fit model M with a_{ini} LVs.

iii. Repeat while adding LVs to M

iii.1) Add a PC to the model M, obtaining model M_1. Compute $\delta(M_1, M)$ from:

$$\delta(M_i, M) = \frac{QPE(M) - QPE(M_i)}{QPE(M_0)} \tag{5.1}$$

$$QPE = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(x_{ijk} - \hat{x}_{ijk})^2 \tag{5.2}$$

where QPE is a quadratic prediction error, for instance the PRESS computed using the algorithms in Chapter 4, and M_0 represents the preprocessing information -the averages and the weights of the variables.

iii.2) Divide the model M in two sub-models with the low-level routine, obtaining model M_2. Compute $\gamma(\delta(M_2, M))$ from:

$$\gamma(\delta(M_i, M)) = k_\gamma \cdot \delta(M_i, M) \tag{5.3}$$

iii.3) If $\delta(M_1, M) > \gamma(\delta(M_2, M))$ and $\delta(M_1, M) > T$, Replace M with M_1.

iii.4) If $\delta(M_1, M) \leq \gamma(\delta(M_2, M))$ and $\gamma(\delta(M_2, M)) > T$, Replace M with M_2 and compute the repeat loop (iii) recursively for both sub-models.

where:

- a_{ini} is the initial number of LVs in the model. $a_{ini} = 0$ is suggested, so that the preprocessing information is cross-validated as if it were some kind of zero LV. This has been also suggested elsewhere [109, 111].
- δ is an improvement index computed by cross-validation.
- γ is a weighting function to make possible the direct comparison between the improvement gained with the addition of a LV and with the division in two of a model. A γ definition with a strong theoretical background is difficult to be found. In (5.3), γ is defined as a linear function of δ, adjusted with gain k_γ. This drastically reduces the complexity and works fairly well.

– T is the improvement threshold.

The combination of T, δ and γ is used in each step of the high-level routine to decide whether a new LV or sub-model should be added or not to the complete model. When applying the MP algorithm for batch process modelling and analysis, one principal issue is to find a compromise between model performance and complexity -number of LVs and phases. This is carried out by properly defining the improvement threshold T, so that the complexity of a model is increased only when enough improvement of performance is gained. Also, T prevents from over-fitting.

The low-level routine -which is called the top-down stage in [3]- identifies the division in phases of the sampling time mode:

i. For each sampling time (k) in the current interval modelled by M

i.1) If the subdivision in k generates two segments of length higher than the minimum established ($minL$):

i.11) Fit model for the two segments.

i.12) Compute β^k from:

$$\beta^k = QRE - QRE^k \tag{5.4}$$

where QRE is the quadratic residual error of model M and QRE^k is the quadratic residual error obtained when M is divided in two sub-models in k.

i.13) If β^k is the highest to the moment ($\beta^k = max_t\ \beta^t : t = minlength, ..., k$), update better subdivision.

where:

– $minL$ is the minimum number of consecutive sampling times included in a phase. For $minL = 1$, the parameter will have no influence in the result. A different $minL$ value should be clearly justified since it establishes a hard constraint in the model.
– β is an improvement index which, contrarily to δ, is computed directly from the residuals instead of using cross-validation. Although cross-validation is supposed to be more reliable, it is very computational demanding. Since the low-level routine performs an extensive search, the use of cross-validation is not recommended.

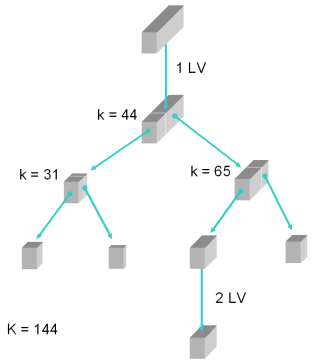


Fig. 5.2. Information-tree of an example of a MP run. The reference for the values of k is the sub-matrix time mode, and not the complete matrix time mode.

In Figure 5.2, an example of a MP run is shown. The algorithm starts from a 0 LV model -represented by the preprocessing information, i.e. the averages and weights of the variables in the unfolded matrix. Firstly, 1 LV is added to the model. In a second recursion, the 1 LV model is divided in two phases, representing the sampling times intervals $[1, 44]$ and $[45, 144]$. The first phase is divided once again yielding phases $[1, 31]$ and $[32, 44]$ and then this branch of the recursion is stopped. The second phase is also split up, yielding phases $[45, 109]$ and $[110, 144]$. Finally, a second LV is added to the sub-model of phase $[45, 109]$. As a result, the batch process has been modelled by four independent sub-models, three of them with 1 LV and the other with 2 LVs. The structure shown in Figure 5.2 is the information-tree of the model. It contains the trace of a MP run.

5.3 Postprocessing

T, δ, γ and the other parameters of the MP algorithm can be used to transform the structure of the resulting multi-phase model according to the designer needs. Nonetheless, as in any data-driven calibration, the final model principally depends on the process itself. This means that, if the process is clearly single-phase, it does not matter the value of the parameters -as long as they are reasonable values- that the resulting model will have only one phase. On the other hand, if several clear phases exists, the algorithm will separate them in different sub-models.

When using the MP algorithm for batch process modelling and analysis, the first issue which should be taken into account is the compromise between modelling performance and complexity. In some cases, the model is wanted to be as accurate as possible. Then, a non-restrictive -low value of- T should be used. In other cases, the model should not be very complex, with a lot

of phases and LVs, for instance because re-calibration may be performed in real-time. Then, a more restrictive T is convenient. The problem pops up when trying to define what it is restrictive and what it is not for a certain process, which is difficult to estimate a-priori without a deeper analysis.

The postprocessing methods here proposed are used to handle the MP model and to extract information from the process in a computationally efficient way, making the most of the information gained with the MP algorithm.

5.3.1 Postprocessing of a single Multi-phase model: Pruning algorithms

The pruning methods are used to reduce the complexity of a single MP model. The aim of these methods is to explore the information-tree supplied by the MP algorithm to prune some of its branches. This information-tree contains the trace of the MP analysis, that is, the order in which the LVs and phases have been added to the model, the cross-validated error values, etc. Take the example of Figure 5.2. Each branch represents the addition of a LV (single branches) or a division in two of a sub-model (double branches) in the MP model of a process. Therefore, if a branch is pruned, the complexity of the MP model -in terms of number of LVs and sub-models- is reduced. The pruning algorithms have one single parameter: T_p. This parameter has the same meaning as parameter T in the MP algorithm.

The decision to validate (maintain) or to prune a branch is similar to what it is done in the MP algorithm. A single branch joins a parent node with a child node. It means the parent node represents a model M with a LVs and the child node a model M_1 with $a+1$ LVs. In that cases, the branch is validated if $\delta(M_1, M) > T_p$, where $\delta(M_1, M)$ was previously stored in the information-tree by the MP algorithm. Otherwise, the branch is pruned. A double branch joins a parent node with two child nodes. It means the model M represented by the former was divided in a pair of sub-models by the MP algorithm, yielding M_2. Then, the double branch is validated if $\gamma(\delta(M_2, M)) > T_p$, where the value $\gamma(\delta(M_2, M))$ was stored in the information-tree by the MP algorithm. Otherwise, the double branch is pruned.

There are two possible ways to explore the information tree: from top to bottom and from bottom to top. In the top-bottom pruning, the algorithm starts from the top node of the information-tree. The single or double branch starting from it is checked. If it is validated, the operation is repeated for its child node(s), immediately bellow. This is repeated until all the possible ways down in the information-tree have being explored or pruned. The top-bottom pruning yields exactly the same result as running the MP algorithm for $T = T_p$. The benefit in using the top-down pruning is merely a reduction of computational cost.

If the information tree is explored in a bottom-top fashion, the result is different to that of the MP algorithm. With this postprocessing, important LVs and divisions in sub-models are maintained in the model although they

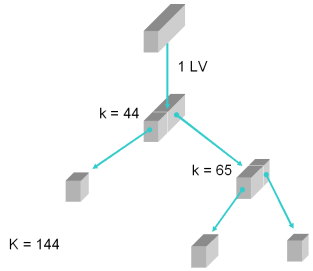


Fig. 5.3. Graphical example of Figure 5.2 post-processed with the bottom-top pruning.

were found after not so important LVs and divisions. To delete a LV or a subdivision -i.e. to join two phases in one- following the bottom-top pruning, they have to fulfil two conditions:

- To be a leaf node of the information tree, i.e. no additional subdivisions or LVs were added by the MP algorithm or, if they were, the bottom-top pruning has already deleted them.
- The branch starting from its parent cannot be validated.

Thus, for instance, if the example of Figure 5.2 corresponds to the input of the bottom-top pruning method, the leaves of the information tree are the subdivision at $k = 31$ and the second LV added to phase $[45, 110]$. Imagine the improvements computed for both the division and the second LV are lower than T_p. In that case, both leaves are deleted reducing the information tree to that of Figure 5.3. Notice that, in that latter tree, the only leaf is the division at sampling time 109 $(44 + 65)$.

Using the pruning methods, the convenient value of T_p in the MP model of a process can be investigated in a computationally efficient way. For that, the MP algorithm is run for a non-restrictive -i.e., a low- T value. Then, the pruning methods are run as many times as desired for several $T_p \geq T$ values.

5.3.2 Postprocessing of a group of Multi-phase models: Merging algorithm

As stated in [3], the MP algorithm is a Greedy approach. This makes it a fast algorithm at the expense of yielding sub-optimal solutions -i.e., sub-optimal MP models. Therefore, to obtain an adequate solution, it is always convenient to make several runs of the MP algorithm for different values of some of its parameters $(a_{ini}, k_\gamma, T, minL)$ or/and different unfolding methods -i.e. different number of LMVs. Once these MP models have been calibrated, the merging algorithm is in charged of mixing them in an unique, optimized MP model.

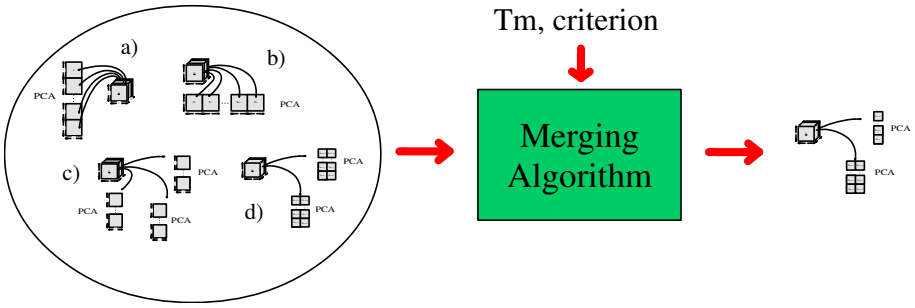


Fig. 5.4. Merging algorithm. Example of use.

An example of use of the merging algorithm is shown in Figure 5.4. The input is composed of four different MP models: a) a variable-wise unfolded model, b) a batch-wise unfolded model, c) a three-phases variable-wise model and d) a two phases, 1 LMV batch dynamic model. The merged MP model obtained in the output is a two phases model, where the first phase is that of model d) whereas the second phase is the third phase of c).

Once again, to reach a compromise between performance and complexity in the merging algorithm, a new threshold T_m is defined. T_m is used in combination with δ and parameter φ, which computes the difference in complexity of two sub-models, according to a predefined criterion to define complexity. In the merging algorithm, sub-model S_1 is considered to be a better solution than S_2 when:

$$\delta(S_1, S_2) > T_m \cdot \varphi(S_1, S_2) \tag{5.5}$$

Thus, according to this equation, the best merged model is yielded using the algorithm presented later on.

The definition of model complexity used to design φ depends very much on the application where it is going to be used. For instance, in many applications, models may benefit of being as parsimonious as possible. To achieve parsimony φ may be defined as follows:

$$\varphi_1(S_1, S_2) = \frac{N_1 - N_2}{max(N_1, N_2)} \tag{5.6}$$

where N_i is the number of parameters of sub-model S_i, which is computed solely with the part of the model applicable to new incoming data. For instance, in PCA the number of parameters is the number of elements in the loadings matrix and in PLS the number of elements in the regressors matrix.

In order to reduce the computational complexity in the merging, the δ values in (5.5) are taken from the information-trees of the MP input models -e.g., Figure 5.2. Therefore, no new cross-validatory computations have to be performed.

The definition of the merging algorithm is specially useful for on-line monitoring and prediction. For those applications, one of the most important parameters to calibrate in the MP models -together with the number of LVs and the intervals of the phases- is the number of LVMs used in the unfolding of the data. The use of the merging algorithm yields MP models where the number of LMVs of different sub-models can be different. This is an important advantage over the approach of [3].

The merging algorithm is defined as follows:

```
Initialize Γ^c with Γ^0
Initialize Γ^t with Γ^0
While Γ^c is not empty
        Initialize M with M^c ϵ Γ^c
        Γ^c = Γ^c ⋂ {M^c}
        For each group of sub-models S^c in M^c (*)
            S = S^c
            For each model M_2^c in Γ^t
                Find a group of sub-models S_2^c for
                  the same interval as S
                If S_2^c exists
                  If δ(S_2^c, S) > T_m · φ(S_2^c, S)
                    S = S_2^c
                  end
                end
            end
            Substitute S^c by S in M
        end
        If M is not in (Γ^t)
          Γ^t = Γ^t ⋃ {M}
          Γ^c = Γ^c ⋃ {M}
        end
end

Initialize M with M^c ϵ Γ^t
For each other model M^c in Γ^t
    If δ(M^c, M) > T_m · φ(M^c, M)
      M = M^c
    end
end

Select M
```

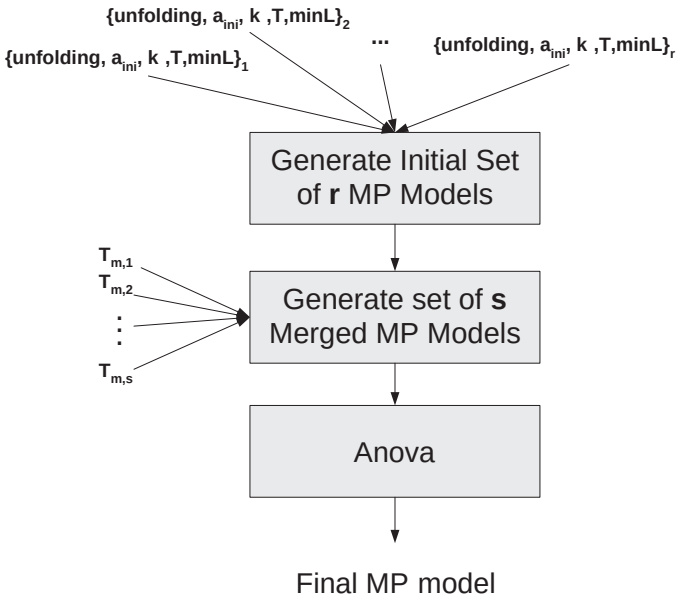


Fig. 5.5. MP analysis framework.

where Γ^0 is the initial set of MP models and Γ^t and Γ^c is the total and the currently used set of MP models in the merging, respectively. To generate a complete set of merged MP models from the initial set, in (*) the algorithm has to start from the single sub-models, continue with combinations of 2 contiguous sub-models, then 3 and so on. The way in which this is accomplished will determine the computational efficiency and completeness of the algorithm.

5.4 Multi-phase Analysis Framework

In this section, a new strategy for modelling and handling batch process data is presented. This strategy, depicted in Figure 5.5, is composed of three steps:

a) First, an initial set of MP models is generated for different values of some MP parameters $(a_{ini}, k_\gamma, T, minL)$ or/and different unfolding methods. The result is an initial group of MP models of enhanced diversity.
b) The second step is devoted to merge the information in the models of the initial group. This can be done according to very different criteria for defining the complexity of a MP model. The criterion should be chosen taking into account the application for which the MP model is being calibrated. The merging algorithm is run for several threshold (T_m) values. The result is a group of merged MP models of different complexity.

c) The third step is to compute an Analysis of Variance (ANOVA) to compare the performance of the models obtained in b) in terms of QPE. For that, the QPE corresponding to each of the calibration batches is computed using the models. The benefit of performing an ANOVA is that a plot with the least significant difference (LSD) intervals can be generated. This plot helps the designer to determine when a simplification of a model does not yield a significant loss of performance. This is a rather useful tool when a compromise between complexity and performance is desired.

5.5 Example of use

Let us obtain a MPPCA model from the data of the Saccharomyces Cerevisiae cultivation process. The parameters used in the first and second steps of the MP framework are listed in Table 5.1. The PRESS computed following the algorithm defined in Chapter 4 is used to identify the proper structure of the model. The MP is run for different values of k_γ in (5.3) and different numbers of LMVs. Once a set of MP models is fitted, these are merged according to different values of T_m and a criterion of minimum number of LMVs. By using this criterion, the aim is to reduce the number of LMVs as much as possible without significant loss of performance.

Table 5.1. Parameters used to fit MPPCA models. M and M_i stand for models of the batch data set. S_i stands for a sub-model of an interval. $\overline{LMV}_i$ stands for the average number of LMVs of sub-model S_i.

Multi-Phase Parameters		
$\delta(M_i, M)$	=	$\frac{PRESS(M)-PRESS(M_i)}{PRESS(M_0)}$
β^k	=	$SSE - SSE^k$
$\gamma(\delta(M_i, M))$	=	$k_\gamma \cdot \delta(M_i, M)$
k_γ	=	[1,3,5]
LMVs	=	[0,1,2,5,10,50]
a_{ini}	=	0
$minL$	=	1
T	=	0.1
Merging Parameters		
T_m	=	$0 \rightarrow 0.5$
$\varphi(S_1, S_2)$	=	$\overline{LMV}_1 - \overline{LMV}_2$

Following the MP framework -Figure 5.5- with the parameters of Table 5.1, five MPPCA models were obtained. An ANOVA from the models was computed, showing statistical significant differences (p-value < 0.05) in the modelling performance of the models. The LSD intervals are shown in Figure 5.6. As it can be seen, there is no statistically significant difference among the first three models ($T_m = 0$, $T_m = 0.02$ and $T_m = 0.08$). The last one

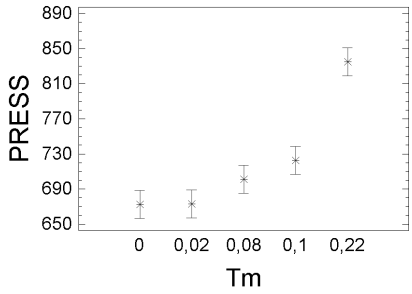


Fig. 5.6. LSD intervals for the PRESS of the merged models with different T_m values. Saccharomyces Cerevisiae cultivation process.

($T_m = 0.22$) clearly presents the poorest performance. From this information, the model suggested is the one for $T_m = 0.08$, because it is more parsimonious than those for $T_m = 0$ and $T_m = 0.02$ with no statistically significant differences in performance.

5.6 Conclusions

In this chapter, the Multi-phase (MP) analysis framework is presented. The MP approach is based on the data-driven identification of the (PCA or PLS) model structure for a batch process. This approach provides the flexibility to adjust this structure to the dynamic nature of the process under study: number of phases, long or short term dynamics, etc... This framework has many advantages:

- The subdivision (low-level) algorithm makes the most of the nature of a batch process, using the time information implicitly.
- The MP algorithm, together with the postprocessing methods, can be used in a straightforward and computationally efficient way to investigate the process under study. The parameters defined for the algorithms allow the designer to handle the structure of the MP model. The appropriate use of the tools within the MP framework helps to improve the process understanding.
- The MP approach allows to obtain a compromise solution between complexity and performance, instead of simply the *"best solution"*.

Part III

Applications.

6 Off-line Monitoring (Post Analysis) of batch processes

Part of the contents of this chapter have been included in the following papers:

[1] J. Camacho and J. Picó. *Monitorización de Procesos por Lotes mediante PCA Multifase*, Revista Iberoamericana de Automatica e Informática Industrial, 3(3):78-91 (2006).

[2] J. Camacho and J. Picó. *Multi-Phase Principal Component Analysis for Batch Processes Modelling*, Chemometrics and Intelligent Laboratory Systems, 81(2):127-136 (2006).

[7] J. Camacho, J. Picó and A. Ferrer. *Multi-Phase Analysis Framework for Handling Batch Process Data*, Submitted to Journal of Chemometrics (2007).

6.1 Introduction

In batch processes, a number of quality measurements are usually collected so that quality constraints imposed in the product can be assessed. Some of these measurements have to be obtained from laboratory analysis. This may take a number of hours after a batch has been processed. Therefore, the quality measurements of a batch may not be available before starting the processing of the following batch. In such cases, an abnormality may affect the product of several batches before being detected. These batches of product sometimes have to be discarded. Moreover, laboratory analysis are commonly invasive and selective, so that only some samples of the batch are assessed and then have to be discarded.

The off-line or end-of-batch monitoring system of a batch process is used to decide whether a batch was produced in-control or out-of-control once its processing has finished. If the monitoring system performs adequately, it can be used to assess the quality of a batch avoiding any delay and destructive analysis. An accurate monitoring system saves time in the detection of production problems [20, 67] and so, it saves money.

This chapter is focused on the end-of-batch monitoring based on Multivariate Statistical Process Monitoring (MSPM). The traditional approach is to develop two complementary monitoring charts -the D-statistic and the Q-statistic charts- from a PCA model fitted from batch-wise unfolded data [45]. The D-statistic is computed from the scores obtained with PCA, whereas the Q-statistic is computed from the residuals. Also, contribution charts are developed to aid in the diagnostic of faults.

After batch-wise unfolding, the resulting number of variables in the unfolded matrix may be very large. Then, the compression performed by PCA is very important in the analysis of the data. Moreover, the resulting set of PCs shows better statistical properties than the original variables. The scores are linear combinations of those variables and so, according to the Central Limit Theorem, they are supposed to be approximately Normal distributed [13]. Also the residuals can be assumed to follow a multi-normal distribution. Therefore, the control limits used to detect faults in the monitoring charts can be computed under the multi-normality assumption of both the scores and residuals. Thus, convenient distributions can be assumed to compute the control limits for the D-statistic [71] and the Q-statistic [69, 70].

Some batch processes show non-linear dynamics in the variation around the average trajectory, which remain in the data after batch-wise unfolding and auto-scaling. Also, the variables collected in separated periods of the batch may be almost -if not completely- independent. When data include independent or non-linearly related groups of variables, a different PLS-based model should be fitted for each of these groups. Non-linear dynamics approximated with a linear combination of variables causes a high prediction and monitoring error in the model of a process. Approximating the non-linearity with a non-linear model can be the solution, but this is a difficult task and

a big and rich data set is needed. When dealing with a huge amount of variables, this is almost impossible. In batch-wise unfolded data, a compromising solution is to model only the dynamics which are well approximated by a linear model. This can be achieved using the tools within the MP framework described in Chapter 5. The MP framework will be applied over batch-wise unfolded data and PCA. Then, from the resulting MPPCA model, the charts of the off-line monitoring system will be generated.

This Chapter is organized as follows: Section 6.2 presents a preliminary study of the modelling performance of MPPCA in batch-wise unfolded data; Section 6.3 shows the generation of the off-line monitoring system based on MPPCA. In Section 6.4, some conclusions are drawn.

6.2 Some preliminary experimentation

A preliminary study of the application of MPPCA to batch-wise unfolded data was presented in [1] and [2]. In these papers, a primitive version of the MPPCA algorithm was used. This algorithm was only able to identify the phases, being the number of PCs a parameter of the algorithm:

i. Unfold the data batch-wise and fit a PCA model M with a_{ini} PCs.

ii. Divide the model M in two sub-models with the low-level routine (Chapter 5), obtaining model M_2. Compute $\beta(M_2, M)$ from:

$$\beta(M_2, M) = \frac{QRE(M) - QRE(M_2)}{QRE(M)} \tag{6.1}$$

iii. If $\beta(M_2, M) > T$, Replace M with M_2 and compute points ii. and iii. recursively for both sub-models.

There are two principal limitations -among others- in this early approach. First, it yields sub-models with the same number of PCs. This feature is not convenient for process modelling, since the number of significant PCs may vary for different phases. Second, the improvement yielded by dividing in two a model is computed from the residuals (QRE) instead of using cross-validation (QPE) as in Chapter 5. The QRE is always reduced when dividing in two a model. Contrarily, the QPE may increase in some cases, showing a division is not convenient at all, and it is more reliable than the QRE.

In [2], the primitive version of MPPCA was compared with other multi-model approaches in terms of modelling performance. The principal aim of

this comparative was to check how appropriate the division in phases performed by MPPCA is. That is to say, the objective was to assess the performance of the low-level routine presented in Chapter 5. This routine is based on a Divide and Conquer Greedy approach, which is fairly fast at the expense of yielding sub-optimal solutions. In spite of the limitations of the version of MPPCA used, this preliminary study is very interesting in order to check whether the low-level routine is well suited for finding the phases of batch-wise unfolded data. Because of that, the study has been included in this section of the Chapter.

6.2.1 Discriminative capability between single-phase and multi-phase processes

A very simple analysis can be performed to detect the single-phase or multi-phase nature of a process. The MPPCA primitive algorithm is run with a_{ini} equal to 1, so that only sub-models with 1 PC are found. With 1 PC it is easier to recognize segments of a batch where the direction of maximum variance is not correctly modelled.

Figure 6.1 shows the result of the MPPCA of the polymerization process with different T (from 0.01 to 0.2 in 0.01 steps) and $minL/K$ values (1/3, 1/4, 1/5, 1/6, 1/7 and 1/8). The prediction power (Figure 6.1(a)) is assessed using the goodness of prediction:

$$Q^2 = 1 - \frac{PRESS_c}{SS}; \tag{6.2}$$

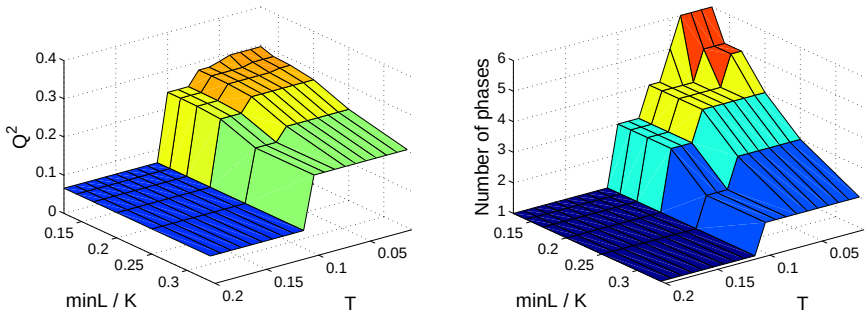
where $PRESS_c$ is the PRESS computed for c PCs and SS is the sum of squares of the data.

Figure 6.1(b) shows the number of phases detected by the algorithm for each pair of the parameters values (T and $minL/K$).

With the pair of figures it is easy to compare the MPPCA batch-wise model with the single-phase batch-wise PCA model. The Q^2 value of the latter can be seen in Figure 6.1(a) for the same parameters for which only 1 phase is found in Figure 6.1(b) -i.e., T values from 0.11 to 0.2. A T equal to 0.06 or 0.07 seems to be convenient for the analysis with MPPCA in this case, because with a lower T some phases included almost do not add prediction power. Also, T equal to 0.08, 0.09 or 0.1 could be a good choice. With a higher T some phases which add prediction power are not found.

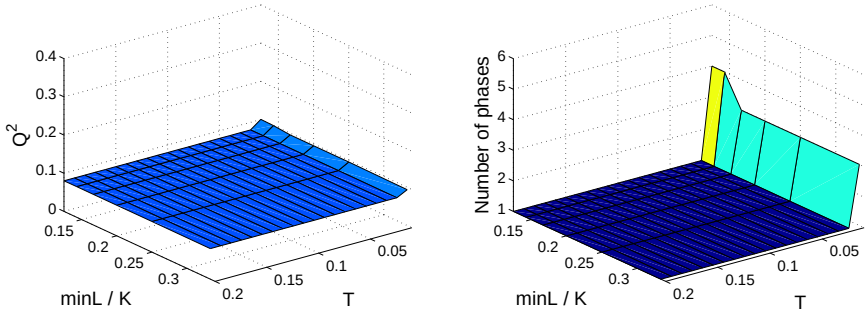
No matter which of these T values is selected, the MP model outperforms the single batch-wise unfolded PCA. For $T = 0.07$, the improvement rises to more than a 320%. For $T = 0.1$, it gets close to a 270%.

In Figure 6.2, results for the etch process are shown. The figures are very different from those commented before. Only for $T = 0.01$ more than a phase is found (Figure 6.2(b)). Moreover, the three or four phases found do not add enough prediction power to be justified (see Figure 6.2(a) for $T = 0.01$). The conclusion is that the etch process is single-phase.



(a) Prediction power (b) Number of phases

Fig. 6.1. MPPCA results with the primitive algorithm of [2] and $a_{ini} = 1$ PC. Nylon 6'6 Polymerization process.

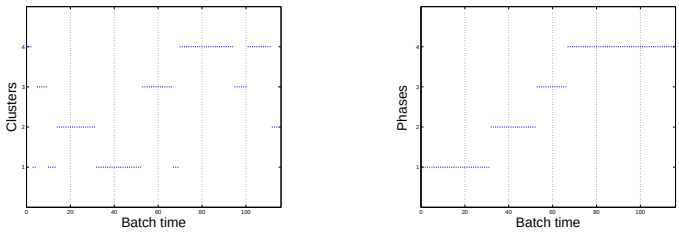


(a) Prediction power (b) Number of phases

Fig. 6.2. MPPCA results with the primitive algorithm of [2] and $a_{ini} = 1$ PC. Etch process.

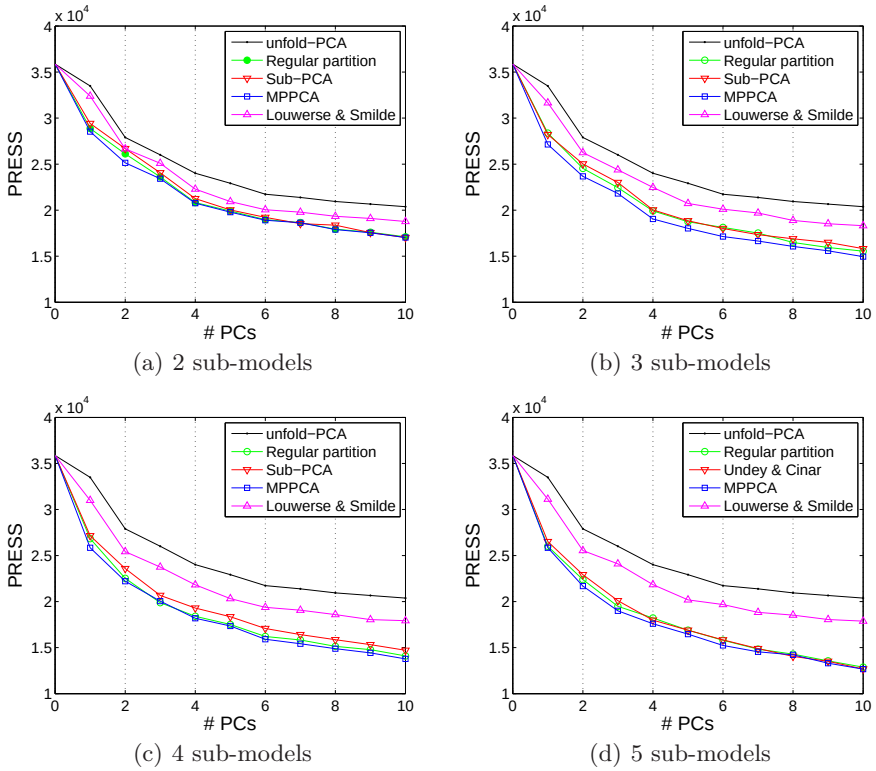
6.2.2 Prediction power

In this section, a comparative study of several division approaches in terms of prediction power is performed. The data set used is that from the polymerization process. Four multi-model methods are compared with the single-phase batch-wise PCA model -for brevity let us call it unfold-PCA or simply uPCA- and MPPCA. The multi-model methods are: splitting the batch in regular intervals, splitting in regular intervals and modelling from the beginning of the batch (evolving modelling), as proposed by Louwerse and Smilde [114],



(a) Sub-PCA output (b) Post-processing

Fig. 6.3. Post-processing of an example of Sub-PCA partition of the polymerization process.



(a) 2 sub-models (b) 3 sub-models

(c) 4 sub-models (d) 5 sub-models

Fig. 6.4. Comparative of multi-model methods. Polymerization process.

splitting according to the known stages of the process, as proposed by Ündey and Çinar [16], and Sub-PCA [54].

Some remarks about Sub-PCA are in due. Sub-PCA was originally proposed for the on-line monitoring of batch processes. It is based on running

a clustering algorithm from the loadings matrices of the local PCA models -remember a local model is fitted exclusively from the data collected at a specific sampling time. Then, clusters are suppose to correspond to intervals of the process where the variables are, more or less, equally related. For this current experimentation, only the partition calculated with Sub-PCA is going to be used, not the modelling method, which is specific for on-line monitoring. Also, since Sub-PCA uses a variation of the k-means and does not include the time information explicitly, the output has to be post-processed to obtain the partition of the process in phases. This is because a cluster may contain sampling times which are not consecutive. Authors do not say anything about how to do this post-processing automatically. Here, the post-processing was achieved by dividing the clusters in phases (where all observations are consecutive), detecting short phases (using the same minimum length parameter for deleting clusters of the k-means algorithm) and merging these with the nearest contiguous phase, using the same distance defined for the k-means. See reference [54] for more details on Sub-PCA. Figure 6.3 shows an example of partition with Sub-PCA and the post-processing effect. Figure 6.3(a) displays the output of Sub-PCA. The clusters include small groups of consecutive sampling times. Figure 6.3(b) shows the output of the post-processing, where there are not phases smaller than the minimum used -equal to 10 sampling times for the example. Note that cluster 1 corresponds to phase 2 and viceversa.

To assess the different multi-model proposals, the multi-models generated with 2, 3, 4 and 5 sub-models have been compared with the uPCA model (Figure 6.4). The minimum length of phase was set to 9 because the shortest stage of the process includes 9 sampling times. T used in MPPCA was set to 0.07. The parameters of Sub-PCA were obtained after extensive experimentation, choosing the best results for 1 PC for each of the first three cases -partition in 2, 3 and 4 sub-models. No partition in 5 sub-models was found with Sub-PCA for a minimum length of phase equal to 9. Therefore, this method is not shown in the last comparative (Figure 6.4(d)). Since the process has 5 stages, the proposal of Ündey and Çinar -partitioning according to the physical stages- only appears in the last comparative.

Again, the comparative shows the multi-phase nature of the polymerization process. The uPCA model gets by far the poorest performance. The proposal of Louwerse and Smilde yields better outcomes, but it is outperformed by the other proposals.

The method of subdividing according to the stages of the batch (Ündey and Çinar) and Sub-PCA are slightly outperformed by the regular partition. Surprisingly, the latter method get really good outcomes. The reason why for such good outcomes may be the following. Imagine a certain interval of a batch process is divided in two, so that a short interval and a long interval are obtained. Then, the improvement in performance expected is low. On one hand, the short interval will be very accurately modelled, but the amount of

predictive error reduced will be low because the interval is short. On the other hand, the long interval will present a similar prediction error than the original one because the former is almost as long as the latter -and thus, almost as imprecise. A priori, the best improvement due to a partition in sub-models is yielded for a regular division. Therefore, a partition has to accurately correspond with the nature of the data to outperform the regular partition.

MPPCA provides the best outcomes in most of the cases. It can be concluded that MPPCA yields the highest prediction power by adjusting the model to the nature of the process.

6.3 Off-line monitoring

To develop the off-line monitoring system of a process, a model of the process is calibrated and from it a group of monitoring charts is designed. Typically, D-statistic and Q-statistic charts (D-charts and Q-charts) are used to detect faults, whereas contribution charts are used to aid in the diagnostic [13]. In the first two charts, control limits at a certain confidence level are developed. Thus, if a batch under monitoring exceeds those limits, the monitoring system generates a fault detection. Commonly, although not the only possibility, these control limits are developed assuming the statistics follow known distributions [69, 70, 71], as commented in Chapter 1. Once a fault has been detected, contribution plots signal the variables related to this fault. Another chart often used is the t-plot, where the distribution of the scores is shown in the latent sub-space.

If the number of charts increases with the number of sub-models, so that a single D-chart and Q-chart is developed for each sub-model, the visual inspection of the process may become challenging when using a multi-model approach. To simplify the task, Ündey and Çinar [16] propose a hierarchical chart system based on a consensus matrix. Once a fault has been detected, the consensus matrix points out which phase is the most related to the fault. Thus, the operator knows in which chart he/she has to look for the fault. Authors also comment this could be fully automated. This approach was proposed to analyze historical data and for end-of-batch monitoring.

Since the division in phases is temporal, the data collected during a phase are available at the end of that phase. Then, the monitoring charts can be updated after each phase has been finished to give early notions of how a batch is evolving. Proceeding that way, nothing is gained by using a hierarchical chart system, since if a fault occurs during a phase it is in the chart of that phase where one should look for it. Therefore, the approach of [16] may be valuable for analyzing historical data, but not for end-of-batch monitoring.

The approach proposed here to reduce the number of charts is to use the same chart to monitor the statistics of the different phases. For instance, if the model has four phases, there will be four values per batch in the D-chart,

four Q-statistic values in the Q-chart and four points in the t-plot. In order to do so, at least in the t-plots, scores belonging to different phases have to be properly scaled so that the control limits of different phases match (see reference [1] for more detail). The result is that the number of charts of a multi-phase model is equal to that for traditional modelling.

The data of the Saccharomyces Cerevisiae cultivation process, the waste-water treatment process and the polymerization of nylon 6'6 will be used in the following experimental study.

6.3.1 Outliers detection

The detection of outliers is an indispensable first step in the PLS-based modelling. If outliers are not eliminated from the data, wrong conclusions can be withdrawn from the analysis. Both the MP algorithm and the PLS-based models can be seriously affected by the presence of outliers. Here we seize the opportunity to shown how to detect outliers in the data with MPPCA.

From experience, it was observed that the structure of the model, as long as it is correctly calibrated, is not very important for outliers detection. Therefore, it is suggested to use predefined -and at some extent arbitrary- values for the parameters of the MP algorithm. In this example, the parameters are set as follows:

Table 6.1. Parameters used to fit MPPCA models for outliers detection. M and M_i stand for models of the batch process. The PRESS is computed with the algorithm proposed in Chapter 4.

Multi-Phase Parameters		
$\delta(M_i, M)$	$=$	$\frac{PRESS(M)-PRESS(M_i)}{PRESS(M_0)}$
β^k	$=$	$SSE - SSE^k$
$\gamma(\delta(M_i, M))$	$=$	$k_\gamma \cdot \delta(M_i, M)$
k_γ	$=$	1
LMVs	$=$	$K - 1$
a_{ini}	$=$	0
$minL$	$=$	1
T	$=$	0.1

Notice that not much importance is being given to the structure of the model. Gross outliers should be detected anyway. This section is only devoted to show an example of outliers detection with MPPCA, but a valid alternative would be to use a single-phase PCA for outliers detection and elimination. Afterwards, MPPCA may be used for the proper modelling of the remaining data.

Only the data sets from the waste-water treatment process and the polymerization of nylon 6'6 were found to present outliers.

Nylon 6'6 polymerization

The MP algorithm presented in Chapter 5 is run for the parameters in Table 6.1. The structure of the resulting MPPCA model is shown in Table 6.2. As it can be seen, not even a single PC has been added to the model. This is because the addition of the first PC reduces the PRESS of the model in less than a 10%, and so the improvement computed in δ does not exceed the threshold imposed in $T = 0.1$. In Figure 6.5, the PRESS computed individually for the batches with the 0 PCs model is shown[a]. A clear outlier, batch #41, can be the cause that no PC is included in the model.

Table 6.2. MPPCA model yielded for parameters in Table 6.1. Nylon6'6 polymerization process. Complete data set.

PRESS	# PCs	Start	End
$4.16 \cdot 10^5$	0	1	116

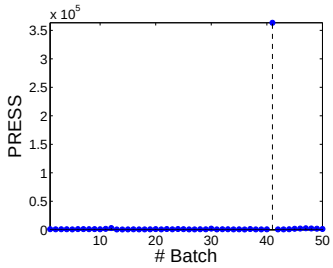


Fig. 6.5. PRESS of the batches in the Nylon6'6 polymerization process. Complete data set.

A MP model is calibrated once again from the data of all batches but #41. Its structure is shown in Table 6.3. Once again, 0 PCs are included in the model. By inspecting the PRESS of the batches, in Figure 6.6, another outlier is found, batch #25. This batch does not present a strange behavior, as it happened with batch #41. It presents a clear deviation from the rest in a single sampling time, probably because of an alignment error. Nonetheless, it should be taken out from the data set for a proper calibration of the MPPCA model or else re-align the whole data set.

[a] To compute the PRESS for 0 PCs, only the preprocessing information is used. In each cross-validatory iteration, one group of batches are separated for the rest. The average and standard deviation is computed from the rest of batches and these are used afterwards to preprocess the separated group. The sum-of-squares of the preprocessed data belonging to each of the batches is the PRESS for 0 PCs.

Table 6.3. MPPCA model yielded for parameters in Table 6.1. Nylon6'6 polymerization process. Data set without batch #41.

PRESS	# PCs	Start	End
$8.32 \cdot 10^5$	0	1	116

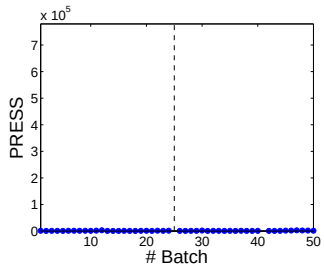


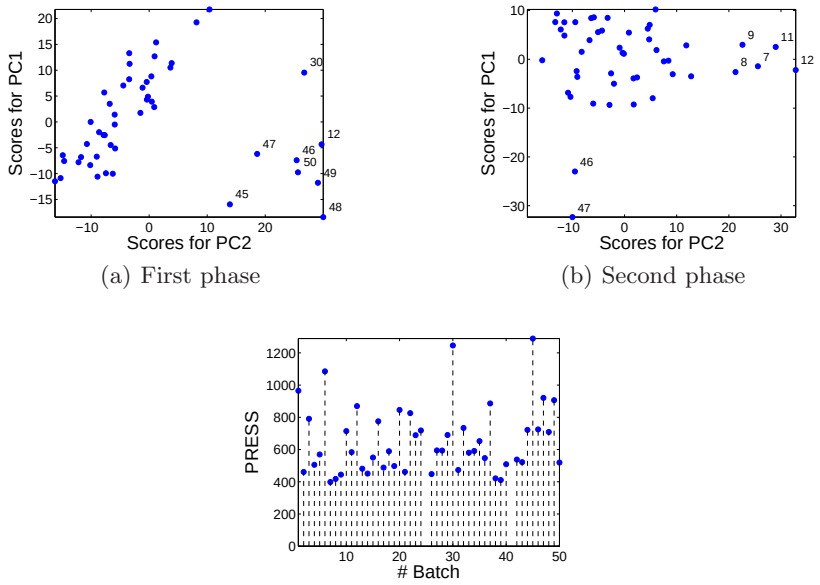
Fig. 6.6. PRESS of the batches in the Nylon6'6 polymerization process. Data set without batch 41#.

Re-calibrating the MPPCA model from the data without batches #41 and #25 yielded a model with several phases and PCs. The MPPCA model obtained is presented in Table 6.4.

Table 6.4. MPPCA model yielded for parameters in Table 6.1. Nylon6'6 polymerization process. Data set without batches #25 and #41.

PRESS	# PCs	Start	End
$2.07 \cdot 10^4$	2	1	68
$1.07 \cdot 10^4$	2	69	116

In Figure 6.7, the distribution of the scores in the PCs sub-space of each sub-model, along with the PRESS values of the batches, are shown. In Figure 6.7(a), a group of 8 batches is located separately from the rest. These are batches #12, #30 and from #45 to #50. In this case, because of the number of batches and their similar behavior, these batches should not be simply eliminated. The experts of the plant may be able to determine if these batches present a common cause of failure or a different mode of normal operation. In the latter case, they should not be discarded, but modelled separately so that batches with similar behavior are not treated as faulty batches by the monitoring system. In Figure 6.7(a), the second PC is hardly modelling useful information for the monitoring, since it is capturing the inter-clusters variability between the two clusters of data. To avoid this, it is necessary to model both clusters separately to perform the analysis. For the sake of simplicity, it will be assumed that the deviation of the 8 batches is a fault of the process and so the batches are simply eliminated from the data.



(a) First phase

(b) Second phase

(c) Residuals

Fig. 6.7. Batches in the Nylon6'6 polymerization process. Data set without batches #25 and #41.

Some additional batches are suspicious of being outliers. It is the case of batches #7, #8, #9 and #11 in the second phase. Again, a deeper analysis or the process knowledge would determine if they are faulty batches or not. Let us assume again they are faulty batches and so they are simply discarded. Re-calibrating the model for the last time gave the model structure of Table 6.5. No additional outlier was found.

Table 6.5. MPPCA model yielded for parameters in Table 6.1. Nylon6'6 polymerization process. Data set without outliers.

PRESS	# PCs	Start	End
$3.06 \cdot 10^4$	2	1	116

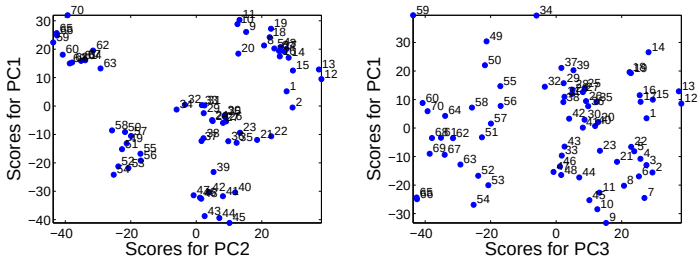
Very similar findings of those presented here were reported by [22]. Notice that we are not giving much importance to the structure of the model. Gross outliers should be determined anyway.

Waste-water treatment

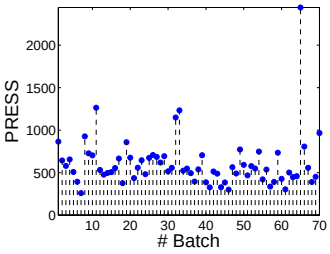
The structure of the MPPCA model obtained for the waste-water treatment process is shown in Table 6.6. One single phase is found with 3 PCs.

Table 6.6. MPPCA model yielded for parameters in Table 6.1. Waste-water treatment process. Complete data set.

PRESS	# PCs	Start	End
$4.25 \cdot 10^4$	3	1	340



(a) Bi-plots



(b) Residuals

Fig. 6.8. Batches in the waste-water treatment process. Complete data set.

In Figure 6.8, two bi-plots of the scores of the model together with the PRESS for each batch, are shown. The batches show a clustered distribution in the latent variables sub-space. This seems to be a consequence of the inter-batch dynamics in the process. An adaptive model is suggested for the proper long-term monitoring of this process. On the other hand, in Figure 6.8(c), batch #65 presents a high PRESS value. This batch was discarded and the model was re-calibrated. The structure of this latter model is presented in Table 6.7. No additional outliers were found.

Table 6.7. MPPCA model yielded for parameters in Table 6.1. Waste-water treatment process. Data set without outliers.

PRESS	# PCs	Start	End
$4.20 \cdot 10^4$	3	1	340

6.3.2 Models generation

Once the outliers have been eliminated, the MP framework can be applied to develop the monitoring system. As it is customary in off-line monitoring [45, 31], data are batch-wise unfolded and PCA is used for modelling. The parameters used in the first and second steps of the MP framework (Section 5.4) are listed in Table 6.8. The PRESS computed with the algorithm of Chapter 4 is used in the definition of δ as the QPE index in the MP algorithm (Section 5.2). The MP algorithm is run three times for different values of k_γ (5.3) to generate the initial set of MP models. Afterwards, the MPPCA models are merged according to the criterion of maximum parsimony and different values of T_m. The last step is to perform an ANOVA from the merged models. Nonetheless, for the cases of study treated, the ANOVA step was not necessary since the merging step returned only one model, no matter the value of T_m used.

Table 6.8. Parameters used to fit MPPCA models for end-of-batch monitoring. M and M_i stand for models of the batch data set. S_i stands for a sub-model of an interval. N_i is the number of parameters of sub-model S_i.

Multi-Phase Parameters		
$\delta(M_i, M)$	$=$	$\frac{PRESS(M)-PRESS(M_i)}{PRESS(M_0)}$
β^k	$=$	$SSE - SSE^k$
$\gamma(\delta(M_i, M))$	$=$	$k_\gamma \cdot \delta(M_i, M)$
k_γ	$=$	[1,3,5]
LMVs	$=$	$K - 1$
n	$=$	0
$minL$	$=$	1
T	$=$	0.1
Merging Parameters		
T_m	$=$	$0 \rightarrow 0.5$
$\varphi(S_1, S_2)$	$=$	$\frac{N_1-N_2}{max(N_1,N_2)}$

Both the data from the Saccharomyces Cerevisiae cultivation process and from the waste-water treatment process were found to be single-phase when batch-wise unfolded. Even in the case of $k_\gamma = 5$ -which is more likely to add phases to the model than for $k_\gamma = 3$ and $k_\gamma = 1$-, the MP algorithm did not divide the model in more than one phase. The Nylon 6'6 polymerization process showed a different result. For $k_\gamma = 1$, one single-phase batch-wise

model was obtained -the model in Table 6.5. Nevertheless, both for $k_\gamma = 3$ and $k_\gamma = 5$, a four-phases batch-wise model was returned. From the merging algorithm it was seen that the model made up of four phases captures better the nature of the data than the single-phase model, being at the same time more parsimonious. Therefore, this is the most convenient model and the ANOVA step is not necessary. The structure of the four phases model is presented in Table 6.9.

Table 6.9. MPPCA model obtained following the MP framework strategy with the parameters of Table 6.8.

PRESS	# PCs	Start	End
$8.60 \cdot 10^3$	1	1	31
$8.42 \cdot 10^3$	1	32	68
$5.91 \cdot 10^3$	1	69	99
$3.26 \cdot 10^3$	1	100	116

6.3.3 Monitoring charts

The statistics used in this chapter are the Q-statistic, which compress the residuals of a batch throughout its processing, and the D-statistic, computed from the scores of the data of a complete batch. With the statistics computed from the calibration batches, control limits at a certain confidence level can be established. A common practice is to develop the limits under the assumption that the statistics follow known distributions, assumption which should be confirmed from the direct observation. Afterwards, new data are monitored against these limits. Thus, if the limits are exceeded by the statistics computed for a new batch, it is catalogued as abnormal.

Both the D-statistic and the Q-statistic for batch i can be computed from the following equations:

$$D_i = \sum_{a=1}^{A} \left(\frac{t_{ai} - \mu_{\mathbf{t}_a}}{\sigma_{\mathbf{t}_a}}\right)^2 \tag{6.3}$$

$$Q_i = \sum_{j=1}^{J} \sum_{k=1}^{K} (e_{ijk})^2 \tag{6.4}$$

where t_{ai} represents the score of the batch in the a-th component, $\mu_{\mathbf{t}_a}$ and $\sigma_{\mathbf{t}_a}$ stand for the mean and the standard deviation of the scores of that component in the calibration data, respectively, and e_{ijk} represents the residual value corresponding to the variable j and sampling time k. The development of control limits for these statistics according to [71] and [70] can be found in Chapter 1.

The only data set which was found to present several phases, according to the MP framework, was that from the polymerization process. This case of study will be used here to compare the traditional off-line monitoring system made up from one single model with that generated with the MPPCA model. Five NOC and five abnormal batches are used to test the monitoring systems.

Table 6.10. Models and performance statistics for the off-line monitoring of the Nylon 6'6 polymerization process: batch-wise (BW) and multi-phase batch-wise (MP-BW) approaches. The number between parenthesis by the NTI (Number of Type I errors) and NTII (Number of Type II errors) is the total number of NOC and abnormal batches tested, respectively. D stands for the D-statistic chart, Q stands for the Q-statistic chart and #MC for the number of miss-classifications.

Model	Structure	PRESS-Ratio	Parameters
MP-BW	4 phases, 1 PC each	**0.93**	**1044**
BW	2 PCs	**0.88**	**2088**

Model	NTI (5 batches)			NTII (5 batches)		
	D	Q	#MC	D	Q	#MC
MP-BW	0	0	**0**	3	0	**0**
BW	0	0	**0**	3	0	**0**

In Table 6.10, both the single-phase and the MPPCA 4-phases batch-wise models are compared. The upper part of the table is devoted to show information of the models. The lower part studies their performance in off-line monitoring. From the models, the structure (number of phases and PCs), the PRESS-ratio -computed dividing the average PRESS of the NOC test data set with that of the calibration data set- and the number of parameters in the loadings matrices of the models are presented. The rate of PRESS is useful to decide whether a model is over-fitted or not. Usually, over-fitted models will tend to inflate the average PRESS of the NOC test set. As long as this

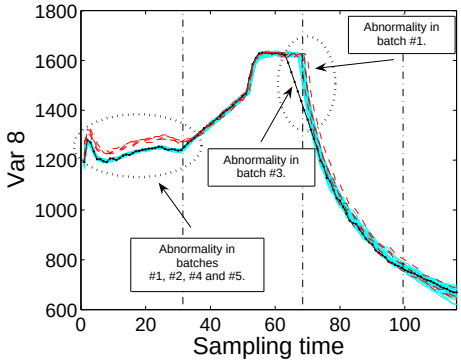
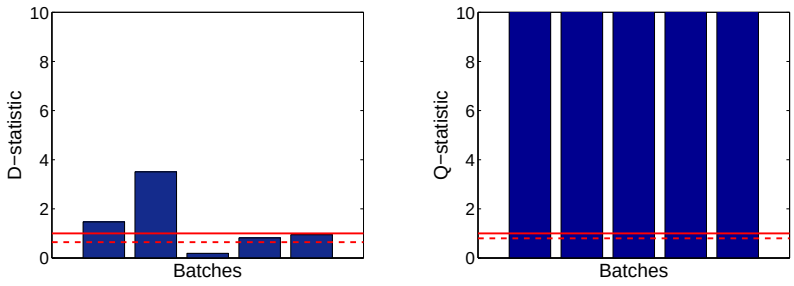


Fig. 6.9. Trajectories for variable 8 in the Nylon 6'6 Polymerization process. The abnormal batches are compared with the calibration data (solid light lines). The first, second, fourth and fifth abnormal batches in the test set are shown with dashed lines. The third abnormal batch is shown with a line with dots. The vertical dashdot lines are the divisions obtained with the MP algorithm.

index is close to 1, models are supposed to be conveniently calibrated. The number of parameters shows that the MP model is more parsimonious than the single-phase one, as it was stated before. The monitoring performance indices shown are the Number of Type I errors (NTI), understood as the number of in-control batches classified as out-of-control, and the Number of Type II errors (NTII), understood as the number of out-of-control batches classified as in-control. Notice that the D-statistic and the Q-statistic charts are complementary. Therefore, to determine if a batch is out-of-control it is enough that it exceeds the limits in any of the charts. As it can be seen, a clear difference in the performance of the models was not found. In both cases, NOC and abnormal batches are correctly identified (number of miss-classifications $\#MC = 0$).

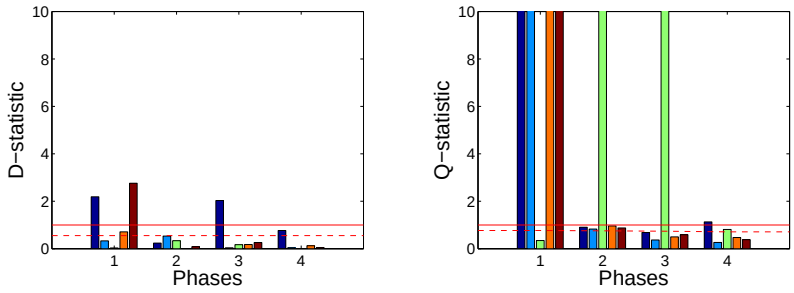
In Figure 6.9, the trajectories of variable #8 for the abnormal and the calibration batches are compared. This variable was chosen since all the abnormalities presented in the test set are noticeable from its view. Also, the division of the sampling time mode obtained with the MP algorithm is shown in the figure. Abnormal batches #1, #2, #4 and #5 present a similar abnormality during the first phase of the batch. Abnormal batch #3 presents an abnormal behavior in the second half of the batch duration.

In Figure 6.10, the D-statistic and Q-statistic charts with the statistics of the five abnormal batches using the single-phase batch-wise model are shown. As it was already commented, the five batches are correctly classified as out-of-control. Nonetheless, from the monitoring charts, there is no evidence of the similarity in the abnormalities in batches #1, #2, #4 and #5. It is necessary to study the contribution of the variables [72, 73] to notice this



(a) (b)

Fig. 6.10. D-statistic chart (a) and Q-statistic chart (b) computed for the batch-wise (single-phase) model from the data of the Nylon 6'6 polymerization process. Off-line monitoring. The batches plotted are the five abnormal batches included in the test set.



(a) (b)

Fig. 6.11. D-statistic chart (a) and Q-statistic chart (b) computed for the MPPCA model from the data of the Nylon 6'6 polymerization process. Off-line monitoring. The batches plotted are the five abnormal batches included in the test set.

similarity and to realize batch #3 presents a different type of fault (plot not shown).

Let us have a look now at Figure 6.11, where the monitoring charts of the MPPCA model are shown. Since the sampling time mode is partitioned in four phases, these charts offer more information regarding the time when a fault occurred than those for the single-phase model. In the Q-statistic chart, batches #1, #2, #4 and #5 behave abnormally during the first phase. After that they evolve under NOC, except for the case of batch #1 which also presents a fault in the D-statistic for the third phase. All this matches with the behavior shown in Figure 6.9. Notice from that figure that the batches #1, #2, #4 and #5 follow separated trajectories from the rest during the

first phase. Additionally one of them (#1) starts the falling trend during the third phase later than the rest. On the other hand, batch #3 presents its fault in the second and third phases, and this is also seen in the Q-statistic chart. Therefore, only by looking at both charts in Figure 6.11, information regarding the time when a fault occurs is obtained and also some typical faults may be identified.

These advantages observed are the consequence of dividing the process in several sub-models. This may lead to the conclusion that using several sub-models is always convenient. Nonetheless, data sets in which the intervals are linearly correlated should not be split up, since the division causes a loss of information in the models. This information, in turn, may be important for the monitoring. Therefore, a mechanism to evaluate when a model should be partitioned in a number of sub-models is necessary. On the other hand, for some processes, the number of sub-models identified may be high. In those cases, charts may get fairly complex to interpret because of the number of values shown for every single batch. A solution is to define the complexity of a model in terms of the number of phases -instead of using parsimony- and use the merging algorithm according to that criterion. By doing so and then studying the results with the ANOVA, it is possible to determine if such a high number of phases is very necessary or it can be reduced with almost no loss of performance.

6.4 Conclusions

In this chapter, the generation of an off-line or end-of-batch monitoring system for a batch process based on batch-wise MPPCA has been studied. This study goes from the investigation of the modelling performance of MP models to the performance of the monitoring system developed from MP models.

In a preliminary study, the low-level routine of the MP algorithm was tested. This routine is devoted to find the appropriate partition in phases of a batch process. From the results, it was found that the MP based on that routine outperforms all the other multi-model approaches investigated. Moreover, the utility of the threshold parameter T in the MP algorithm to distinguish single-phase from multi-phase processes was proven.

After this preliminary study, the generation of an off-line monitoring system from the MP approach was studied. The MP framework provides a complete toolkit for analyzing processes, including the elimination of outliers, the detection of the phases and the analysis of the impact of model simplifications in the modelling quality. Also, it was observed that the MPPCA approach yields more informative monitoring charts than those of single-phase PCA, in which some information regarding the time when a fault occurs is provided. This, in turn, makes possible the recognition of common failure patterns directly on the charts.

7 On-line Monitoring of batch processes

Part of the contents of this chapter have been included in the following papers:

[3] J. Camacho and J. Picó. *Online Monitoring of Batch Processes using Multi-Phase Principal Component Analysis*, Journal of Process Control, 10(16):1021-1035 (2006).

[7] J. Camacho, J. Picó and A. Ferrer. *Multi-Phase Analysis Framework for Handling Batch Process Data*, Submitted to Journal of Chemometrics (2007).

7.1 Introduction

The monitoring system is especially useful if it is able to monitor a batch not only at the end of the processing, but during its processing. The online monitoring procedure of [13] is based on the same unfolding method of end-of-batch monitoring, the batch-wise unfolding. Thus, the data of a complete batch are needed to compute the monitoring statistics. This makes necessary to perform some kind of predictions or assumptions to estimate the monitoring statistics online. The quality of these approximations is not supposed to affect the performance of the monitoring system because both the statistics used to develop the control limits and those for new batches are calculated in the same way [3, 77]. The model obtained after batch-wise unfolding has the nice feature of capturing the dynamics of the process. Nonetheless, it has been shown that these dynamics can be captured by including only a few lagged variables [47, 2].

In [31], Wold et *al.* developed a very complete monitoring framework. In a first step, they perform an alternative unfolding, the variable-wise unfolding. Nevertheless, some researchers have claimed that the proposed online monitoring system has poor detection capability when a multi-phase process is modelled with a single model [77]. Batch Dynamic PCA (BDPCA) [47] is based on the variable-wise unfolding plus the addition of LMVs.

An alternative strategy is based on generating a model for every sampling time of the batch duration -i.e. the K-models approach. If each model includes only the data of a sampling time, then it is called a local model [49]. If each model incorporates the measurements from the beginning of the batch to a sampling time, then it is called an evolving model [50, 49]. Hierarchical PCA [48] follows the latter method but giving different weight to the current measurements of the variables. This is done by calculating the scores in a hierarchical fashion. Finally, if only the immediate part of the past measurements is included in each model together with the current measurements, the procedure is called Moving Window PCA (MWPCA) [51]. Time-Varying State Space Modelling (TVSS) [76] applies the same latter philosophy in the generation of a state space model, instead of a PCA model.

In this chapter, an online monitoring system based on the MP framework is presented. As in the previous chapter, the work presented here corresponds to several years of study, during which the MP approach has evolved. A preliminary study was performed in [3]. Some of the experiments and conclusions of that investigation are presented here. Please, refer to the cited paper for more detail.

This chapter is organized as follows: Section 7.2 presents some preliminary studies, which justify the approach of the chapter. Section 7.3 shows the application of the MP framework to the monitoring of three case studies. In Section 7.4, the results are discussed.

7.2 Some preliminary results

The on-line monitoring of batch processes using PCA models commonly relies in two control charts built from the D-statistic and the SPE [13]:

$$D_{i,k} = \sum_{a=1}^{A} \left(\frac{t_{aik} - \mu_{\mathbf{t}_{ak}}}{\sigma_{\mathbf{t}_{ak}}}\right)^2 \tag{7.1}$$

$$SPE_{i,k} = \sum_{j=1}^{J} (e_{ijk})^2 \tag{7.2}$$

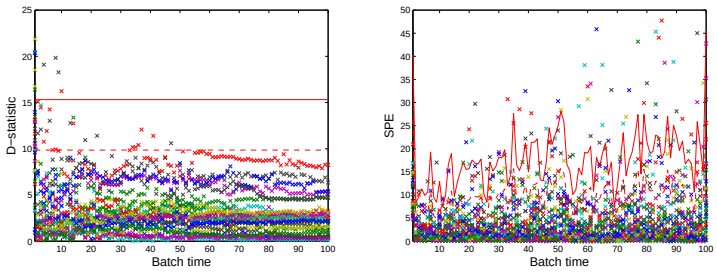
where t_{aik} represents the score of the batch in the a-th component obtained at sampling time k, $\mu_{\mathbf{t}_{ak}}$ and $\sigma_{\mathbf{t}_{ak}}$ stand for the mean and the standard deviation of the scores of that component and sampling time in the calibration data, respectively, and e_{ijk} represents the residual value corresponding to the variable j and sampling time k.

Commonly, control limits for both charts are computed according to a specific imposed significance level (ISL) using statistical distributions. Here, the D-statistic is assumed to be F-distributed [71] and the limits of the SPE are computed following the approach of [70]. The development of control limits can be found in Chapter 1. A fault in an on-line monitoring chart is commonly signaled when a run chart of three or more consecutive values of the statistic exceeds the control limit.

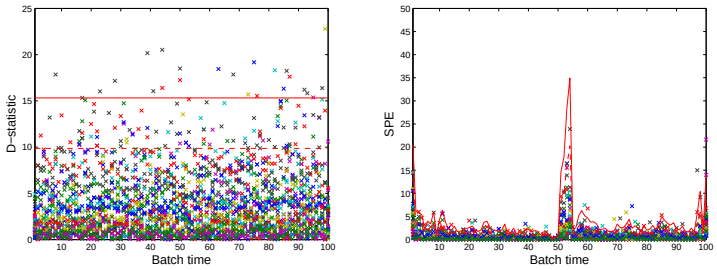
In this section, some of the effects in the performance of the monitoring charts caused by the modelling structure are analyzed. The principal modelling structures included in the experiments are the batch-wise ($\underline{\mathbf{X}}^{(K-1)}$), the variable-wise ($\underline{\mathbf{X}}^{(0)}$), batch dynamic ($\underline{\mathbf{X}}^{(n)}$ for $n = \{k-1 \quad : k \in \{1, 2, .., K\}\}$), local ($\{\mathbf{X}_k : k = 1, ..., K\}$) and evolving ($\{\underline{\mathbf{X}}_{1:k}^{(k-1)} : k = 1, ..., K\}$). This study is not intended to be exhaustive, but illustrative of the possible consequences of using one approach or another. In all the cases, the average trajectory is subtracted and data are scaled to unit variance around it. In order to apply the batch-wise model in on-line monitoring, the approaches for imputing future observations used are zero deviations, current deviations and Projection to Model Plane (PMP) [13].

7.2.1 On the auto-correlation of the statistics and the modelling structure

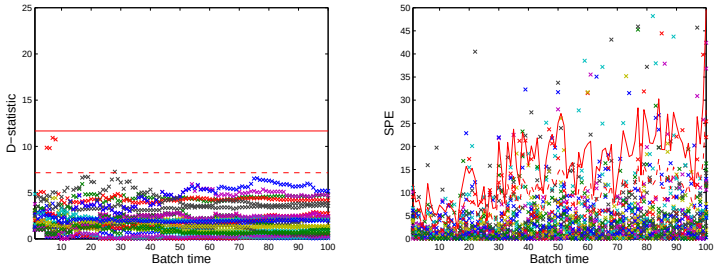
In Figure 7.1, the distribution of the D-statistic and SPE values in the Saccharomyces Cerevisiae cultivation process, for a batch-wise PCA model, a variable-wise PCA model, evolving and local PCA models are shown. Let us focus on the first column of the figure, where the D-statistic charts are shown. For batch-wise and evolving modelling the D-statistics seem to be very auto-correlated, whereas for variable-wise and local models they do not. The



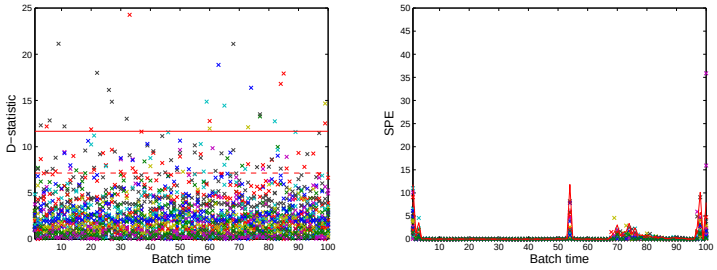
(a)



(b)



(c)



(d)

Fig. 7.1. D-statistic and SPE control limits and distributions: (a) batch-wise PCA model with current deviations, (b) variable-wise PCA model, (c) evolving PCA models, (d) local PCA models. Control limits for a 95% confidence level are represented by a dashed line and for a 99% confidence level by a solid line.

correlograms of the D-statistic for the four modelling approaches corresponding to the same batch are shown in Figure 7.2. This figure confirms that for batch-wise and evolving modelling the D-statistics are auto-correlated. For variable-wise and local models the null hypothesis of non auto-correlation cannot be rejected. The auto-correlation of the D-statistic in batch-wise and evolving models is caused by the use of all the measurements from the beginning of the batch to a specific sampling time to compute the D-statistic of that sampling time. The autocorrelation can affect the overall type I risk (OTI) defined in Section 7.2.2 and it leads to the over-estimation of the control limits as shown in the D-charts in Figures 7.1(a) and 7.1(c). Also, some of the NOC batches may exceed those limits for several consecutive sampling times. In Figures 7.1(b) and 7.1(d), the D-statistics are not auto-correlated. The consequence is that the controls limits are better adjusted. Also, consecutive sampling times exceeding the limits for the same NOC batch are unlikely to appear.

In [47], the batch dynamic approach is presented. Authors use the data of the Nylon 6'6 polymerization process in that article. There, the three-way matrix of data was unfolded with 8 LMVs, i.e. according to $\underline{\mathbf{X}}^{(8)}$. In Figure

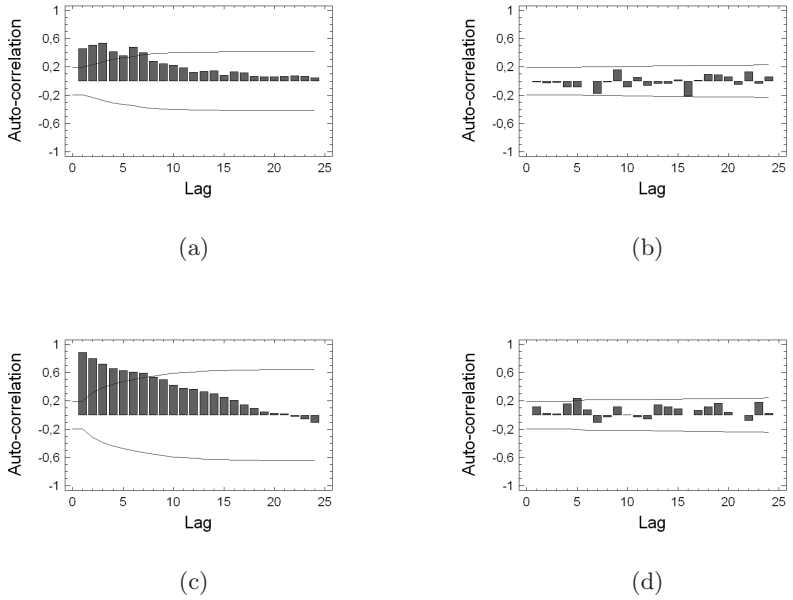


Fig. 7.2. Correlograms of the D-statistics of a batch, 95% confidence level: (a) batch-wise PCA model with current deviations, (b) variable-wise PCA model, (c) evolving PCA models, (d) local PCA models.

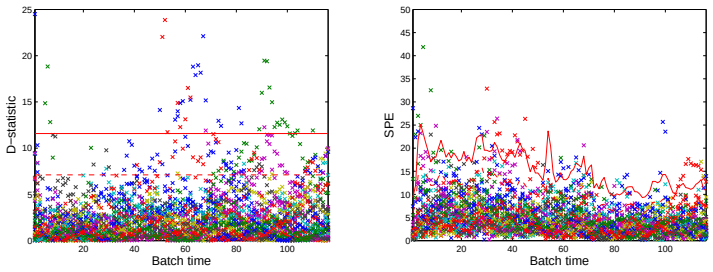
7.3 it is shown how the inclusion of such a number of LMVs causes the auto-correlation of the D-statistic and the SPE. The trajectories of the statistics of Figure 7.3(b) are much smoother than those of Figure 7.3(a). The number of lags for which the null hypothesis of non auto-correlation is rejected is increased with the addition of LMVs, as shown in Figure 7.4 for the SPE.

Now let us investigate how the monitoring of a number of batches is carried out by different modelling structures. In Figure 7.5, the SPE monitoring charts developed for the Nylon 6'6 polymerization process, using a variable-wise PCA model (Figure 7.5(a)) and a batch-wise PCA model with missing values (Figure 7.5(b)), are shown[a]. A NOC batch and an abnormal batch are being monitored. As it can be seen, both models make clear the difference between the behavior of the two batches. As a matter of fact, the SPE of the abnormal batch reaches to values of order 10^4.

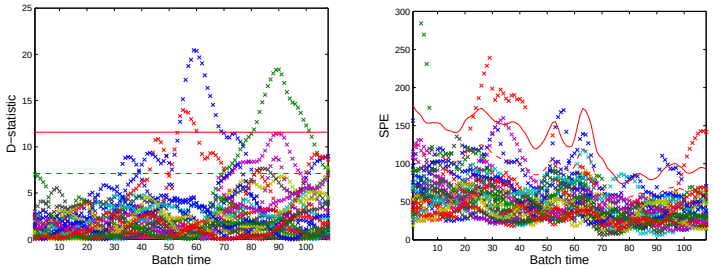
In Figures 7.6(a) and 7.6(b), the trajectories of variables 6 and 8, respectively, of the calibration batches and the abnormal batch monitored in Figure 7.5 are compared. Those variables were selected from the study of the contributions of the variables to the deviation of the abnormal batch from NOC. In both figures, the deviation goes from the beginning of the batch to sampling time 34. Batch-wise models and evolving models tend to prolong the fault detection for several sampling times after the real disturbance has finished. This fact can be clearly seen by comparing Figures 7.5(b) and 7.6. The same behavior was observed for similar abnormal batches. A theoretical explanation of this phenomenon is offered in Appendix C. The variable-wise model, BDPCA and local models were very precise to detect the end of the fault.

In Figure 7.7, the SPE monitoring charts for a variable-wise PCA model (Figure 7.7(a)) and a BDPCA model (Figure 7.7(b)), are shown. Once again, the statistics for one NOC and one abnormal batches are shown. The abnormal batch was used in [47] to compare the monitoring performance of BDPCA with that of a batch-wise approach based on PMP. The outcomes presented here are coherent with those in that paper, with the difference that the batches were aligned with a total duration of 100 sampling times in [47]. The consequence of adding 8 LMVs is the smoothing of both the control limits and the statistics computed for a batch, as it can be seen by comparing Figures 7.7(a) and 7.7(b). Additionally, the inclusion of LMVs leads to detect a higher number of consecutive values exceeding the control limits (as it is claimed in [47]), but it has the drawback that the detection may be delayed. Moreover, this increment in the number of consecutive faults distorts the length of the deviation observed in the charts. Figures 7.8(a) and 7.8(b) show the trajectories of variables 4 and 7 for the calibration data set and the abnormal batch. Those variables were pointed out as those which contribute more to the deviation of the abnormal batch. In both figures, the interval

[a] Only the SPE chart is used here for the sake of simplicity. Nonetheless, for monitoring purposes both the D-statistic and the SPE have to be used together.

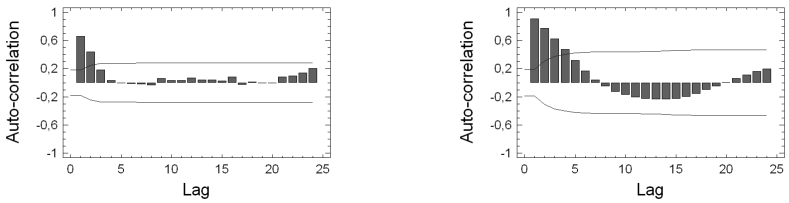


(a)



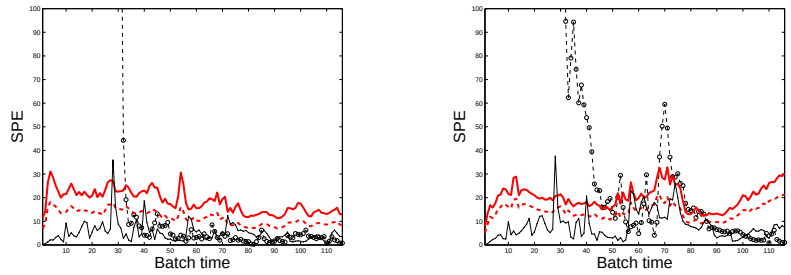
(b)

Fig. 7.3. D-statistic and SPE control limits and distributions: (a) variable-wise PCA model, (b) BDPCA model with 8 LMVs. Control limits for a 95% confidence level are represented by a dashed line and for a 99% confidence level by a solid line.



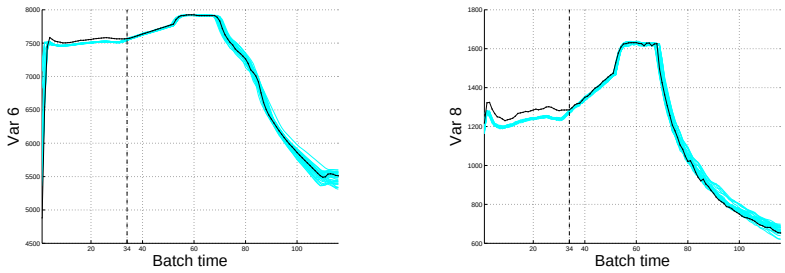
(a) (b)

Fig. 7.4. Correlograms of the SPE of a batch, 95% confidence level: (a) variable-wise PCA model, (b) BDPCA model with 8 LMVs.



(a) (b)

Fig. 7.5. SPE charts: (a) variable-wise PCA model, (b) batch-wise PCA model with missing values. The thick lines are the control limits: 95% (dashed line) and 99% (solid line) confidence levels. The thin lines show the statistics for an abnormal batch (dashed circle line) and a NOC batch (solid dot line).

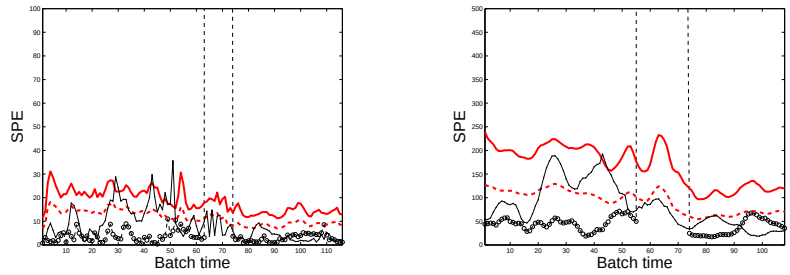


(a) (b)

Fig. 7.6. Variables 6 (a) and 8 (b) of the process. The light lines represent the batches of the calibration set. The dark dot line represents the abnormal batch.

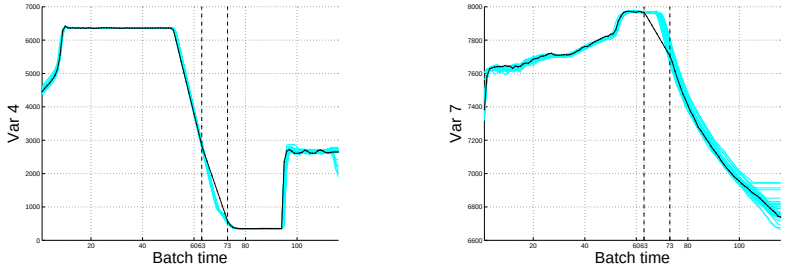
where the latter batch shows an abnormal behavior is approximately from sampling time 63 to 73. The variable-wise model detected the fault from 64 to 73 whereas the BDPCA model detects it from 64 to 82[b]. Imagine a correction has been performed in order to lead the batch back to NOC. Then, the monitoring system should be able to detect the batch has come back to NOC as soon as that happens, so that the special correction is stopped. Therefore, this delay in the BDPCA approach should not be seen as an advantage, but

[b] Notice in the BDPCA monitoring chart, the statistic at time k corresponds to the sampling time $k + 8$, since 8 LMVs are used. Therefore, since the fault is signaled in the chart at time 56, the fault is supposed to start at sampling time 56+8=64.



(a) (b)

Fig. 7.7. SPE charts: (a) variable-wise PCA model, (b) BDPCA model. The thick lines are the control limits: 95% (dashed line) and 99% (solid line) confidence levels. The thin lines show the statistics for an abnormal batch (dashed circle line) and a NOC batch (solid dot line).



(a) (b)

Fig. 7.8. Variables 4 (a) and 7 (b) of the process. The light lines represent the batches of the calibration set. The dark dot lines represent the abnormal batch.

as a drawback. As a conclusion, care should be taken in order not to include more LMVs than necessary.

7.2.2 On the adjustment of control limits and the modelling structure

Control limits computed from statistical distributions

As commented before, control limits in the monitoring charts are commonly developed according to known distributions [71, 70]. To assess the validity of control limits, the Overall Type I risk (OTI) can be computed from the

data. Following the definition in [13], the OTI is the percentage of faults in the NOC calibration batches:

$$OTI = 100 \cdot \frac{nf}{I \cdot K}\% \tag{7.3}$$

where nf is the total number of faults (i.e., single sampling times where the statistic computed for a batch crosses the limit) in the NOC calibration data. Ideally, the OTI should match the ISL value.

The OTI computed for different modelling structures in the Saccharomyces Cerevisiae cultivation process is shown in Table 7.1. The values of OTI are obtained for 95% and 99% confidence limits (ILS equal to 0.05 and 0.01, respectively), both for the D-statistic and the SPE. The modelling structures included are batch-wise, variable-wise, batch dynamic with 1 LMV ($\underline{\mathbf{X}}^{(1)}$), local, evolving and a 2-phases MP model from variable-wise unfolded data ($\{\underline{\mathbf{X}}^{(0)}_{1:54}, \underline{\mathbf{X}}^{(0)}_{55:100}\}$).

Table 7.1. Overall type I risk (OTI) of the control limits for the D-statistic and the SPE charts in the *Saccharomyces cerevisiae* cultivation process. Modelling approaches are batch-wise modelling plus zero deviations (BW-ZD), current deviations (BW-CD) and missing values (BW-MV), evolving modelling, local modelling, variable-wise modelling (VW), batch dynamic (BD) with 1 LMV and Multi-phase from variable-wise unfolding (MP-VW).

OTI	D-st	SPE	D-st	SPE
ISL	0.0500	0.0500	0.0100	0.0100
BW-ZD	0.0052	0.0714	0.0010	0.0224
BW-CD	0.0141	0.0848	0.0041	0.0300
BW-MV	0.0052	0.1041	0.0010	0.0386
Evolving	0.0017	0.0966	0.0000	0.0341
Local	0.0400	0.0955	0.0097	0.0466
VW	0.0455	0.0634	0.0100	0.0138
BD(1 LMV)	0.0411	0.0603	0.0122	0.0132
MP-VW	0.0355	0.0690	0.0066	0.0176

From this Table, the differences due the model structure are clear. The results for evolving models are very similar to those for batch-wise models. It can be concluded, then, that the number of sub-models needed to create the evolving monitoring system is not justified. D-statistic charts based on batch-wise and evolving models present very low OTI values. This is a consequence of the overestimation of the control limits caused, as discussed before, by the auto-correlation of the D-statistics. Moreover, this makes the D-chart almost useless [75], being all the detection capabilities centered in the SPE. Contrarily, the SPE of those models presents an OTI much higher than the ISL. The SPE of local models also presents an OTI much higher than the

ISL. Batch-wise, local and evolving models need of a huge number of parameters to be fitted. The high OTI in the SPE can be seen as a consequence of this over-parametrization. Because of the number of parameters, part of the information included in the models is noisy, causing a reduction in the amount of noise identified in the calibration batches. Therefore, new NOC batches under monitoring present a higher noise level than expected. As a consequence, the number of Type I faults of the SPE is too high.

For the variable-wise model of these data, the OTI is close to the ISL. A similar behavior is observed in the batch dynamic model with 1 LMV. This is coherent with the discussion in Chapter 3, according to which if the number of LMVs is low, the behavior of the batch dynamic model is expected to be similar to that of a variable-wise model. Finally, the OTI-ISL mismatch in the MP-VW model is higher than that for the variable-wise model. Since they only differ in that the MP-VW has two sub-models, at first glance the division in two sub-models does not seem to be appropriate. Anyway, Let us study this more deeply.

The information tree of the MP-VW model is shown in Figure 7.9(a). First, 2 PCs were included in the model. Afterwards, since the division of the model in two outperformed the addition of a third PC, the first choice was selected. The control limits for the SPE of the 2 PCs single variable-wise model is shown in Figure 7.9(b). The result of adding a new PC appears in Figure 7.9(c) whereas the result of dividing appears in Figure 7.9(d). Figure 7.9(c) corresponds to the SPE chart of the variable-wise model of Table 7.1 whereas Figure 7.9(d) corresponds to the chart of the MP-VW model in the same table. Notice that, when the single model has 2 PCs, it represents the first interval of the batch better than the second interval -from sampling time 55 to 100 in Figure 7.9(b). This causes the sharp change in the control limit of the SPE chart. Adding a new PC overcomes the problem, but the model does not fit the change of correlation structure of the process. Dividing the model leads to a better representation of the data with only 2 PCs (a value of $Q^2 = 93.8\%$). Moreover, if the scores of a batch in the variable-wise model of 2 PCs are plotted in the latent sub-space (Figure 7.10), it can be seen that the scores are arranged in a pair of clusters in accordance with the division found by the MP algorithm.

Therefore, whereas the OTI computed in Table 7.1 tell us the VW model is a good choice, the analysis of the data suggests to divide in two sub-models. Strictly speaking, the OTI is telling us to what extent the assumptions regarding the distribution of the D-statistics and the SPE are valid for a specific case of study and a modelling structure. For instance, if the OTI is very different to the ISL for the D-statistics of batch-wise models is because the assumed F-distribution is not appropriate. It is important to point out that the best model structure for modelling the data (in this case, the MP-VW model according to Figures 7.9 and 7.10) does not have to be the best modelling structure for monitoring, nor when a specific set of assumptions are accepted.

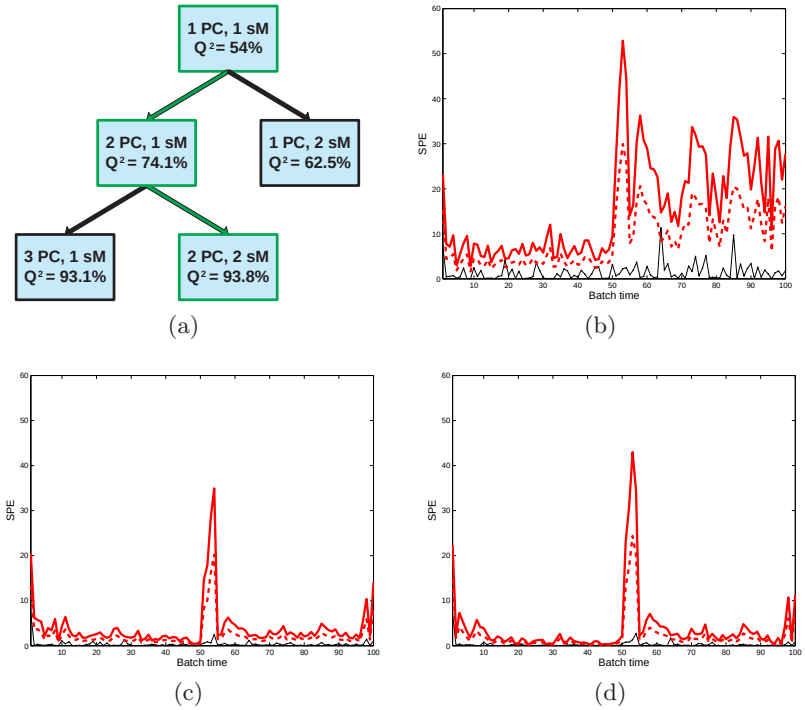


Fig. 7.9. Calibration of the MP-VW model: (a) information-tree, (b) SPE plot for a 2 PCs variable-wise model, (c) SPE plot for a 3 PCs variable-wise model, (d) SPE plot for a 2 PCs, 2 sub-models variable-wise model. $Q^2 = 1 - \frac{PRESS}{SSE}$ is the goodness of prediction. The thick solid lines are the 99% confidence limit; the thick dashed lines represent the 95% confidence limit; the thin dot lines represent the residuals of a normal batch.

Nonetheless, notice that most approaches -including the one presented in this Thesis- use the PRESS to fit a model for monitoring. Alternative statistics should be defined in the future to optimize the structure of a model for its use in monitoring.

Control limits readjusted from the OTI

If in a monitoring system the OTI is too high, the number of false alarms is as well too high. This happens, for instance, with the SPE of local models (see Table 7.1). If the OTI is too low, the chart only detects clear faults at the expense of not detecting slight faults. This happens with the D-statistic of the batch-wise and evolving approaches. If the OTI is not close to the ISL, the limits are meaningless and the monitoring performance of the different

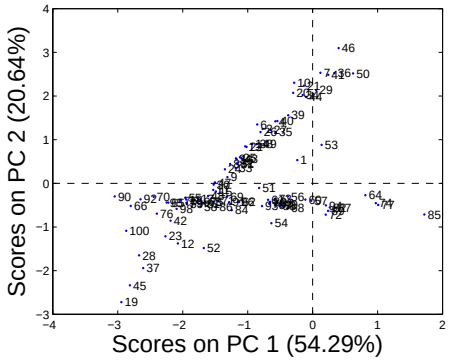


Fig. 7.10. Scores of a batch for the 2 first PCs of a variable-wise model from data of the *Saccharomyces cerevisiae* cultivation process.

approaches cannot be compared. To achieve an adequate performance of the monitoring charts, it is highly recommended to readjust the control limits using the calibration data. Limits are raised or lowered so that the OTI matches the ISL.

If control limits are not readjusted, the monitoring charts tend to be very dependent on the structure of the model used, as shown in Table 7.1. Even when they are readjusted it can be seen that this structure is still important. In Table 7.2, the OTI of the 14 batches in the test set of the Saccharomyces Cerevisiae cultivation process is computed from the adjusted monitoring charts. To perform this adjustment, after the models are fitted, control limits are readjusted so that the OTI computed for the calibration data matches the ISL. Now, the study is focused on the batch-wise model with zero deviations (BW-ZD), the local models and the variable-wise models (VW). This experiment is also devoted to investigate how the OTI of the modelling structures varies with the number of PCs in the models.

Table 7.2. Overall type I risk (OTI) computed for the NOC test set for different models -batch-wise with zero deviations (BW-ZD), local and variable-wise (VW)- and number of PCs in the on-line monitoring of the Cultivation of Saccharomyces Cerevisiae. Imposed Significance Level (ISL) set to 1%

PCs	BW-ZD		Local		VW	
	D	SPE	D	SPE	D	SPE
1	3.71%	**7.29**%	2.64%	**7**%	4.79%	**5.5**%
2	3.71%	**6.86**%	3.43%	**8.79**%	**5.71**%	4.07%
3	0.07%	**7.5**%	2.71%	**12.36**%	**7.79**%	4%
4	0%	**8.86**%	3.36%	**13.86**%	**8.21**%	5%
5	0.29%	**9.71**%	4.5%	**16.43**%	**13.14**%	4.21%
6	0.5%	**11**%	**6.79**%	**20.71**%	**14.79**%	3.64%

Although control limits have been readjusted, some of the results observed in Table 7.1 also appear in this experiment. First, for batch-wise models, the D-statistic presents too low OTI values and the SPE presents too high OTI values. Second, local models present high OTI values in the SPE. In the case of variable-wise models, the D-statistic for the test set tends to present an OTI much higher than the ISL. This was unnoticed from Table 7.1. It may be seen as a consequence of imposing the same correlation structure for the whole batch. Let us take the example shown in Figure 7.9. In that case, there is an evident change of correlation structure at the middle point of the batch duration. This means the relationships of the process variables found for the first phase do not hold for the second phase. Nevertheless, the variable-wise PCA model is constructed under the assumption that those relationships for the first phase, captured by the first 2 PCs, are true for the whole batch processing. The effect of this incorrect assumption is observed in Table 7.2 in the high OTI of the D-statistic. Notice that all the problems caused by using an inappropriate modelling structure become more dramatic as the number of PCs increases. For this reason, as a general rule it is recommended not to use many PCs unless necessary.

The OTI-ILS mismatch is much relaxed if the control limits are readjusted using a leave-one-out procedure [3, 75]. Following this procedure, a batch is separated each time and the model and control limits are computed from the remaining batches. Then the batch is monitored and the number of faults is recorded. This procedure is repeated for the complete set of batches. If the OTI is closed to the ISL, the procedure stops. If not, according to the OTI computed, the limits are raised or lowered and everything is repeated. The comparatives presented from here on have been performed using this procedure.

7.3 Application of the MP framework

As it was seen in the preliminary study, the use of batch-wise data for on-line monitoring may present some disadvantages, such as the auto-correlation of the D-statistics and a distorted representation of the faulty intervals. The on-line monitoring approach of this chapter is based on applying the MP framework over batch dynamic unfolded data with a reduced number of LMVs and PCA. If a reduced number of LMVs is used, several sub-models need to be fitted in order to model changes in the correlation structure.

The parameters used in the first and second steps of the MP framework (Figure 5.5) are listed in Table 7.3. The PRESS is used to identify the proper structure of the model. The MP is run for different values of k_γ in (5.3) and different numbers of LMVs. Once a set of MP models is fitted, these are merged according to different values of T_m and the criterion of minimum number of LMVs. As stated in [3], the number of LMVs should be as low as possible to avoid the auto-correlation in the statistics and the distortion

of the faulty interval of an out-of-control batch. Therefore, instead of using parsimony as in off-line monitoring, it was preferred to define complexity in terms of the number of LMVs. Thus, the objective is to reduce this number as much as possible without significant loss of modelling performance.

Table 7.3. Parameters used to fit MPPCA models for on-line monitoring. M and M_i stand for models of the batch data set. S_i stands for a sub-model of an interval. $\overline{LMV}_i$ stands for the average number of LMVs of sub-model S_i.

Multi-Phase Parameters		
$\delta(M_i, M)$	=	$\frac{PRESS(M)-PRESS(M_i)}{PRESS(M_0)}$
β^k	=	$SSE - SSE^k$
$\gamma(\delta(M_i, M))$	=	$k_\gamma \cdot \delta(M_i, M)$
k_γ	=	[1,3,5]
LMVs	=	[0,1,2,5,10,50]
n	=	0
$minL$	=	1
T	=	0.1
Merging Parameters		
T_m	=	$0 \rightarrow 0.5$
$\varphi(S_1, S_2)$	=	$\overline{LMV}_1 - \overline{LMV}_2$

Saccharomyces Cerevisiae cultivation

Let us start with the analysis of the Saccharomyces Cerevisiae cultivation data. Although the model design was used as example in Chapter 5, it is repeated here. Following the MP framework -Figure 5.5- with the parameters of Table 7.3, five MPPCA models were obtained. An ANOVA from the models was computed, showing statistical significant differences (p-value < 0.05) in the modelling performance of the models. The LSD intervals are shown in Figure 7.11. As it can be seen, there is no statistically significant difference among the first three models ($T_m = 0$, $T_m = 0.02$ and $T_m = 0.08$). The last one ($T_m = 0.22$) clearly presents the poorest performance. From this information, the model suggested is the one for $T_m = 0.08$, because it is more parsimonious than those for $T_m = 0$ and $T_m = 0.02$ with no statistically significant differences in performance. Nonetheless, to see whether this selection procedure is adequate, the performance of the MPPCA models for $T_m = 0$, $T_m = 0.08$ and $T_m = 0.22$ will be compared. Also, a batch-wise model, a local model and a variable-wise model are included in the comparison.

The results of the comparison are shown in Table 7.4. First, let us focus on the upper part of the table, where only the features of the models are presented: the structure (number of phases, LMVs and PCs), the PRESS-ratio -computed dividing the average PRESS of the NOC test data set with that of

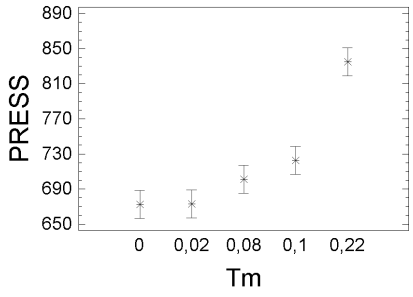


Fig. 7.11. LSD intervals for the PRESS of the merged models with different T_m values. Saccharomyces Cerevisiae cultivation process.

Table 7.4. Models and performance statistics for the on-line monitoring of the Cultivation of Saccharomyces Cerevisiae process. $\overline{LMV}$ stands for the average number of LMVs and $\overline{PC}$ stands for the average number of PCs. The statistics shown are the Overall Type I (OTI) risk, for a Imposed Significance Level (ISL) of 1%, the Average Run Length (ARL) and the Number of Type I (NTI) and Number of Type II (NTII) errors. The number between parenthesis by the NTI and NTII is the total number of NOC and abnormal batches tested, respectively. D stands for the D-statistic and #MC for the number of miss-classifications.

Model	Structure	PRESS Ratio Parameters	
MP_0	11 phases, 0.8 $\overline{LMV}$, 2.1 $\overline{PC}$	**1.02**	**410**
$MP_{0.08}$	11 phases, 0.5 $\overline{LMV}$, 2 $\overline{PC}$	**1.04**	**340**
$MP_{0.22}$	7 phases, 0 $\overline{LMV}$, 1.7 $\overline{PC}$	**1.08**	**120**
BW	2 PCs	**1.04**	**2000**
Local	2 PCs	**1.04**	**2000**
VW	2 PCs	**1.10**	**20**

Model	OTI (ISL=1%)		ARL	NTI (14 batches)			NTII (20 batches)		
	D	SPE		D	SPE	#MC	D	SPE	#MC
MP_0	1.36%	1.79%	**28.6**	1	0	**1**	7	9	**4**
$MP_{0.08}$	1.07%	1.64%	**28.7**	1	0	**1**	7	9	**5**
$MP_{0.22}$	1.29%	1.71%	**34.4**$*$	1	0	**1**	7	4	**3**
BW-ZD	0%	2.43%	**26.8**	0	2	**2**	12	4	**4**
Local	1.93%	2.43%	**36.2**$*$	2	1	**2**	6	6	**3**
VW	1.5%	1.71%	**34.2**$*$	1	0	**1**	7	3	**3**

$*$ Statistically significant different in terms of ARL to the BW-ZD model, p-value<0.05.

the calibration data set- and the number of parameters in the loadings matrices of the models. The principal difference among models is the number of parameters used. Both local and batch-wise produce highly over-parameterized models. Their number of parameters are one order of magnitude higher than that of the MP models and two comparing to the variable-wise model, which is the most parsimonious. Nonetheless, the PRESS of the models (not shown) is of the same order of magnitude, showing that the models are capturing - more or less- the same amount of structured information.

Now let us focus on the performance indices of the on-line monitoring system developed for every model (lower part of Table 7.4). Remember control limits in the charts have been readjusted following a leave-one-out approach. The OTI -computed for the NOC test set- is shown together with the Average Run Length (ARL), the Number of Type I errors (NTI) and the Number of Type II errors (NTII). The ARL is the average number of sampling times a monitoring system needs to detect an out-of-control signal. It will be considered an on-line monitoring chart signals an out-of-control situation when three consecutive values of the statistic (D-statistic or SPE) exceed the corresponding control limit. To compare the models in a fair scenario, only the abnormal batches detected by all the approaches are used to compute the ARL. The NTI is the number of in-control batches classified as out-of-control, and the NTII is the number of out-of-control batches classified as in-control.

By comparing the OTI values in Table 7.4 with those of Table 7.2, it can be seen that the OTI is closer to the ISL when using a leave-one-out correction of the limits than when a direct correction is used. The batch-wise approach still presents a very low OTI value for the D-statistic in Table 7.4. This feature, which has been reported many times in the literature [13, 3], makes the D-statistic chart almost useless for batch monitoring [75], since it is very unlikely to find a fault in the D-statistic and not in the SPE.

The MPPCA models for $T_m = 0$ and $T_m = 0.08$ together with the batch-wise model gave the best response in terms of ARL. The other three models are statistically significantly slower (p-value < 0.05) in detecting the faults. With respect to the NTI, it must be stated that the differences among the models are not substantial. For instance, by carefully inspecting the 2 NOC batches detected as faulty by the Local model, it can be seen that the trajectory of the batches do present some slight deviations from the rest of the batches. Therefore, it is difficult to say whether this is a wrong detection or, in fact, it is correct. For similar reasons, the results in the NTII are quite similar among the batches. Nonetheless, an important issue is which chart (D-statistic or SPE) signals the faults. In the Multivariate Statistical Process Monitoring (MSPM) of continuous processes, the faults which are consistent with the model are detected in the D-statistic while those not consistent are signaled by the Q-statistic. Table 7.4 shows that the batch-wise model detects all the abnormal batches in the SPE. Therefore, little can be said about the nature of the fault. Local, VW and MP models treat the sampling times as

objects, instead of the whole batch, thus presenting a monitoring philosophy more close to that used for continuous processes.

Focusing on the MP models, the selection of the model of $T_m = 0.08$ is understood to be adequate. First, the MP models for $T_m = 0$ and $T_m = 0.08$ yield very similar outcomes, presenting the second a lower number of LMVs -which is the objective in the merging. Second, the MP model with $T_m = 0.22$, which was discarded from the view of the LSD intervals, presents a statistically significant slower response to faults -higher ARL. Nonetheless, notice its monitoring performance in terms of NTI and NTII is quite good.

Nylon 6'6 polymerization

Following the MP framework of Figure 5.5 with the parameters of Table 7.3, two MPPCA models were obtained -for $T_m = 0$ and $T_m = 0.06$. Both MPPCA models are single-phase, so that it can be concluded that during the batch processing no significative changes in the correlation structure occur. At least this is true for the data set collected. Similar findings were presented in [3]. An ANOVA from both models was computed, showing statistical significant differences (p-value < 0.05) in the modelling performance of both models. Since such a difference exists, the MPPCA model for $T_m = 0$ is preferred. It consists of a batch dynamic model with 1 LMV, 1 phase and 2 PCs. For $T_m = 0.06$, the MPPCA model coincides with the variable-wise model (0 LMVs).

In Table 7.5, the MPPCA model is compared with a batch-wise model, a local model and the variable-wise model. All models are correctly fitted, according to the PRESS Ratio. Again, both local and batch-wise produce highly over-parameterized models. Regarding the monitoring performance, no differences where observed among the approaches. In this case, the faults in the out-of-control batches are so evident that they were detected with no delay by all the approaches.

Waste-water treatment

Following the MP framework of Figure 5.5 with the parameters of Table 7.3, four MPPCA models were obtained. An ANOVA from the models was computed, showing statistical significant differences (p-value < 0.05) in the modelling performance. The LSD intervals are shown in Figure 7.12. As it can be seen, there is no statistically significant difference among the two first models ($T_m = 0$ and $T_m = 0.02$). The other two ($T_m = 0.06$ and $T_m = 0.22$) present poorer performance. From this information, the model suggested would be the one for $T_m = 0.02$. To see whether this selection procedure is adequate, the performance of the MPPCA models for $T_m = 0$, $T_m = 0.02$ and $T_m = 0.06$ will be compared. Again, a batch-wise model, a local model and a variable-wise model are included in the comparison.

Table 7.5. Models and performance statistics for the on-line monitoring of the Nylon 6'6 polymerization process. The statistics shown are the Overall Type I (OTI) risk, for a Imposed Significance Level (ISL) of 1%, the Average Run Length (ARL) and the Number of Type I (NTI) and Number of Type II (NTII) errors. The number between parenthesis by the NTI and NTII is the total number of NOC and abnormal batches tested, respectively. D stands for the D-statistic and #MC for the number of miss-classifications.

Model	Structure			PRESS Ratio		Parameters
MP_0	1 phase, 1 LMV, 2 PCs			**1.03**		**36**
BW	3 PCs			**0.93**		**3132**
Local	3 PCs			**0.93**		**3132**
VW ($MP_{0.6}$)	2 PCs			**1.03**		**18**
Model	OTI (ISL=1%)		ARL	NTI (5 batches)			NTII (5 batches)		
	D	SPE		D	SPE	#MC	D	SPE	#MC
MP_0	0%	0.86%	**0**	0	0	**0**	0	0	**0**
BW-ZD	0%	0.17%	**0**	0	0	**0**	1	0	**0**
Local	0.34%	0.34%	**0**	0	0	**0**	0	0	**0**
VW ($MP_{0.6}$)	0%	0.86%	**0**	0	0	**0**	0	0	**0**

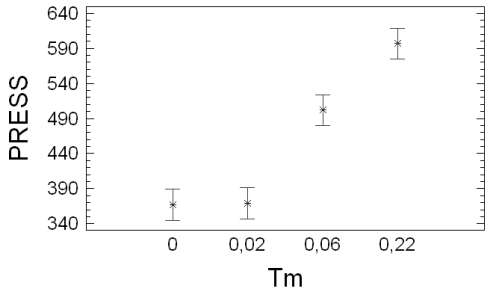


Fig. 7.12. LSD intervals for the PRESS of the merged models with different T_m values. Waste-water treatment process.

In Table 7.6 it can be observed, once again, the different number of parameters of the models. The variable-wise model is the most parsimonious. For the MPPCA models, the number of parameters is one order of magnitude higher. Both for the batch-wise and the local model, this number is two orders of magnitude higher. Regarding the monitoring performance, the two first MPPCA models are statistically significantly faster than the rest (p-value < 0.05). They both present no clear differences in terms of ARL, NTI and NTII. Therefore, it could be stated that the use of the ANOVA and LSD intervals is adequate once again to select the MP model. On the other hand, the batch-wise model presents a very good performance specially in

terms of NTI, unfortunately yielding a slow response to faults -high ARL. Finally, the detection capability of the variable-wise model was poor -high NTII. Notice the MPPCA model for $T_m = 0.06$ do not present a high NTII. Since this model has almost no LMVs included, its only difference with the variable-wise model is the number of phases -4 in front of 1. Therefore, the poor detection of the variable-wise model may be caused by the constant correlation structure imposed in the model. Similar findings were presented in [77].

Table 7.6. Models and performance statistics for the on-line monitoring of the Waste-water treatment process. $\overline{LMV}$ stands for the average number of LMVs and $\overline{PC}$ stands for the average number of PCs. The statistics shown are the Overall Type I (OTI) risk, for a Imposed Significance Level (ISL) of 1%, the Average Run Length (ARL) and the Number of Type I (NTI) and Number of Type II (NTII) errors. The number between parenthesis by the NTI and NTII is the total number of NOC and abnormal batches tested, respectively. D stands for the D-statistic and #MC for the number of miss-classifications.

Model	Structure	PRESS Ratio	Parameters
MP_0	4 phases, 5 $\overline{LMV}$, 3.5 $\overline{PC}$	**0.96**	**420**
$MP_{0.02}$	5 phases, 2.5 $\overline{LMV}$, 4.5 $\overline{PC}$	**0.97**	**270**
$MP_{0.06}$	4 phases, 0.25 $\overline{LMV}$, 4 $\overline{PC}$	**1.01**	**100**
BW	3 PCs	**0.9**	**5100**
Local	2 PCs	**0.51**	**3400**
VW	2 PCs	**1.05**	**10**

Model	OTI (ISL=1%)		ARL	NTI (35 batches)			NTII (40 batches)		
	D	SPE		D	SPE	#MC	D	SPE	#MC
MP_0	0.49%	1.32%	**28.8**	2	3	**4**	8	11	**6**
$MP_{0.02}$	0.38%	1.49%	**29.1**	2	4	**5**	8	10	**6**
$MP_{0.06}$	0.38%	1%	**36.3**$*$	2	3	**4**	6	8	**5**
BW-ZD	0.4%	0.86%	**43.3**$*$	1	1	**2**	19	6	**6**
Local	0.87%	0.91%	**40.6**$*$	2	4	**5**	7	9	**6**
VW	0.3%	0.89%	**41.3**$*$	2	3	**4**	9	10	**9**

$*$ Statistically significant different in terms of ARL to the MP_0 model, p-value<0.05.

7.4 Conclusions

The first conclusion withdrawn from the results of this chapter is that one of the principal steps to obtain a good on-line monitoring system is to adjust the control limits adequately. This step is as important -if not more important- than the structure of the model used to capture the data. A good approach is to correct the limits computed from theory [69, 71, 70] in a left-one-out basis [3, 75].

Regarding the structure of the model, batch-wise models present several drawbacks. First, the D-statistic is almost useless, being all the detection capabilities centered in the SPE. This has the additional problem that the monitoring system cannot distinguish between faults which are consistent with the model from those which are not consistent. On the other hand, the statistics computed from batch-wise models are evolving in the sense that data from preceding sampling times have some influence in both the D-statistic and the SPE of a particular sampling time. This causes the auto-correlation of the statistics and the distortion of the faulty interval in the charts. Nonetheless, this feature is interesting for detecting faults which are more evident in accumulated form. This may be the reason why the batch-wise model was very fast in the Saccharomyces Cerevisiae cultivation process, where many faults were induced with slight changes in the constants of the first principles model used for simulation. Contrarily, in the waste-water treatment process, the batch-wise model presented a slow response to faults.

The local models were found to present slow detection capabilities. This is consistent with the results presented in [49], but not with those of [3]. In both the latter paper and this document, the first principles model of Saccharomyces Cerevisiae have been used. Nonetheless, the data sets treated are very different. In [3], most of the faults were sudden changes in the variables. Here, most faults are slight modifications of the constants which change the evolving behavior of the process. Therefore, it must be concluded that the local models, with no memory, are efficient to detect sudden changes, but they are not to detect sustained changes in the process.

The results obtained with the batch-wise and local models are similar to those found for Q-score charts for monitoring white noise signals [115]. The structure of batch-wise models is specially suited for the detection of sustained signals, similarly as Cumulative Sum (CUSUM) charts are. On the other hand, local models present similar performance as Shewhart charts. As a matter of fact, both the D-statistic and SPE charts used in batch process monitoring are Shewhart charts. Therefore, equivalent Exponentially Weighted Moving Average (EWMA) or CUSUM charts can be developed to improve the detection of slight sustained changes in the process signals.

The variable-wise models gave a poor response when the correlation structure of the process was changing. In that case, it is strongly suggested to use several sub-models [77]. Also, the variable-wise approach was seen to be slow in detecting faults.

Different processes present different features and thus, they must be modelled differently. It has been shown that the MP framework, conveniently used, develop monitoring systems with a fast response to faults. Nonetheless, notice that the criterion for model calibration does not coincide with the objective of the application in this case. The criterion for model fitting is to minimize the PRESS whereas the objective of the monitoring is to distinguish faulty from NOC batches. Although these two objectives are intuitively

close, the models which optimize each of them do not necessarily match. Future work is deserved to find alternative loss functions to fit the models taking into account explicitly they are going to be used for monitoring, instead of using the PRESS.

8 On-line Quality Prediction in batch processes

Part of the contents of this chapter have been included in the following publications:

[6] J. Camacho, J. Picó and A. Ferrer. *Bilinear modelling of batch processes. Part II: PLS comparative*, Submitted to Journal of Chemometrics (2007).

[7] J. Camacho, J. Picó and A. Ferrer. *Multi-Phase Analysis Framework for Handling Batch Process Data*, Submitted to Journal of Chemometrics (2007).

[12] J. Camacho, J. Picó and A. Ferrer. *Multi-Phase Analysis Framework for Handling Batch Process Data*, 10th Scandinavian Symposium on Chemometrics (2007).

8.1 Introduction

In many batch processes, some quality measurements are performed after the processing to discard products which do not comply with the specifications. If these measurements can be somehow predicted during the batch processing, batches of poor quality can be discarded without finishing their processing or control actions can be performed to readjust the trajectory of the manipulable variables. Additionally, in some processes like waste-water treatment processes or bio-reactors, the trajectory of a variable which is expensive or difficult to measure may be estimated for control and monitoring purposes. In the on-line estimation of -the trajectory of- a variable, the objective at each sampling time of the batch is to estimate the value of that variable at that current sampling time. When a batch has finished, the complete trajectory has been estimated.

Both on-line final quality prediction and trajectories estimation need the availability of a model of the process. The quality of that model, which is very related to its capacity to capture the dynamics of a process, is very important from an economic point of view. PLS-based methods, like PCA and PLS, have proven to be powerful modelling tools. The approaches for modelling batch processes with PLS-based methods in an on-line application can be roughly classified in two types: The single model approach and the multi-model approach. A comparative study is preformed here.

The principal aim of the investigation in this chapter is to highlight the differences in the information captured by several on-line modelling approaches. This is carried out by studying the on-line prediction with PLS. Some of the conclusions presented can be extrapolated to PCA, especially those related with the modelling of the dynamics of a process. Since some of the approaches under study are defined only for PCA, their extensions to PLS are presented here for the first time. Notice that, although the approaches were proposed for PCA-based on-line monitoring, this study is by no means an attempt to compare them in that concrete application.

This chapter is split in four sections: Section 8.2 introduces the materials and methods used in the comparative; Section 8.3 is devoted to the comparison of single-model approaches and Section 8.4 to the multi-model approaches; Section 8.5 presents the general conclusions of the chapter.

8.2 Materials and Methods

The different approaches under study are compared using data from the Saccharomyces Cerevisiae cultivation and the waste-water treatment processes. The performance of the models is studied for two different problems: the on-line estimation of trajectories and the on-line final quality prediction.

The comparative is performed in terms of the Mean Squared Error (MSE) through the batch processing:

$$\mathbf{MSE} = \frac{\sum_{i=1}^{I}\sum_{k=1}^{K}(y_{ik} - \hat{y}_{ik})^2}{I \cdot K} \tag{8.1}$$

where y_{ik} is the value of the variable to be predicted in batch i at sampling time k, and $\hat{y}_{ik}$ its prediction.

The comparative presents several ANOVAs performed on the MSE values. The LSD intervals are shown when statistically significant differences are found among the approaches (p-value < 0.05). When observing positive skewness of the residuals, the logarithmic transformation is applied on the MSE.

For the Saccharomyces Cerevisiae cultivation process, data are split into calibration (52 batches, 66%) and test sets (26 batches, 33%). The purpose is to predict the biomass concentration from 2 process variables: the specific oxygen uptake rate and the specific carbon dioxide evolution rate. The batch duration comprises 100 sampling times. For the estimation of the biomass trajectory, the data structure is:

$$calibration : \underline{\mathbf{X}}(52 \times 2 \times 100) \rightarrow \mathbf{Y}(52 \times 100) \tag{8.2}$$

$$test : \underline{\mathbf{X}}(26 \times 2 \times 100) \rightarrow \mathbf{Y}(26 \times 100) \tag{8.3}$$

and for end-quality prediction:

$$calibration : \underline{\mathbf{X}}(52 \times 2 \times 100) \rightarrow \mathbf{y}(52 \times 1) \tag{8.4}$$

$$test : \underline{\mathbf{X}}(26 \times 2 \times 100) \rightarrow \mathbf{y}(26 \times 1) \tag{8.5}$$

The number of batches of the waste-water treatment process is low (18 batches). Therefore, no splitting in calibration and test is performed. The value of $\hat{y}_{ik}$ in (8.1) is obtained using a left-one-out cross-validation procedure. Five process variables are used to predict the phosphorus content. A total of 255 sampling times conform the sampling time mode of the process data, whereas only 18 samples of the phosphorus content per batch are used[a]. For the estimation of the phosphorus content trajectory, the data structure is:

$$\underline{\mathbf{X}}(18 \times 5 \times 18) \rightarrow \mathbf{Y}(18 \times 18) \tag{8.6}$$

and for end-quality prediction:

$$\underline{\mathbf{X}}(18 \times 5 \times 255) \rightarrow \mathbf{y}(18 \times 1) \tag{8.7}$$

[a] Actually, 19 samples of the phosphorus content per batch are collected. Nevertheless, only 18 are used since the true variables are substituted by the increments from the initial value, as commented in Chapter 2.

8.3 Single model approach

In the single model approach, a single PLS-based model is generated for the whole process. For the application of a bilinear PLS model, the three-way matrix of data has to be unfolded in two dimensions [44, 45].

In this section of the chapter, the influence of several issues in the quality of an unfold PLS model for prediction[b] purposes is studied, including:

- How are dynamics built in the model?
- How does the imputation method affect the performance of the prediction system?
- How does the preprocessing method affect the performance of the prediction system?

8.3.1 Issues under study

Relationship between the dynamic information built in the model and the unfolding direction

If batch-wise unfolding ($\underline{\mathbf{X}}^{(K-1)}$) is used for on-line prediction, the measurements which are not available have to be imputed at every sampling time. The methods designed to impute missing data from a PCA and PLS model [102, 103] can be used for this purpose. If variable-wise unfolding ($\underline{\mathbf{X}}^{(0)}$) is used no imputation is necessary, but the dynamics of the process in the form of auto-covariances and lagged cross-covariances among variables are not captured [46]. The batch dynamic unfolding method [47] can be seen as a generalization of the unfolding methods for batch data, including batch-wise and variable-wise unfolding as special cases:

$$\mathbf{X} = \underline{\mathbf{X}}^{(n)} \tag{8.8}$$

where n stands for the number of LMVs included as variables and:

$$n = \{k-1 \quad : \quad k \,\epsilon\, \{1, 2, .., K\}\} \tag{8.9}$$

The dynamics built in a model depend on the unfolding performed on the data, as discussed in Chapter 3. The influence of this unfolding in the prediction performance of a model is part of the study of this section.

[b] For the sake of simplicity, the term prediction and its derivatives will be used for both prediction of the final quality and estimation of the trajectory of a variable.

Missing data imputation

For the on-line application of batch-wise models, the imputation of future values is necessary. In the case of batch dynamic models, only the measurements of the initial period needs to be imputed. For instance, take the general batch dynamic model $\underline{\mathbf{X}}^{(n)}$. An object of matrix $\underline{\mathbf{X}}^{(n)}$ is composed of $n+1$ measurement-vectors, i.e. $(n+1)\cdot J$ variables. Thus, at the beginning of the batch, when $m < n+1$ sampling times have passed, the remaining future $n+1-m$ measurement-vectors have to be imputed. The length of the period in which imputation is necessary depends on the number of LMVs included in the model. Notice that, with this philosophy, batch dynamic models work in two modes: imputation mode -at the beginning- and complete-object mode. From here on, when mentioning BDPLS models, the approach of [47] extended with the imputation of the first samples is understood.

When comparing the performance of the different modelling methods, an important factor which should be taken into account is the algorithm for future data imputation in batch-wise and batch dynamic models. The algorithm used has a big influence on the quality of the prediction of the scores [103, 102], and so on the prediction of the quality variables.

[116] presents a comparative study of several imputation methods. In that paper, Trimmed Scores Regression (TSR) along with Conditional Mean Replacement (CMR) are recommended. In [117], authors demonstrate that the different CMR approximations and TSR are particular cases of a general framework and present similar features, being TSR the simplest method from an algorithmic point of view.

In this comparative study, TSR is compared with the traditional imputation method used for online batch monitoring [13], named Projection to Model Plane (PMP). TSR was first used for PLS prediction in [101], but the theoretical formulation was not treated. This is carried out here. For more details concerning the application of the PMP and TSR methods with PLS models, see [118] and Appendix D, respectively.

Preprocessing

Batch process data are normally preprocessed previous to PCA or PLS modelling so that the unfolded data matrix has $\mathbf{0}$ mean. Additionally, scaling to variance 1 gives the same importance or weight to all variables in the resulting model. This is in principle a good scaling method when nothing is known about the process.

Here, two preprocessing methods are distinguished: auto-scaling and trajectory scaling. The term auto-scaling stands for the centering and scaling of a two-way matrix to mean 0 and variance 1. For three-way data, the auto-scaling operation is the centering and scaling -mean 0 and variance 1- of the unfolded matrix. Notice that the resulting data after auto-scaling depend on the unfolding procedure. After batch-wise unfolding, the auto-scaling means

the subtraction of the average trajectories of the variables and the scaling of the deviations around these trajectories. Resulting data for each single variable and sampling time have 0 mean and variance 1. After variable-wise unfolding, the auto-scaling operation means the subtraction of the grand mean of the variables -i.e. the mean values of the variables taking into account all the measurements throughout the batch- and the scaling of the deviations from these grand means. Resulting data for each single variable, including all sampling times, have 0 mean and variance 1.

The subtraction of the average trajectories of the variables, independently of the unfolding method used -batch-wise, variable-wise and batch dynamic-, yields a two-way matrix of $\mathbf{0}$ mean. In this document, the preprocessing method consisting in subtracting the average trajectory and scaling to variance 1, independently of the unfolding method, is named trajectory scaling.

Trajectory scaling yields exactly the same preprocessed data as auto-scaling for batch-wise models, so that it can be stated that both preprocessing methods coincide for that unfolding. Nonetheless, the prediction performance of variable-wise models, as well as batch dynamic models, can be affected by the choice of the preprocessing method. In the case of batch dynamic models the difference of both preprocessing methods decreases as the number of LMVs increases.

8.3.2 Strategies under study

Table 8.1. Strategies using a single PLS model of a batch process for on-line prediction.

Acronym	Explanatory text
BW-PMP	Batch-wise unfolding [45] + Projection to Model Plane [118]
BW-TSR	Batch-wise unfolding [45] + Trimmed Scores Regression [103]
BD-TS	Batch Dynamic [47] + Trajectory Scaling + Trimmed Scores Regression [103]
BD-AS	Batch Dynamic [47] + Auto-Scaling + Trimmed Scores Regression [103]
VW-TS	Variable-wise unfolding + Trajectory Scaling [119]
VW-AS	Variable-wise unfolding + Auto-Scaling (Level 1 in [31])
VW-BW	VW-AS + Batch-wise unfolding (Level 2 in [31]) + Trimmed Scores Regression [103]

The strategies studied in this section are listed in Table 8.1. The approaches *BW-PMP* and *BW-TSR* are batch-wise models where the imputation methods used are PMP and TSR, respectively. *BD-TS* and *BD-AS* are batch dynamic models where data were preprocessed using trajectory scaling and auto-scaling, respectively. In both cases, the imputation is performed with TSR[c]. To choose the number of LMVs for the batch dynamic models, a set of models of the form $\underline{\mathbf{X}}^{(n)}$ for different n values -from 0 to $K-1$- are

[c] In a preliminary study, TSR was observed to outperform PMP. Thus, TSR was chosen as the general imputation method. Strictly, these approaches should be

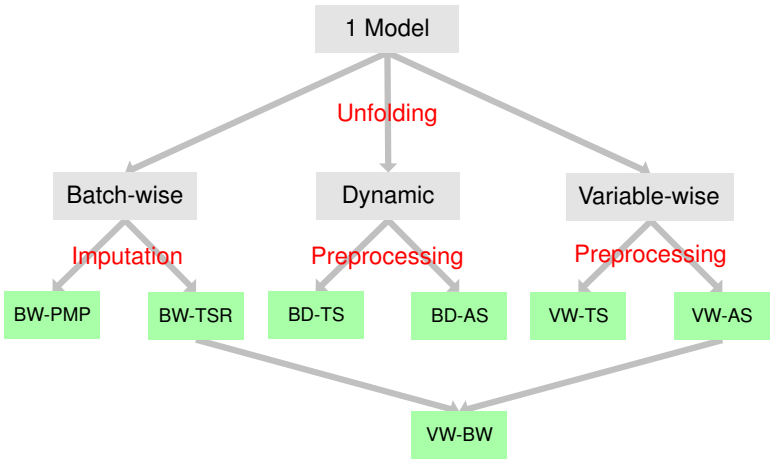


Fig. 8.1. Classification of the strategies. Note the batch dynamic approach is understood as a general parametrization of the unfolding where batch-wise and variable-wise unfolding methods are included as extreme cases.

calibrated and that with the best performance according to cross-validation is chosen. *VW-TS* represents the model obtained after trajectory scaling and variable-wise unfolding [119]. *VW-AS* represents the model obtained after variable-wise unfolding and auto-scaling. This latter approach coincides with the level 1 proposed in [31]. In the cited paper, authors also comment the possibility of generating a batch-wise model (level 2, batch trace level) from the scores obtained in the level 1 "*to derive sharper bounds for new batches taking the auto-correlation structure of the scores into account*". In order to understand the effect of generating a batch-wise model from these scores, instead of from the original data, the approach *VW-BW* is included in the comparative. Again, the imputation method used is TSR[d]. Thus, the convenience of generating the batch-wise unfolding model from the scores of the *VW-AS* model can be clearly determined by comparing the results of the *BW-TSR* and *VW-BW* approaches.

The strategies under study can be classified as shown in Figure 8.1. First, strategies are distinguished by the unfolding procedure. Batch-wise models are further classified by the imputation method. Notice that for those models, auto-scaling and trajectory scaling yield the same preprocessed data and so the preprocessing is not specified. Batch dynamic and variable-wise models

named *BD-TS-TSR* and *BD-AS-TSR*, but *BD-TS* and *BD-AS* have been chosen for briefness.

[d] Strictly, once again, the name *VW-AS-BW-TSR* should be used, but *VW-BW* was preferred for briefness.

are classified by the preprocessing method. The *VW-BW* strategy cannot be classified in any of the preceding groups. Since it is built from the batch-wise unfolding of the *VW-AS* scores, using TSR for imputation, the model is represented as the combination of the *VW-AS* and *BW-TSR* approaches.

As it can be seen, the aim of this comparative is to study the influence of the preprocessing, unfolding and imputation methods on the prediction power of a PLS model of a batch process. At the same time, the quality of the prediction is related to the quality of the modelling of the dynamic information. The modelling of the dynamics is of especial interest for a monitoring and control system, in order to generate a dynamical pattern of a process. Thus, how dynamics are built in the models is an essential part of this study.

8.3.3 Experimental Study

Cultivation of *Saccharomyces cerevisiae*

The parameters of the calibrated models, representing each of the approaches under study and for both applications, are shown in Table 8.2. The LSD intervals for the MSE outcomes are presented in Figures 8.2 and 8.3. The MSE values of the method labelled *AT* are calculated assuming that the predicted trajectory is the average trajectory. This information is added as a base line.

Table 8.2. Structure of the models in the estimation of the trajectory of the biomass concentration (first row) and the prediction of its final value (second row). *Saccharomyces cerevisiae* cultivation process. The percentage in batch dynamic models $\underline{\mathbf{X}}^{(n)}$ is computed as $100 * (n+1)/K$.

BW-PMP	BW-TSR	BD-TS	BD-AS	VW-TS	VW-AS	VW-BW
9 LVs	9 LVs	9 LVs, 99 LMVs (100%)	15 LVs, 91 LMVs (92%)	1 LV	2 LVs	2 LVs, 9 LVs
2 LVs	4 LVs	14 LVs, 91 LMVs (92%)	13 LVs, 91 LMVs (92%)	2 LVs	2 LVs	2 LVs, 4 LVs

The outcomes presented for both the calibration and test sets are very similar. This similarity was expected since both sets represent the same operation point of the process and a sufficient number of samples was generated. Additionally, it means that the models were not over-fitted.

In general terms, the best approaches are *BW-TSR*, *BD-TS*, *BD-AS* and *VW-BW*. No statistically significant differences (p-value < 0.05) were found among them in any of the cases. It should be stressed the high number of lagged variables in the batch dynamic models (Table 8.2). This high number means that both dynamic models are very similar to *BW-TSR*, and so their

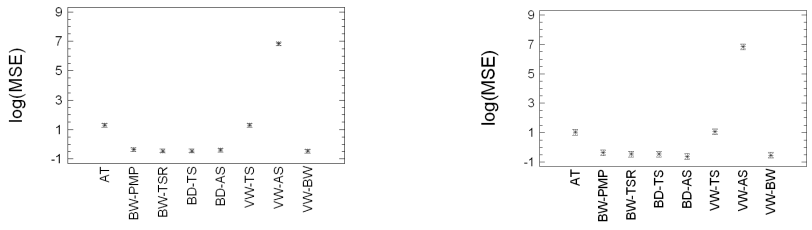


Fig. 8.2. LSD intervals for the MSE outcomes in the estimation of the trajectory of the biomass concentration in the *Saccharomyces cerevisiae* cultivation process: calibration set (left) and test set (right). The acronym *AT* means "average trajectory".

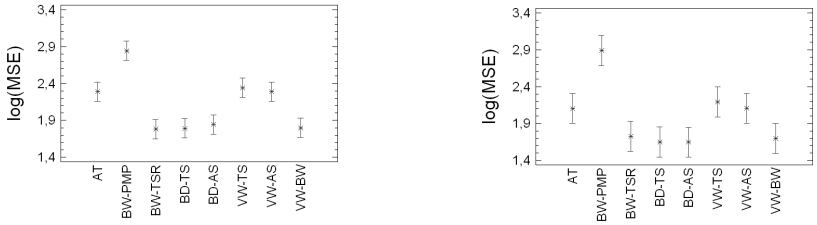


Fig. 8.3. LSD intervals for the MSE outcomes in the prediction of the final biomass concentration in the *Saccharomyces cerevisiae* cultivation process: calibration set (left) and test set (right). The acronym *AT* means "average trajectory".

performance is also similar. This also means that the batch-wise unfolding is close to the optimum unfolding method for prediction in this process, since such a high number of LMVs was determined to be the best choice in the calibration of the dynamic models.

From Figure 8.2, the way in which dynamics are built in the models can be deduced. On the one hand, a big part of the dynamics are captured by the average trajectory of $\mathbf{Y}$, which represents the general tendency in the evolution of the variables. The auto-scaling operation in the *VW-AS* approach does not subtract this trajectory. That is the reason why it yields, by far, the worst outcomes. Notice that the *BD-AS* model includes such a high number of LMVs that the auto-scaling operation almost matches the trajectory scaling operation.

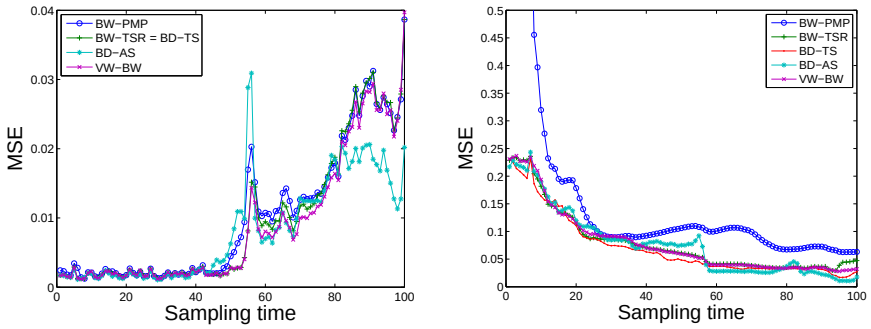
On the other hand, the remaining linear dynamics after the preprocessing are represented by the auto-covariances and cross-covariances of the variables. If this is not taken into account, part of the dynamics are not built in

the models. From the theoretical study in Chapter 3, it was deduced that the structure of the variable-wise models cannot capture these dynamics. Therefore, the only dynamics built in the *VW-TS* model are those subtracted with the average trajectory in the preprocessing. Additionally, the instantaneous relationships modelled in *VW-TS* are constrained to be constant throughout the batch. If the process presents changes in the correlation structure -as this is the case, see later in the following section- the instantaneous relationships modelled after variable-wise unfolding are meaningless. Those are the reasons why *AT* and *VW-TS* present no statistically significant difference. The inclusion of LMVs makes possible to capture the -possibly changing- dynamics which were not modelled with the average trajectory. Thus, the models which take into account both the general tendency and the additional dynamics present the best outcomes.

In the results obtained in the prediction of the final biomass concentration, shown in Figure 8.3, two differences can be found from the results of Figure 8.2: *VW-AS* presents a comparable performance to that of *VW-TS* or *AT*, whereas the performance of *BW-PMP* turns to be the worst. These differences are the consequence of the different content of the y-block in both applications. For trajectory estimation, matrix $\mathbf{Y}$ contains a trajectory for every batch. For the prediction of the final quality, vector $\mathbf{y}$ only contains a single value per batch. Therefore, in the latter case, both trajectory scaling and auto-scaling performs the same centering operation in the y-block. That is why the outcomes of the *VW-AS* approach are not statistically significantly different to those of *AT* and *VW-TS*.

BW-PMP performs as good as the best approaches for trajectory estimation. Nevertheless, its performance for final quality prediction is the poorest, being clearly outperformed by *BW-TSR*. This can be explained by looking at Figure 8.4, where the MSE of the best approaches along with *BW-PMP* is presented as a function of the sampling time. As stated in [13], the worst performance of the PMP imputation is presented during the first sampling times. The biomass concentration is close to 0 for that sampling times, with almost null variability. Thus, in the trajectory estimation (see Figure 8.4(a)), the poor performance of the PMP during the first samples is masked by this low variability. Nonetheless, in the prediction of the final biomass concentration, $\mathbf{y}$ is the same throughout the batch processing and equal to the final biomass concentration. The initial variability in the y-block is high and thus PMP (*BW-PMP*) presents a high initial MSE (Figure 8.4(b)).

From Figure 8.4(a), the behavior of *BD-AS* should also be stressed. *BD-AS* presents a sharp peak in the MSE around sampling time 55. All approaches present that sharp peak, but that of *BD-AS* is higher than the rest by more than a 50%. On the other hand, *BD-AS* outperforms the rest approaches during the last 20 sampling times (approximately). Both features can be explained by looking at the special nature of *BD-AS*. This model has 91 LMVs, thus the unfolded matrix presents a total of 184 variables -the



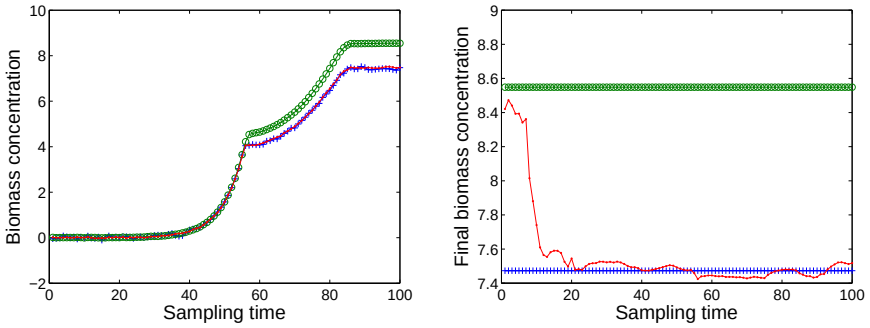
(a) (b)

Fig. 8.4. MSE in the estimation of the biomass concentration trajectory (a) and MSE in the prediction of its final value (b) along the batch processing. Approaches shown are BW-PMP (circles), BW-TSR (pluses), BD-TS (points), BD-AS (stars) and VW-BW (crosses)

two original variables times 92 sampling times. The samples of each of these 184 variables are the measurements collected for every batch, during 9 consecutive sampling times. This gives a total of 468 calibration samples in a variable -52 batches, 9 samples per batch. Thus, the unfolded matrix $\underline{\mathbf{X}}^{(91)}$ is 468×184. The assumption beneath the PLS model from $\underline{\mathbf{X}}^{(91)}$ is that the dynamics not subtracted after the auto-scaling operation, are sufficiently slow so that they can be considered invariant for an interval of 9 sampling times. At sampling time 55, the process variables (not shown) experience a sharp change in their evolution. This sharp change refuse the assumption of slow dynamics and affect the performance of batch dynamic models. A similar behavior was observed for a *BD-TS* model with 91 LMVs.

The good performance of *BD-AS* at the final phase of the batch was not observed in any *BD-TS* model fitted from the data. Thus, not only the batch dynamic structure is responsible of this good performance, but the preprocessing is also involved. During this final phase, the process variables as well as the biomass content have a constant value. The *BD-AS* approach is thus useful for that kind of stationary periods.

In Figure 8.5(a), the estimation of the biomass concentration trajectory, performed for a batch of the test set with the *BW-TRS* approach, is compared with the average trajectory and the real trajectory. The same comparison is shown for the prediction of the final value in Figure 8.5(b). The good performance of the PLS model following a batch-wise prediction/estimation approach is clear. Similar results are observed for the *BD-TS*, *BD-AS* and *VW-BW* models.



(a) (b)

Fig. 8.5. Example of the estimation of the biomass concentration trajectory (a) and of the prediction of its final value (b). Real trajectory is represented by the line with pluses. The average trajectory of the calibration set is represented by a line with circles. The estimation by the BW-TSR approach is represented by a dotted-line.

Waste-water treatment

The parameters of the models, representing each of the approaches under study and for both applications, are shown in Table 8.3. The LSD intervals for the MSE outcomes are presented in Figures 8.6 and 8.7. Again, the MSE of AT is added as a base line.

Table 8.3. Structure of the models in the estimation of the trajectory of the phosphorus content (first row) and the prediction of its final value (second row). The percentage in batch dynamic models $\underline{\mathbf{X}}^{(n)}$ is computed as $100 * (n+1)/K$.

BW-PMP	BW-TSR	BD-TS	BD-AS	VW-TS	VW-AS	VW-BW
3 LVs	4 LVs	4 LVs, 16 LMVs (94%)	7 LVs, 16 LMVs (94%)	1 LV	3 LVs	3 LVs, 4 LVs
1 LV	1 LV	1 LV, 226 LMVs (89%)	1 LV, 255 LMVs (100%)	1 LV	1 LV	5 LV, 2 LVs

The principal conclusion observed in the results is that the best approaches in the previous process, *BW-TSR*, *BD-TS*, *BD-AS* and *VW-BW*, still present the best performance here in general terms. Although some differences can be found, these are not statistically significant (p-value < 0.05). Also, the batch dynamic models have a high number of LMVs (observe the percentages in Table 8.3) and so they are very similar to *BW-TSR*. In the

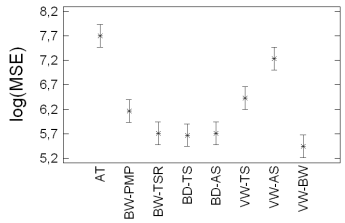


Fig. 8.6. LSD intervals for the MSE outcomes in the estimation of the trajectory of the phosphorus content in the waste-water treatment process. The acronym *AT* means "average trajectory".

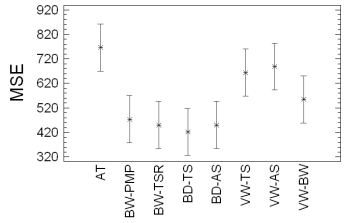


Fig. 8.7. LSD intervals for the MSE outcomes in the prediction of the final phosphorus content in the waste-water treatment process. The acronym *AT* means "average trajectory".

concrete case of the *BD-AS* model for quality prediction, the model is exactly equal to *BW-TSR* ($\underline{\mathbf{X}}^{(n)}$ with $n = K - 1$).

In Figure 8.6, *AT* yields the worst performance, closely followed by *VW-AS*. This is clearly different from what it was observed in the previous process. Note that, in the current process, the variability around the average trajectory is high, with similar magnitude to that of the trajectory itself (this can be seen later in Figure 8.9). *VW-TS* outperforms both *AT* and *VW-AS*, since it takes into account both the information in the average trajectory and the instantaneous relationships -in this case, the instantaneous relationships modelled are not so affected by the changes in the correlation structure. Finally, *BW-PMP* bears no statistically significant difference with *VW-TS*, being both approaches slightly inferior than the best ones.

The outcomes in the final quality prediction, Figure 8.7, are very similar to that of the previous process, except for the good performance of *BW-PMP*. In Figure 8.8, the MSE of all approaches for final quality prediction is presented as a function of the sampling time. In this figure, *BW-PMP* shows a high MSE

value during first samples. Nonetheless, since the number of total samples is large (255), this initial high MSE does not produce a significant increase in the total MSE. This is why *BW-PMP* performance is comparable to that of the best approaches. Figure 8.8 also shows that the process data collected at the first stage of the process -up to sampling time 85 approximately- are more linearly related to the final quality than those collected at the other stage of the batch. This can be observed in the ascending MSE trajectory of *VW-TS* from sampling time 85. Finally, the *VW-BW* approach presents poorer performance than the best approaches in a generalized way through the batch. Therefore, the use of the scores obtained from the *VW-AS* model, instead of the original data, does not seem to be convenient in this case. Nonetheless, no statistically significant differences were observed in the LSD intervals between *VW-BW* an the other best approaches (Figure 8.7).

In Figure 8.9(a), an example of the estimation of a phosphorus content trajectory by the *BW-TSR* approach is shown and compared with the average trajectory and the real trajectory. In Figure 8.9(b), the same is shown for the prediction of the final phosphorus content. The performance of the PLS models is not so good as in the previous process -only half of the variability is more or less captured.

The impact of the preprocessing method in the batch dynamic models is not clear neither from the results shown here, nor from those of the previous process, since in both cases batch dynamic models presented a high number of LMVs. A further analysis to compare the two preprocessing methods studied -trajectory scaling and auto-scaling- can be performed. In Figure 8.10, the MSE in the estimation of the phosphorus content trajectory, of a *BD-TS* and a *BD-AS* model, is evaluated as a function of the number of LVs and the number of LMVs. The phosphorus content trajectory for the last 8 sampling times has been modelled using different numbers of LMVs (from 0 to 10). By modelling only the second half of the batch duration and limiting the number of LMVs, the differences between both preprocessing methods can be observed without any side-effect. First, no imputation of missing data needs to be done. Second, the auto-scaled data are independent of the number of LMVs - i.e., the mean and standard deviation of each variable in the unfolded matrix is computed from the same data belonging to 8 consecutive sampling times, no matter the number of LMVs.

In Figures 8.10(a) and 8.10(b) it can be seen, once again, that the addition of LMVs is relevant for the estimation of trajectories. Models with 0 LMVs present a higher MSE value than those with more LMVs. Auto-scaling (Figure 8.10(b)) obtains more benefit from the inclusion of lagged data, since the dynamics contained in the average trajectory are not subtracted in the preprocessing. After trajectory scaling (Figure 8.10(a)), most of the dynamics of the process are subtracted and the improvement gained when adding LMVs is lower. The same results were observed for the case of the *Saccha-*

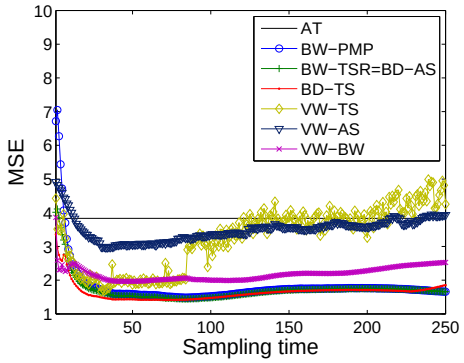
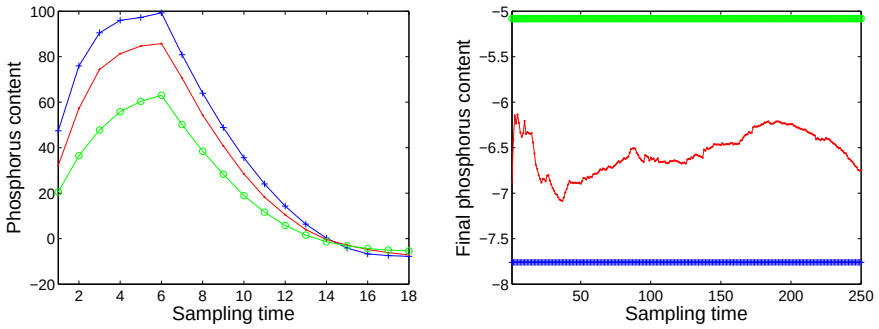


Fig. 8.8. MSE in the prediction of the final phosphorus content along the batch processing.



(a) (b)

Fig. 8.9. Example of the estimation of the trajectory of the phosphorus content (a) and of the prediction of its final value (b). Real trajectory is represented by the line with pluses. The average trajectory of the calibration set is represented by a line with circles. Estimation by the BW-TSR approach is represented by a dotted-line.

romices Cerevetisiae cultivation process when performing a similar analysis (not shown).

It should be stressed that after a certain number of LMVs has been added (two in Figure 8.10(a) and three in Figure 8.10(b)), a significative improvement is not gained with further additions. This shows that with only a few LMVs, the dynamics of the process are built in the model. Nonetheless, the optimum BD models in Table 8.3 present a much higher number of LMVs. Therefore, such a number of LMVs may be optimum for a different reason apart from the modelling of the dynamics. Notice the more LMVs, the longer

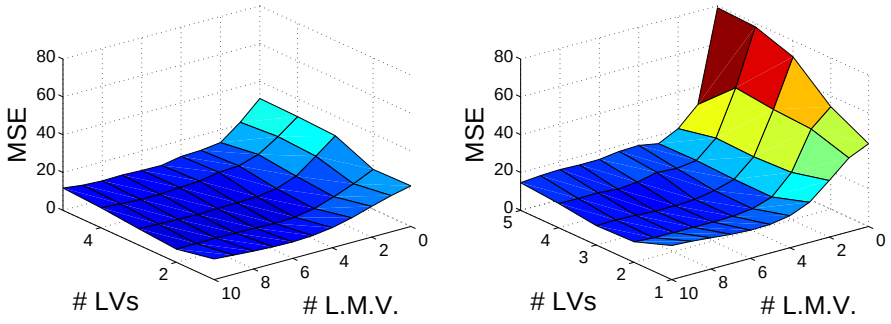


Fig. 8.10. Evaluation of the influence of the lagged information in the prediction power of a BD-PLS model: estimation of the phosphorus content trajectory after trajectory scaling (a) and after auto-scaling (b).

the interval in which imputation is used and also the shorter the interval where the correlation structure is imposed to be constant. For the first period of the batch, the use of an imputation method, such as PMP or TSR, may work better than using a low number of LMVs in some cases. Also, processes with changing dynamics may benefit of the inclusion of many LMVs, as discussed in Chapter 3. Then, in some cases, the principal drawback of the use of batch dynamic models with a low number of LMVs is the constant correlation structure imposed, and not a real necessity of incorporating more dynamic information.

8.3.4 Conclusions

In this first comparative of the chapter, the quality of several approaches for the on-line application of a single PLS model, in batch processes, was compared. The on-line estimation of trajectories and the on-line prediction of the final quality for two batch processes were considered.

The principal aim of this investigation is to study the influence of several issues in the performance of a PLS model used on-line. These issues include:

- The arrangement of the three-way data in an unfolded matrix, taking into account both batch-wise and variable-wise unfolding methods as well as intermediate cases: the batch dynamic models.
- The preprocessing method, including the subtraction of the average trajectory independently of the unfolding method, and the auto-scaling of the unfolded data.
- The method for imputing future measurements, including Projection to Model Plane (PMP) and Trimmed Scores Regression (TSR). In order to include the latter method in the comparative, the basis of TSR were extended to the PLS modelling. This is shown in Appendix D.

The following conclusions can be withdrawn from the results obtained:

- Variable-wise models are clearly outperformed by the other models. As shown in the theoretical study of Chapter 3, the formers do not capture the auto-covariances and lagged cross-covariances of the variables, which are part of the dynamics of the process. Also, time-varying correlation structures cannot be modelled with variable-wise models. These limitations drastically reduce the estimation performance.
- No statistically significant differences were observed between batch dynamic and batch-wise models. Notice that the batch dynamic unfolding is a general unfolding mechanism, where the number of lagged measurement vectors (LMVs) is an additional degree of freedom in the model calibration. Batch dynamic models presented an optimum number of LMVs close to that of batch-wise models in both cases studied. Therefore, it could be stated that the batch-wise unfolding is almost optimum -when using a single model of the process- for prediction purposes. At least this was true for the cases of study treated.
- Batch dynamic models take advantage from slowly changing dynamics. The less LMVs included, the less that changing dynamics can be properly modelled. The more LMVs included, the more flexible to changing dynamics the model is. The extreme cases are variable-wise and batch-wise models, respectively. If the dynamics of the process change slowly, the batch dynamic model can be fitted with a low number of LMVs, taking advantage from the benefits of calibrating a more parsimonious model -less number of parameters to be calibrated- from the same amount of data. Of course, this is always possible if the information in the auto-covariances and lagged cross-covariances discarded by the model is negligible.
- The TSR method outperforms the PMP method when used for prediction-estimation purposes. Therefore, it is recommended the use of TSR instead of PMP. The principal difference between both methods is found at the beginning of the batch processing. Nevertheless, under some conditions -when the initial high prediction error of PMP is somehow masked- no statistically significant difference was observed between both methods.
- None of the preprocessing methods clearly outperforms the other. It should be noted that the subtraction of the average trajectory subtracts a big part of the dynamics of the process. Thus, a lower number of LMVs is needed to fit the dynamics. For variable-wise models, where no LMVs are included, an important improvement of performance can be gained by subtracting the average trajectory, instead of auto-scaling.
- No general statistically significant difference was observed between the generation of a batch-wise model from the scores of a previous variable-wise model or from raw data. Following the first strategy, data are variable-wise unfolded and a PLS model is generated. The effect of this modelling is similar to a filtering operation. In a second step, the resulting scores are rearranged as in batch-wise unfolding and modelled once again. With

this rearranging, the dynamic information is built in the model. The other strategy directly unfolds the original data batch-wise and calibrates a PLS model. Thus, the only difference between both approaches is the filtering performed in the first approach. If this filtering successfully subtracts the part of the data which is not useful for predicting, then it is convenient. If part of useful data is subtracted, then it is not convenient. It depends on the current process and application under study.

8.4 Multi model approach

The use of a single model in an on-line prediction application presents some drawbacks. Depending on the structure of the model, future measurements need to be imputed or the correlation structure is constrained to be constant throughout the batch.

An alternative way to design an on-line prediction system for a batch process is to fit a PLS model for every single sampling time of the process. Following this approach, future measurements do not need to be imputed and changing dynamics can be effectively modelled. Several proposals can be found in the literature, which mainly vary in the data used to generate the sub-models.

One of the most appreciated benefits of using PLS-based technics is that, by conveniently studying the calibrated model, the process understanding can be improved [22]. Although the calibration of a sub-model for each sampling time can be advantageous for the on-line application, such a number of sub-models can be complex to handle and difficult to interpret. To reduce the number of sub-models, the MP framework can be used.

In this section, the use of several PLS models for a single process, i.e. the multi-model approach, is investigated. Some of the conclusions can be helpful to better understand the features of the methods.

8.4.1 Modelling of the Process Dynamics

The prediction power of a model is highly related with its capacity to capture the dynamics of a process. This was the intuitive premise used in the preceding section and then proved experimentally. Also, this was discussed in Chapter 3 from the theoretical point of view. The modelling of the dynamics using PLS-based methods is carried out in two stages: preprocessing and calibration. In the second stage, the dynamics remaining in the preprocessed data -in the form of auto-covariances and cross-covariances- may be included in the model to improve the prediction performance.

There are two principal strategies to model the dynamic relationships of the variables when using multiples models. On one hand, LMVs can be explicitly included in the data used to fit the PLS-based models. On the other

hand, an adaptive hierarchical modelling approach can be used, so that the past information is implicitly taken into account in the models.

Explicit inclusion of past information

The K-models approach (see Chapter 3) is based on generating a single sub-model for every sampling time. Local models [49] are the simplest example of this approach, where the sub-model associated to a sampling time is computed from the data collected at that sampling time alone:

$$M = \{PLS(\mathbf{X}_k, \mathbf{y}_k, a_k) : k = 1, ..., K\} \tag{8.10}$$

Nonetheless, as it was seen in the previous section, the inclusion of the dynamics of the process can be crucial for a good prediction performance. A straightforward alternative to local modelling is evolving modelling [50, 49], where all the possible LMVs are included in a sub-model as additional variables:

$$M = \{PLS(\underline{\mathbf{X}}_{1:k}^{(k-1)}, \mathbf{y}_k, a_k) : k = 1, ..., K\} \tag{8.11}$$

More sophisticated approaches can be used. Not all the lagged information may be of interest or at least be given the same importance in every sub-model. Uniformly Weighted Moving Window (UWMW) or Exponentially Weighted Evolving Window (EWEW) models may be defined in order to better adjust the multi-model to the nature of the process. UWMW modelling uses the current data along with those of the immediate n_k LMVs to generate the current sub-model:

$$M = \{PLS(\underline{\mathbf{X}}_{k-n_k:k}, \mathbf{y}_k, a_k) : k = 1, ..., K\} \tag{8.12}$$

n_k is *the size of the window*, a calibration parameter of the UWMW model.

An EWEW sub-model includes all the lagged measurements up to the current sampling time:

$$M = \{PLS(\underline{\mathbf{X}}_{1:k} \odot \mathbf{W}_k, \mathbf{y}_k, a_k) : k = 1, ..., K\} \tag{8.13}$$

where $\mathbf{W}_k$ is a weighting matrix and $\odot$ stands for the Hadamard (element to element) product. $\mathbf{W}_k$ is constructed according to an exponentially decreasing value, the *forgetting factor* $\lambda_k \epsilon [0, 1]$, so that each measurement losses importance as the process advances. The weight of the measurement-vector collected at time $k - d$, for the generation of the sub-model at time k, is $(\lambda_k)^d$, being the weight of the current measurements always $(\lambda_k)^0 = 1$. The parameters n_k and λ_k may be calibrated independently for every single sub-model -that is why they subscript k is used-, for instance by using cross-validation.

Additionally, there are two ways to carry out the explicit inclusion of lagged measurements: as additional variables or as additional objects (see Chapter 3). Both arrangements can be respectively noted as:

$$M = \{PLS(\underline{\mathbf{X}}^{(n_k)}_{k-n_k:k}, \mathbf{y}_k, a_k) : k = 1, ..., K\} \tag{8.14}$$

and

$$M = \{PLS(\underline{\mathbf{X}}^{(0)}_{k-n_k:k}, \mathbf{y}_k, a_k) : k = 1, ..., K\} \tag{8.15}$$

for UWMW models, and:

$$M = \{PLS(\underline{\mathbf{X}}^{(k-1)}_{1:k} \odot \mathbf{W}'_k, \mathbf{y}_k, a_k) : k = 1, ..., K\} \tag{8.16}$$

and

$$M = \{PLS(\underline{\mathbf{X}}^{(0)}_{1:k} \odot \mathbf{W}''_k, \mathbf{y}_k, a_k) : k = 1, ..., K\} \tag{8.17}$$

for EWEW models, being $\mathbf{W}'_k$ a $I \times (J \cdot k)$ weighting matrix and $\mathbf{W}''_k$ a $(I \cdot k) \times J$ weighting matrix.

Implicit inclusion of past information

In [48], an adaptive modelling approach is presented to capture the dynamic information of a batch process. The idea beneath this approach is to compress the lagged information in the super-scores of a hierarchical PCA (HPCA) model and combine them with the local data of the current sampling time.

When translating HPCA to PLS modelling, two main alternatives are possible: the hierarchical PLS (HPLS) and the multi-block PLS (MBPLS). The main difference between them is that HPLS performs a PCA loop with the data of the single blocks and the PLS is only performed in the super-level. On the other hand, MBPLS performs a PLS loop for both levels of the model. A review of the algorithms is carried out in [62].

In EWEW and UWMW models, the parameters are determined locally. That is to say, every sub-model has a value of λ_k or n_k calibrated independently from the rest of sub-models. Also, the number of LVs a_k is locally set. Nonetheless, in HPLS and MBPLS the sub-models are interrelated in a hierarchical structure. This implies that each local model cannot be calibrated independently.

The algorithms which combine the batch adaptive modelling approach along with HPLS and MBPLS can be found in the Appendix E. As far as the authors know, this is the first time the adaptive modelling approach of [48] has been extended to PLS.

8.4.2 Strategies under study

The strategies studied in this section are listed in Table 8.4. These strategies can be classified as shown in Figure 8.11. Firstly, the MP model is distinguished from the rest, the K-models. The UWMW and EWEW models with lagged variables -i.e. LMVs included as variables, (8.14) and (8.16)- are named *UWMW-1* and *EWEW-1*. The UWMW and EWEW models with lagged objects -i.e. LMVs included as objects, (8.15) and (8.17)- are named *UWMW-2* and *EWEW-2*. Since they have been recently proposed for on-line monitoring in [49], the local (*Loc*) and evolving (*Evo*) approaches are included, although they are a particular case of the others. Finally, Hierarchical (*Hier*) and Multi-block (*MB*) adaptive PLS models are also included. As it can be seen, the objectives of this study are to compare the different ways of including lagged information into a multi-model, in order to capture the dynamics of a process, and to evaluate the performance of the MP modelling approach.

Table 8.4. Strategies using multiple PLS models of a batch process for on-line prediction. Last strategy (BW-TSR) is used to compare the multi-models with one of the best single model approaches

Acronym	Explanatory text
MP	Multi-Phase modelling [7]
Evo	Evolving models [50]
UWMW-1	Uniformly Weighted Moving Window (Variables) models
EWEW-1	Exponentially Weighted Evolving Window (Variables) models
Hier	Adaptive Hierarchical model [48]
MB	Adaptive [48] + Multi-block with super-score deflation [120]
UWMW-2	Uniformly Weighted Moving Window (Objects) models
EWEW-2	Exponentially Weighted Evolving Window (Objects) models
Loc	Local models [49]
BW-TSR	Batch-wise unfolding [118] + Trimmed Scores Regression [103]

To obtain the MP model, the parameters used in the first and second steps of the MP framework (Figure 5.5) are listed in Table 8.5. The prediction with PLS is less complicated than the PCA based on-line monitoring. First, the criterion for model calibration coincides with the objective of the application: the models are calibrated to minimize the prediction error, exactly the objective of the end-quality prediction or trajectories estimation system. Second, the computation of the prediction error is more reliable for PLS than for PCA. This is because PLS models are predictive models, an thus the prediction error is easily computed by comparing the true quality values with those predicted with the model. PCA models are compression models and it is not completely clear how to compute this error. Both facts

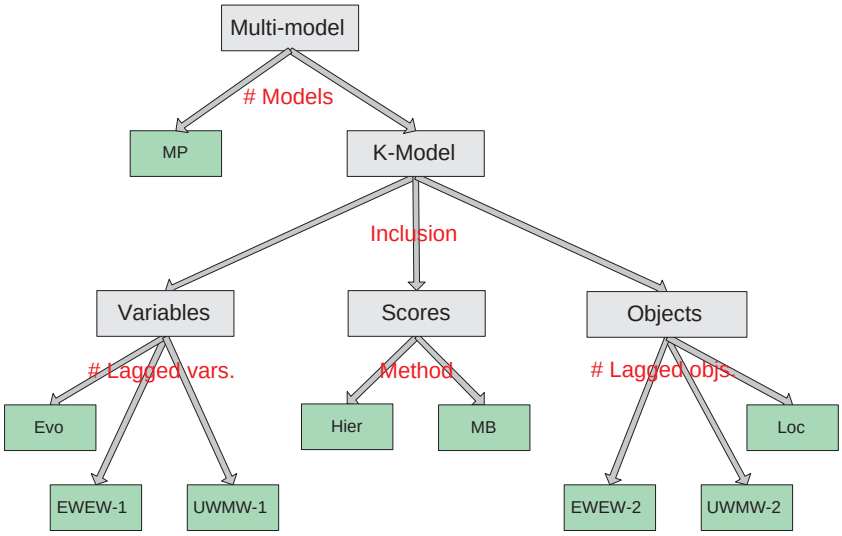


Fig. 8.11. Classification of the strategies.

make the selection of the parameters much simpler than for monitoring. First, k_γ can be set to 1. Nonetheless, using more values may be reasonable due to the heuristic nature of the MP algorithm. The number of LMVs has to be varied as much as possible, since it is unknown how much dynamic information is necessary to predict the quality. This depends on the specific nature of the process. T_m can be set to 0, since in this case it is preferred to use the best predictive model found. Also, a possibility is to play a little with this parameter to see if the complexity of the model can be reduced.

Table 8.5. Parameters used to fit MPPLS models for on-line prediction. M and M_i stand for models of the batch data set. S_i stands for a sub-model of an interval.

Multi-Phase Parameters		
$\delta(M_i, M)$	$=$	$MSE(M) - MSE(M_i)$
β^k	$=$	$SSE - SSE^k$
$\gamma(\delta(M_i, M))$	$=$	$k_\gamma \cdot \delta(M_i, M)$
k_γ	$=$	1
LMVs	$=$	$0 \rightarrow K-1$
n	$=$	0
$minL$	$=$	1
T	$=$	0
Merging Parameters		
$T_m = 0$		

To connect the results obtained with those of the preceding section, one of the best single-model approaches, *BW-TSR*, is included in the comparative. This approach is based on batch-wise unfolding and the imputation of future measurements by Trimmed Scores Regression [103].

8.4.3 Experimental Study

Cultivation of *Saccharomyces cerevisiae*

In this section, the estimation of the trajectory of the biomass concentration and the prediction of its final value, using the data of the cultivation of *Saccharomyces cerevisiae*, are investigated. The structure of the models is shown in Table 8.6. Only global parameters are shown. The LSD intervals for the MSE outcomes are presented in Figures 8.12 and 8.13. The MSE values of the method labelled *AT* are calculated assuming that the predicted trajectory is the average trajectory. This information is added as a base line.

Table 8.6. Global calibration parameters of the models in the estimation of the trajectory of the biomass concentration (first row) and the prediction of its final value (second row). *Saccharomyces cerevisiae* cultivation process. The acronym *sub* stands for the number of "sub-models". The percentage in multi-phase models is computed as $100 * sub/K$, with K the number of sampling times in a batch.

MP	Hier	MB	BW-TSR
33 sub (33%)	2 LVs, $\lambda = 0.97$	2 LVs, $\lambda = 1$	9 LVs
54 sub (54%)	2 LVs, $\lambda = 0.95$	2 LVs, $\lambda = 1$	4 LVs

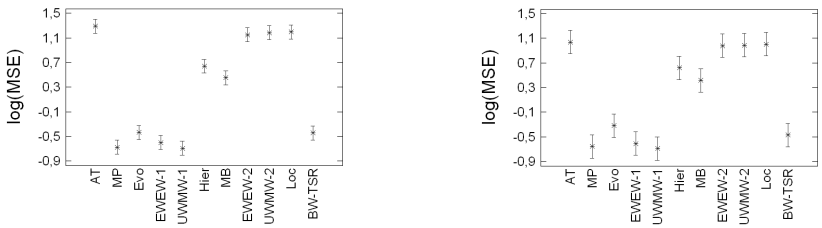


Fig. 8.12. LSD intervals for the MSE outcomes in the estimation of the trajectory of the biomass concentration in the *Saccharomyces cerevisiae* cultivation process: calibration set (left) and test set (right). The acronym *AT* means "average trajectory".

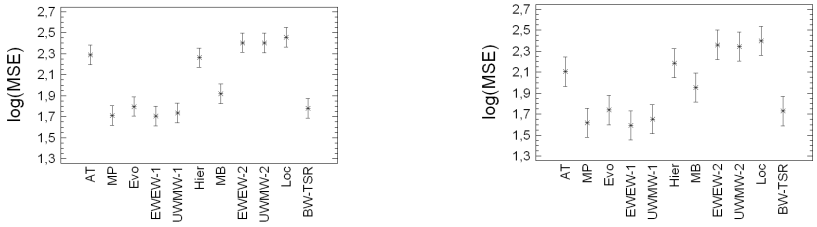


Fig. 8.13. LSD intervals for the MSE outcomes in the prediction of the final biomass concentration in the *Saccharomyces cerevisiae* cultivation process: calibration set (left) and test set (right). The acronym *AT* means "average trajectory".

The outcomes presented for both the calibration and test sets are very similar. This is indicative of a correct calibration of the models. In general terms, the best performance is presented by five models: *MP*, *Evo*, *EWEW-1*, *UWMW-1* and *BW-TSR*. In Figure 8.14, the MSE of these approaches is presented as a function of the sampling time. *MP*, *EWEW-1* and *UWMW-1* outperform *Evo* and *BW-TSR* during some periods of the batch. Nevertheless, the improvement is not enough to cause a statistically significant difference in the LSD intervals. *Evo* and *BW-TSR* show a very similar performance. Therefore, as in online monitoring (Chapter 7) the use of several sub-models does not necessarily imply a benefit in terms of performance.

Multi-block and hierarchical approaches are less efficient in capturing the dynamic information of this process. As it was explained before, these approaches are less flexible than EWEW or UWMW in the sense that the parameters have to be globally set. Nonetheless, this is not their only drawback. In Figure 8.15, the adaptive approaches are compared with the *EWEW-1* approach with a number of LVs and λ globally set for the whole batch duration -i.e. $a_k = a$ and $\lambda_k = \lambda$ for $k = 1, ..., K$ in (8.16). Even for the same number of LVs than the adaptive approaches (2 LVs), the *EWEW-1* model yields better performance. The adaptive methods compress the past information in the scores and combine it with the current information. Part of this past information may be lost in the process. If the information discarded is useful in subsequent sampling times, the adaptive approaches present poorer performance than *EWEW-1* models. This may explain the results observed.

EWEW-2, *UWMW-2* and *Loc* models present the poorest performance. The conclusion arising is that the dynamics are not captured by including LMVs as additional objects -as expected from the theoretical discussion of Chapter 3 and the results shown in the previous section. Also, since the three approaches perform very similarly, a clear benefit of including lagged objects is not observed.

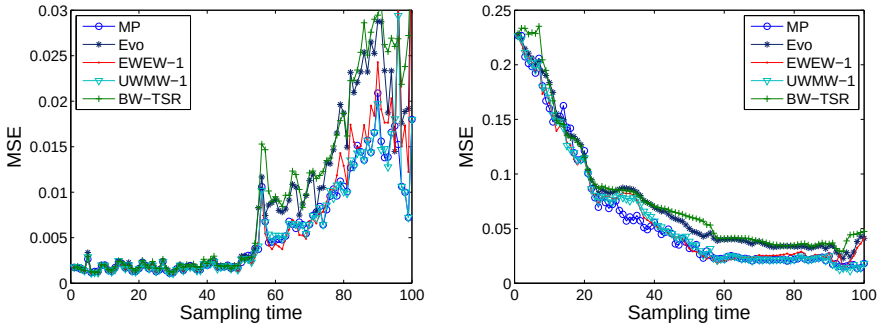


Fig. 8.14. MSE in the estimation of the biomass concentration trajectory (a) and prediction of its final value (b) along the batch processing. Approaches shown are MP (circles), Evo (asterisks), EWEW-1 (points) , UWMW-1 (triangles-down) and BW-TSR (pluses).

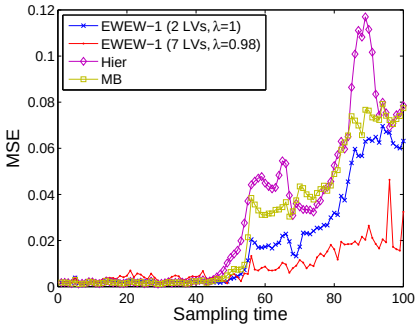


Fig. 8.15. MSE in the estimation of the biomass concentration trajectory. Approaches shown are EWEW-1 with globally set parameters (crosses and points), Hier (diamonds) and MB (squares).

Now let us focus on the *MP* approach and its capability to handle the process data. The MP framework is able to reduce the number of sub-models to a third part in trajectory estimation and to a half part for final quality prediction (see Table 8.6), without loss of performance with respect to *UWMW-1* and *EWEW-1*. Moreover, by conveniently defining the criterion of model complexity in the merging algorithm (Section 5.3.2), the number of sub-models can be further reduced using $T_m > 0$. Imagine φ in the merging algorithm is defined as:

$$\varphi(S_1, S_2) = phases_1 - phases_2 \tag{8.18}$$

where $phases_i$ is the number of sub-models included in sub-model S_i. Also, the number of LMVs can be reduced. For that, φ can be defined as:

$$\varphi(S_1, S_2) = \overline{LMV}_1 - \overline{LMV}_2 \tag{8.19}$$

where $\overline{LMV}_i$ is the average number of LMVs of sub-model S_i.

For instance, the *MP* model for the estimation of the biomass trajectory, computed with T_m set to 10^{-4} and (8.18), finds out only 16 sub-models, instead of the 33 obtained for $T_m = 0$. The parameters of the resulting 16 sub-models are presented in Table 8.7. The first column specifies the MSE of each sub-model computed for the interval it represents. This MSE is growing with the sampling time, consistently with the MSE trajectory in Figure 8.14(a). The third column shows the large number of LMVs used in the sub-models of the phases. In the preceding section, it was seen that when using a single batch dynamic PLS (BDPLS) model, a high number of LMVs may be needed to model changing dynamics. That high number may be necessary even when the long-term dynamics of the process are negligible. Therefore, the reason of the need of LMVs is not clearly distinguishable with a single model. Since in the *MP* model changing dynamics are modelled using different sub-models, it can be stated that the large number of LMVs is showing how important is the long-term dynamic information for the prediction in this process. Notice that the first sub-model, which represents the phase from sampling time 1 to 56, has 51 LMVs. Therefore, during the first 51 sampling times, the model has to impute the future measurements. This is performed with the TSR imputation method (Appendix D). From 52 to 56, the sub-model works with complete objects. Notice that, following this philosophy, the *MP* approach includes the *BW-TSR* approach as a special case.

Let us perform a further analysis of the process using the MP framework. The performance of several MP models obtained for different T_m values and for (8.18) and (8.19) will be compared using an ANOVA. The values used for T_m, as well as the resulting number of sub-models and the $\overline{LMV}$ included in them, are presented in Table 8.8. The table shows that, in this process, a reduction in the number of sub-models implies an increase in the past information needed ($\overline{LMV}$). This is caused by the changing dynamics of the process. When the number of sub-models is forced to reduce, the changing dynamics have to be modelled with an initial batch dynamic model with a high number of LMVs. Like in *MP-A1* (Table 8.7), models *MP-A2* and *MP-A3* present a long first phase where the number of LMVs is almost the number of sampling times.

The LSD intervals for the MSE outcomes included in the ANOVA are presented in Figure 8.16. Only the analysis of the test set is shown, since the calibration set presented similar results. *UWMW-1* and *BW-TSR* have been included in the analysis, since both are extreme cases -extreme in the number of sub-models- included in the *MP* approach.

Table 8.7. MPPLS model for $T_m = 10^{-4}$ and (8.18) for the estimation of the biomass concentration trajectory in the *Saccharomyces cerevisiae* cultivation process (Total MSE=0.006).

MSE	LVs	LMVs	Beginning	End
0.0022	4	51	1	56
0.0058	8	51	57	61
0.0063	7	51	62	65
0.0064	6	51	66	67
0.0078	7	51	68	74
0.0093	6	61	75	77
0.0108	7	61	78	82
0.0141	5	61	83	84
0.0150	6	61	85	86
0.0187	9	81	87	90
0.0150	8	81	91	94
0.0163	8	81	95	96
0.0106	8	71	97	97
0.0100	8	71	98	98
0.0072	8	71	99	99
0.0180	7	61	100	100

Table 8.8. T_m used for the calibration of several MPPLS models along with the number of sub-models and the average number of LMVs. Estimation of the biomass concentration trajectory in *Saccharomyces cerevisae* process. Models named MP-An use (8.18) in the definition of φ. Models named MP-Bn use (8.19) in the definition of φ.

Name	T_m	Sub-models	$\overline{LMV}$
MP	0	33	49.79
MP-A1	10^{-4}	16	62.88
MP-A2	$1.5 \cdot 10^{-4}$	5	81
MP-A3	$2 \cdot 10^{-4}$	2	86
MP-B1	$2 \cdot 10^{-4}$	44	30.77
MP-B2	$4 \cdot 10^{-4}$	42	23.62
MP-B3	$6 \cdot 10^{-4}$	39	21

Firstly, let us focus our attention again on the first four *MP* models (*MP*, *MP-A1*, *MP-A2* and *MP-A3*). In these models, the objective was the reduction of the number of sub-models (8.18). As it can be seen, this reduction does not have a big impact in the prediction performance. This is because changing dynamics can be conveniently modelled using a high number of lagged variables. That is the reason why the performance of the *BW-TSR* approach is so close to that of the *UWMW-1* approach.

In the last four *MP* models (*MP*, *MP-B1*, *MP-B2* and *MP-B3*) the effect of reducing the number of LMVs (8.19) is shown. As it can be seen, the decreasing of the number of LMVs has a clear impact in the performance. This

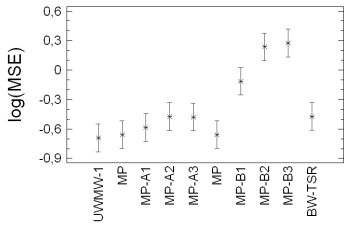


Fig. 8.16. LSD intervals for the MSE outcomes in the estimation of the trajectory of the biomass concentration in the *Saccharomyces cerevisiae* cultivation process: Test set.

was expected from the high number of lagged variables in the *MP* models, which shows the importance of the dynamic information in the performance of a predictive model of this process.

A clear benefit of *MP* is that the structure of the PLS model is automatically identified. Thus, changing dynamics are successfully distinguished from long-term dynamics. Additionally, relevant information for process understanding is gained by conveniently using the MP framework.

Waste-water treatment

In this section, the estimation of the phosphorus content trajectory and the prediction of its final value, in a waste-water treatment process [100], are investigated. The global parameters of the models for both applications are shown in Table 8.9. The LSD intervals for the MSE outcomes are presented in Figures 8.17 and 8.18. The MSE of AT is added as a base line. Nonetheless in Figure 8.18, AT was eliminated after a first analysis (not shown), since it yielded such a poor performance (average MSE equal to 770) that the differences among the other approaches could not be appreciated.

Table 8.9. Calibration parameters of the models in the estimation of the trajectory of the phosphorus content (first row) and the prediction of its final value (second row). Waste-water treatment process. The acronym *sub* stands for the number of "sub-models". The percentage in multi-phase models is computed as $100 * sub/K$, with K the number of sampling times in a batch.

MP	Hier	MB	BW-TSR
2 sub (11%)	4 LVs, $\lambda = 0.7$	3 LVs, $\lambda = 0.9$	4 LVs
164 sub (65%)	1 LVs, $\lambda = 1$	1 LVs, $\lambda = 1$	1 LVs

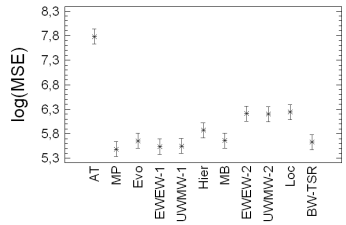


Fig. 8.17. LSD intervals for the MSE outcomes in the estimation of the trajectory of the phosphorus content in the waste-water treatment process. The acronym *AT* means "average trajectory".

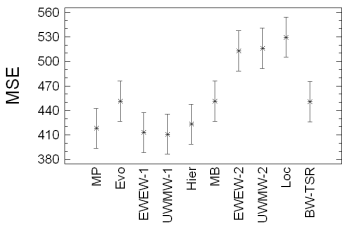


Fig. 8.18. LSD intervals for the MSE outcomes in the prediction of the final phosphorus content in the waste-water treatment process. The acronym *AT* means "average trajectory".

As in the previous process, the approaches that include the dynamics as lagged objects (*EWEW-2* and *UWMW-2*) together with *Loc*, show very similar performance and were outperformed by the rest PLS approaches. Again, this implies that the inclusion of lagged objects does not generally improve the performance of the models.

In Figure 8.19, The MSE of the other approaches in the prediction of final phosphorus content are plotted as a function of the sampling time. The estimation of the trajectory is not shown since no significative differences are found. The figure shows a zoom from sampling time 1 to 150, so that differences among approaches can be better observed. In general, all approaches present a very similar shape in the MSE. Initially, the MSE falls fast and reaches its minimum at sampling time 85 (approximately). Then, it follows a smooth rising trend for the rest of the batch. A similar behavior was observed in the previous section for the single-model approaches. The shape of the MSE trajectories evidences that the principal information regarding the final phosphorus content is collected during the first 85 sampling times -at

least, the principal information linearly related to the final phosphorus content. This interval coincides with the anaerobic stage of the process, where the trajectory of the phosphorus content reaches its highest values. It is reasonable to think that the maximum phosphorus content registered by a batch is related with the residual content at the end of that batch. This is coherent with the biological and physicochemical interpretation given in [121]. From sampling time 85 onwards, the collected measurements have a noisy effect on the prediction. The *Hier* approach seems to be more robust to that noise than the rest approaches, since its performance during the last 100 sampling times is slightly superior than the rest.

The fact that the principal information for prediction is at the beginning of the batch can be also seen by looking at the λ_k parameter set for the *EWEW-1* multi-model, in Figure 8.20. The parameter is set to 1 from sampling time 100 to almost the end of the batch. This means that the past information is very important from that sampling time onwards.

The *MP* approach showed a very surprising result: for the first 85 sampling times interval a single model with 18 LMVs is identified, whereas during the rest of the batch, almost as many sub-models as sampling times are necessary. Apart from the first sub-model -from 1 to 85- the rest present a high number of LMVs -almost all possible LMVs are included- and higher PRESS values. This is revealing that the information from sampling time 86 onwards present little predictive power, making thus necessary to include lagged information from the first stage in the sub-models.

By comparing the model obtained with $T_m = 0$ with that for $T_m = 0.01$ using criterion (8.19), it is seen that the lagged information -the 18 LMVs- can be discarded from the sub-model with a negligible reduction of the performance. As a conclusion, it could be stated that the final phosphorus content can be conveniently predicted with a single variable-wise model of the first 85 sampling times.

In Figure 8.21, the loadings of the first LV of the sub-model for interval [0-85] and 0 LMVs are compared with those of the batch-wise unfolded model of the process data. The latter are shown in Figure 8.21(a), where the vertical subdivision separates the initial 85 sampling times from the rest. During the first interval, the variables present reasonably stable loadings. From that point onwards, the loadings of variables 1, 3 and 4 are changing. The loadings of the variables in the MPPLS model of the first interval (8.21(b)) are very similar (in proportion) to the stable loads shown in Figure 8.21(a). The MPPLS model presents the information in a much more parsimonious form, specially suited for control purposes, yielding also a better performance in terms of MSE.

Now let us focus our attention on the phosphorus content trajectory estimation. As shown in Table 8.10, the *MP* model for trajectory estimation has only 2 sub-models. Nonetheless, its $\overline{LMV}$ is 11, being the length of the batch 18. It could be interesting to investigate if the performance is very dependent

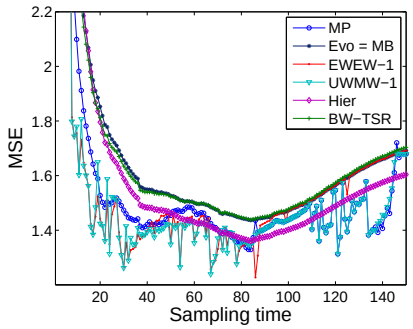


Fig. 8.19. MSE in the prediction of the final phosphorus content from sampling time 1 to 150. Approaches shown are MP (circles), Evo (asterisk, coinciding with MB) , EWEW-1 (points), UWMW-1 (triangles-down), Hier (diamonds) and BW-TSR (pluses).

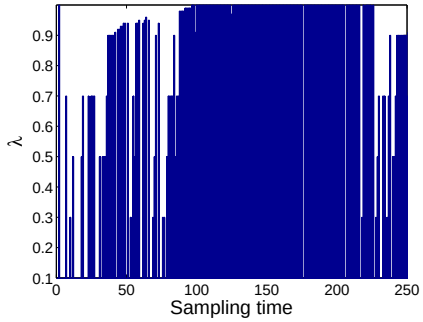
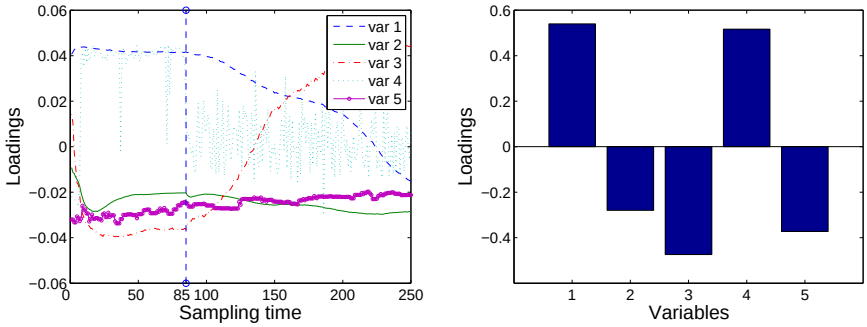


Fig. 8.20. λ parameter along the batch processing, set for the EWEW-1 approach. Prediction of the final phosphorus content.

on the past information, as it happened in the previous process, or if the number of LMVs can be reduced. With the MP framework, four *MP* models were obtained. Their global parameters are shown in Table 8.10, along with those of the original *MP* model. The models are compared in Figure 8.22. It can be seen that the $\overline{LMV}$ can be forced to be below 3 (*MP-B1* and *MP-B2*), with almost no loss of performance. If the $\overline{LMV}$ is reduced below 2, the performance is significantly worsened (Figure 8.22). Note that the evaluation of the LSD intervals is very important to guarantee the good performance of the final model. It can be concluded that, unlike in the previous process, a high number of LMVs is not needed for the trajectory estimation of the phosphorus content.



(a) (b)

Fig. 8.21. Loadings of first latent variable in the batch-wise model (a) and the MP model ($T_m = 0.01$) for interval $[1, 85]$ (b). Prediction of the final phosphorus content.

Table 8.10. T_m used for the calibration of several MPPLS models along with the number of sub-models and the average number of LMVs. Estimation of the trajectory of the phosphorus content in the waste-water treatment process.

Name	T_m	Sub-models	$\overline{LMV}$
MP	0	2	11
MP-B1	0.1	8	2.94
MP-B2	1	7	2
MP-B3	10	6	1
MP-B4	100	2	0

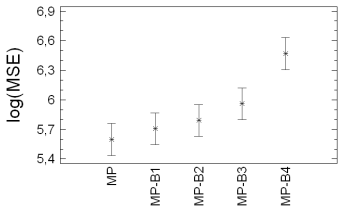


Fig. 8.22. LSD intervals for the MSE outcomes in the estimation of the trajectory of the phosphorus content in the waste-water treatment process.

8.4.4 Conclusions

This is the second one of a pair of comparatives devoted to the study of the on-line application of PLS models in batch processes. In the previous section, the approaches based on a single PLS model were treated. Here, the multi-model approach was investigated.

The comparative included: evolving and local models; exponentially and uniformly weighted models including LMVs as variables or as objects; adaptive hierarchical and multi-block models; multi-phase models; and a batch-wise unfolded model using Trimmed Scores Regression (TSR) imputation of future samples. This latter strategy was observed to yield a good prediction performance in the preceding section.

As far as the authors know, the adaptive hierarchical and multi-block approaches have been adapted for PLS for the first time in this document. Additionally, some modifications for improving the performance of these approaches, based on extensive experimentation, are suggested in Appendix E.

The following conclusions are drawn:

- The past information in the form of LMVs should be included as additional variables in the models. Exponentially and uniformly weighted models doing so were among the best approaches in all the cases, both presenting similar performance. When including the LMVs as objects, the models were not improved in general terms. Theoretically it was deduced that with the inclusion of lagged objects, the dynamics of the process are not built in the models, whereas with lagged variables they are.
- The principal drawback of the adaptive hierarchical and multi-block models, in comparison with the models including LMVs, is a consequence of the hierarchical structure of the formers. This structure makes it very difficult to set the parameters of the sub-models locally. The solution in this study was to set the number of LVs and the weighting factor globally for the whole batch in the adaptive approaches. This may be a limitation for modelling certain types of time varying correlation structures, as shown in the first case of study.
- The adaptive models performed differently for the two processes. Their performance in the first process were much poorer than in the second, in comparison with the rest approaches. Differences between the hierarchical and multi-block approaches were found, but none of both generally outperformed the other. The adaptive models compress the past information in the super-scores. When this compression eliminates data useful afterwards, their performance is reduced.
- The evolving and local models are especial cases of exponentially and uniformly weighted models. Evolving models outperformed local models since no dynamic information is included in the latter.
- The evolving approach presented a very similar performance to that of the single batch-wise unfolded model with TSR imputation. Therefore, using a larger number of sub-models alone do not necessarily implies any

improvement. Exponentially and uniformly weighted models, as well as multi-phase models fitted with the MP framework, slightly outperformed both the evolving and batch-wise approaches in some periods of the batch. This improvement is achieved by adjusting locally the amount of past information included in the sub-models. Anyway, statistical significant differences (p-value < 0.05) in the MSE computed for the complete batch were not found among the five approaches.

8.5 General Conclusions

The MP approach is a generalization of many of the approaches studied in this investigation, including: batch-wise unfolded models, variable-wise unfolded models, batch dynamic models, evolving and local models and uniformly weighted models. In principle, it could be stated that the benefits of using one approach or another depends on the nature of the process under study. The advantage of using the MP is that the structure of the convenient model for each case of study can be inferred directly from the process data.

The PLS models have the capacity of discovering certain features of a process, as it was shown for the two processes studied. Different arrangements of the three-way data in the models highlight different features. The MP framework can be used to perform a general, in deep, analysis of a process. Throughout the experiments performed it has been shown that the MP framework not only yields good outcomes in terms of prediction performance, but it is an interesting tool for the analysis of batch processes.

9 Optimization of batch processes

Part of the contents of this chapter have been included in the following paper:

[4] J. Camacho, J. Picó and A. Ferrer. *Self-tuning run to run optimization of fed-batch processes using unfold-PLS*, AIChE Journal, 53(7):1789-1804 (2007).

9.1 Introduction

Batch and fed-batch processing is the general trend to manufacture low quantities of high value products [122]. The biotechnological industry is an example of this kind of production. A large portion of bio-reactors 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 [14]. Additionally, fed-batch is believed to outperform batch processing in fermentation processes [123]. In fed-batch fermentations, the feeding strategy needs to be somehow optimized since both underfeeding and overfeeding lead to a decrease of product formation [124].

The optimization and advanced control of this kind of processes is a challenging task due to the lack of measurements of principal variables [79]. The uncertainties in the process behavior and their non-linear nature and slow response complicate this task [80]. The design of first principles models is an important research line [19] and the fundamental knowledge is commonly used in the control system design. Jobé et *al.* [81] regulate the substrate feeding according to an estimation of the metabolic state of the process. Picó-Marco et *al.* [82, 83] 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 [84].

The reliance on the process knowledge has the drawback that some of the assumed dynamic relationships may not be valid in all cases, i.e. 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 (e.g., 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 [27, 67]. A more radical approach is to rely completely on input-output data, like in Iterative Learning Control methods [88, 92, 93, 94], which take advantage of the repetitive nature of batch processes. Also, according to Lee et *al.* [124], 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 [95, 96] 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.* [97]:

- *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 [23] or combined with fundamental knowledge [98]. 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 to the gradient law computed with a model [25].
- *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 [97].

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, PLS-based methods are powerful empirical modelling tools. The utility of batch-wise unfold PLS (or simply uPLS, for briefness) for the optimization of control trajectories in batch processes has already been reported [125].

In this chapter, a self-tuning extremum-seeking controller [126] based on uPLS is proposed to optimize the feeding profile of a fed-batch process. Extremum-seeking controllers can be roughly classified as model-based [127, 128], which rely on assumptions on the dynamics of the process and the objective function, and model free [129, 130]. 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 is assumed. The optimization is performed on a gradient basis using data from previously processed batches. The uPLS 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 chapter is organized as follows: Section 9.2 describes the optimization procedure proposed in this chapter. In section 9.3, the first principles model used for simulation is explained. Section 9.4 presents the experimental results. Section 9.5 gives some conclusions and remarks.

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

9.2 Self-tuning Extremum uPLS

Self-tuning systems combine a recursive estimation algorithm with a synthesis algorithm [126]. In this approach, the estimation algorithm is an adaptive uPLS 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 [126]:

$$\mathbf{u}_{t+1} = \mathbf{u}_t + c \cdot \frac{dy}{d\mathbf{u}} \tag{9.1}$$

with:

$$\frac{dy}{d\mathbf{u}} = \left[\frac{dy}{du_1}, ..., \frac{dy}{du_J} \right]^T \tag{9.2}$$

where $\mathbf{u}_t$ 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 $\mathbf{u}$ represents the feeding profile. By constructing an uPLS 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 uPLS model. Notice that the batch-wise unfolding procedure 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 uPLS model (see Chapter 3).

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 $\mathbf{y}$. Matrix $\underline{\mathbf{U}}$ is batch-wise unfolded and the resulting matrix $\mathbf{U}$ and vector $\mathbf{y}$ are auto-scaled -i.e. 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:

$$\frac{dy}{d\mathbf{u}} = \frac{dy}{dy^n} \cdot \frac{dy^n}{d\mathbf{u}^n} \odot \frac{d\mathbf{u}^n}{d\mathbf{u}} \tag{9.3}$$

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

$$\frac{dy}{dy^n} = \sigma_y \tag{9.4}$$

$$\frac{d\mathbf{u}^n}{d\mathbf{u}} = \mathbf{1}_{JK} \oslash \boldsymbol{\sigma}_{\mathbf{u}} \tag{9.5}$$

$$\boldsymbol{\sigma}_{\mathbf{u}} = \left[\sigma_{u_{1,1}}, ..., \sigma_{u_{1,K}}, ..., \sigma_{u_{J,1}}, ..., \sigma_{u_{J,K}}\right]^T \tag{9.6}$$

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 uPLS model corresponds to the matrix of regressors $\hat{B}_{PLS}$ (Section 1.3.1). 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:

$$\left(\frac{\hat{dy^n}}{d\mathbf{u}^n}\right) = \mathbf{W} \cdot (\mathbf{P}^T \cdot \mathbf{W})^{-1} \cdot \mathbf{q}^T \tag{9.7}$$

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, equation (9.3) conveys all the information required to modify the complete feeding law so as to improve the performance of next batch. Combining equations (9.3), (9.4), (9.5) and (9.7) yields:

$$\left(\frac{\hat{dy}}{d\mathbf{u}}\right) = \sigma_y \cdot (\mathbf{W} \cdot (\mathbf{P}^T \cdot \mathbf{W})^{-1} \cdot \mathbf{q}^T) \oslash \sigma_{\mathbf{u}} \tag{9.8}$$

The run to run optimization algorithm proposed follows the common steps of a self-tuning algorithm [126], being:

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

Step 2: Rebuild the uPLS 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 uPLS 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 uPLS model. Thus, the control law (9.1) must be modified to:

$$\mathbf{u}_{i+1} = \bar{\mathbf{u}}_i + c \cdot \left(\frac{\hat{dy}}{d\mathbf{u}}\right)_i \tag{9.9}$$

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

$$\left(\frac{\hat{dy}}{d\mathbf{u}}\right)_i = \sigma_{y,i} \cdot (\mathbf{W}_i \cdot (\mathbf{P}_i^T \cdot \mathbf{W}_i)^{-1} \cdot \mathbf{q}_i^T) \oslash \boldsymbol{\sigma}_{\mathbf{u},i} \tag{9.10}$$

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

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 inter-batch 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 model. An alternative to the calibration of the model using the last I batches is the use of an exponentially weighted PLS model [104, 64]. The other estimation parameters, such as the standard deviations in (9.10), are also recomputed with the last I batches.

In order to guarantee the convergence to a (local) extremum, a dither signal is added to (9.9) 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:

$$\mathbf{u}_{i+1} = \bar{\mathbf{u}}_i + c \cdot \left(\frac{\hat{dy}}{d\mathbf{u}}\right)_i + \boldsymbol{\rho}_{i+1} \tag{9.11}$$

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 uPLS model.

9.2.1 Heuristic rules

In practice the strategy proposed may not work properly when the objective function y is not convex with respect to the variables $\mathbf{u}$, 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 (9.10) can be estimated and used to limit the step calculated in (9.11).

[b] 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.

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 [71] (see Chapter 6), can be computed for the feeding law and be used to restrict the solution of (9.11). 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 (9.11) by means of the quality predicted with the uPLS 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 uPLS model. The prediction of the quality for batch $i+1$ with the uPLS model follows:

$$\hat{y}_{i+1} = \bar{y}_i + c \cdot \hat{y}_{\mathbf{u},i+1} + \hat{y}_{\boldsymbol{\rho},i+1} \tag{9.12}$$

where $\bar{y}_i$ is the average quality of the batches used to fit the uPLS 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:

$$\hat{y}_{\mathbf{u},i+1} = \sigma_{y,i} \cdot \hat{y}^n_{\mathbf{u},i+1} \tag{9.13}$$

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

$$\hat{y}^n_{\mathbf{u},i+1} = (\left(\frac{\hat{dy}}{d\mathbf{u}}\right)_i \oslash \boldsymbol{\sigma}_{\mathbf{u},i})^T \cdot \left(\frac{\hat{dy_n}}{d\mathbf{u}_n}\right)_i \tag{9.14}$$

The first term in (9.14) performs the scaling operation. The second one performs the prediction using the PLS model.

The approach used in this paper 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 $\bar{y}_i \pm 3 \cdot \sigma_{y,i}$ 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 follows:

$$\left(\frac{\tilde{dy}}{d\mathbf{u}}\right)_i = \begin{cases} \left(\hat{\frac{dy}{d\mathbf{u}}}\right)_i & \hat{y}^n_{\mathbf{u},i+1} \leq 3 \\ \frac{3}{\hat{y}^n_{\mathbf{u},i+1}} \cdot \left(\hat{\frac{dy}{d\mathbf{u}}}\right)_i & \hat{y}^n_{\mathbf{u},i+1} > 3 \end{cases} \tag{9.15}$$

The behavior of the optimization can be further improved by tuning the parameter c of (9.11) conveniently. Here, the following adaptive rule is used:

$$\tilde{c}_i = 1 - \left(\frac{PRESS_A}{PRESS_0}\right)_i \tag{9.16}$$

where $PRESS_A$ represents the PRESS computed for the uPLS model with A LVs, $PRESS_0$ is PRESS computed only for the average output -i.e., 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 $(1 - PRESS_A/PRESS_0)$[c] is an indicator of how well the uPLS model is predicting the data. 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 non-linear 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, i.e. 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):

[c] 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 auto-scaling 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 $[0, 1]$.

$$\tilde{\boldsymbol{\rho}}_{i+1} = \left(\frac{PRESS_A}{PRESS_0}\right)_i \cdot g \cdot \boldsymbol{\rho}_{i+1} \tag{9.17}$$

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 next expression:

$$\mathbf{u}_{i+1} = \bar{\mathbf{u}}_i + \tilde{c}_i \cdot \left(\frac{\tilde{dy}}{d\mathbf{u}}\right)_i + \tilde{\boldsymbol{\rho}}_{i+1} \tag{9.18}$$

9.2.2 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 uPLS 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:

$$\mathbf{U}_{aug,i} = [\mathbf{Z}_i, \mathbf{U}_i] \tag{9.19}$$

with $\mathbf{Z}_i$ representing a $I \times L$ matrix of L initial measurements performed over I samples (batches). Afterwards, the PLS model is computed for normalized matrices $\mathbf{U}_{aug,i}$ and $\mathbf{y}_i$. 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 (9.15). 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:

$$\hat{y}^n_{\mathbf{u},i+1} = (\left(\frac{\hat{dy}}{d\mathbf{u}}\right)_i \oslash \boldsymbol{\sigma}_{\mathbf{u},i})^T \cdot \mathbf{W}_{\mathbf{U},i} \cdot (\mathbf{P}_i^T \cdot \mathbf{W}_i)^{-1} \cdot \mathbf{q}_i^T \tag{9.20}$$

$$\hat{y}^n_{\mathbf{z},i+1} = ((\mathbf{z}_{i+1} - \bar{\mathbf{z}}_i) \oslash \boldsymbol{\sigma}_{\mathbf{z},i})^T \cdot \mathbf{W}_{\mathbf{Z},i} \cdot (\mathbf{P}_i^T \cdot \mathbf{W}_i)^{-1} \cdot \mathbf{q}_i^T \tag{9.21}$$

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 $\mathbf{U}_i$.

Then, the effect of the initial variability can be used to adjust the correction of the gradient in (9.15):

$$\left(\frac{\tilde{dy}}{d\mathbf{u}}\right)_i = \begin{cases} \left(\frac{\hat{dy}}{d\mathbf{u}}\right)_i & \hat{y}^n_{\mathbf{u},i+1} + \hat{y}^n_{\mathbf{z},i+1} \leq 3 \\ \mathbf{0}_{JK} & \hat{y}^n_{\mathbf{z},i+1} > 3 \\ \frac{3-\hat{y}^n_{\mathbf{z},i+1}}{\hat{y}^n_{\mathbf{u},i+1}} \cdot \left(\frac{\hat{dy}}{d\mathbf{u}}\right)_i & otherwise \end{cases} \tag{9.22}$$

This extension implies that the initial state $\mathbf{z}_{i+1}$ should be measured before processing the next batch $i+1$ to compute (9.21). This might be difficult in some applications. In practice, the initial state can be taken into account to fit the uPLS model -using $\mathbf{U}_{aug,i}$- but not to constrain the gradient. Thus, $\hat{y}^n_{\mathbf{z},i+1} = 0$ can be assumed, making equations (9.15) and (9.22) 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 non-linear processes.

Several performance indices

If a number of M performance indices are to be optimized, each one with an associated weighting factor $\psi_1, \psi_2, ..., \psi_M$, 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 have to be normalized, but incorporating the weight information:

$$\mathbf{Y}_i^{n(\boldsymbol{\psi})} = (\mathbf{Y}_i - \mathbf{1}_I \cdot \bar{\mathbf{y}}_i^T) \odot (\mathbf{1}_I \cdot (\boldsymbol{\psi} \oslash \boldsymbol{\sigma}_{\mathbf{y},i})^T) \tag{9.23}$$

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 $\boldsymbol{\psi} = [\psi_1, \psi_2, ..., \psi_M]$. To use the self-tuning optimization procedure, the step or *weighted gradient* is computed as:

$$\left(\frac{\hat{d\mathbf{y}}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} = (\left(\frac{\hat{d\mathbf{y}^{n(\boldsymbol{\psi})}}}{d\mathbf{u}^n}\right)_i^T \cdot (\boldsymbol{\sigma}_{\mathbf{y},i} \oslash \boldsymbol{\psi})) \oslash \boldsymbol{\sigma}_{\mathbf{u},i} \tag{9.24}$$

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 uPLS model:

$$\left(\frac{d\hat{\mathbf{y}^{n(\boldsymbol{\psi})}}}{d\mathbf{u}^n}\right)_i = (\mathbf{W}_i \cdot (\mathbf{P}_i^T \cdot \mathbf{W}_i)^{-1} \cdot \mathbf{Q}_i^T)^T \tag{9.25}$$

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

$$\mathbf{u}_{i+1} = \bar{\mathbf{u}}_i + \tilde{c}_i \cdot \left(\frac{\hat{d\mathbf{y}}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} + \tilde{\boldsymbol{\rho}}_{i+1} \tag{9.26}$$

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

$$\hat{\mathbf{y}}_{\mathbf{u},i+1}^{n(\boldsymbol{\psi})} = \left(\frac{d\hat{\mathbf{y}^{n(\boldsymbol{\psi})}}}{d\mathbf{u}^n}\right)_i \cdot \left(\left(\frac{\hat{d\mathbf{y}}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} \oslash \boldsymbol{\sigma}_{\mathbf{u},i}\right)^T \tag{9.27}$$

$$\hat{\mathbf{y}}_{\mathbf{u},i+1} = \hat{\mathbf{y}}_{\mathbf{u},i+1}^{n(\boldsymbol{\psi})} \odot \boldsymbol{\sigma}_{\mathbf{y},i} \oslash \boldsymbol{\psi} \tag{9.28}$$

In order to keep the same idea of (9.15), 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 $\hat{\mathbf{y}}_{\mathbf{u},i+1}^{n(\boldsymbol{\psi})}$ is computed as follows:

$$\ell_{i+1} = \sqrt{\hat{\mathbf{y}}_{\mathbf{u},i+1}^{n(\boldsymbol{\psi})T} \cdot \mathbf{S}_i^{-1} \cdot \hat{\mathbf{y}}_{\mathbf{u},i+1}^{n(\boldsymbol{\psi})}} \tag{9.29}$$

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:

$$\left(\frac{\tilde{d\mathbf{y}}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} = \begin{cases} \left(\frac{\hat{d\mathbf{y}}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} & \ell_{i+1} \leq 3 \\ \frac{3}{\ell_{i+1}} \cdot \left(\frac{\hat{d\mathbf{y}}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} & \ell_{i+1} > 3 \end{cases} \tag{9.30}$$

And the optimization law:

$$\mathbf{u}_{i+1} = \bar{\mathbf{u}}_i + \tilde{c}_i \cdot \left(\frac{\tilde{d\mathbf{y}}}{d\mathbf{u}}\right)_{\boldsymbol{\psi},i} + \tilde{\boldsymbol{\rho}}_{i+1} \tag{9.31}$$

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. A**n** 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 (9.30). 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 -e.g. $\psi_1 >> \psi_2$.

Constraints Handling

Assume soft constraints of the form:

$$\mathbf{f}(\mathbf{x}) \leq \mathbf{0} \tag{9.32}$$

where $\mathbf{x}$ is a vector of process -or quality- variables collected during the batch processing which are not manipulable[d].

The first step is to create a uPLS 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:

$$\left(\frac{d\hat{x}^n}{d\mathbf{u}^n}\right)_i = \mathbf{W}_{x,i} \cdot (\mathbf{P}_{x,i}^T \cdot \mathbf{W}_{x,i})^{-1} \cdot \mathbf{q}_{x,i}^T \tag{9.33}$$

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

$$\left(\frac{\hat{dx}}{d\mathbf{u}}\right)_i = \sigma_{x,i} \cdot \left(\frac{d\hat{x}^n}{d\mathbf{u}^n}\right)_i \oslash \boldsymbol{\sigma}_{\mathbf{u},i} \tag{9.34}$$

Using this, the value of x_{i+1} can be predicted for the control law $\mathbf{u}_{i+1}$ returned by the optimization algorithm (9.18):

$$\hat{x}_{i+1} = \bar{x}_i + \left(\frac{\hat{dx}}{d\mathbf{u}}\right)_i^T \cdot (\mathbf{u}_{i+1} - \bar{\mathbf{u}}_i) \tag{9.35}$$

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:

[d] If they were manipulable, the constraint could be handled ad-hoc.

$$\Delta_{\hat{x}_{i+1}} \; s.t. \; f(\hat{x}_{i+1} - \Delta_{\hat{x}_{i+1}}) \leq \lambda \leq 0 \tag{9.36}$$

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:

$$\Delta_{\mathbf{u}_{i+1}} = \Delta_{\hat{x}_{i+1}} \cdot \mathbf{h} \oslash \left(\frac{d\hat{x}}{d\mathbf{u}} \right)_i \tag{9.37}$$

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 (uPLS) gradient $\left(\frac{d\hat{x^n}}{d\mathbf{u}^n}\right)_i$ is recommended here. Following this approach equation (9.37) yields:

$$\Delta_{\mathbf{u}_{i+1}} = \frac{\Delta_{\hat{x}_{i+1}}}{\sigma_{x,i} \cdot \sum \left(\frac{d\hat{x^n}}{d\mathbf{u}^n}\right)_i} \cdot \boldsymbol{\sigma}_{\mathbf{u},i} \tag{9.38}$$

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:

$$\tilde{\mathbf{u}}_{i+1} = \mathbf{u}_{i+1} - \Delta_{\mathbf{u}_{i+1}} \tag{9.39}$$

Two comments about this approach for constraints handling are in due. Firstly, this approach should not be used for hard -safety- constraints because the violation of these cannot be guaranteed. Secondly, if cross-validation of the uPLS model of (9.33) suggests 0 latent variables -i.e. the feeding law is not correlated with the constrained variable-, no gradient information is available. Therefore no adjustment can be done -(9.38) cannot be computed. This makes reasonable in practice to modify (9.17) by:

$$\tilde{\boldsymbol{\rho}}_{i+1} = (1 - (1 - \left(\frac{PRESS_A}{PRESS_0} \right)_i) \cdot (1 - \left(\frac{PRESS_{x,A'}}{PRESS_{x,0}} \right)_i)) \cdot g \cdot \boldsymbol{\rho}_{i+1} \tag{9.40}$$

with $PRESS_{x,A'}/PRESS_{x,0}$ computed from cross-validation of model (9.33).

The extension to handle several constraints can be performed either using a single uPLS 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.

9.3 Cultivation of *Saccharomyces cerevisiae*

The first principles model of the cultivation of *Saccharomyces cerevisiae* presented by Lei et *al.* [19] is used to test the optimization procedure proposed

in this chapter. The model includes 12 stoichiometry reactions and 9 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 paper.

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.* [131]. These were also used by Lei et *al.* [19] to test the first principles model in fed-batch simulation. The initial solution has 7 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 [19]). 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 hours 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-dimensional space. Notice that this is a much more ambitious optimization than those pursued in many other papers.

The simulations performed present a $\pm 1\%$ of measurement noise in the final biomass concentration as well as in the initial conditions. Initial variable conditions were simulated by adding a $\pm 5\%$ 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 [131] and [19] -an exponential phase of 3 hours followed by a constant phase of 7 hours- 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 0. 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.

9.4 Simulation Study

9.4.1 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 9.1. This optimization was performed starting from scratch -i.e. null feeding law- with a uPLS model fitted from the last 20 batches. The first 20 circles in Figure 9.1(a) represent the initial set of batches. The optimization is very fast until the volume reaches the maximum established (see Figure 9.1(d)). 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 9.1(a)). 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 [131] and [19], which obtains a performance close to 10 g/l.

The evolution of the feeding law can be seen in Figure 9.1(b). 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 9.1(c). 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 9.1(a), 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 the maximum value of the performance index[e], 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

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

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 9.1(a), the optimum found by the GA is that of the extremum controller.

Now lets pay attention to the number of batches used to generate the model. Figure 9.2 shows the results of an optimization performed with the same parameters as those used in Figure 9.1, 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 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 9.2(b)) is not exactly equal to that of the previous one (Figure 9.1(b)). This is expected to happen because of the local optimality of the solution found and the uncertainty associated to it. This last due 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, etc. 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 paper, 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

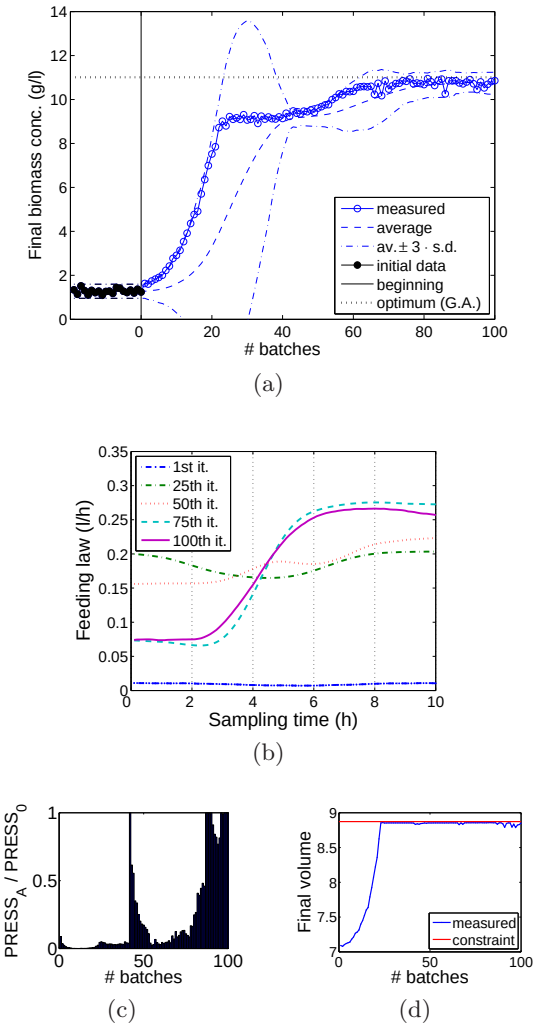
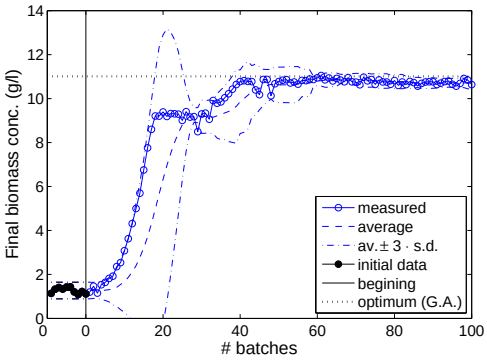
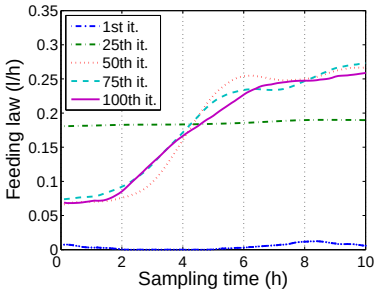


Fig. 9.1. Simulation performed with an adaptive model generated with 20 batches. Initial variability of ±5% (initial values included in the model) and measurement noise of ±1%. PRBS signal of adaptive amplitude ($g = 0.03$). The Figure includes: (a) performance of the batches along with the evolution of the average and ± 3 times the standard deviation. The vertical line marks the beginning of the optimization. The horizonal dotted line marks the result obtained with a genetic algorithm; (b) feeding law for different iterations; (c) evolution of index $PRESS_A/PRESS_0$; (d) evolution of the final volume.



(a)



(b)

Fig. 9.2. 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 horizonal dotted line marks the result obtained with a genetic algorithm; (b) feeding law for different iterations.

variables in a batch process [2]. 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 *Saccharomyces cerevisiae* example. Twenty optimizations were performed for each of the five cases presented in Table 9.1 an three values of the amplitude of the excitation signal -g equal to 0.01, 0.03 and 0.05. The options of Ta-

ble 9.1 allow to study separately the influence of g in the ideal case (Option 1), the effect of non-explained 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).

Table 9.1. Different options for the simulation.

Opt. 1: No initial variability nor measurement noise.
Opt. 2: Initial variability ($\pm 5\%$) and measurement noise ($\pm 1\%$)
Opt. 3: **Opt. 2** + Filtering of the PRBS, the gradient and the final feeding law.
Opt. 4: **Opt. 3** + Addition of initial biomass conc. ($\hat{y}^n_{\mathbf{z},i+1} = 0$ assumed in (9.22))
Opt. 5: **Opt. 3** + Addition of initial biomass conc.

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 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 ANOVA was performed on the d value, showing statistical significative differences (p-value ≤ 0.05) due to g, the measurement noise and their interaction. The least significative difference (LSD) intervals for a 95% confidence are shown for the interaction plot in Figure 9.3.

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 non-reliable 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 9.3, 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 non-linear 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 9.3). 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 9.1) 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, 5 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 significant. Yet, the two-order interactions between the measurement noise an the other two factors were significant. The LSD intervals of a 95% confidence are shown for these interactions in Figure 9.4.

From Figure 9.4(a) 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 significantly 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 9.4(b), 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.

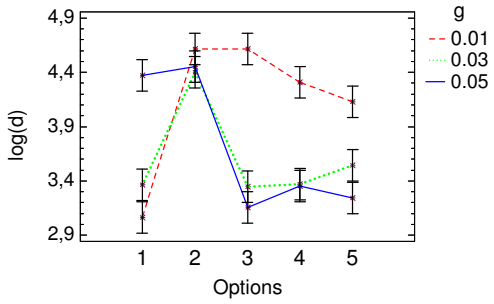
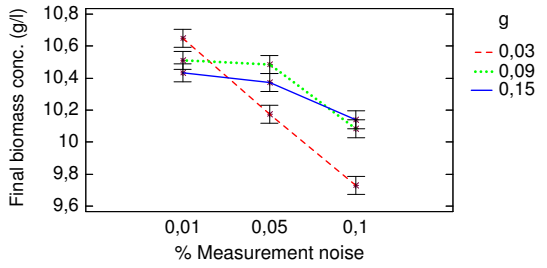
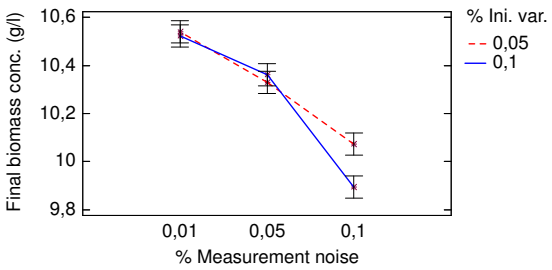


Fig. 9.3. Analysis of variance of the influence of the options in Table 9.1 in the performance of the optimization process (d, number of batches to reach a significant improvement) for different g values. The logarithm of d was used to correct for normality. Least significative difference (LSD) intervals of a 95% confidence are shown for the interaction plot between g and the options.



(a)



(b)

Fig. 9.4. 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.

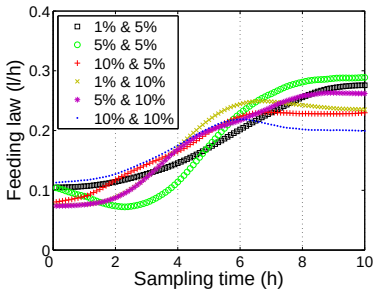


Fig. 9.5. Examples of feeding laws optimized for different percentages of measurement noise and of initial variability.

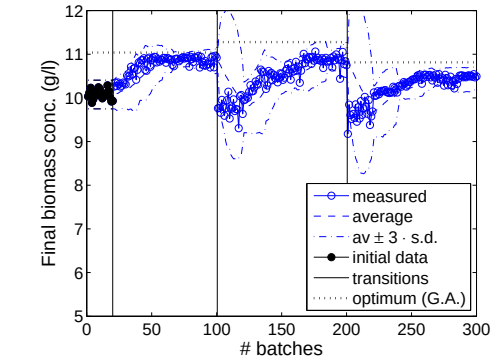
It is clear that the optimization algorithm is not completely robust to noise nor to initial variability. Therefore, the repetition of quality measurements to reduce the noise and the tight control in the initial conditions seems justified in some critical cases. Finally, it is worth noticing that in all cases the self-tuning optimizer identified feeding profiles of similar shape (see Figure 9.5).

9.4.2 Adaptation to process changes

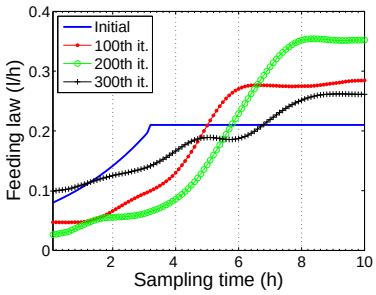
One of the additional advantages of the presented approach is that the feeding law is adapted when state changes or transitions in the process occur. This kind of transitions can be due to changes in the environmental conditions, degradation of the process and maintenance operations. Transitions can 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 [19]. 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.

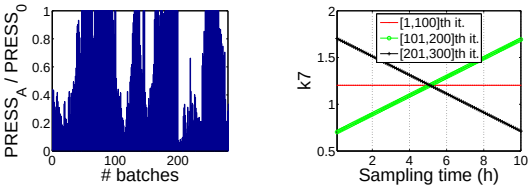
In Figure 9.6, a simulation with several sudden transitions is presented. The simulation is started using the feeding law proposed in [131]. This contains an exponential phase of 3 hours followed by a constant phase of 7 hours (see Figure 9.6(b)). The first part of the optimization (up to batch number 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



(a)



(b)



(c) (d)

Fig. 9.6. 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 horizonal 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 $PRESS_A/PRESS_0$; (d) value of $k7$ during the batch for every period.

of the batch time (see Figure 9.6(d)). 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 9.6(b)). 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 9.6(c).

9.4.3 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:

$$J_1 = -(12 - \chi_f)^2 \tag{9.41}$$

$$J_2 = \frac{\chi_f}{v_f} \tag{9.42}$$

where χ_f is the final biomass concentration and v_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 (9.41) and (9.42) as:

$$J = J_1 + J_2 \tag{9.43}$$

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

The result of the optimization of J is shown in Figure 9.7. After convergence (approximately from batch number 60), the following batches present a final biomass concentration close to 12 g/l with a volume slightly bigger than 9.2 l. 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 uPLS model. The result is shown in Figure 9.8. Although some slight differences may be seen between both cases (Figures 9.7 and 9.8), a more detailed investigation showed no general differences in the velocity of convergence nor in the final error.

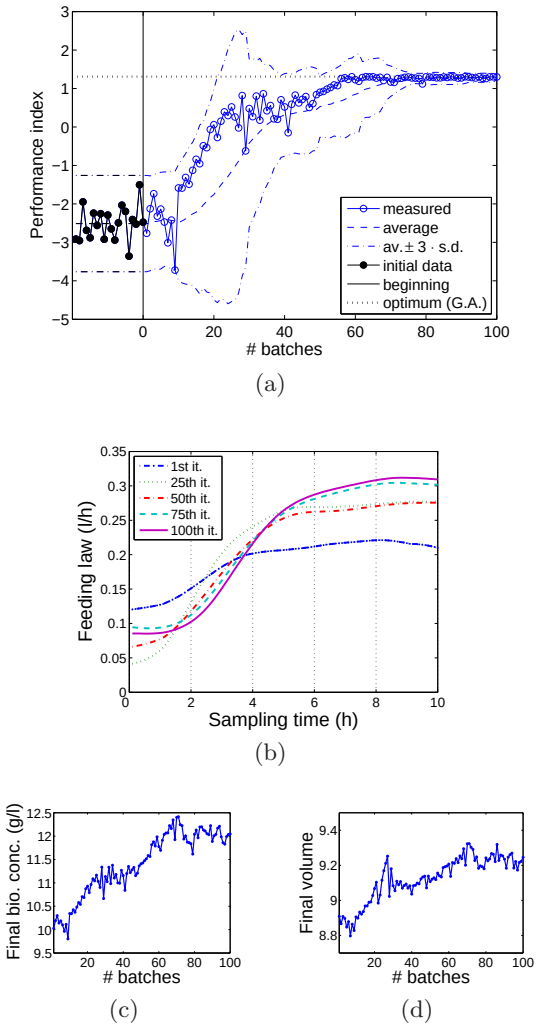


Fig. 9.7. Simulation for the combination of two performances indices in one. Optimization performed with an adaptive model generated with 20 batches. Initial variability of ±5% (initial values included in the model) and measurement noise of ±1%. PRBS signal of adaptive amplitude ($g = 0.03$). The Figure 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 horizonal dotted line marks the result obtained with a genetic algorithm; (b) feeding law for different iterations; (c) evolution of the final biomass concentration; (d) evolution of the final volume.

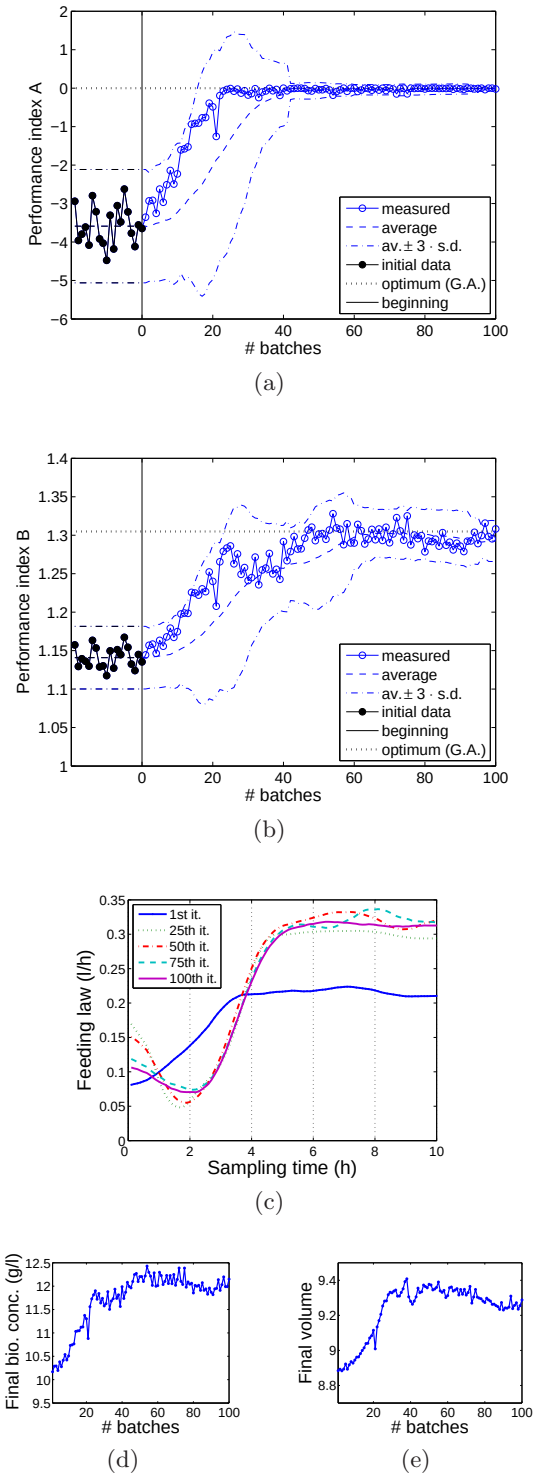
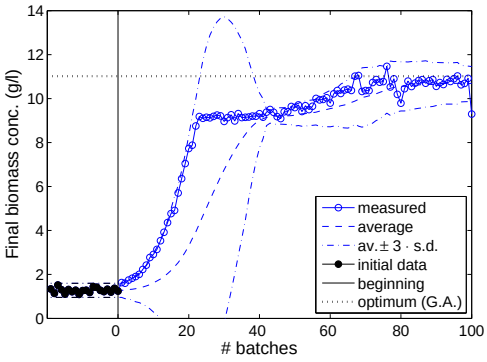
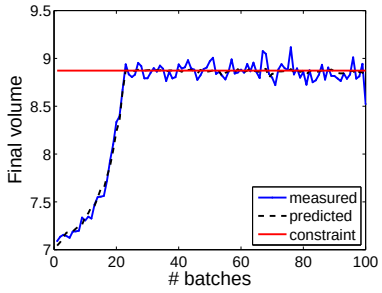


Fig. 9.8. Simulation for two performances indices. Optimization performed with an adaptive model generated with 20 batches. Initial variability of ±5% (initial values included in the model) and measurement noise of ±1%. PRBS signal of adaptive amplitude ($g = 0.03$). The Figure includes: (a) first performance 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 horizonal 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; (d) evolution of the final volume.



(a)



(b)

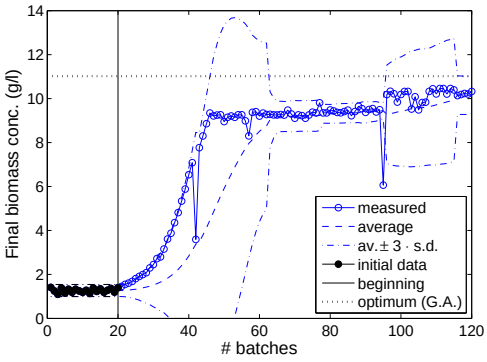
Fig. 9.9. Simulation performed with two adaptive models generated with 20 batches for both optimization and to constrain the volume. Initial variability of ±5% (initial values included in the model) and measurement noise of ±1%. PRBS signal of adaptive amplitude ($g = 0.03$) and null uncertainty margin ($\lambda = 0$). 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 horizonal dotted line marks the result obtained with a genetic algorithm; (b) evolution of the final volume.

9.4.4 Constraints handling

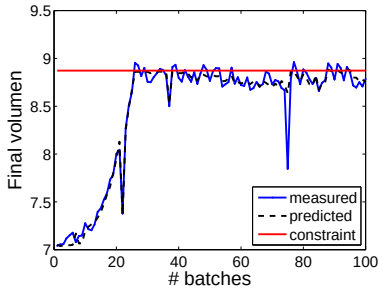
In order to check the performance under presence of constraints, the experiment of Figure 9.1 was repeated using the information of a uPLS model for handling the volume limitation:

$$f(v) = v - v_{max} \tag{9.44}$$

Figure 9.9 shows an optimization trial for $\lambda = 0$ -no security margin due to uncertainty is used. The performance obtained is equivalent to that of Figure 9.1. The constraint is violated several times but the process is maintained around it. Such a result is very interesting since this approach



(a)



(b)

Fig. 9.10. 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 horizonal dotted line marks the result obtained with a genetic algorithm; (b) evolution of the final volume.

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 9.10, the optimization is repeated for:

$$\lambda_i = -3 \cdot (1 - \left(\frac{PRESS_{v,A'}}{PRESS_{v,0}} \right)_i) \cdot \sigma_{v,i} \tag{9.45}$$

where the idea is to maintain the volume at a distance to the maximum proportional to the reliability of the uPLS model. If all the variability in the

volume is explained by the uPLS model -i.e., $(PRESS_{v,A'}/PRESS_{v,0})_i \simeq 0$-, 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 9.10 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.

9.5 Conclusions

In this chapter, 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 batch-wise PLS model. The approach has proven to be very effective and versatile, low sensitive to non-linearities 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 multi-variate problems but inherits the limitations of self-tuning extremum-seeking controllers, i.e. 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.

Part IV

Conclusions.

10 Conclusions

This Thesis is devoted to the study and application of methods based on the projection to latent structures (PLS-based methods), in particular Principal Component Analysis (PCA) and Partial Least Squares (PLS), to batch processes. First, a theoretical study was performed in order to understand the principal problems in the application of bilinear PLS-based methods to batch data. Second, a number of algorithms were developed for the modelling of batch processes overcoming these problems. Finally, some industrial applications in which a model of the process is necessary were investigated, including the off-line and on-line monitoring, the on-line end-quality prediction, the on-line estimation of the trajectory of a variable and the optimization of batch processes.

Here, the main conclusions of the work are summarized, organized according to the objectives presented at the beginning of the document:

a) Identification of the limitations in the modelling of batch processes with PLS-based methods.

The work presented in this document was restricted to the application of bilinear models, in particular of PCA and PLS, to batch data. Approaches for modelling batch data based on PCA or PLS can be classified in two main groups: single model and multi-model approaches. The latter, in turn, can be divided in K-models approaches and multi-phase approaches. The single-model approaches use the same model for all the batch duration. The K-models approaches fit a single model per sampling time. Finally, the multi-phase approaches fit a different model for certain periods of the batch, given a criterion for the identification of the periods which should be modelled separately.

In single model approaches, the three-way matrix of data is unfolded into a two-way matrix and then PCA or PLS is applied. The most well-known unfolding directions for batch process data, i.e. batch-wise unfolding and variable-wise unfolding, can be seen as special cases of a general unfolding method: the batch dynamic unfolding. In the batch dynamic unfolding, a number of lagged measurement vectors (LMVs) are added to the current measurement vector to form the object of the unfolded two-way matrix. If no LMV is added, the resulting matrix is the same yielded with variable-wise

unfolding. If all possible LMVs are added, then the unfolding matrix is the same yielded with batch-wise unfolding. The advantage of the batch dynamic unfolding is that the number of LMVs of the model can be optimized for a particular process. This is, a-priori, a more powerful modelling approach than using an extreme case, like in batch-wise or variable-wise unfolding.

There are two different reasons for the addition of LMVs in the single model approach: the modelling of the dynamics of a certain order and/or the modelling of changing correlation structures[a]. The more LMVs added, the more dynamic information included in the model. Therefore, variable-wise models do not capture the dynamics of the process whereas batch-wise models do. Moreover, modelling with a reduced number of LMVs assumes a constant correlation structure during the batch. The more LMVs included in the model, the shorter the interval where the correlation structure is imposed to be constant and the more capable the model is to capture changes in the dynamics. At the same time, the dynamics built in the model are more dependent on the precise sampling time. On the other hand, for the sake of parsimony of the resulting model[b], the less the number of variables in the unfolded matrix the better. Also, the inclusion of independent or non-linearly related variables in the same object is not recommendable for PCA nor for PLS. As a conclusion, the number of LMVs in a single-model approach should be kept as low as possible, but being high enough to capture the -possibly changing- dynamics of the process. This compromising solution tends to yield over-parameterized models.

The K-models approach is based on the calibration of one PCA or PLS model per sampling time. This approach is able to capture changing dynamics and avoids problems with non-linearity and independence between different phases of the process. Nonetheless, its principal drawback is the high number of sub-models, which makes the calibration and interpretation challenging. This is also the less parsimonious approach and so a larger data set is needed for a proper calibration.

Multi-phase models overcome the limitations of single-model approaches with a reduced number of sub-models. As commented, the addition of LMVs in a single model approach may be necessary to capture dynamics of a certain order and/or changing correlation structures. This makes difficult to interpret these models and to draw conclusions regarding the dynamic behavior of the process. Moreover, these models tend to be over-parameterized. In multi-phase models, dynamics of a certain order are effectively modelled by adding a sufficient number of LMVs and changing correlation structures are captured

[a] See Appendix A for an explanation of what is understood as changing correlation structures.

[b] Taking into account solely the part of the model applicable to new incoming data. For instance, in PCA the number of parameters is understood as the number of elements in the loadings matrix and in PLS the number of elements in the regressors matrix.

by using different sub-models for different periods of the batch. Therefore, contrarily to what happens with the single model approaches, the structure of the model corresponds to the dynamic nature of the process.

The multi-phase is the most flexible and potent modelling approach of those studied here. Nonetheless, its calibration may be a challenging task since the appropriate partition of the processes in several sub-models has to be identified. Therefore, the convenience of this approach relies on the existence and performance of a calibration algorithm for multi-phase models.

b) Proposal of new methods to overcome the limitations found in the previous step

To calibrate multi-phase models, both a performance function and a calibration algorithm have to be defined. The calibration algorithm is an optimization algorithm where the objective is to find the multi-phase model which maximizes the performance function for a specific process or data-set.

A typical performance function in data modelling is the -quadratic- prediction error. Computing of the prediction error is straightforward in PLS, but not in PCA. This is because PLS models are predictive models, thus the prediction error is easily computed by comparing the true output -quality- values with those predicted with the model. PCA models are compression models and it is not completely clear how to compute this error. Two novel cross-validation algorithms have been proposed in this Thesis to estimate the prediction error in a PCA model. The algorithms were compared with some well-known approaches using both simulated and real data, yielding very good results. Subsequently, the algorithms to compute the prediction error for PCA and PLS models were extended for their application to batch three-way data. These extended algorithms can be used to compute the performance function for the calibration of multi-phase PCA and PLS models.

Regarding the calibration algorithm for multi-phase models, a new analysis framework has been developed: the Multi-phase (MP) analysis framework. This approach provides the flexibility to adjust the model structure to the dynamic nature of the process under study: number of phases, long or short term dynamics, etc. The terms *"analysis framework"* refer to the fact that the proposed approach is more than a calibration algorithm. It can be used in a straightforward and computationally efficient way to investigate the process under study. Therefore, it is also a process understanding and investigation tool.

The MP analysis framework is based on the data-driven identification of the (PCA or PLS) model structure, using the combination of a calibration algorithm, a set of post-processing algorithms and analysis of variance (ANOVA). The first algorithm, named MP algorithm, is the basic tool for the calibration of multi-phase models. It is based on a Divide and Conquer Greedy approach. The post-processing algorithms are defined to handle the multi-phase models in an computationally efficient way. Those algorithms

used in combination with the ANOVA allow to obtain a compromise solution between model complexity and performance, instead of simply the *"best solution"*. This is interesting for model simplification and process understanding.

c) Application of the developed methods in different real problems.

Several problems have been addressed in this work: the off-line and on-line process monitoring, the on-line end-quality prediction, the on-line estimation of batch trajectories and the process optimization. All these problems share the common feature that batch processes are difficult to understand and to model, making thus the application of data-based methods appropriate, in particular of PLS-based methods. Also, each of the problems studied presents its particularities, which need to be addressed independently.

The first application investigated was the off-line or end-of-batch monitoring. In a preliminary study, the performance of the MP algorithm to identify the adequate division in sub-models of a process was tested. It was found that the MP algorithm outperforms all the other multi-model approaches investigated. Moreover, the ability of the MP algorithm to distinguish single-phase from multi-phase processes was proven. After this preliminary study, the generation of an off-line monitoring system from the MP approach was studied. The MP framework provides a complete tool for analyzing processes, including the elimination of outliers, the detection of the phases and the analysis of the impact of model simplifications in the modelling quality. Also, it was observed that the MP modelling based on PCA yields more informative monitoring charts than the traditional off-line monitoring approach. In the MP charts, some information regarding the time when a fault occur is additionally provided. This, in turn, makes possible the recognition of common failure patterns directly on the monitoring charts.

Regarding the on-line monitoring of batch processes, it was found that the use of inappropriate modelling structures might cause undesirable effects on the monitoring performance. Batch-wise models, variable-wise models, local and evolving models presented some drawbacks in their application. In both batch-wise and evolving models one of the monitoring charts, the D-statistic, was almost useless being all the detection capabilities centered in the other -the SPE chart. This is a drawback since both charts are thought to be complementary, so that faults found in them are supposed to point out different types of abnormalities. Moreover, the statistics computed from batch-wise and evolving models were auto-correlated and caused the distortion of the faulty interval in the charts. Nonetheless, this feature is interesting for detecting faults which are more evident in accumulated form. Batch-wise, evolving and local models presented a higher overall type I risk than expected as a consequence of their over-parametrization. The local models were found to present slow detection capabilities. Finally, the variable-wise models gave a poor response when the correlation structure of the process was changing

and they were also slow in detecting faults. The MP framework developed monitoring systems with a fast response to faults and reasonable overall type I risk values. Nonetheless, it was stressed that the criterion used for model calibration in the MP framework -i.e. the performance function- does not coincide with the objective of the application. The criterion for model fitting is to minimize the prediction error whereas the objective of the monitoring is to distinguish faulty from NOC batches. Although these two objectives are intuitively close, the models which optimize each of them do not necessarily match. The use of new criteria for model fitting, more directly related to the objective in monitoring, would further improve the performance of the on-line monitoring system.

Several approaches were compared in the on-line end-quality prediction and the on-line estimation of batch trajectories. In a first study, the single model approaches were investigated, including the batch-wise, variable-wise and the batch dynamic models. The preprocessing and the method for imputing future measurements, concretely Projection to Model Plane and Trimmed Scores Regression, were also studied. A second comparison, devoted to the multi-model approaches, included: evolving and local models; exponentially and uniformly weighted models; adaptive hierarchical and multi-block models; and multi-phase models. The principal aim of these investigations was to understand how dynamics are built in the models. The main conclusions stated from the theoretical analysis were proved experimentally in the comparatives. From the results, it is advised to add LMVs as new variables -columns- to capture the dynamic information. Models doing so yielded the best modelling performance. The principal drawback of the adaptive approaches, in comparison with the models including LMVs, is the difficulty to set the parameters of the sub-models locally. The application of the MP framework with PLS modelling was among the best approaches in all the cases, providing more parsimonious and understandable models. Moreover, throughout the experiments performed it was shown that the MP framework not only yields good outcomes in terms of prediction performance, but it is an interesting tool for the analysis of batch processes.

Finally, a new procedure to optimize batch processes was proposed. This procedure follows a self-tuning extremum-seeking algorithm performed in combination with an adaptive batch-wise unfolded PLS model. For this current application, the batch-wise PLS was chosen for the sake of simplicity. The optimization approach proved to be very effective and versatile, low sensitive to non-linearities 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 control law is 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

multi-variate problems but inherits the limitations of self-tuning extremum-seeking controllers, i.e. 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- were discussed.

Future lines

This Thesis opens some interesting research lines:

- The application of optimization and pattern recognition methods in combination with multivariate statistical methods for the modelling of batch processes, so that more complex dynamical structures can be identified. This is interesting not only for improving the performance in certain applications, but also for model interpretability and process understanding. The aim in the application of optimization and pattern recognition methods should never be the fully automation of the process modelling, but the development of tools to aid in the analysis, modelling and interpretation of the data. This is seen here in the use of the algorithms within the MP framework in combination with the ANOVA and the LSD intervals plots.
- The development of new loss indices in the calibration of batch process models for their application to monitoring. It is customary to fit models so as to minimize the prediction error. Nonetheless, the performance of a monitoring system would be improved if the model is fitted directly for monitoring, i.e. to maximize the fault detection capability of the system. These new loss indices should not assume a set of common faults is available (unsupervised approach).
- The development of EWMA and CUSUM charts for the on-line monitoring of batch processes. The effect of using these type of charts, instead of the traditional Shewhart charts, may be similar to the use of batch-wise or evolving models. Therefore, cumulative faults are expected to be more evident in EWMA and CUSUM charts than in Shewhart charts. It seems to be reasonable to fit a model so that it meets the structure of the dynamics of the process. Then, Shewhart, EWMA and CUSUM charts may be used for the optimum detection of different types of faults.
- The use of fuzzy transitions or gaussian mixture models between the phases of a batch process, which may improve the modelling of nonlinear dynamics and the robustness to alignment errors. The use of fuzzy logic for multi-phase modelling was originally proposed in [2] and has recently been investigated in [106]. Nonetheless, there is much work to be done in this matter.
- The extension of the MP framework to the alternative multi-phase partition based on disjoint sub-matrices of data (Appendix B). This partition has the nice feature to use both past and future information for prediction.

- The extension of the MP framework to three-way modelling methods. Three-way modelling methods present similar limitations to bilinear methods. Therefore, a MP approach is also suitable. Nonetheless, the excessive computation time may be a serious difficulty to overcome.
- The use of the MP framework within a predictive control approach.
- The application of the optimization algorithm developed in Chapter 9 in other optimization problems. A specially interesting investigation may be to use this algorithm in the modelling of batch processes within the Multi-Phase framework.

Part V

Appendices.

Notation

General Notation

- $\underline{\mathbf{A}}$: Tree-way matrix.
  - $\underline{\mathbf{A}}(B \times C \times D)$: Tree-way matrix of dimension $B \times C \times D$.

- $\underline{\mathbf{A}}^{(n)}$: Three-way matrix of batch data unfolded with n lagged measurement vectors added as variables.

- $\underline{\mathbf{A}}_{k_i:k_e}$: Sub-matrix of $\underline{\mathbf{A}}$ containing the batch data from sampling time k_i to k_e.

- $\mathbf{A}$: Two-way matrix.
  - $\mathbf{A}_b$: Two-way matrix from $\underline{\mathbf{A}}$ specified by b. Two-way matrix specific to b.
  - $\mathbf{A}(B \times C)$: Two-way matrix of dimension $B \times C$.

- $\mathbf{a}$: Vector (column vector by default).
  - $\mathbf{a}_b$: Vector from $\mathbf{A}$ specified by b. Vector specific to b.
  - $\mathbf{a}_{bc}$: Vector from $\underline{\mathbf{A}}$ specified by b and c. Vector specific to b and c.
  - $\mathbf{a}(B)$: Vector of length B.

- A: Constant or Statistic.
  - A_b: Constant or Statistic specific to b.

- $A(b)$: Constraint imposed on b.

- $a(b)$: Function of parameter b.

- a: Scalar.
  - a_b: Scalar from $\mathbf{a}$ specified by b.
  - a_{bc}: Scalar from $\mathbf{A}$ specified by b and c.
  - a_{bcd}: Scalar from $\underline{\mathbf{A}}$ specified by b, c and d.

- $\odot$: Hadamard, Schur or element-wise matrix product.

– $\oslash$: Hadamard, Schur or element-wise matrix division.
– $*$: Khatri-Rao product.
– $\otimes$: Kronecker product.
– $'+'$: Moore-Penrose inverse.

Specific Notation

– $\mathbf{Z}$: Matrix containing the measurements of the initial conditions for a set of batches of a process.
– $\underline{\mathbf{X}}$: Aligned matrix containing the measurements of the process variables at different sampling times for a set of batches of a process.
– $\underline{\mathbf{U}}$: Matrix containing the manipulable variables at different sampling times for a set of batches of a process.
– $\mathbf{Y}$: Matrix containing the measurements of the quality variables at the end of the batch for a set of batches of a process.
– $\underline{\mathbf{Y}}$: Aligned matrix containing the measurements of the quality variables at different sampling times for a set of batches of a process.
– A: Number of LVs in a PLS-based model.
– a: LV index.
– I: Number of batches in a data set.
– i: Batch index.
– L: Number of measurements of the initial conditions in a batch. Number of sub-models in a multi-model.
– J: Number of process variables in a batch.
– j: Process variable index.
– M: Number of quality variables in a batch.
– m: Quality variable index.
– K: Number of sampling times in a batch, specific for $\underline{\mathbf{X}}$.
– k: Sampling time index.
– K_y: Number of sampling times in a batch, specific for $\underline{\mathbf{Y}}$.
– n: Number of lagged measurement-vectors included in a model. Superscript for normalized data.

Glossary

The sense in which one should understand some of the terms and acronyms used in this document is as follows:

- **ALS:** Alternating Least Squares.
- **ANN:** Artificial Neural Networks.
- **ANOVA:** ANalysis Of VAriance.
- **ARL:** Average Run Length.
- **AS:** Auto-Scaling.
- **AT:** Average Trajectory.
- **BD:** Batch Dynamic.
- **BMPC:** Batch Model Predictive Control.
- **CGF:** Criterion of Goodness of Fit.
- **CMR:** Conditional Mean Replacement.
- **CUSUM:** CUmulative SUM.
- **D-chart:** D-statistic monitoring chart.
- **EWMA:** Exponentially Weighted Moving Average.
- **EWEW:** Exponentially Weighted Evolving Window.
- **GA:** Genetic Algorithm.
- **HPCA, HPLS:** Hierarchical Principal Component Analysis, Hierarchical Partial Least Squares.
- **ILC:** Iterative Learning Control.
- **ISL:** Imposed Significant Level.
- **LMV:** Lagged Measurement Vector.
- **LSD:** Least Significant Difference.
- **LV:** Latent Variable.
- **MBO:** Model-Based Optimization.
- **MB:** Multi-Block.
- **MC:** Miss-Classifications.
- **MIMO:** Multiple-Input Multiple-Output.
- **MISO:** Multiple-Input Single-Output.
- **MP:** Multi-Phase.
- **MPC:** Model Predictive Control.
- **MSE:** Mean Squared Error.
- **MSPM:** Multivariate Statistical Process Monitoring.
- **MW:** Moving Window.

– **NIPALS:** Non-linear Iterative PArtial Least Squares.
– **NOC:** Normal Operation Conditions.
– **NTI:** Number of Type I errors.
– **NTII:** Number of Type II errors.
– **N-way matrix:** N-dimensional matrix.
– **OLS:** Ordinary Least Squares.
– **OTI:** Overall Type I risk.
– **PARAFAC:** PARAllel FACtor analysis.
– **PC:** Principal Component.
– **PCA:** Principal Components Analysis.
– **PCR:** Principal Components Regression.
– **PID:** Proportional-Integral-Derivative.
– **PLS:** Partial Least Squares.
– **PLS-Based:** Projection to Latent Structures-Based methods.
– **PMP:** Projection to Model Plane.
– **PRESS:** PRediction Errors Sum-of-Squares.
– **QBMPC:** Quality control-combined Batch Model Predictive Control.
– **Q-chart:** Q-statistic monitoring chart.
– **QPE:** Quadratic Prediction Error.
– **QRE:** Quadratic Residual Error.
– **R2R:** Run-To-Run.
– **RMSPE:** Root Mean of Square Prediction Errors.
– **SPE:** Squared Prediction Error.
– **SVD:** Singular Value Decomposition.
– **TS:** Trajectory Scaling.
– **TSR:** Trimmed Scores Regression.
– **TVSS:** Time-Varying State Space.
– **UCL:** Upper Control Limit.
– **UWMW:** Uniformly Weighted Moving Window.
– **lagged objects:** LMVs included as additional objects -rows- in the unfolded matrix of data.
– **lagged variables:** LMVs included as additional variables -columns- in the unfolded matrix of data.
– **mode:** each of the dimensions of a N-dimensional matrix of data.
– **model (of a batch process) structure:** arrangement in two-way matrices of the batch data together with the number of LVs used to model each matrix.
– **multi-model:** the complete model of a batch process represented by one or several sub-models.
– **multi-stage model:** multi-model where sub-models correspond to stages.
– **multi-phase model:** multi-model where sub-models correspond to phases.
– **object:** each single row of a two-way matrix.
– **phase:** batch process interval data-driven recognized.
– **partition:** division of the batch duration in intervals.

– **sample:** each single value of a two-way matrix.
– **stage:** batch process interval identified by process knowledge.
– **sub-model:** model for the data of a batch process interval.
– **uPCA, uPLS:** unfold Principal Component Analysis, unfold Partial Least Squares.
– **variable:** each single column of a two-way matrix.

A Some notes on the dynamics in autoregressive models

Let the bivariate, time-discrete signal $\mathbf{s}(k) = [x(k), y(k)]$ be approximated by the following linear model of order p:

$$a_0 \cdot x(k) + \sum_{n=1}^{p} a_n \cdot x(k-n) = b_0 \cdot y(k) + \sum_{n=1}^{p} b_n \cdot y(k-n) + e(k) \qquad \text{(A.1)}$$

where some of the coefficients a_n and b_n can be null and $e(k)$ is white noise.

When the values of signal $\mathbf{s}$ at time $k-d$ have influence on the values at time k, for $d > p$, the order of the model has to be augmented to properly capture the dynamics. Then, the order of the model should match the order of the dynamics of the signal. On the other hand, when the autocorrelation and cross-correlation in signal $\mathbf{s}$ changes with the time, different coefficients $a_n(k)$ and $b_n(k)$ must be identified for different time periods. This is referred throughout this Thesis as time-varying or changing correlation structure, understanding correlation structure as the way variables are related with each other and in time. Notice in some contexts, the time-varying correlation structure may be understood differently.

Imagine the signal $\mathbf{s}$ in (A.1) represents the behavior of a batch process, where the duration of the batch processing is K sampling times. Imagine also I batches are processed from the same initial conditions, yielding data matrices $\{\mathbf{X}_1(2 \times K), \mathbf{X}_2(2 \times K), ..., \mathbf{X}_I(2 \times K)\}$ arranged in a three-way matrix of data $\underline{\mathbf{X}}(I \times 2 \times K)$. Then, the variable-wise unfolded matrix of data is:

$$\mathbf{X} = \begin{bmatrix} x_1(1) & , & y_1(1) \\ x_2(1) & , & y_2(1) \\ & ... & \\ x_I(1) & , & y_I(1) \\ \\ x_1(2) & , & y_1(2) \\ x_2(2) & , & y_2(2) \\ & ... & \\ x_I(2) & , & y_I(2) \\ \\ & ... & \\ & ... & \\ \\ x_1(K) & , & y_1(K) \\ x_2(K) & , & y_2(K) \\ & ... & \\ x_I(K) & , & y_I(K) \end{bmatrix} \tag{A.2}$$

If, for simplicity, one single PC is fitted for $\mathbf{X}$ in (A.2) with PCA, the loadings -the model- will have the form:

$$\mathbf{p} = [p_x,\ p_y] \tag{A.3}$$

and the information retained by the PCA model is:

$$\begin{bmatrix} t_1(1) \\ t_2(1) \\ ... \\ t_I(1) \\ \\ t_1(2) \\ t_2(2) \\ ... \\ t_I(2) \\ \\ ... \\ ... \\ \\ t_1(K) \\ t_2(K) \\ ... \\ t_I(K) \end{bmatrix} = \begin{bmatrix} p_x x_1(1) + p_y y_1(1) \\ p_x x_2(1) + p_y y_2(1) \\ ... \\ p_x x_I(1) + p_y y_I(1) \\ \\ p_x x_1(2) + p_y y_1(2) \\ p_x x_2(2) + p_y y_2(2) \\ ... \\ p_x x_I(2) + p_y y_I(2) \\ \\ ... \\ ... \\ \\ p_x x_1(K) + p_y y_1(K) \\ p_x x_2(K) + p_y y_2(K) \\ ... \\ p_x x_I(K) + p_y y_I(K) \end{bmatrix} \tag{A.4}$$

This model structure is only able to model signals of the form of (A.1) for $p = 0$. Moreover, time-varying coefficients cannot be modelled. This modelling approach assumes a signal of the form:

$$a_0 \cdot x(k) = b_0 \cdot y(k) + e(k) \tag{A.5}$$

Now, imagine a batch dynamic unfolding with 1 LMVs is applied. Then, the resulting unfolded matrix is:

$$\mathbf{X} = \begin{bmatrix} x_1(1), & y_1(1), & x_1(2), & y_1(2) \\ x_2(1), & y_2(1), & x_2(2), & y_2(2) \\ \quad ... & & & \\ x_I(1), & y_I(1), & x_I(2), & y_I(2) \\ & & & \\ x_1(2), & y_1(2), & x_1(3), & y_1(3) \\ x_2(2), & y_2(2), & x_2(3), & y_2(3) \\ \quad ... & & & \\ x_I(2), & y_I(2), & x_I(3), & y_I(3) \\ & & & \\ \quad ... & & & \\ \quad ... & & & \\ & & & \\ x_1(K-1), & y_1(K-1), & x_1(K), & y_1(K) \\ x_2(K-1), & y_2(K-1), & x_2(K), & y_2(K) \\ \quad ... & & & \\ x_I(K-1), & y_I(K-1), & x_I(K), & y_I(K) \end{bmatrix} \tag{A.6}$$

The loadings of a single PC have the form:

$$\mathbf{p} = [p_{x-1},\ p_{y-1},\ p_x,\ p_y] \tag{A.7}$$

and the information retained by the PCA model is:

$$\begin{bmatrix} t_1(2) \\ t_2(2) \\ ... \\ t_I(2) \\ \\ ... \\ ... \\ \\ t_1(K) \\ t_2(K) \\ ... \\ t_I(K) \end{bmatrix} = \begin{bmatrix} p_{x-1}x_1(1) + p_{y-1}y_1(1) + p_x x_1(2) + p_y y_1(2) \\ p_{x-1}x_2(1) + p_{y-1}y_2(1) + p_x x_2(2) + p_y y_2(2) \\ ... \\ p_{x-1}x_I(1) + p_{y-1}y_I(1) + p_x x_I(2) + p_y y_I(2) \\ \\ ... \\ ... \\ \\ p_{x-1}x_1(K-1) + p_{y-1}y_1(K-1) + p_x x_1(K) + p_y y_1(K) \\ p_{x-1}x_2(K-1) + p_{y-1}y_2(K-1) + p_x x_2(K) + p_y y_2(K) \\ ... \\ p_{x-1}x_I(K-1) + p_{y-1}y_I(K-1) + p_x x_I(K) + p_y y_I(K) \end{bmatrix} \tag{A.8}$$

Notice this approach is able to capture signals of the form:

$$a_0 \cdot x(k) + a_1 \cdot x(k-1) = b_0 \cdot y(k) + b_1 \cdot y(k-1) + e(k) \tag{A.9}$$

Therefore, the order of the dynamics which can be modelled is augmented by adding LMVs as variables. Also, notice in (A.8) coefficients p_x and p_y are identified from data corresponding to sampling times from 2 to K, instead of from 1 to K as in (A.4). Thus, the interval where the correlation structure is imposed to be constant is reduced by adding LMVs.

Finally, the batch-wise unfolded matrix of data -equivalent to adding $K-1$ LMVs, is:

$$\mathbf{X} = \begin{bmatrix} x_1(1), & y_1(1), & ... & x_1(K), & y_1(K) \\ x_2(1), & y_2(1), & ... & x_2(K), & y_2(K) \\ & & ... & & \\ x_I(1), & y_I(1), & ... & x_I(K), & y_I(K) \end{bmatrix} \tag{A.10}$$

The loadings of the first PC have the form:

$$\mathbf{p} = [p_{x(1)}, \ p_{y(1)}, \ ... \ p_{x(K)}, \ p_{y(K)}] \tag{A.11}$$

and the information retained by the model is.

$$\begin{bmatrix} t_1 \\ t_2 \\ ... \\ t_I \end{bmatrix} = \begin{bmatrix} p_{x(1)}x_1(1) + p_{y(1)}y_1(1) + ... + p_{x(K)}x_1(K) + p_{y(K)}y_1(K) \\ p_{x(1)}x_2(1) + p_{y(1)}y_2(1) + ... + p_{x(K)}x_2(K) + p_{y(K)}y_2(K) \\ ... \\ p_{x(1)}x_I(1) + p_{y(1)}y_I(1) + ... + p_{x(K)}x_I(K) + p_{y(K)}y_I(K) \end{bmatrix} \tag{A.12}$$

This structure, using a sufficient number of PCs, is able to model signals of the form of (A.1) for $p < K$ and constant initial conditions. Moreover, time-varying coefficients $a_n(k)$ and $b_n(k)$ can be effectively modelled, although they do not appear explicitly in the model of (A.11).

B Alternative multi-phase partition from batch dynamic unfolded data

The definition of a multi-phase partition from batch dynamic unfolded data used throughout this document is represented by:

$$\mathbf{X} = \{\underline{\mathbf{X}}^{(n_l)}_{max(k_{il}-n_l,1):k_{el}} : l = 1, ..., L\} \tag{B.1}$$

with:

$$\begin{gathered} n_l = \{k-1 \quad : \quad k \,\epsilon\, \{1, 2, .., k_{el}\}\} \\ s.t.\ k_{il} \leq k_{el},\ k_{i(l+1)} = k_{el} + 1,\ k_{i1} = 1,\ k_{eL} = K \end{gathered} \tag{B.2}$$

This definition is suited for the use on-line based on current and lagged information.

An alternative multi-phase partition from batch dynamic unfolded data is:

$$\begin{gathered} \mathbf{X} = \{\underline{\mathbf{X}}^{(n_l)}_{k_{il}:k_{el}} : l = 1, ..., L\} \\ n_l = \{k-1 \quad : \quad k \,\epsilon\, \{1, 2, .., k_{el} - k_{il} + 1\}\} \\ s.t.\ k_{il} \leq k_{el},\ k_{i(l+1)} = k_{el} + 1,\ k_{i1} = 1,\ k_{eL} = K \end{gathered} \tag{B.3}$$

The difference with (B.1) is that here the data matrices corresponding to the sub-models are disjoint. This definition has two nice properties, comparing to that in (B.1):

– It is a generalization of both the multi-phase partitions from variable-wise and batch-wise unfolded data.

– The model fitted from $\mathbf{X}$ in (B.2) uses both past and future information for prediction. Nonetheless, this means it needs of imputation in all the sampling times but the last one of each phase.

(B.1) has been used throughout the Thesis, but (B.2) deserves future study.

C Distortion of faults in batch-wise models: Theoretical explanation.

In this appendix, a theoretical explanation of the reason why batch-wise unfolded models may distort in an on-line monitoring system the interval where a batch is behaving abnormally is offered.

Let us imagine that, in a batch process, NOC is represented by I batches, J variables and K sampling times. For the sake of simplicity, the imputation of future values will be done with zero deviations [13], but notice that a similar analysis could be performed with current deviations or any of the missing values techniques available [102, 103]. Notice, as well, that the cause of the distortion is the direction of the unfolding and not the imputation method. Thus, evolving models produce a similar phenomenon.

Let $\mathbf{z}^{99}$ represent the mean-centered and auto-scaled data of a batch for which the D-statistic and SPE values exactly match the 99% confidence levels computed for those statistics at every sampling time. After batch-wise unfolding and assuming zero deviations at sampling time k we get:

$$\mathbf{z}_k^{99} = [x_{11}^{99}, ..., x_{J1}^{99}, ..., x_{1k}^{99}, ..., x_{Jk}^{99}, 0, ..., 0]^t \tag{C.1}$$

where x_{jn}^{99} is the value of the variable j at sampling time n of batch $\mathbf{z}^{99}$. The scores and the error values are then obtained:

$$\boldsymbol{\tau}_k^{99} = \mathbf{P}^t \cdot \mathbf{z}_k^{99} \tag{C.2}$$

$$e_{jk}^{99} = x_{jk}^{99} - \mathbf{p}_{jk} \cdot \boldsymbol{\tau}_k^{99} \tag{C.3}$$

where $\mathbf{P}$ is the loadings matrix of the batch-wise unfolded PCA model obtained from NOC and $\mathbf{p}_{jk}$ is the loadings row vector for variable j and sampling time k.

The values for the D-statistic and SPE can be computed as follows:

$$D_k^{99} = (\boldsymbol{\tau}_k^{99})^t \cdot \mathbf{S}_k^{-1} \cdot \boldsymbol{\tau}_k^{99} \tag{C.4}$$

$$SPE_k^{99} = \sum_{j=1}^{J} (e_{jk}^{99})^2 \tag{C.5}$$

where $\mathbf{S}_k$ is the covariance matrix estimated from the scores of NOC assuming zero deviations at sampling time k. D_k^{99} and SPE_k^{99} match the corresponding 99% confidence limits in each of the statistics.

Now, let a new batch noted by $\mathbf{z}^{new}$ be under monitoring. This batch has behaved abnormally from the beginning to a certain sampling time k', but during the rest of the batch is going to behave under normal conditions. An accurate monitoring system should detect this return to NOC fast.

Imagine the variables of $\mathbf{z}^{new}$ present, for instance, c_1 times the value of those corresponding in batch $\mathbf{z}^{99}$ until sampling time k' and for the rest of the batch duration are going to present $\frac{1}{c_2}$ times the value of corresponding variables in batch $\mathbf{z}^{99}$, with $c_1 > 1$ and $c_2 > 1$. For the sake of simplicity and easy understanding and without loss of generality, let us assume the values of c_1 and c_2 are equal for all variables.

After batch-wise unfolding and auto-scaling and assuming zero deviations at sampling time $k' + 1$ we get:

$$\begin{gathered}\mathbf{z}_{k'+1}^{new} = [c_1 \cdot x_{11}^{99}, ..., c_1 \cdot x_{J1}^{99}, ..., c_1 \cdot x_{1k'}^{99}, ..., \\ c_1 \cdot x_{Jk'}^{99}, \tfrac{1}{c_2} \cdot x_{1(k'+1)}^{99}, ..., \tfrac{1}{c_2} \cdot x_{J(k'+1)}^{99}, 0, ..., 0]^t\end{gathered} \tag{C.6}$$

$$\begin{gathered}\boldsymbol{\tau}_{(k'+1)}^{new} = c_1 \cdot \mathbf{P}_{(1:k')}^t \cdot \mathbf{x}_{(1:k')}^{99} + \\ + \tfrac{1}{c_2} \cdot \mathbf{P}_{(k'+1)}^t \cdot \mathbf{x}_{(k'+1)}^{99}\end{gathered} \tag{C.7}$$

$$D_{k'+1}^{new} = (\boldsymbol{\tau}_{k'+1}^{new})^t \cdot \mathbf{S}_{k'+1}^{-1} \cdot \boldsymbol{\tau}_{k'+1}^{new} \tag{C.8}$$

$$SPE_{k'+1}^{new} = \sum_{j=1}^{J} (\frac{1}{c_2} \cdot x_{j(k'+1)}^{99} - \mathbf{p}_{j(k'+1)} \cdot \boldsymbol{\tau}_{k'+1}^{new})^2 \tag{C.9}$$

where $(b : c)$ means an interval from sampling time b to c.

If the preceding abnormal interval has been long ($k' >> 1$) and/or the deviation from NOC has been sufficiently high ($c_1 >> 1$) , its influence on $\boldsymbol{\tau}_{k'+1}^{new}$ is also high and for every PC a-th we have:

$$|\tau_{a(k'+1)}^{new}| = |c_1 \cdot \tau_{a(k')}^{99} + \frac{1}{c_2} \cdot (\tau_{a(k'+1)}^{99} - \tau_{a(k')}^{99})| > |\tau_{a(k'+1)}^{99}| \tag{C.10}$$

where τ_{ak} is the score of the a-th PC obtained at sampling time k. Then:

$$D_{k'+1}^{new} > D_{k'+1}^{99} \tag{C.11}$$

This will continue happening for following sampling times until the abnormal interval in the batch is compensated by the following normal interval. The closer the batch is to the mean trajectory during the normal phase, the faster the D-statistic recovers from the past information influence. Of course, this is not the behavior that is expected from the monitoring system and so the charts can be badly interpreted.

The high value of the scores has an similar influence in the SPE statistic that can be as well misunderstood. The error of the model at sampling time k' follows next expression:

$$e^{new}_{j(k'+1)} = x^{new}_{j(k'+1)} - \mathbf{p}_{j(k'+1)} \cdot \boldsymbol{\tau}^{new}_{(k'+1)} \tag{C.12}$$

From (C.7) and (C.12) it can be concluded that the modelling error used in the SPE statistic depends on both the past information and the current information. Again, if the preceding abnormal interval has been long ($k' >> 1$) and/or the deviation from NOC has been sufficiently high ($c_1 >> 1$), the prediction of the value of $x^{new}_{j(k'+1)}$ may be almost determined solely by the past information. If the prediction is not good due to the preceding abnormal interval, the value of the SPE can be outside the confidence limits even when the normal interval has started and until the influence of the abnormal interval is compensated.

With a similar reasoning it could be concluded that short deviations from NOC, following a normal interval, could be not detected if the intensity of the deviation is not sufficiently high to overcome the influence of a previous normal interval.

D TSR for PLS models

The TSR method [103] estimates the value of the scores from the trimmed scores, i.e. the scores obtained imputing with zero deviations [13]. Zero deviations is based on the assumption that the future values of a batch under processing will follow exactly the average trajectory of the variables. This means that after the substraction of the average trajectory and batch-wise unfolding, assuming zero deviations, the batch data are completed with zeros. Thus, at sampling time k, the in-filled data of a new batch $\mathbf{z}$ are:

$$\mathbf{z}_k = [z_{11}, ..., z_{J1}, ..., z_{1k}, ..., z_{Jk}, 0, ..., 0]^T \tag{D.1}$$

where z_{jn} is the value of the variable j at sampling time n of batch $\mathbf{z}$.

The trimmed scores at sampling time k are calculated from this estimation of the data of the batch. For instance, in PCA:

$$\boldsymbol{\tau}^*_{1:A,k} = \mathbf{P}^T_{1:A} \cdot \mathbf{z}_k \tag{D.2}$$

where $\mathbf{P}_{1:A}$ is the loadings matrix of the PCA model retaining the first A principal components. The latter equation can be re-expressed in an equivalent form:

$$\boldsymbol{\tau}^*_{1:A,k} = \mathbf{P}^{*T}_{1:A,k} \cdot \mathbf{z}^*_k \tag{D.3}$$

with:

$$\mathbf{P}^*_{1:A,k} = \begin{bmatrix} p_{1,11}, ..., p_{1,J1}, ..., p_{1,1k}, ..., p_{1,Jk} \\ ... \\ p_{A,11}, ..., p_{A,J1}, ..., p_{A,1k}, ..., p_{A,Jk} \end{bmatrix}^T \tag{D.4}$$

$$\mathbf{z}^*_k = [z_{11}, ..., z_{J1}, ..., z_{1k}, ..., z_{Jk}]^T \tag{D.5}$$

where $p_{a,jn}$ is the loading corresponding to the variable j at sampling time n in the principal component a.

In PLS, the scores of a new batch $\mathbf{z}$ are calculated as follows:

$$\boldsymbol{\tau}_{1:A} = \mathbf{R}^T_{1:A} \cdot \mathbf{z} \tag{D.6}$$

$$\mathbf{R}_{1:A} = \mathbf{W}_{1:A} \cdot (\mathbf{P}_{1:A}^T \cdot \mathbf{W}_{1:A})^{-1} \tag{D.7}$$

where $\mathbf{W}_{1:A}$ is the weights matrix and $\mathbf{P}_{1:A}$ the loadings matrix of the PLS model with A latent variables. The trimmed scores are calculated with a similar philosophy as in PCA:

$$\boldsymbol{\tau}_{1:A,k}^* = \mathbf{R}_{1:A,k}^{*T} \cdot \mathbf{z}_k^* \tag{D.8}$$

$$\mathbf{R}_{1:A,k}^* = \mathbf{W}_{1:A,k}^* \cdot (\mathbf{P}_{1:A}^T \cdot \mathbf{W}_{1:A})^{-1} \tag{D.9}$$

$$\mathbf{W}_{1:A,k}^* = \begin{bmatrix} w_{1,11}, ..., w_{1,J1}, ..., w_{1,1k}, ..., w_{1,Jk} \\ ... \\ w_{A,11}, ..., w_{A,J1}, ..., w_{A,1k}, ..., w_{A,Jk} \end{bmatrix}^T \tag{D.10}$$

where $w_{a,jn}$ is the weight corresponding to the variable j at sampling time n in the latent variable a. The complete matrices $\mathbf{P}_{1:A}$ and $\mathbf{W}_{1:A}$ are used in the inversion in (D.9). Thus, the information of the complete model is used, improving the prediction capability. Also problems of invertibility are avoided.

To estimate the scores of a batch from its trimmed scores computed at a certain sampling time k, TSR uses the following approximation [103]:

$$\mathbf{T}_{1:A} = \mathbf{T}_{1:A,k}^* \cdot \mathbf{B}_k + \mathbf{E} \tag{D.11}$$

where $\mathbf{T}_{1:A}$ is the score matrix of the data used to create the model and $\mathbf{T}_{1:A,k}^*$ the trimmed score matrix computed from the same data -at that certain sampling time k. These matrices are computed from:

$$\mathbf{T}_{1:A} = \mathbf{Z} \cdot \mathbf{R}_{1:A} \tag{D.12}$$

$$\mathbf{T}_{1:A,k}^* = \mathbf{Z}_k^* \cdot \mathbf{R}_{1:A,k}^* \tag{D.13}$$

where $\mathbf{Z}$ is a data set of calibration batches batch-wise unfolded and $\mathbf{Z}_k^*$ contains the data collected for the batches up to sampling time k. From least squares, $\mathbf{B}_k$ can be estimated:

$$\hat{\mathbf{B}}_k = (\mathbf{T}_{1:A,k}^{*T} \cdot \mathbf{T}_{1:A,k}^*)^{-1} \cdot \mathbf{T}_{1:A,k}^{*T} \cdot \mathbf{T}_{1:A} \tag{D.14}$$

Alternatively, a similar derivation procedure to that followed in [103] can be used to obtain an equivalent expression to (D.14). Applying equalities in (D.12) and (D.13) to (D.14) yields:

$$\hat{\mathbf{B}}_k = (\mathbf{R}_{1:A,k}^{*T} \cdot \mathbf{Z}_k^{*T} \cdot \mathbf{Z}_k^* \cdot \mathbf{R}_{1:A,k}^*)^{-1} \cdot (\mathbf{R}_{1:A,k}^{*T} \cdot \mathbf{Z}_k^{*T} \cdot \mathbf{Z} \cdot \mathbf{R}_{1:A}) \tag{D.15}$$

The following equalities are also true:

$$\mathbf{Z}_k^{*T} \cdot \mathbf{Z}_k^* = \mathbf{P}_k^* \cdot \mathbf{\Theta} \cdot \mathbf{P}_k^{*T} \tag{D.16}$$

$$\mathbf{Z}_k^{*T} \cdot \mathbf{Z} = \mathbf{P}_k^* \cdot \mathbf{\Theta} \cdot \mathbf{P}^T \tag{D.17}$$

where $\mathbf{\Theta}$ is a diagonal matrix containing the eigenvalues of the covariance matrix of $\mathbf{Z}$, ordered by their value. Combining (D.15), (D.16) and (D.17), it yields:

$$\hat{\mathbf{B}}_k = (\mathbf{R}_{1:A,k}^{*T} \cdot \mathbf{P}_k^* \cdot \mathbf{\Theta} \cdot \mathbf{P}_k^{*T} \cdot \mathbf{R}_{1:A,k}^*)^{-1} \cdot (\mathbf{R}_{1:A,k}^{*T} \cdot \mathbf{P}_k^* \cdot \mathbf{\Theta} \cdot \mathbf{P}^T \cdot \mathbf{R}_{1:A}) \tag{D.18}$$

From the deflation in PLS, we know the data modelled by two different components are independent, thus:

$$\mathbf{P}_k^* \cdot \mathbf{\Theta} \cdot \mathbf{P}^T = \mathbf{P}_{1:A,k}^* \cdot \mathbf{\Theta}_{1:A} \cdot \mathbf{P}_{1:A}^T + \mathbf{P}_{A+1:K,k}^{*T} \cdot \mathbf{\Theta}_{A+1:K} \cdot \mathbf{P}_{A+1:K}^T \tag{D.19}$$

Then, from (D.18) and (D.19):

$$\begin{gathered}\hat{\mathbf{B}}_k = (\mathbf{R}_{1:A,k}^{*T} \cdot \mathbf{P}_k^* \cdot \mathbf{\Theta} \cdot \mathbf{P}_k^{*T} \cdot \mathbf{R}_{1:A,k}^*)^{-1} \cdot \\ \cdot (\mathbf{R}_{1:A,k}^{*T} \cdot (\mathbf{P}_{1:A,k}^* \cdot \mathbf{\Theta}_{1:A} \cdot \mathbf{P}_{1:A}^T + \mathbf{P}_{A+1:K,k}^* \cdot \mathbf{\Theta}_{A+1:K} \cdot \mathbf{P}_{A+1:K}^T) \cdot \mathbf{R}_{1:A})\end{gathered} \tag{D.20}$$

From (D.7) the following holds:

$$\mathbf{P}_{1:A}^T \cdot \mathbf{R}_{1:A} = \mathbf{P}_{1:A}^T \cdot \mathbf{W}_{1:A} \cdot (\mathbf{P}_{1:A}^T \cdot \mathbf{W}_{1:A})^{-1} = \mathbf{I}_A \tag{D.21}$$

Then from (D.20):

$$\begin{gathered}\hat{\mathbf{B}}_k = (\mathbf{R}_{1:A,k}^{*T} \cdot \mathbf{P}_k^* \cdot \mathbf{\Theta} \cdot \mathbf{P}_k^{*T} \cdot \mathbf{R}_{1:A,k}^*)^{-1} \cdot \\ \cdot (\mathbf{R}_{1:A,k}^{*T} \cdot \mathbf{P}_{1:A,k}^* \cdot \mathbf{\Theta}_{1:A} + \mathbf{R}_{1:A,k}^{*T} \cdot \mathbf{P}_{A+1:K,k}^* \cdot \mathbf{\Theta}_{A+1:K} \cdot \mathbf{P}_{A+1:K}^T \cdot \mathbf{R}_{1:A})\end{gathered} \tag{D.22}$$

One of the features of PLS models is that $\mathbf{P}^T \cdot \mathbf{W}$ is upper triangular. Therefore, the following holds:

$$\mathbf{P}_{A+1:K}^T \cdot \mathbf{W}_{1:A} = 0 \tag{D.23}$$

and so from (D.7):

$$\mathbf{P}_{A+1:K}^T \cdot \mathbf{R}_{1:A} = 0 \tag{D.24}$$

and combining (D.22) and (D.24), it yields:

$$\hat{\mathbf{B}}_k = (\mathbf{R}_{1:A,k}^{*T} \cdot \mathbf{S}_k^{**} \cdot \mathbf{R}_{1:A,k}^*)^{-1} \cdot \mathbf{R}_{1:A,k}^{*T} \cdot \mathbf{P}_{1:A,k}^* \cdot \mathbf{\Theta}_{1:A} \tag{D.25}$$

where $\mathbf{S}_k^{**} = \mathbf{P}_k^* \cdot \mathbf{\Theta} \cdot \mathbf{P}_k^{*T}$ is the covariance matrix of the variables from sampling time 1 to k and $\mathbf{\Theta}_{1:A}$ the covariance matrix of the scores. Finally,

to estimate with TSR the value of the scores of a new batch at time k, the following expression is used:

$$\hat{\boldsymbol{\tau}}_{1:A,k} = \hat{\mathbf{B}}_k^T \cdot \mathbf{R}_{1:A,k}^{*T} \cdot \mathbf{z}_k^* \tag{D.26}$$

E Adaptive PLS algorithms

Adaptive Hierarchical PLS

This algorithm combines the Hierarchical PLS algorithm according to [62] with the adaptive method of [48]. Experimentally, it was determined that the normalization of the loadings in the internal PCA loop and the weights in the super loop lead to a much better performance than the normalization of the scores.

The initialization of the scores for sampling time $k = 1$ is performed with PCA. No general differences were observed by doing this initialization with a PLS loop. Since the hierarchical algorithm is not a PCA nor a PLS, but a combination of both, the prediction at sampling time $k = 1$ is recalculated after the initialization.

The parameter λ multiplies the lagged score, instead of the current score, so that it is coherent with its definition in the EWEW approach used in Chapter 8.

For each LV ($a = 1...A$)
- choose start $\mathbf{t}_{a0}$
- loop until convergence of $\mathbf{t}_{a0}$
  - $\mathbf{p}_{a0} = \mathbf{X}_1^T \cdot \mathbf{t}_{a0} / \mathbf{t}_{a0}^T \cdot \mathbf{t}_{a0}$
  - normalize to $\|\mathbf{p}_{a0}\| = 1$
  - $\mathbf{t}_{a0} = \mathbf{X}_1 \cdot \mathbf{p}_{a0} / \mathbf{p}_{a0}^T \cdot \mathbf{p}_{a0}$
- end
- For each sampling time ($k = 1...K$)
  - $\mathbf{t}_{ak} = \mathbf{t}_{a(k-1)}$
  - loop until convergence of $\mathbf{t}_{ak}$
    - $\mathbf{p}_{ak} = \mathbf{X}_k^T \cdot \mathbf{t}_{ak} / \mathbf{t}_{ak}^T \cdot \mathbf{t}_{ak}$
    - normalize to $\|\mathbf{p}_{ak}\| = 1$
    - $\mathbf{r}_{ak} = \mathbf{X}_k \cdot \mathbf{p}_{ak}$
    - $\mathbf{R}_{ak} = [\lambda \cdot \mathbf{t}_{a(k-1)}, \mathbf{r}_{ak}]$
    - $\mathbf{q}_{ak} = \mathbf{Y}_k^T \cdot \mathbf{t}_{ak} / \mathbf{t}_{ak}^T \cdot \mathbf{t}_{ak}$
    - $\mathbf{u}_{ak} = \mathbf{Y}_k \cdot \mathbf{q}_{ak} / \mathbf{q}_{ak}^T \cdot \mathbf{q}_{ak}$
    - $\mathbf{w}_{ak} = \mathbf{R}_{ak}^T \cdot \mathbf{u}_{ak} / \mathbf{u}_{ak}^T \cdot \mathbf{u}_{ak}$
    - normalize to $\|\mathbf{w}_{ak}\| = 1$
    - $\mathbf{t}_{ak} = \mathbf{R}_{ak} \cdot \mathbf{w}_{ak} / \mathbf{w}_{ak}^T \cdot \mathbf{w}_{ak}$

end
$\mathbf{X}_k = \mathbf{X}_k - \mathbf{t}_{ak} \cdot \mathbf{p}_{ak}^T$
$\mathbf{Y}_k = \mathbf{Y}_k - \mathbf{t}_{ak} \cdot \mathbf{q}_{ak}^T$
end

Adaptive Multi-block PLS

This algorithm combines the Multi-block PLS algorithm according to [62] with the adaptive method of [48]. Super score deflation is used as recommended in [120]. Initialization of the scores for sampling time $k = 1$ is performed by a PLS, used as well to predict the quality value at that sampling time. The parameter λ multiplies the lagged score for the same reason as in the previous algorithm.

For each LV ($a = 1...A$)
choose start $\mathbf{u}_{a1}$
loop until convergence of $\mathbf{t}_{a1}$
$\mathbf{w}_{a1} = \mathbf{X}_1^T \cdot \mathbf{u}_{a1} / \mathbf{u}_{a1}^T \cdot \mathbf{u}_{a1}$
normalize to $\|\mathbf{w}_{a1}\| = 1$
$\mathbf{t}_{a1} = \mathbf{X}_1 \cdot \mathbf{w}_{a1} / \mathbf{w}_{a1}^T \cdot \mathbf{w}_{a1}$
$\mathbf{q}_{a1} = \mathbf{Y}_1^T \cdot \mathbf{t}_{a1} / \mathbf{t}_{a1}^T \cdot \mathbf{t}_{a1}$
$\mathbf{u}_{a1} = \mathbf{y}_1 \cdot \mathbf{q}_{a1}$
end
$\mathbf{p}_{a1} = \mathbf{X}_1^T \cdot \mathbf{t}_{a1} / \mathbf{t}_{a1}^T \cdot \mathbf{t}_{a1}$
$\mathbf{X}_1 = \mathbf{X}_1 - \mathbf{t}_{a1} \cdot \mathbf{p}_{a1}^T$
$\mathbf{Y}_1 = \mathbf{Y}_1 - \mathbf{t}_{a1} \cdot \mathbf{q}_{a1}^T$
For each sampling time ($k = 2...K$)
$\mathbf{u}_{ak}$ first column of $\mathbf{Y}_k$
$\mathbf{t}_{ak} = \mathbf{t}_{a(k-1)}$
loop until convergence of $\mathbf{t}_{ak}$
$\mathbf{v}_{ak} = \mathbf{X}_k^T \cdot \mathbf{u}_{ak} / \mathbf{u}_{ak}^T \cdot \mathbf{u}_{ak}$
normalize to $\|\mathbf{v}_{ak}\| = 1$
$\mathbf{r}_{ak} = \mathbf{X}_k \cdot \mathbf{v}_{ak}$
$\mathbf{R}_{ak} = [\lambda \cdot \mathbf{t}_{a(k-1)}, \mathbf{r}_{ak}]$
$\mathbf{w}_{ak} = \mathbf{R}_{ak}^T \cdot \mathbf{u}_{ak} / \mathbf{u}_{ak}^T \cdot \mathbf{u}_{ak}$
normalize to $\|\mathbf{w}_{ak}\| = 1$
$\mathbf{t}_{ak} = \mathbf{R}_{ak} \cdot \mathbf{w}_{ak} / \mathbf{w}_{ak}^T \cdot \mathbf{w}_{ak}$
$\mathbf{q}_{ak} = \mathbf{Y}_k^T \cdot \mathbf{t}_{ak} / \mathbf{t}_{ak}^T \cdot \mathbf{t}_{ak}$
$\mathbf{u}_{ak} = \mathbf{Y}_k \cdot \mathbf{q}_{ak} / \mathbf{q}_{ak}^T \cdot \mathbf{q}_{ak}$
end
$\mathbf{p}_{ak} = \mathbf{X}_k^T \cdot \mathbf{t}_{ak} / \mathbf{t}_{ak}^T \cdot \mathbf{t}_{ak}$
$\mathbf{X}_k = \mathbf{X}_k - \mathbf{t}_{ak} \cdot \mathbf{p}_{ak}^T$
$\mathbf{Y}_k = \mathbf{Y}_k - \mathbf{t}_{ak} \cdot \mathbf{q}_{ak}^T$
end

References

1. J. Camacho and J. Picó. Monitorización de procesos por lotes mediante pca multifase. *Revista Iberoamericana de Automatica e Informática Industrial*, 3(3):78–91, 2006.
2. J. Camacho and J. Picó. Multi-phase principal component analysis for batch processes modelling. *Chemometrics and Intelligent Laboratory Systems*, 81(2):127–136, 2006.
3. J. Camacho and J. Picó. Online monitoring of batch processes using multi-phase principal component analysis. *Journal of Process Control*, 10(16):1021–1035, 2006.
4. J. Camacho, J. Picó, and A. Ferrer. Self-tuning run to run optimization of fed-batch processes using unfold-pls. *AIChE Journal*, 53(7):1789–1804, 2007.
5. J. Camacho, J. Picó, and A. Ferrer. Bilinear modelling of batch processes. part i: Theoretical discussion. *Submitted to Journal of Chemometrics*, 2007.
6. J. Camacho, J. Picó, and A. Ferrer. Bilinear modelling of batch processes. part ii: Pls comparative. *Submitted to Journal of Chemometrics*, 2007.
7. J. Camacho, J. Picó, and A. Ferrer. Multi-phase analysis framework for handling batch process data. *Submitted to Journal of Chemometrics*, 2007.
8. J. Camacho, J. Picó, and A. Ferrer. Leave-n-samples-out cross-validation in pca for missing data imputation and measurement noise reduction. *In elaboration for Chemometrics and Intelligent Laboratory Systems*, 2007.
9. J. Camacho, J. Picó, and A. Ferrer. A new look at the dynamic covariance structure of various approaches for batch process modelling. In *10th Scandinavian Symposium on Chemometrics*, 2007.
10. J. Camacho, J. Picó, and A. Ferrer. New cross-validation methods in principal component analysis. In *10th Scandinavian Symposium on Chemometrics*, 2007.
11. J. Camacho, J. Picó, and A. Ferrer. A new algorithm for selecting the unfolding method and the number of sub-models in batch process modelling with pca. In *10th Scandinavian Symposium on Chemometrics*, 2007.
12. J. Camacho, J. Picó, and A. Ferrer. Multi-phase analysis framework for handling batch process data. In *10th Scandinavian Symposium on Chemometrics*, 2007.
13. P. Nomikos and J.F. MacGregor. Multivariate spc charts for monitoring batch processes. *Technometrics*, 37(1):41–59, 1995.
14. T.F. Edgar. Control and operations: when does controllability equal profitability? *Computers and Chemical Engineering*, 29:41–49, 2004.
15. I.C. Trelea, G. Trystarm, and F. Courtois. Optimal constrained non-linear control of batch processes: Application to corn drying. *Journal of Food Engineering*, 31:403–422, 1997.

16. C. Ündey and A. Çinar. Statistical monitoring of multistage, multiphase batch processes. *IEEE Control System Magazine*, 22(5):40–52, 2002.
17. K.J. Åström. *Introduction to Stochastic Control Theory*. Dover, New York, 2006. Reprint. Originally published by Academic Press 1970.
18. D. Bonvin. Optimal operation of batch reactors-a personal view. *Journal of Process Control*, 8:355–368, 1998.
19. F. Lei, M. Rotbøll, and S.B Jørgensen. A biochemically structured model for saccharomyces cerevisiae. *Journal of Biotechnology*, 88:205–221, 2001.
20. T. Kourti. Process analysis and abnormal situation detection: from theory to practice. *IEEE Control Systems Magazine*, 22(5), 2002.
21. A. Singhal and D.E. Seborg. Pattern matching in historical batch data using pca. *IEEE Control System Magazine*, 22(5):53–63, 2002.
22. K.A. Kosanovich, K.S. Dahl, and M.J. Piovoso. Improved process understanding using multiway principal component analysis. *Engineering Chemical Research*, 35:138–146, 1996.
23. Z. Xiong and J. Zhang. A batch-to-batch iterative optimal control strategy based on recurrent neural network models. *Journal of Process Control*, 15:11–21, 2005.
24. A. Bakhtazad, A. Palazoglu, and J.A. Romagnoli. Detection and classification of abnormal process situations using multidimensional wavelet domain hidden markov trees. *Computers and Chemical Engineering*, 24:769–775, 2000.
25. D. Dong, T.J. McAvoy, and E. Zafiriou. Batch-to-batch optimization using neural networks models. *Industrial Engineering and Chemical Research*, 35:2269–2276, 1996.
26. J.C. Wong, K.A. McDonald, and A. Palazoglu. Classification of abnormal plant operations using multiple process variable trends. *Journal of Process Control*, 11:409–418, 2001.
27. C.W. Ng and M.A. Hussain. Hybrid neural network–prior knowledge model in temperature control of a semi-batch polymerization process. *Chemical Engineering and Processing*, 43:559–570, 2004.
28. E.N.M. van Sprang, H. Ramaker, J.A. Westerhuis, A.K. Smilde, and D. Wienke. Statistical batch process monitoring using gray models. *AIChE Journal*, 51:931–945, 2005.
29. F.J. Doyle III, Ch.A. Harrison, and T.J. Crowley. Building inferential prediction models of batch processes using subspace identification. *Journal of Process Control*, 13:397–406, 2003.
30. A. Kassidas, J.F. MacGregor, and P.A. Taylor. Synchronization of batch trajectories using dynamic time warping. *AIChE Journal*, 44:864–875, 1998.
31. S. Wold, N. Kettanch, H. Friden, and A. Holmberg. Modelling and diagnostics of batch processes and analogous kinetic experiments. *Chemometrics and Intelligent Laboratoy Systems*, 44:331–340, 1998.
32. B.M. Wise, N.B. Gallagher, and E.B. Martin. Application of parafac2 to fault detection and diagnosis in semiconductor etch. *Journal of Chemometrics*, 15:285–296, 2001.
33. N. Lu, F. Gao, Y. Yang, and F. Wang. Pca-based modeling and on-line monitoring strategy for uneven-length batch processes. *Industrial and Engineering Chemical Research*, 43:3343–3352, 2004.
34. M. Fransson and S. Folestad. Real-time alignment of batch process data using cow for on-line process monitoring. *Chemometrics and Intelligent Laboratory Systems*, 84:56–61, 2006.

35. A.K. Smilde, R. Bro, and P. Geladi. *Multi-way Analysis, Application in the Chemical Sciences*. John Wiley & Sons, England, 2003.
36. R. Bro. Parafac. tutorial and applications. *Chemometrics and Intelligent Laboratory Systems.*, 38:149–171, 1997.
37. L.R. Tucker. The extension of factor analysis to three-dimensional matrices. in: Frederiksen n, gulliksen h. contributions to mathematical psychology. new york: Holt, rinehart and winston. pages 110–162, 1964.
38. R. Bro. Multiway calibration. multi-linear pls. *Journal of Chemometrics*, 10:47–61, 1996.
39. H. Wold and E. Lyttkens. Nonlinear iterative partial least squares (nipals) estimation procedures. In *Bull. Intern. Statist. Inst. Proc., 37th session, London*, pages 1–15, 1969.
40. H. Martens and T. Næs. *Multivariate Calibration*. John Wiley & Sons, 1992.
41. J.E. Jackson. *A User's Guide to Principal Components*. Wiley-Interscience, England, 2003.
42. P. Geladi and B.R. Kowalski. Partial least-squares regression: a tutorial. *Analytica Chimica Acta*, 185:1–17, 1986.
43. A.E. Hoerl and R.W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(3):55–67, 1970.
44. S. Wold, P. Geladi, K. Esbensen, and J. Ohman. Multi-way principal components and pls analysis. *Journal of Chemometrics*, 1:41–56, 1987.
45. P. Nomikos and J.F. MacGregor. Monitoring batch processes using multiway principal components analysis. *AIChE Journal*, 40(8):1361–1375, 1994.
46. T. Kourti. Multivariate dynamic data modeling for analysis and statistical process control of batch processes, start-ups and grade transitions. *Journal of Chemometrics*, 17:93–109, 2003.
47. J. Chen and K. Liu. On-line batch process monitoring using dynamic pca and dynamic pls models. *Chemical Engineering Science*, 57:63–75, 2002.
48. S. Rännar, J.F. MacGregor, and S. Wold. Adaptive batch monitoring using hierarchical pca. *Chemometrics and Intelligent Laboratory Systems*, 41:73–81, 1998.
49. H. Ramaker, E.N.M. van Sprang, J.A. Westerhuis, and A.K. Smilde. Fault detection properties of global, local and time evolving models for batch process monitoring. *Journal of Process Control*, 15(7):799–805, 2005.
50. P. Nomikos. *Statistical process control of batch processes*. PhD Thesis, McMaster University, Hamilton, ON, 1995.
51. B. Lennox, G.A. Montague, H.G. Hiden, G. Kornfeld, and P.R. Goulding. Process monitoring of an industrial fed-batch fermentation. *Biotechnology and Bioengineering*, 74:125, 2001.
52. C. Ündey, S. Ertunç, and A. Çinar. Online batch/fed-batch process performance monitoring, quality prediction, and variable-contribution analysis for diagnosis. *Industrial and Engineering Chemical Research*, 42:4645–4658, 2003.
53. B. Lennox, H.G. Hiden, G.A. Montague, G. Kornfeld, and P.R. Goulding. Application of multivariate statistical process control to batch operations. *Computers and Chemical Engineering*, 24:291–296, 2000.
54. N. Lu, F. Gao, and F. Wang. Sub-pca modeling and on-line monitoring strategy for batch processes. *AIChE Journal*, 50(1):255–259, 2004.
55. R.A. Harshman. Foundations of the parafac procedure: Models and conditions for an 'explanatory' multi-modal factor analysis. *UCLA Working Papers in Phonetics*, 16:1–84, 1970.

56. J.D. Carroll and J. Chang. Analysis of individual differences in multidimensional scaling via an v-way generalization of 'eckart-young' decomposition. *Psychometrika*, 45:3–24, 1980.
57. R.B. Cattell. 'parallel proportional profiles' and other principles for determining the choice of factors by rotation. *Psychometrika*, 9:267–283, 1944.
58. R. Bro and H.A.L. Kiers. A new efficient method for determining the number of components in parafac models. *Journal of Chemometrics.*, 17:274–286, 2003.
59. R. Bro. Three-way analysis course. kvl, copenhagen. 2007.
60. R. Bro and A.K. Smilde. Centering and scaling in component analysis. *Journal of Chemometrics.*, 17:16–33, 2003.
61. D. Dong and T.J. McAvoy. Batch tracking via non-linear principal component analysis. *AIChE Journal*, 42(8):2199–2208, 1996.
62. J.A. Westerhuis, T. Kourti, and J.F. MacGregor. Analysis of multiblock and hierarchical pca and pls models. *Journal of Chemometrics*, 12:301–321, 1998.
63. A.K. Smilde, J.A. Westerhuis, and S. de Jong. A framework for sequential multiblock component methods. *Journal of Chemometrics*, 17:323–337, 2003.
64. S.J. Qin. Recursive pls algorithms for adaptive data modeling. *Computers and Chemical Engineering*, 22:503–514, 1998.
65. W. Li, H. Yue, S. Valle-Cervantes, and S.J. Qin. Recursive pca for adaptive process monitoring. *Journal of Process Control*, 10:471–486, 2000.
66. D.S. Lee and P.A. Vanrolleghem. Monitoring of a sequencing batch reactor using adaptive multiblock principal component analysis. *Biotechnology and Bioengineering*, 82(4):489–497, 2003.
67. Y. Rotem, A. Wachs, and D.R. Lewin. Ethylene compressor monitoring using model-based pca. *AIChE Journal*, 46(9):1825–1835, 2000.
68. H. Hotelling. *Multivariate Quality Control. Techniques of Statistical Analysis*. MacGraw-Hill, New York, 1947.
69. G.E.P. Box. Some theorems on quadratic forms applied in the study of analysis of variance problems: Effect of inequality of variance in one-way classification. *The Annals of Mathematical Statistics*, 25:290–302, 1954.
70. J.E. Jackson and G.S. Mudholkar. Control procedures for residuals associated with principal component analysis. *Technometrics*, 21:331–349, 1979.
71. N.D. Tracy, J.C. Young, and R.L. Mason. Multivariate control charts for individual observations. *Journal of Quality Technology*, 24(2):88–95, 1992.
72. J.F. MacGregor, C. Jaeckle, C. Kiparissides, and M. Koutoudi. Process monitoring and diagnosis by multiblock pls methods. *AIChE Journal*, 40(5):826–838, 1994.
73. T. Kourti and J.F. MacGregor. Multivariate spc methods for process and product monitoring. *Journal of Quality Technology*, 28(4), 1996.
74. A. Ferrer. Multivariate statistical process control based on principal component analysis (mspc-pca): Some reflections and a case study in an autobody assembly process. *Quality Engineering (In PRESS)*, 2007.
75. H. Ramaker, E.N.M. van Sprang, J.A. Westerhuis, S.P. Gurden, A.K. Smilde, and F.H. van der Meulen. Performance assessment and improvement of control charts for statistical batch process monitoring. *Statistica Neerlandica*, 60(3):339–360, 2006.
76. A. Simoglou, P. Georgieva, E.B. Martin, A.J. Morris, and S. Feyo de Azevedo. On-line monitoring of a sugar crystallization process. *Computers and Chemical Engineering*, 29:1411–1422, 2005.

77. D. Aguado, A. Ferrer, J. Ferrer, and A. Seco. Multivariate spc of a sequencing batch reactor for wastewater treatment. *Chemometrics and Intelligent Laboratory Systems*, 85:82–93, 2007.
78. J. Flores-Cerrillo and J.F. MacGregor. Within-batch and batch-to-batch inferential-adaptive control of semibatch reactors: A partial least squares approach. *Industrial and Engineering Chemical Research*, 42:3334–3345, 2003.
79. H. Zhang and B. Lennox. Integrated condition monitoring and control of fed-batch fermentation processes. *Journal of Process Control*, 14:41–50, 2004.
80. K. Shimizu. An overview on the control system design of bioreactors. *Advances in Biochemical Engineering/Biotechnology*, 50:65–84, 1993.
81. A.M. Jobé, C. Herwiq, M. Surzyn, B. Walker, I. Marison, and U. von Stockar. Generally applicable fed-batch culture concept based on the detection of metabolic state by on-line balancing. *Biotechnology and Bioengineering*, 82:627–639, 2003.
82. E. Picó-Marco, J. Picó, and H. De Battista. Sliding mode scheme for adaptive specific growth rate control in biotechnological fed-batch processes. *International Journal of Control*, 78:128–141, 2005.
83. H. De Battista, J. Picó, and E. Picó-Marco. Globally stabilizing control of fed-batch processes with haldane kinetics using growth rate estimation feedback. *Journal of Process Control*, 16(8):865–875, 2006.
84. S. Valentinotti, B. Srinivasan, U. Holmberg, D. Bonvin, C. Cannizzaro, M. Rhiel, and U. von Stockar. Optimal operation of fed-batch fermentations via adaptive control of overflow metabolite. *Control Engineering Practice*, 11:665–674, 2003.
85. S.A. Russell, D.G. Robertson, J.H. Lee, and B.A. Ogunnaike. Control of product quality for batch nylon 6,6 autoclaves. *Chemical Engineering Science*, 53(21):3685–3702, 1998.
86. N.M. Lakshmanan and Y. Arkun. Estimation and model predictive control of non-linear batch processes using linear parameter varying models. *International Journal of Control*, 72:659–675, 1999.
87. H.J. Lee K.S. Lee, I. Chin and J.H. Lee. Model predictive control technique combined with iterative learning for batch processes. *AIChE Journal*, 45(10):2175–2187, 1999.
88. K.S. Lee and J.H. Lee. Iterative learning control-based batch process control technique for integrated control of end product properties and transient profiles od process variables. *Journal of Process Control*, 13:607–621, 2003.
89. A.W. Dorsey and J.H. Lee. Hybrid model-based approach to batch-to-batch control of particle size distribution in emulsion polymerization. *Computers and Chemical Engineering*, 27:1153–1163, 2003.
90. Y. Yabuki, T. Nagasawa, and J.F. MacGregor. An industrial experience with product quality control in semi-batch processes. *Computers and Chemical Engineering*, 24:585–590, 2000.
91. J. Flores-Cerrillo and J.F. MacGregor. Control of particle size distributions in emulsion semibatch polymerization using mid-course correction policies. *Industrial and Engineering Chemical Research*, 41:1805–1814, 2002.
92. F. Gao, Y. Yang, and C. Shao. Robust iterative learning control with applications to injection molding process. *Chemical Engineering Science*, 56:7025–7034, 2001.

93. M. Mezghani, G. Roux, M. Cabassud, B. Dahhou, M.V. Le Lann, and G. Casamatta. Robust iterative learning control of an exothermic semi-batch chemical reactor. *Mathematics and Computers in Simulation*, 57:367–385, 2001.
94. P. Fu and J.P. Barford. Simulation of an iterative learning control system for fed-batch cell culture processes. *Cytotechnology*, 10:53–62, 1992.
95. D. Levišauskas, V. Galvanauskas, R. Simutis, A. Žilinskas, and J. Žilinskas. Optimization of biomass production in fed-batch culture by feed and dilution control actions. *Information Technology and Control*, 35(4):383–390, 2006.
96. E. Franco-Lara and D. Weuster-Botz. Estimation of optimal feeding strategies for fed-batch bioprocesses. *Bioprocess and Biosystems Engineering*, 27:255–262, 2005.
97. B. Srinivasan, C.J. Primus, D. Bonvin, and N.L. Ricker. Run-to-run optimization via control of generalized constrains. *Control Engineering Practice*, 9:911–919, 2001.
98. A.P. Teixeira, J.J. Clemente, A.E. Cunha, M.J.T. Carrondo, and R. Oliveira. Bioprocess iterative batch-to-batch optimization based on hybrid parametric/nonparametric models. *Biotechnology progress*, 22:247–258, 2006.
99. B.M. Wise, N.B. Gallagher, S.W. Butler, D.D. White Jr., and G.G. Barna. A comparizon of principal component analysis, multiway principal component analysis, trilinear decomposition and parallel factor analysis for fault detection in a semiconductor etch process. *Journal of Chemometrics*, 13:379–396, 1999.
100. D. Aguado. *Aplicación de Métodos Estadísticos Multivariantes para la Modelación y Monitorización de un Reactor Discontinuo Secuencial para el Tratamiento de Aguas Residuales, PhD thesis, Universidad Politécnica de Valencia (in spanish)*. 2005.
101. D. Aguado, A. Ferrer, A. Seco, and J. Ferrer. Comparison of different predictive models for nutrient estimation in a sequencing batch reactor for wastewater treatment. *Chemometrics and Intelligent Laboratory Systems*, 84:75–81, 2006.
102. P.R.C. Nelson, P.A. Taylor, and J.F. MacGregor. Missing data methods in pca ans pls: score calculations with incomplete observations. *Chemometrics and Intelligent Laboratory Systems*, 35:45–65, 1996.
103. F. Arteaga and A. Ferrer. Dealing with missing data in mspc: several methods, different interpretations, some examples. *Journal of Chemometrics*, 16:408–418, 2002.
104. B.S. Dayal and J.F. MacGregor. Recursive exponentially weightes pls and its applications to adaptive control and prediction. *Journal of Process Control*, 7:169–179, 1997.
105. M.E. Tipping and Ch.M. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation, MIT Press*, 11(2):443–482, 1999.
106. C. Zhao, F. Wang, N. Lu, and M. Jia. Stage-based soft-transition multiple pca modeling. *Journal of Process Control, doi:10.1016/j.jprocont.2007.02.005*, 2007.
107. H.T. Eastment and W.J. Krzanowski. Cross-validatory choice of the number of components from a principal component analysis. *Technometrics*, 24(1):73–77, 1982.
108. D. Giancarlo and Ch. Tommasi. Cross-validation methods in principal component analysis: a comparison. *Statistical Methods and Applications*, 11:71–82, 2002.

109. S. Wold. Cross-validatory estimation of the number of components in factor and principal components. *Technometrics*, 20(4):397–405, 1978.
110. B.M. Wise, N.B. Gallagher, R. Bro, J.M. Shaver, W. Windig, and R.S. Koch. *PLSToolbox 3.5 for use with Matlab*. Eigenvector Research Inc., 2005.
111. D.J. Louwerse, A.K. Smilde, and H.A.L. Hiers. Cross-validation of multiway component models. *Journal of Chemometrics*, 13:491–510, 1999.
112. W.O. McReynolds. Characterization of some liquid phases. *Journal of Chromatography Science*, 8:685–691, 1970.
113. S. Wold and K. Andersson. Major components influencing retention indices in gas chromatography. *Journal of Chromatography*, 80:43–59, 1973.
114. D.J. Louwerse and A.K. Smilde. Multivariate statistical process control of batch processes based on three-way models. *Chemical Engineering Science*, 55:1225–1235, 2000.
115. G.E.P. Box and A. Luceno. *Statistical Control by Monitoring and Feedback Adjustments*. John Wiley & Sons, New York, 1997.
116. S. García-Mu noz, T. Kourti, and J.F. MacGregor. Model predictive monitoring for batch processes. *Industrial and Engineering Chemical Research*, 43:5929–5941, 2004.
117. F. Arteaga and A. Ferrer. Framework for regression-based missing data imputation methods in on-line mspc. *Journal of Chemometrics*, 19:439–447, 2005.
118. P. Nomikos and J.F. MacGregor. Multi-way partial least squares in monitoring batch processes. *Chemometrics and Intelligent Laboratory Systems*, 30:97–108, 1995.
119. J.M. Lee, C.K. Yoo, and I.B. Lee. Enhanced process monitoring of fed-batch penicillin cultivation using time-varying and multivariate statistical analysis. *Journal of Biotechnology*, 110:119–136, 2004.
120. J.A. Westerhuis and P.M.J. Coenegracht. Multivariate modelling of the pharmaceutical two-step process of wet granulation and tableting with multiblock partial least squares. *Journal of Chemometrics*, 11:379–392, 1997.
121. D. Aguado, M. Zarzo, A. Seco, and A. Ferrer. Process understanding of a wastewater batch reactor with block-wise pls. *Environmetrics (In PRESS)*.
122. D. Bonvin, B. Srinivasan, and D. Hunkeler. Control and optimization of batch processes - improvements of process operation in the production of specialty chemicals. *IEEE Transactions on Control Systems Magazine*, 26:34–45, 2006.
123. T. Yamanè and S. Shimizu. Fed-batch techniques in microbial processes. *Advances in Biochemical Engineering/Biotechnology*, 30:147–194, 1984.
124. J. Lee, S.Y. Lee, S. Park, and A.P.J. Middelberg. Control of fed-batch fermentations. *Biotechnology Advances*, 17:29–48, 1999.
125. C. Duchesne and J.F. MacGregor. Multivariate analysis and optimization of process variable trajectories for batch processes. *Chemometrics and Intelligent Laboratory Systems*, 5:125–137, 2000.
126. P.E. Wellstead and M.B. Zarrop. *Self Tuning Systems Control and Signal Processing*. John Wiley & Sons, England, 1991.
127. M. Titica, D. Dochain, and M. Guay. Adaptive extremum seeking control of fed-batch bioreactors. *European Journal of Control*, 9:618–631, 2003.
128. D. DeHaan and M. Guay. Extremum-seeking control of state-constrained nonlinear systems. *Automatica*, 41:1567–1574, 2005.
129. K.B. Ariyur and M. Krstić. *Real-Time Optimization by Extremum-Seeking Control*. John Wiley & Sons, England, 2003.

130. H.H. Wang, M. Krstić, and G. Bastin. Optimizing bioreactors by extremum seeking. *International Journal of Adaptive Control and Signal Processing*, 13:651–669, 1999.
131. H.T.B. Pham, G. Larsson, and S. Enfors. Growth and energy metabolism in aerobic fed-batch cultures of saccharomyces cerevisiae: Simulation and model verification. *Biotechnology and Bioengineering*, 60(4):474–482, 1998.