1 2 9 0

UNIVERSIDADE D
COIMBRA

Ricardo Alberto Reis Silva Rendall

Monitorização e Previsão Da Qualidade em Processos Descontínuos

Desenvolvimento de Metodologias de Elevada Granularidade

Tese no âmbito do Doutoramento em Engenharia Química orientada pelo Professor Doutor Marco Paulo Seabra dos Reis e apresentada ao departamento de Engenharia Química da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Maio de 2019

Faculty of Sciences and Technology

BATCH PROCESS MONITORING AND QUALITY PREDICTION
Development of Coarse-Grained Data-Driven Methodologies

Ricardo Alberto Reis Silva Rendall
(Master in Chemical Engineering)

Doctoral Thesis in Chemical Engineering under the supervision of Professor Doctor Marco Paulo Seabra dos Reis and presented to the Department of Chemical Engineering, Faculty of Sciences and Technology, University of Coimbra.

May 2019

1 2 9 0

UNIVERSIDADE Ð COIMBRA

Acknowledgments

I would like to express my gratitude to all who have contributed to my journey and made it so worthwhile and fulfilling. I am grateful to my supervisor, Professor Doctor Marco Reis, for his guidance, insights, ideas, and a positive attitude that is unpaired, an example that goes beyond scientific supervision. Moreover, I thank him for introducing me to the "data-driven world" and equipping me with "lenses" that gave me a new perspective on solving Chemical Engineering problems. For this, I will be forever grateful.

I thank my colleagues and friends that enhanced my experience during these years, providing discussions, feedback, and an environment that was both stimulating and enriching. I am grateful to the friends from the research lab C16: Ana Lopes, Carolina Vertis, José Granjo, Paulo Dias, Tiago Rato, and Veronique Gomes. I also thank João Costa, a friend since my undergraduate studies, and Kelvin Fortes, a childhood friend that still accompanies me today.

I would like to acknowledge the support from Dow and the people that offered unique industrial insights and suggestions. I am grateful to Alix Schmidt, Bo Lu, Brigit Braun, Ivan Castillo, Leo Chiang, Lloyd Colegrove, and Swee-Teng Chin.

I am grateful to my family for their exemplary values of love and unity. I thank my mother, Augusta Reis, for always supporting and advising me on everything that is really important in life. I thank my uncle Nelson Reis for his always thoughtful attitude and my aunts, Elsie Maia, and Nazaré Mariano, for their continued support.

Lastly, to all others that helped me throughout this journey but whose name I did not mention, I offer my deepest apologies and a big thank you!

Abstract

The first holistic approaches for batch data analysis were proposed around two decades ago, with an emphasis on statistical process control and quality prediction. In the subsequent years, more methods were proposed, with varying degrees of success and acceptance by practitioners. A detailed and comprehensive analysis of these contributions reveals different complexity levels, both in terms of the degrees of freedom used for setting up the techniques (modeling complexity) and the expertise/training required by practitioners to autonomously apply them in concrete real world applications (implementation complexity). Both dimensions decisively contribute to the impact of a given proposal in industry but, analyzing carefully the technical literature, it is possible to notice a rather clear trend towards increasingly complex batch data analysis methods. These methods are named fine-grained approaches since the time dimension is explicitly transposed and incorporated in the data-driven model. To this end, they require intricate pre-processing techniques (high implementation complexity) and a high number of model parameters (high modeling complexity). On the other hand, alternative approaches that are simpler, but also more robust and potentially effective have been largely overlooked. These methods often summarize the batch evolution in fewer quantities called features and compress or remove the time dimension from data-driven model building. Therefore, they are named coarse-grained approaches. Coarse-grained methods, and feature-oriented methods in particular, constitute the main focus of this thesis and an effort is made to develop and unify all the available methodologies belonging to this category. More specifically, a novel feature-oriented method, named profile-driven features (PdF), is developed, tested, and compared to benchmark alternatives. The comparison considers many of the important tasks that are routinely conducted in batch data analysis, such as exploratory data analysis, process monitoring, and quality prediction. It is shown that PdF extracts specific features from the trajectories of the process variables and can provide results that are either on par or better when compared to those obtained with more complex methods. Afterwards, feature-oriented methods are integrated and unified into a batch analytics framework called FOBA, demonstrating how these methods can be utilized in assisting practitioners with their daily activities. Furthermore, an extension of FOBA is also developed to increase the flexibility of feature-oriented methods and identify important periods of the batch evolution.

Another topic discussed in this thesis refers to the need to employ efficient feature selection procedures when analyzing batch datasets. This stems from the fact that there can be many features available for data-driven model building; however, many of them are highly noisy or may be even irrelevant to the predictive task at hand. In this context, a new two-stage approach, called wide spectrum feature selection (WiSe), is proposed to remove noisy and irrelevant features. In the first stage of WiSe, various metrics of feature importance are employed in order to detect relevant predictors and select them. In the second stage, predictive methods that have built-in mechanisms for feature selection are utilized to further reduce the number of features. Consequently, the developed models tend to be more parsimonious, with the consequent advantages on increased robustness and improved prediction performance. This was confirmed by a series of case studies that demonstrate the method's effectiveness and suitability in high-dimensional scenarios.

The last topic discussed in this thesis covers the development of a predictive analytics comparison framework (PAC) for assessing the performance of predictive methods. The features computed from batch trajectories can be utilized for predicting important quality parameters; however, during data-driven model building, practitioners face the challenging task of selecting the best method for their application. This selection process is frequently constrained by limited *a priori* knowledge about the characteristics and mechanisms generating the data and, therefore, prone to be sub-optimal. In such scenarios, practitioners often choose their preferred method(s) without a proper assessment of other methods (and other classes of methods) that may bring predictive advantages. PAC is a platform for these scenarios since it considers a wide variety of methods and assesses their performance using robust comparison metrics based on statistical hypothesis testing. Furthermore, PAC provides output results in a user-friendly fashion, allowing the identification of the best method(s) and the most important features influencing quality. PAC was applied to a variety of case studies, covering simulated and real world datasets, and the results demonstrate its benefits in providing insights into the prediction problem at hand as well as speeding up the process of model screening and development.

Resumo

As primeiras abordagens holísticas para análise de dados de processos descontínuos foram propostas há cerca de duas décadas e focavam-se essencialmente no controlo estatístico de processos e na previsão da qualidade. Desde então, novos métodos foram propostos, com diferentes níveis de sucesso e aceitação prática. Uma análise detalhada destas contribuições revela que elas possuem diferentes níveis de complexidade, tanto em termos dos graus de liberdade utilizados no desenvolvimento dos modelos (complexidade do modelo) como em termos do conhecimento necessário para aplicá-los autonomamente (complexidade de implementação). Estas dimensões contribuem decisivamente para o impacto de uma dada metodologia na indústria, no entanto, a literatura recente da análise de dados de processos descontínuos tende a favorecer métodos cada vez mais complexos. Esses métodos denominam-se por abordagens de baixa granularidade uma vez que a dimensão do tempo é explicitamente incorporada no modelo e requerem técnicas de pré-processamento complexas (alta complexidade de implementação) e conduzem a modelos com um elevado número de parâmetros (modelos complexos). Por outro lado, abordagens alternativas mais simples, mas que também são robustas e eficazes, têm sido em grande parte negligenciadas. Estes métodos sumariam a evolução do lote em algumas quantidades denominadas por *features*, compactando ou mesmo removendo a dimensão temporal. Assim sendo, eles denominam-se por abordagens de elevada granularidade e constituem o tema principal desta tese. Mais especificamente, um novo método, denominado por *profile-driven features* (PdF) foi desenvolvido, testado e comparado com abordagens alternativas. A comparação aborda muitas das tarefas típicas da análise de dados, nomeadamente a análise exploratória de dados, a monitorização de processos e a previsão da qualidade. É demonstrado que o método PdF extrai características específicas das trajetórias das variáveis e conduz a resultados similares ou superiores aos obtidos com métodos mais complexos. Os métodos baseados em *features* são também integrados e unificados numa plataforma analítica (designada FOBA, *feature-oriented batch analytics framework*) que demonstra como esses métodos podem ser utilizados para auxiliar profissionais em suas atividades diárias.

Outro tópico discutido nesta tese refere-se à necessidade de aplicar métodos eficientes para a seleção de *features*. Esta necessidade advém do facto de muitas *features* conterem elevados níveis de ruído ou serem até irrelevantes num dado contexto de previsão. Assim sendo, uma nova abordagem para a seleção de *features* é proposta. Esta abordagem denomina-se por *wide spectrum feature selection* (WiSe). Na primeira etapa, várias métricas de importância são utilizadas para detetar e selecionar preditores relevantes. Na segunda etapa, métodos preditivos que também possuem mecanismos automáticos para a seleção de features são utilizados. Consequentemente, os modelos desenvolvidos tendem a ser mais parcimoniosos, robustos e com melhor desempenho na previsão. Estas vantagens são verificadas num conjunto de casos de estudos que demonstram a eficácia do método em diferentes cenários de aplicação.

O último tópico discutido nesta tese aborda o desenvolvimento de uma plataforma de seleção métodos de previsão denominada *predictive analytics comparison framework* (PAC). As *features* calculadas a partir das trajetórias das variáveis de processo são frequentemente utilizadas na previsão de parâmetros de qualidade. No entanto, o desenvolvimento de modelos é frequentemente condicionado pela difícil tarefa de selecionar o melhor método para uma dada aplicação. Este processo de seleção é frequentemente dificultado por um escasso conhecimento prévio sobre as características e os mecanismos que geram os dados e, portanto, a escolha tende a ser subótima. Em tais cenários, o utilizador muitas vezes seleciona o seu método preferido sem uma análise prévia de outros métodos (e outras classes de métodos), potencialmente melhores. PAC é uma plataforma concebida para lidar com esses cenários, pois inclui uma ampla variedade de métodos e avalia os seus desempenhos utilizando métricas de comparação robustas baseadas em testes estatísticos de hipóteses. Além disso, a PAC produz resultados intuitivos, permitindo a identificação do melhor método e quais as *features* mais importantes que influenciam a resposta. Esta metodologia foi aplicada a uma variedade de casos de estudo, contemplando dados simulados e reais, e os resultados demonstraram que o processo de desenvolvimento do modelo é mais expedito e eficaz, e que em paralelo, permite obter mais informação relevante sobre o problema.

Abbreviations

Abbreviations

AR	Auto-regressive
ARPCA	Auto-regressive principal component analysis
BaRT	Bagging of regression trees
BoRT	Boosting of regression trees
BDA	Batch data analysis
BDPCA	Batch dynamic principal component analysis
BS	Best subset
BWU	Batch-wise unfolding
COW	Correlation optimized warping
CPI	Chemical Processing Industry
DTW	Dynamic time warping
DDTW	Derivative dynamic time warping
DPCA	Dynamic principal component analysis
DPCA-DR	Dynamic principal component analysis with decorrelated residuals
EN	Elastic net
ESI	Equally spaced intervals
ESID	Equally spaced intervals for consistent dynamic variables
FOBA	Feature-oriented batch analytics framework
FN	False negatives
FP	False positives
FSM	Feature selection methods
FSR	Forward stepwise regression
GA	Genetic algorithms
HPLC	High-performance liquid chromatography
IV	Indicator variable
KPI	Key performance indicator
Lasso	Least absolute shrinkage and selector operator
LSD	Least significant differences
MICA	Multi-way independent component analysis
MPCA	Multi-way principal component analysis
MPLS	Multi-way partial least squares
MRQP	Multiresolution quality prediction

OLS	Ordinary least squares
NOC	Normal operating conditions
PAC	Predictive analytics comparison framework
PCA	Principal component analysis
PCR	Principal component regression
PdF	Profile-driven features
PLS	Partial least squares
rPLS	Recursive weighted partial least squares
RF	Random forests
SFFS	Sequential forward float selection
SPA	Statistical pattern analysis
SNV	Standard normal variate
SVR	Support vector regression
TIME-F	Translation-invariant multiscale energy-based features
TN	True negatives
TP	True positives
USID	Unequally spaced intervals for consistent dynamic variables
VIP	Variable importance in projection
VWU	Variable-wise unfolding
WiSe	Wide spectrum feature selection

Table of Contents

List of Figures

List of Tables

Chapter I
Introduction

1 Introduction

This introductory chapter is divided into four sections that altogether provide the context for the work presented in this thesis. The first section introduces the topics of batch processes and batch data analysis, highlighting their prevalence in modern industry. These topics constitute the backbone of this thesis and the underlying application environments. In the second section, the pursued goals are presented, whereas in the third section, the contributions made in order to achieve such goals are enumerated. The last section describes the thesis' organization, detailing the topics addressed in each chapter.

1.1 Scope and motivations

Batch processes play a central role in modern industry, being ubiquitous in a wide variety of sectors such as chemicals, oil & refining, biofuels, pharmaceutical, food & beverages, semiconductors, etc. They are characterized by the cyclic repetition of pre-defined operational procedures (known as "batch recipes"), which are often grouped in different batch stages (e.g., load reactants, set reaction conditions, control reaction time to achieve the desired conversion, set reactor discharge conditions, clean the vessel) and may involve different process times (from seconds to several years, as in vintage wines production), unit operations (e.g., in chemical and pharmaceutical processes) and even different companies (as in semiconductors). The batch operation mode offers great flexibility to change process conditions, production volumes, and allows for a better control of the batch evolution. Therefore, batch systems are the preferred solution for plants with lower and varying production rates and/or producing multiple products. However, this added flexibility comes with a cost. The operational flexibility offers more opportunities for natural variability to enter the process, as well as diverse special causes, leading to increased variability or problems in batch evolution, if not properly monitored and controlled. Furthermore, batch processes are intrinsically dynamic and non-stationary, requiring proper data-driven techniques that can correctly take into account these features in a multivariate (often high-dimensional) context. Thus, methods with greater flexibility are required to process batch data, when compared to those employed for treating data arising from continuous processes.

Typical tasks conducted in the scope of batch data analysis (BDA) include process monitoring and control (Nomikos and MacGregor 1994, Gallagher, Wise et al. 1996, Undey and Cinar 2002, Cinar, Parulekar et al. 2003, Kourti 2003, Camacho and Picó 2006, Marjanovic, Lennox et al. 2006, Camacho, Picó et al. 2009, Das, Maiti et al. 2012), fault detection and diagnosis (Kourti and MacGregor 1995, Dunia and Joe Qin 1998, García-Muñoz, Kourti et al. 2003, Ündey, Ertunç et al. 2003, Miletic, Quinn et al. 2004), batch-end quality prediction (Boqué and Smilde 1999, Gurden, Westerhuis et al. 2001, Ündey, Tatara et al. 2004, Lu and Gao 2005), optimization (MacGregor and Cinar 2012), among others. These data-driven activities will benefit from the recent trends of big data analytics and Industry 4.0 initiatives (García-Muñoz and MacGregor 2016, Reis, Braatz et al. 2016, Chiang, Lu et al. 2017), which are expected to have a significant impact on plant performance, bringing data analytics to the forefront of process management. Given the pervasiveness and complexity of batch processes, it is not surprising that a variety of data-driven methods have been proposed and applied for BDA. These methods have different *a priori* assumptions and their suitability is dependent on the goals, the data available and whether the methods find good adherence to the characteristics of collected data, which are often unknown beforehand (Reis and Kenett 2018). The current trend in BDA has favored more complex methodologies (Ge, Song et al. 2013, Wang, Wang et al. 2016) that aim to model intricate relationships found in batch data. These methodologies occupy the top spectrum of complexity because they employ complex and cumbersome data pre-processing techniques, combined with very flexible models. Moreover, the insights obtained are often limited due to challenges in interpreting such intricate models and robust assessment metrics are required to avoid overfitting. On the other hand, the development of effective and simpler approaches (i.e. methods with low complexity) for BDA has been largely overlooked. This is one of the main motivations for the research efforts undertaken in this thesis. Simple methods for BDA can be adopted in a more straightforward fashion by a larger number of users, often produce a good baseline performance to be used as benchmark, and produce easily interpretable outcomes. The suitability of such methods is even more pronounced for offline applications (e.g., root-cause analysis, troubleshooting) since these are rare events, where employing all the machinery necessary for complex methods may not be the most cost-effective solution, as similar conclusions can often be achieved in a more time-efficient manner with simpler methods.

Batch datasets are also high-dimensional, with many variables being collected and used for data-driven modeling (either the original measured variables or features derived from them). In this context, selecting, in an efficient way, the key factors affecting batch quality is of paramount importance because they constitute the critical-to-quality drivers to scrutinize and optimize. This constitutes a second motivation for the work developed in this thesis, which relates to the need for selecting the right features/variables for BDA tasks: separating the vital few from the trivial many, using Juran's well-known terminology.

Finally, after employing simple and effective methods for BDA and identifying the important batch quality factors, one is often interested in building a model that relates the predictors with the quality parameters. To this end, a vast literature on predictive methods exist (Hastie, Tibshirani et al. 2001, Chatterjee and Hadi 2015) that, tacitly or explicitly, assume a variety of data-generating mechanisms that may be dictating the characteristics of the collected datasets. These mechanisms may consider the distribution of the predictors (process variables or features), the distribution of the quality variable(s), and the relationship between predictors and quality variables. In scenarios where *a priori* knowledge cannot be utilized for selecting a suitable predictive method, comparison studies and comparison frameworks are needed to guide the selection process. These frameworks should assess the potential of different predictive methods in a robust fashion and produce results that are easy to interpret. As an additional benefit, these frameworks avoid the background bias introduced by practitioners that often choose to use their preferred method in most applications. The development of a robust framework for assessing and comparing the predictive ability of a carefully selected pool of methods covering the analytics domain in a non-redundant way, is the third and last motivation for the work developed in this thesis.

1.2 Goals

The present research focuses on developing new data-driven methodologies that can be utilized for extracting information from datasets collected during the operation of batch processes. Therefore, the following goals were specified to overcome some of the current limitations of the methodologies employed for BDA:

i) Develop methodologies for BDA that occupy the low complexity spectrum. These methodologies should be effective, simple to apply, and robust. Practitioners would greatly benefit from the availability of simpler methods that can provide insights into the process operation and the main factors driving quality, without the need to apply cumbersome and complex pre-processing techniques;

ii) Develop a feature-oriented framework for batch data analysis that supports a structured approach for addressing a variety of goals, namely quality prediction and process monitoring. This framework should contain capabilities of feature generation, selection and model building.

These general goals can be achieved through the development of novel methodologies and frameworks that take into account the challenges of batch datasets. These methodologies are envisioned to be applied in real world scenarios and, therefore, should comply with the requirements of being simple, robust, interpretable, and scalable.

1.3 Contributions

The main contributions of this thesis are the following:

i) A thorough and systematic literature review focused on methods for batch data analysis, with a special emphasis on process monitoring and quality prediction. We adopt a taxonomy that allows grouping different methodologies into classes that share similar elements in terms of their complexity levels. In this context, we consider two dimensions of complexity: modeling complexity, which is related to the ability of a given method to fit a dataset, and implementation complexity, which ascertain the difficulty that practitioners face when testing and implementing a particular method. This taxonomy helps practitioners locate where their current preferred method resides in the complexity spectrum and consider alternatives by weighting their associated potential added value and complexity;

ii) The development of a methodology and a framework for batch data analysis that occupies the low spectrum of complexity. This methodology does not require intricate pre-processing techniques compared to traditional approaches but is able to extract relevant information from the process variables' profiles and summarize their dynamic content by a few quantities, called features. The methodology is named profile-driven features (PdF) and can be employed to transform the dataset collected from a batch process into a matrix of features. The framework is called feature-oriented batch analytics framework (FOBA) and encompasses several feature-oriented methods in an integrated and unifying fashion;

iii) An extension of FOBA, called FOBA 2.0, was also developed that enhances the time-resolution characteristics of feature-oriented methods and allows extracting localized information from the batch evolution. In particular, FOBA 2.0 has built-in mechanisms to divide the complete batch trajectory into smaller sub-intervals, which are later analyzed to identify batch periods that are critical to quality;

iv) The proposal of a new variable/feature selection scheme suitable for the high-dimensional datasets found in batch processes. The method is named wide spectrum feature selection (WiSe) and it is efficient and scalable to large datasets. The scalability is a consequence of a two-stage approach: the first stage removes irrelevant variables, whereas in the second stage model building is conducted in a more efficient manner and a final selection round is performed;

v) The development and application of a predictive analytics comparison framework (PAC) for assessing and comparing the prediction performance of different classes of predictive methods. Selected methods from each class were considered in order to enlarge the framework's potential to properly model the relationship between features and quality parameters. The outputs of PAC provide critical information in scenarios characterized by limited *a priori* knowledge, where selecting a suitable predictive method is not a straightforward task.

The contributions ii-v) can be summarized in a workflow that converts a batch dataset into a set of features (using FOBA or FOBA 2.0), whose dimensionality can be reduced by feature selection (using WiSe) and that can be finally utilized for predictive modeling (with PAC). This workflow is presented in Figure 1.1 and each of the depicted stages will be detailed in the specific chapters of this thesis. The scientific papers associated with each contribution are presented in Table 1.1.

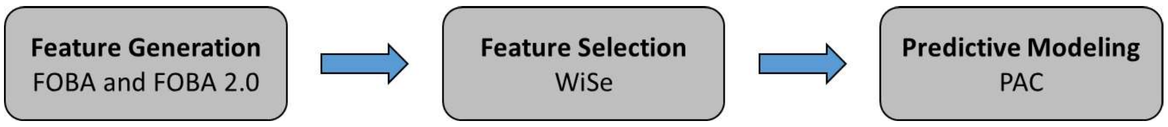


Figure 1.1. The topics discussed in this thesis and their logical sequence.

Table 1.1. Published papers associated with each of the contributions made in this thesis.

Contribution	Reference
i)	R. Rendall, L.H. Chiang, M.S. Reis, *Data-driven Methods for Batch Data Analysis – A Critical Overview and Mapping on a Complexity Scale,* Computers and Chemical Engineering (2019).
ii)	R. Rendall, B. Lu, I. Castillo, S.-T. Chin, L.H. Chiang, M.S. Reis, *A Unifying and Integrated Framework for Feature-Oriented Analysis of Batch Processes*, Industrial & Engineering Chemistry Research (2017).
iv)	R. Rendall, I. Castillo, A. Schmidt, S.-T. Chin, L.H. Chiang, M. Reis, *Wide Spectrum Feature Selection (WiSe) for Regression Model Building*, Computers & Chemical Engineering (2018).
v)	R. Rendall, M.S. Reis, *Which regression method to use? Making informed decisions in "data-rich/knowledge poor" scenarios – The Predictive Analytics Comparison framework (PAC)*, Chemometrics and Intelligent Laboratory Systems (2018).

1.4 Thesis organization

This thesis is divided into seven chapters that are presented in Figure 1.2.

Chapter I provides an introduction to the thesis, presenting the scope and motivations for this work, as well as the goals that were established. Additionally, the main contributions achieved in pursuing such goals are listed.

Chapter II presents a structured state-of-the-art review of methodologies utilized for batch data analysis and maps available methods according to their modeling and implementation complexities. This mapping provides a topological organization that highlights different classes of methods and helps to identify gaps that may be pursued in future research projects.

Chapter III addresses feature-oriented approaches for batch data analysis and provides a detailed description of available methods. The chapter describes single-granularity methods whose time-resolution is limited by the batch duration or stage duration. In particular, a new feature-oriented method named profile-driven features is proposed and compared to other alternative approaches in a series of simulated and real-world case studies.

Chapter IV extends feature-oriented methods in order to extract localized information from the batch evolution. A set of alternatives are tested to enhance the time-resolution characteristics of feature-oriented methods, allowing the identification of critical periods of the batch. The performance of these alternatives are assessed and compared in a simulated case study that validates their advantage over the standard approaches.

Chapter V focuses on feature selection methodologies that identify and select important features, whereas irrelevant features are discarded. These methodologies are important tools that are used to reduce the dimensionality of the collected datasets, leading to more parsimonious and robust models.

Chapter VI describes a comparison framework for predictive analytics that can provide insights into datasets arising in prediction problems and may lead to improved prediction performance. This framework contemplates methods from different classes in order to model various relationships between predictors and response variable, as well as a robust comparison scheme to assess and compare the methods' performance.

Finally, Chapter VII summarizes the main conclusions of this thesis and enumerates directions for future research.

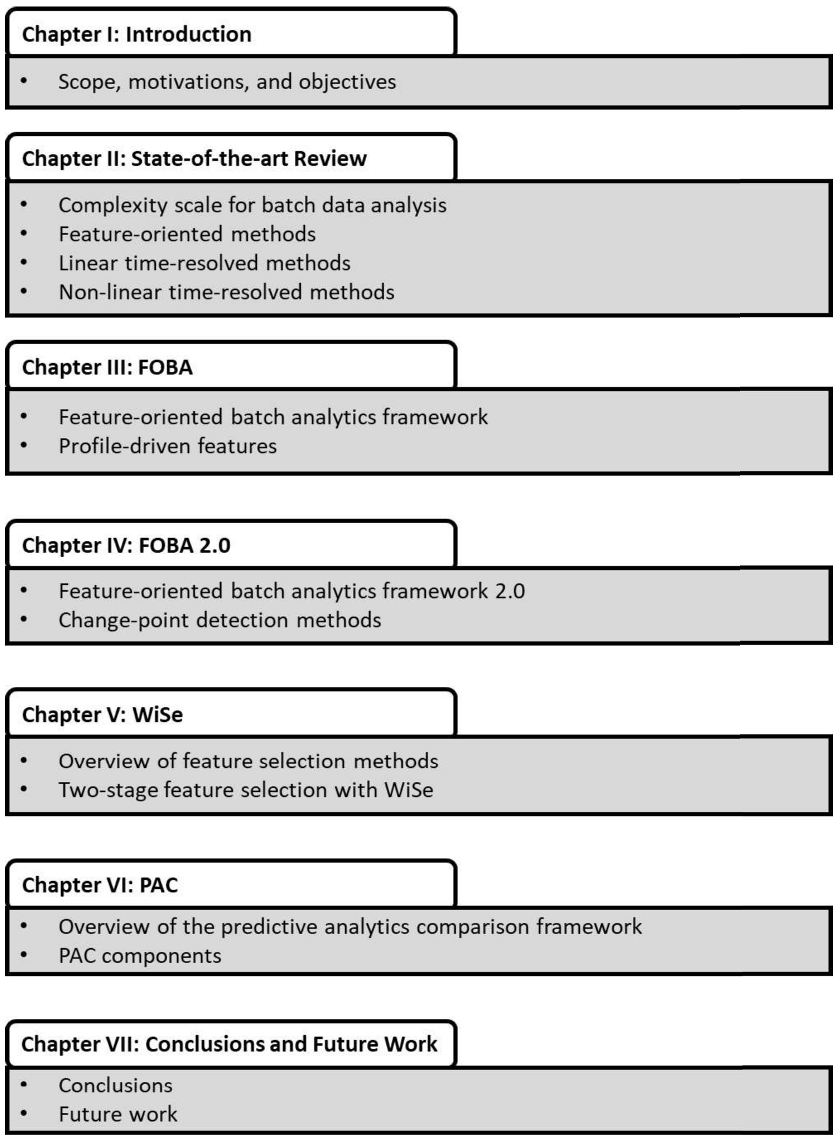

Chapter I: Introduction
• Scope, motivations, and objectives
Chapter II: State-of-the-art Review
• Complexity scale for batch data analysis
• Feature-oriented methods
• Linear time-resolved methods
• Non-linear time-resolved methods
Chapter III: FOBA
• Feature-oriented batch analytics framework
• Profile-driven features
Chapter IV: FOBA 2.0
• Feature-oriented batch analytics framework 2.0
• Change-point detection methods
Chapter V: WiSe
• Overview of feature selection methods
• Two-stage feature selection with WiSe
Chapter VI: PAC
• Overview of the predictive analytics comparison framework
• PAC components
Chapter VII: Conclusions and Future Work
• Conclusions
• Future work

Figure 1.2. Overview of the organization of this thesis.

Chapter II
State-of-the-Art Review

2 State-of-the-Art Review

This chapter presents the state-of-the-art review regarding methods employed for batch data analysis (BDA), with a particular emphasis on process monitoring and quality prediction applications. These two tasks constitute a major portion of the research on BDA and their importance has been widely recognized in the literature, given the large number of publications dedicated to them. In this context, the following sections provide a critical overview of methods utilized in BDA, where four main topics are discussed:

i) A complexity scale for BDA methods is presented, which allows practitioners to easily identify where their current preferred methods are located and to compare them to other available alternatives. Three classes of BDA methods are identified: feature-oriented methods, linear time-resolved methods, and non-linear time-resolved methods;

ii) Feature-oriented methods are presented and the main proposals available in the literature are described. These methods occupy the lower positions on the complexity spectrum;

iii) Linear time-resolved methods are a class of methods that transpose the time dimension to the modeling stage and, therefore, have increased flexibility. However, the risk of overfitting is higher, and their implementation requires synchronization of batch profiles, which represents a significant increase in terms of implementation complexity;

iv) Finally, non-linear time-resolved methods are presented. These approaches extend linear methods by being able to model non-linear relationships. They lie on the top of the complexity spectrum and have the highest flexibility and also the largest potential for overfitting.

The last section of this chapter presents a methodologic workflow to aid practitioners in the task of selecting a suitable method for their particular applications. This workflow is inspired by the Occam's razor principle, where complexity is gradually increased as long as increments in performance are observed that justify their adoption.

2.1 Complexity scale for batch data analysis

The performance of data-driven methods for BDA is governed by the well-known bias and variance trade-off (Hastie, Tibshirani et al. 2001) and its consequences. Unless there is a perfect match between the data generating mechanism and the model structure, fewer degrees of freedom imply more biased estimates, given the limitations for modeling the existing relationships between process variables (either input variables only, or input-output/quality relationships). On the other hand, a model with too many degrees of freedom leads to high variance estimates, as it is prone to fit spurious correlations in the training dataset, i.e., overfitting data. In statistical theory (Hastie, Tibshirani et al. 2001), this reasoning can be expressed as the partition of the mean squared error (MSE) of an estimator $\hat{\theta}$ into a bias and a variance component:

$$E\left[\left(\hat{\theta}-\theta\right)^2\right]=\left(E\left[\hat{\theta}\right]-\theta\right)^2+E\left[\left(\hat{\theta}-E\left[\hat{\theta}\right]\right)^2\right] \qquad (1)$$

The first term in the right-hand side of eq. (1) corresponds to the squared bias and accounts for the deviation between the expected value of the estimator $\hat{\theta}$ and the actual value for the parameter θ. The second term in the right-hand side of eq. (1) is the variance of the estimator and measures the dispersion of the estimator around the expected value. To achieve a small MSE $\left(E\left[\left(\hat{\theta}-\theta\right)^2\right]\right)$, the right complexity level of BDA methods has to be found, by balancing bias and variance. Within the scope of a given BDA method, this balance is often established recurring to cross-validation or similar techniques (Vitale, Westerhuis et al. 2017) that emulate or estimate the methods' performance under testing conditions. However, before this tuning stage, it is important to select the right BDA method to employ, which must also take into consideration the complexity inherent to the method. Therefore, it is important to define more rigorously what is meant by the term complexity.

In this work, we consider the complexity of BDA methods as a multidimensional entity with two major contributors: modeling complexity and implementation complexity. The former type of complexity regards the flexibility of a BDA method to fit data and can be inferred from the number of model parameters that need to be estimated to set up the technique. Modeling complexity is closely related to the bias-variance trade-off (more complex methods tend to fit better the training data, leading to estimates with low bias and higher variance). More specifically, the number of degrees of freedom associated with a given method specifies its modeling complexity, and more complex methods contain a higher number of degrees of freedom. This fact suggests that BDA methods can be ordered in accordance with their modeling complexity. However, the number of degrees of freedom can seldom be computed from theoretical considerations only, and alternatives such as the pseudo-degrees of freedom (van der Voet 1999) can be utilized as a proxy for the model's "true" degrees of freedom. This alternative provides a quantifiable measure of complexity and can be used to compare different BDA methods. The second type of complexity, implementation complexity, considers the amount of resources required to operationalize the method (mainly know-how, but also software/hardware resources). In scientific publications, the analysis focus is usually entirely restricted to modeling complexity, but implementation complexity is decisive for transferring a method to the shop floor and strongly affects its acceptance by companies. Very complex methods, requiring highly specialized personnel and very specific software solutions, often meet strong difficulties when migrating to the shop floor, and are frequently discarded in favor of simpler, wide spectrum approaches, unless they prove to bring a large amount of added value. We would like to point out that, while assessing different aspects, these two dimensions of complexity usually present strong dependencies: more complex modeling frameworks tend to present higher implementation complexity, and the inverse is true for simpler methods, which have less modeling complexity and require less technical and computational resources to implement.

Figure 2.1 illustrates the complexity scale. It contains classes of methods used in BDA aligned according to an increasing ordering of complexity (from left to right), as well as some representative methods from each class. One should note that Figure 2.1 is not exhaustive in terms of the available methods for BDA, but it portraits a taxonomy that can help practitioners navigating through the space of solutions at their disposal, in a systematic way. We classify BDA methods into three main classes: feature-oriented methods, linear time-resolved methods, and non-linear time-resolved methods. This grouping provides a useful topological organization for practitioners to identify where their currently preferred methods stand in terms of both modeling and implementation complexities. It also highlights other promising approaches worthwhile considering in the same class. Moreover, it may also motivate them to explore other classes of methods, in accordance with the parsimony principle.

Feature-oriented methods present lower complexity levels. Representative methods from this class include landmark features, profile-driven features (PdF) (Rendall, Lu et al. 2017), statistical pattern analysis (SPA) (He and Wang 2011), and translation-invariant multiscale energy-based features (TIME-F) (Rato, Blue et al. 2017). The premise of feature-oriented methods is to transform a batch profile into a set of features that capture relevant aspects of the batch evolution. These features are often few in number, creating conditions for developing models with a small number of parameters (i.e. low modeling complexity). Furthermore, their implementation complexity is also low, because they do not require batch synchronization. In fact, many classical methods for treating two-way tables can be readily applied to analyze the tables of features generated by these methods.

As more complex methods are considered, linear time-resolved methods are the next class in the complexity scale. Common examples include 2-way variable-wise unfolding (2-way VWU) (Wold, Kettaneh et al. 1998), dynamic methods such as auto-regressive principal component analysis (ARPCA), batch dynamic PCA (BDPCA) (Chen and Liu 2002), 3-way methods (PARAFAC and Tucker3) (Louwerse and Smilde 2000), and 2-way batch-wise unfolding (2-way BWU) (Nomikos and MacGregor 1994, Nomikos and MacGregor 1995, Nomikos and MacGregor 1995). Time-resolved methods preserve time resolution for BDA and often result in models that contain a high number of parameters and, consequently, higher modeling complexity. Furthermore, they require batch synchronization, which represents an increase in implementation complexity (see Figure 2.1). In some particular batch processes, the batch-to-batch variability is very small and alignment can be ignored by resorting to ad-hoc techniques (e.g., cutting longer batches or extending smaller batches by repeating the last measurement). However, in the more general setting, these ad-hoc techniques are not applicable and will have negative consequences on model development, since unsynchronized datasets exhibit excessive variability. The objectives of synchronization are two-fold: to ensure that the key events of the batch occur at similar time points and to make all batches have the same duration. This is a critical step in order to apply time-resolved methods and impacts their performance (Gonzalez-Martinez, Vitale et al. 2014).

In the top of the complexity scale lie the non-linear time-resolved methods. These methods extend the linear-time resolved approaches in order to accommodate non-linear relationships. These are the most flexible methods, which are often based on kernels that map samples into high dimensional non-linear spaces where the modeling implicitly takes place. However, the risk of overfitting is significant and a high number of samples are usually required to obtain robust estimates of model parameters.

Data-driven methods for BDA can also be assessed from other points of view. For instance, they can be grouped based on whether they can be applied online and offline, or offline only. Most methods reviewed in this paper fall in the former category or can be straightforwardly adapted to online applications with little or no adjustments. However, the simpler feature-oriented methods have been only applied in an offline manner, since computing features requires knowing the full extent of the batch. Nevertheless, one should also note that offline BDA is of great utility in a variety of tasks. In fact, many batch processes (e.g., in the semiconductor industry) occur at very small time scales, where online BDA is not possible or relevant since no corrective actions could be taken in useful time. Furthermore, many times the batch-end properties take too long to obtain and require lengthy, complex and expensive laboratorial methods. To have a method at our disposal that immediately provides a reliable estimate at the end of the batch, with no extra cost, is a valuable strategic asset. For instance, it allows anticipating corrective actions for the next batch and speeds up the release of the current one, increasing operational efficiency, while reducing batch-to-batch variability and therefore improving product quality.

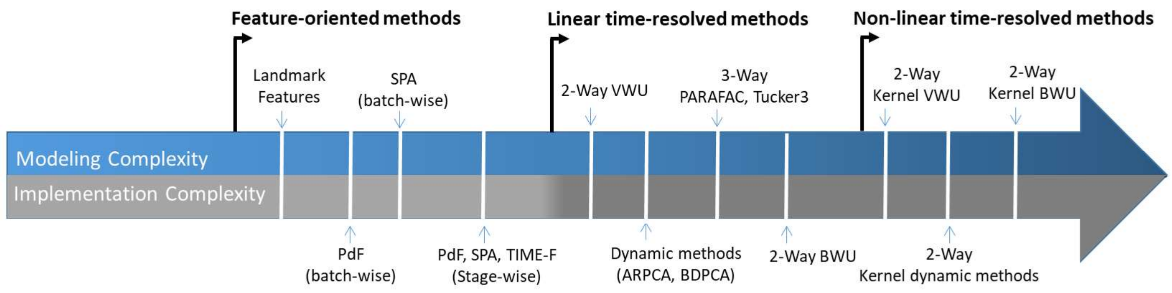

Feature-oriented methods
Linear time-resolved methods
Non-linear time-resolved methods
Landmark Features
SPA (batch-wise)
2-Way VWU
3-Way PARAFAC, Tucker3
2-Way Kernel VWU
2-Way Kernel BWU
Modeling Complexity
Implementation Complexity
PdF (batch-wise)
PdF, SPA, TIME-F (Stage-wise)
Dynamic methods (ARPCA, BDPCA)
2-Way BWU
2-Way Kernel dynamic methods

Figure 2.1. Modeling and implementation complexity of BDA methods. The gradients in coloring indicate that modeling complexity steadily increases, whereas the increase in implementation complexity shows a sudden jump when moving to time-resolved methods, requiring synchronization operations.

A class of methods not discussed in the scope of Figure 2.1 is the class of multi-block methods. It has been shown that a more meaningful analysis of batch data can be obtained when the data is divided into its natural blocks (Kourti, Nomikos et al. 1995). In particular, three blocks are often available: batch initial conditions, the trajectories of process variables during the batch, and quality parameters measured at the end of the batch. Multi-block methods can exploit this structure, giving results that are easier to interpret, both in terms of inner-relations within each block and outer-relations between blocks. Throughout this thesis, the focus is on single-block methods capable of handling the information extracted from the trajectories of process variables during the batch. Another topic that is also not covered concerns stage identification methods. Batch processes contain multiple stages, and the variables' mean levels and correlation structure differ from stage to stage. Stage-wise models can be combined, for instance, using multi-block methods, leading to alternative methods for BDA. The reader is referred to an overview by Yao and Gao (2009) and the references therein (Camacho and Picó 2006, Camacho, Picó et al. 2008) for a discussion on multi-stage methods.

2.2 Feature-oriented methods

As in the scope of pattern recognition (Pal and Mitra 2004), feature-oriented methods for BDA can also be viewed as a transformation from a measurement space *M* to the feature space *F* and finally to the decision space *D*:

$$M \rightarrow F \rightarrow D$$

The measurement space *M* contains the trajectory of process variables measured during process operation (measurements are arranged in a 3-way array $\underline{\mathbf{X}}^{raw}$, containing *J* process variables measured at K_i time intervals for *N* batches. Note that the trajectories in $\underline{\mathbf{X}}^{raw}$ may not be synchronized and may have different durations, since each *i*th batch has a length of K_i. The space *M* is mapped into the feature space *F*, following a feature extraction procedure, $\alpha : M \rightarrow F$ (also known as a mapping function or feature dictionary; a dictionary is a finite set of features computed according to a well-defined procedure – see below for some examples). The set *D* is discrete and finite in a classification problem and contains the response variable for each sample in *M*. In a regression setting, *D* is a continuous set containing the values of the response variable. An important advantage of feature-oriented methods is that batch trajectories don't need to be aligned. Instead, features are computed directly from $\underline{\mathbf{X}}^{raw}$. Variability in non-aligned batches is translated into features variability, and treated through conventional methods developed for handling two-way tables. This advantage makes their application more straightforward because synchronization can be a rather cumbersome task and greatly increases the implementation complexity of some methods (see Figure 2.1). Research on feature-oriented methods focuses on devising a suitable dictionary α in order to extract relevant features from measured data. In batch processes, the feature space tends to have small dimensionality and when correctly constructed, it should capture the main sources of structured variation in the batch. Once suitable features are computed, a modeling formalism is adopted to model their relationships with the response. A latent variable modeling framework is often applied for this purpose, as presented in eq. (2) and eq. (3):

$$\mathbf{X} = \mathbf{T}\mathbf{P}^T + \mathbf{E} \tag{2}$$

$$\mathbf{y} = \mathbf{T}\mathbf{c} + \mathbf{f} \tag{3}$$

where **X** is a $N \times m_f$ feature matrix, m_f is the number features extracted for each batch, **T** is a $N \times a$ score matrix, *a* is the number of latent variables and **P** is a $m_f \times a$ matrix of loadings. The response variable for each batch is arranged in a vector **y** with dimension $N \times 1$. **c** is a vector of coefficients relating scores (**T**) and the response variable, and **E** and **f** are residuals arrays. The model can be estimated using Partial Least Squares (PLS) or Principal Component Regression (PCR). When no quality parameters are available, principal component analysis (PCA) is often utilized to describe the correlations in the feature space (eq. (2) only). The PCA model will extract principal components that explain the maximum amount of variability in **X**, modeling the correlation structure between features. In the context of feature-oriented methods, PCA models are mainly applied for offline process monitoring in order to detect process upsets and identify their root causes. When quality parameters are available, a partial least squares (PLS) modeling formalism is adopted so that the scores (**T** in eq. (2) and eq. (3)) contain information regarding both the feature space and the output variables **y**. PLS modeling can be used for both process monitoring and quality prediction. For batch process monitoring, two complementary statistics are commonly used, namely the T^2 statistic and the *Q* statistic. The T^2 statistic measures deviations from normal operating conditions (NOC) in the PCA subspace. The T^2 statistic can be computed for the *i*th batch according to eq. (4).

$$T_i^2 = \mathbf{t}_i \mathbf{\Lambda}^{-1} \mathbf{t}_i^T \tag{4}$$

Where $\mathbf{t}_i$ is the *i*th row of the matrix **T**, and $\mathbf{\Lambda}$ is an $a \times a$ diagonal covariance matrix containing the largest eigenvalues in descendant order as the entries of its main diagonal.

The *Q* statistic monitors the variability around the PCA subspace, by computing the orthogonal distance from a given observation to the subspace defined by the first *a* latent variables:

$$Q_i = \mathbf{e}_i^T \mathbf{e}_i \tag{5}$$

where $\mathbf{e}_i$ is the model residual for the *i*th batch $\left(\mathbf{e}_i = \mathbf{x}_i - \mathbf{t}_i \mathbf{P}^T\right)$. It is usually considered that a batch presenting a *Q* statistic above its control limit undergone a change that broke some aspect of the NOC correlation structure.

In the literature of BDA, four feature-oriented methods can be identified: landmark features, profile-driven features (PdF) (Rendall, Lu et al. 2017), statistics pattern analysis (SPA) (He and Wang 2011), and translation-invariant multiscale energy-based features

(TIME-F) (Rato, Blue et al. 2017). These methods can be employed batch-wise, when the complete batch trajectory is considered, or stage-wise, when features are extracted for each batch stage. Stage-wise feature extraction tends to be preferred, since it is more flexible, allowing one to better describe the batch evolution.

Landmark features are based on major batch milestones and capture the behavior of process variable(s) at specific time intervals. Typical examples of landmark features are the total amount of reactant fed to a reactor unit, the amount of heat transferred during a reaction stage or the amount of water used in a washing stage. These features are very specific and one of their main advantages is that the dimensionality of **X** is likely to be small when compared to other methods. On the other hand, the features obtained are highly problem-specific and require a significant amount of process knowledge. This limitation hinders broader and scalable applications of landmark features, but the flexibility of feature-oriented methods allows combining landmark features with other feature dictionaries that are more systematic and scalable (e.g., PdF, SPA).

As we move forward in the complexity scale (Figure 2.1), PdF occupies the next position. PdF was recently proposed by Rendall, Lu et al. (2017) and consists of a finite set of object-profiles, representing typically trajectories found in batch operations. These object-profiles are archetype profiles that, combined with data, produce an estimated trajectory for the process variable. Once the estimated trajectory is available, features specifically designed for that object-profile can be computed, constituting the **X** matrix. The distinguishing characteristics of this dictionary are that the set of object-profiles capture the backbone of non-stationary behavior in data in a parsimonious way, and the extracted features are specifically tailored for each object-profile, resulting in a small set of specific features (**X** will have dimensionality of $\approx 5J$). Additional object-profiles can be concatenated to the original set, extending the dictionary as more shapes are found in new applications. This dictionary will be described in detail in the next chapter (Section 3.1) as it is one of the contributions of this thesis.

SPA was proposed by He and Wang (2011) and is based on the statistical moments of process variables. Typical moments include the mean, variance, skewness, kurtosis, and other higher-order statistics that may be used to better characterize the process variables' profiles. Furthermore, the covariance between variables also constitutes a typical SPA feature, whereas the auto- and lagged-correlations are included in cases where they are expected to be relevant (Wang and He 2010). Therefore, applying SPA will result in a feature matrix **X** that has dimensionality $\approx J(J+7)/2$, which contains the mean, variance, skewness, kurtosis, and the covariance between all pairs of variables. By default, SPA extracts the same types of features for every variable in order to characterize their statistical distribution. However, some features (e.g., variance, kurtosis, and skewness) are invariant to data shuffling of batch samples and therefore completely ignore their time sequence, limiting the amount of dynamic information that is captured. Nevertheless, SPA is a simple and effective method and was successfully applied to monitor a continuous simulated system through the use of a window-based scheme (Wang and He 2010) and to the monitoring of an industrial semiconductor batch process (He and Wang 2011).

TIME-F was proposed by Rato, Blue et al. (2017) and characterizes the multiscale dynamics of each batch by applying the wavelet transform in order to decompose each process variable profile into its contributions at different scales, i.e., the measured variables are decomposed in J_{dec} scales. The wavelet coefficients at each scale represent the time-frequency content of a variable, and their energy is defined as the median of the squared sum of wavelet coefficients. The final set of features for a given batch corresponds to the energies for all scales of all variables, and the dimensionality of **X** is $J\left(J_{dec}+1\right)$, where J_{dec} is typically 5. As a drawback of this method, one can note that the same decomposition scale is used for all variables, which might not be the most adequate solution. Thus, a further reduction in the dimensionality of **X** can be achieved if variables were decomposed to an optimum scale, instead of using the same decomposition depth for all variables.

All the aforementioned feature-oriented methods offer great flexibility since they can be easily combined by concatenating feature matrices obtained from other dictionaries. In particular, combining landmark features with other dictionaries is a promising alternative since one takes advantage of both process knowledge and data-driven induction. Feature-oriented methods are also able to model the correlation between process variables. Although feature extraction is applied variable-wise, the resulting features characterize, in more or less detail, the behavior of the variable over a batch. It is therefore expected that highly correlated variables will also present highly correlated features; thus, multivariate methods can be applied in order to model multivariate relationships in the feature space. Nevertheless, localized time information is inevitably lost when process data is converted into features. For instance, important correlations at a certain localized period in time may not be well captured. In those scenarios, time-resolved methods may be more suitable, although they often entail higher implementation and modeling complexities.

As previously referred, feature-oriented methods have only been applied offline and, therefore, a first gap can be immediately identified in Figure 2.1, which concerns adapting them for online applications without requiring batch synchronization. Such

progress would enable these methods to be extended to a more general class of problems, still enjoying the low modeling and implementation complexities.
Analyzing again the complexity scale (Figure 2.1), one can also note that complexity increases as one moves from feature-oriented methods to linear time-resolved methods, and then to non-linear time-resolved methods. This transition helps to identify a second gap in the complexity scale: the lack of feature-oriented methods combined with non-linear modeling methods. The aforementioned dictionaries (PdF, SPA, and TIME-F) were combined with linear methods such as PCA and PLS. However, it would be worth assessing whether non-linear relationships exist and if they can be effectively exploited using non-linear methods applied to the feature matrix **X**.

2.3 Linear time-resolved methods

Time-resolved methods are a class of BDA approaches that preserve the time information and incorporate it in the modeling step. In contrast to feature-oriented methods, where the analysis conducted is either stage-wise or batch-wise (i.e. a feature describes the overall evolution of a variable over a stage or an entire batch, and its values are relative to the complete duration of these periods), time-resolved methods are based on a fine time grid. In other words, they use a time grid with fine granularity (high resolution), in contrast to feature-oriented methods, whose features present high granularity or very low time resolution. Therefore, time-resolved methods have higher modeling complexity and present the flexibility to capture more subtle patterns localized in specific periods of the batch. When applied to process monitoring, they also provide more accurate information about the onset of disturbances. This higher complexity and flexibility is achieved by a considerably higher number of modeling degrees of freedom that need to be estimated from data. Two classes of linear time-resolved approaches are discussed in this section: multi-resolution and full-resolution methods. Full-resolution methods utilize the native time resolution at which process measurements are collected (Figure 2.2.a), whereas multiresolution methods (or multi-granularity methods, Figure 2.2.b) may flexibly adopt coarser time grids, if advantageous for the purpose of the analysis – the new coarser values result from the time aggregation of the high resolution raw data, over the new grid with higher granularity. The granularity can be quantified by the aggregating time-support, i.e., the period during which raw data is accumulated before giving rise to a low resolution value. As an example, variable x_J in Figure 2.2.b has a time support of 4 and each sample may contain the mean from measurements within its time support. Multi-resolution methods have the advantage of implicitly handling multiresolution/multi-granularity datasets. Moreover, they can also be used to find out the optimal resolution for analyzing full-resolution datasets. Before presenting these methods in more detail, synchronization of batch trajectories is first considered since it is an integral part of the modeling effort and a requirement of all time-resolved methods.

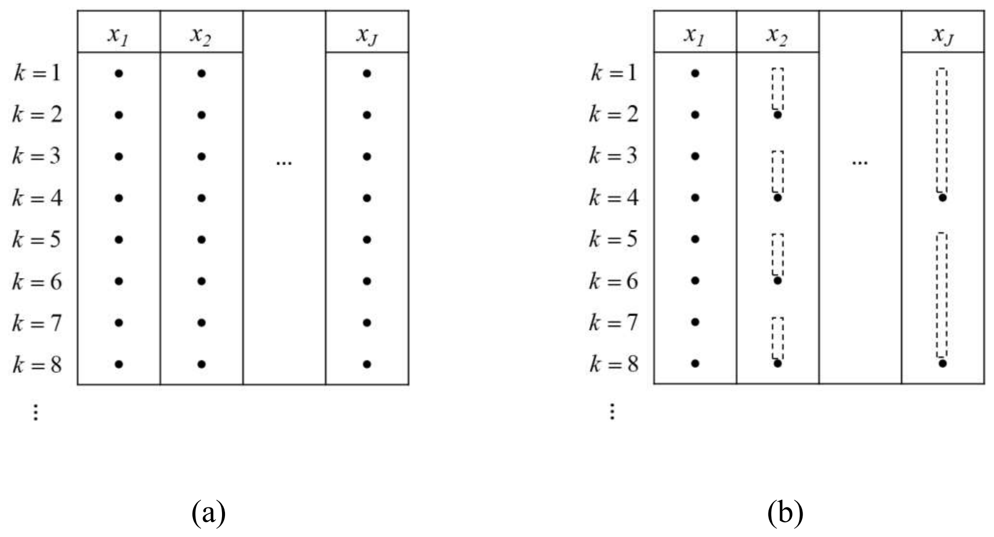


Figure 2.2. Time grid of full-resolution methods (a) where all variables are analyzed at the finest granularity. Multiresolution or multi-granularity methods (b) may use the default resolution or aggregate multiple measurements of a variable into coarser versions of it. Each dot represents the time at which a value is saved, which is then used for modeling. At lower resolutions, the values are only saved after the aggregation periods end.

2.3.1 Batch synchronization

Synchronizing batch trajectories is a preliminary step employed in BDA time-resolved methods, which ensures that different batches have the same duration and the key events happening throughout the batch are properly aligned. The effect of synchronization is depicted in Figure 2.3. Since different batches typically have different durations, the length of the measurements obtained from a process variable measured over two batches

(Figure 2.3.a) is not constant. By applying synchronization, all batches will present the same number of samples and the major batch events become synchronized (Figure 2.3.b). When considering raw measurements obtained for a set of batches $\left(\underline{\mathbf{X}}^{raw}\right)$, the length of the time grid (k) is dependent on the batch duration (Figure 2.3.c). Thus, synchronization outputs a 3-way array $\underline{\mathbf{X}}$ where all batches have the same duration K (Figure 2.3.d).

Two commonly used approaches for synchronizing batch trajectories are the indicator variable method (IV) (Nomikos and MacGregor 1995, Kourti, Lee et al. 1996) and dynamic time warping (DTW) (Kassidas, MacGregor et al. 1998). However, other methods and variants are available, such as correlation optimized warping (COW) (Fransson and Folestad 2006), multisynchro (González-Martínez, De Noord et al. 2014), and hybrid derivative dynamic time warping (Gins, Van den Kerkhof et al. 2012). IV was first proposed by Nomikos and Macgregor (Nomikos and MacGregor 1995) and relies on the availability of a monotonically increasing or decreasing variable that correlates with batch maturity. The IV indexing variable must also present the same starting and end values for all batches and show low levels of noise. In order to synchronize a new batch, process variables are plotted against the IV and interpolated at fixed regular or irregular increments of the IV. Special care must be taken in order to avoid destroying the correlation structure in the data, which can occur by an excessive number of interpolations or by averaging too many data points that may occur between increments of the IV (Kourti 2003). Good results have been reported using the IV approach (García-Muñoz, Polizzi et al. 2011), both in terms of their ability to synchronize variables, as well as improved process understanding resulting from a more detailed analysis of synchronized data. When an IV is not available, DTW is an alternative common solution. DTW is a more general synchronization technique, originally developed in the speech recognition community. It was adapted to BDA by Kassidas, MacGregor et al. (1998), who proposed an iterative approach to synchronize batches. In brief terms, DTW builds a time grid between a reference batch and the new batch to be aligned and optimizes the warping of the time domain while considering several constraints that stabilize and guide the synchronization process (endpoint, local, and global constraints). DTW searches for a path that minimizes the distance between the reference batch and the batch to be aligned, which is achieved by translating, expanding, and contracting segments of the batch to be aligned. Further improvements to DTW have been proposed to speed up its online implementation (González-Martínez, Ferrer et al. 2011) and to apply the warping information for monitoring purposes (González-Martínez, Westerhuis et al. 2013). Other improvements include derivative DTW (DDTW) (Keogh and Pazzani 2001) and a robust version of DDTW (Zhang, Lu et al. 2013) in order to overcome the singularity problem, which occurs when a point in a reference trajectory is mapped to multiple points in the trajectory to be aligned or vice-versa.

Synchronization of batch processes constitutes an integral modeling step that significantly affects performance in time-resolved BDA methods (Gonzalez-Martinez, Vitale et al. 2014). For instance, it has been observed that batch synchronization influences the detection strength (Rato, Rendall et al. 2016) and detection speed (Rato, Rendall et al. 2018) of monitoring methods. In other words, performance in signaling a given fault is dependent on the synchronization method utilized. Therefore, synchronization adds another layer of complexity to BDA, which can affect the modeling task in a positive or negative way. Nevertheless, the complexity associated with synchronization is mainly due to the implementation dimension, requiring trained personnel and process-knowledge to effectively align process data.

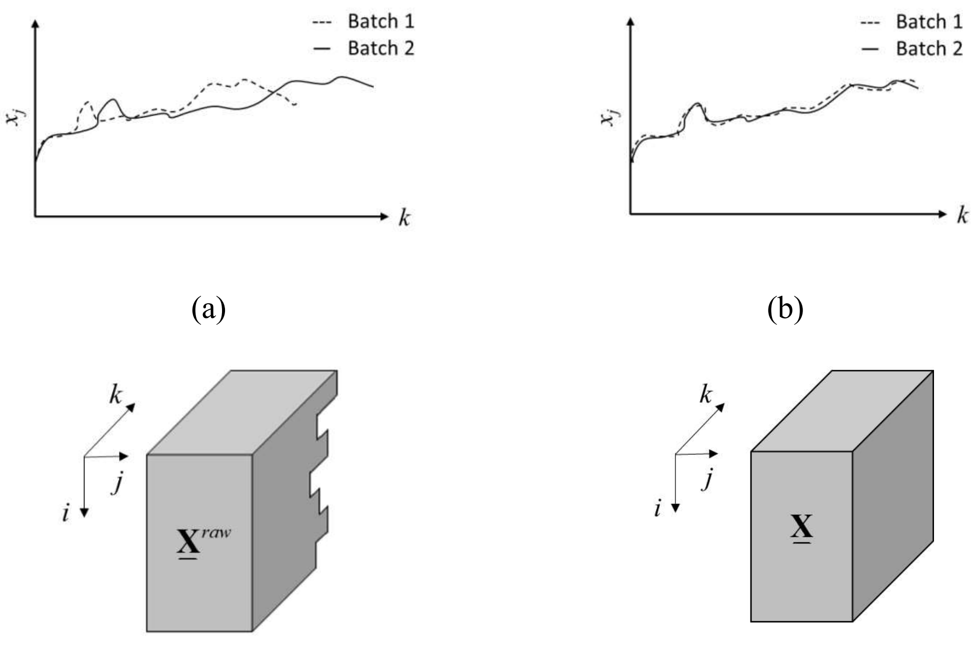


Figure 2.3. The effect of synchronizing a batch dataset: (a) the trajectory of process variables over two batches may have different durations. After synchronization, all the batches have the same duration (b). The general effect of synchronization converts batches with different durations (c) to a 3-way array where the batch duration is the same (d).

2.3.2 Full-resolution Methods

Full-resolution methods conduct BDA using the finest-grained time-grid, i.e., data is analyzed at the native resolution of the measurements. Methods in this category are commonly used in practice but require synchronization of batch trajectories. The simplest approach employs variable-wise unfolding, which reshapes $\underline{\mathbf{X}}$ to a matrix $\mathbf{X}^{VW}$ with dimensions $NK \times J$, as exemplified in Figure 2.4.a. The procedure proposed by Wold and co-authors (Wold, Kettaneh et al. 1998, Wold, Kettaneh-Wold et al. 2009) then builds a PLS model between $\mathbf{X}^{VW}$ and the local batch time (or a batch maturity index). Typically, auto-scaling is employed so that the scaled variables have zero mean and unit variance. Thus, the model parameters correspond to the J means and J standard deviations needed to auto-scale the data, and the additional degrees of freedom consumed by the bilinear model. The first latent variable of the PLS model contains variables that vary monotonically with time and the subsequent latent variables will capture more intricate relationships. In a second analysis stage, the scores are reshaped batch-wise and analysis can be conducted at the batch level. A third level of analysis may include initial conditions and batch quality variables ($\mathbf{y}$).

The variable-wise unfolding (VWU) solution does not favor the extraction of dynamic information, as the time mode is overlapped with the batch mode. In order to extract dynamic information, more complex methods are required that explicitly incorporate the auto-correlation and lagged cross-correlation in BDA. This can be achieved with batch dynamic PCA (BDPCA) or batch dynamic PLS (Chen and Liu 2002) methods proposed by Chen and Liu, which use lagged variables to model the auto- and lagged cross-correlations. For each batch *i*, an extended matrix $\tilde{\mathbf{X}}^{(i)}$ with L lags is constructed as follows:

$$\tilde{\mathbf{X}}^{(i)} = \left[\mathbf{X}(0) \quad \mathbf{X}(1) \quad \dots \quad \mathbf{X}(L)\right] \tag{6}$$

where $\mathbf{X}(l)$ is an $(K-L)\times J$ matrix containing shifted variables with l lags. For each batch, the sample covariance matrix is estimated for the BDPCA model according to eq. (7).

$$\mathbf{S}^{(i)} = \frac{\tilde{\mathbf{X}}^{(i)\mathrm{T}}\tilde{\mathbf{X}}^{(i)}}{K-L-1} \tag{7}$$

All covariance matrices for all batches are then combined in a pooled sample covariance, from which a PCA model can be obtained. By including lagged variables, the residual statistics tend to be less correlated, improving the ability to detect process upsets. The number of lags can be estimated by the method proposed by Ku, Storer et al. (1995).

Another method suitable for modeling dynamic behavior is auto-regressive PCA (ARPCA) (Choi, Morris et al. 2008). ARPCA uses lagged measurements to predict the current sample based on an auto-regressive model. PCA is then applied to the residuals between predicted and observed values for monitoring purposes, and the scores from the dynamic model are monitored as well. Similarly, DPCA with decorrelated residuals (DPCA-DR) (Rato and Reis 2013) was also applied to monitor the residuals obtained from an implicit one-step-ahead prediction operation. However, in DPCA-DR the prediction is made with resort to the existing DPCA model through a missing data technique, instead of using an additional AR time series model. In developing dynamic models, degrees of freedom are consumed by the model itself and additional model parameters are estimated for scaling the data (e.g., $J(L+1)$ means and $J(L+1)$ standard deviations for auto-scaling).

When the number of lags to be considered in the model increases, one eventually gets to the point where all the possible lags $(K-1)$ are incorporated. In this limiting case, each line of the extended matrix, eq. (6), is composed by all observations of all variables (i.e., a $1\times JK$ row vector). This corresponds to nothing more than the batch-wise unfolding (BWU) procedure proposed by Nomikos and MacGregor (Nomikos and MacGregor 1994, Nomikos and MacGregor 1995, Nomikos and MacGregor 1995), where $\underline{\mathbf{X}}$ is reshaped into a matrix $\mathbf{X}^{BW}$ with dimensions $N\times JK$ (see Figure 2.4.b). Afterwards, PCA or PLS are applied to $\mathbf{X}^{BW}$. These methods are usually known as multi-way PCA (MPCA) and multi-way PLS (MPLS), respectively. As more lags are included, the modeling complexity of these approaches is higher than that for batch dynamic methods. This allows covering situations where the dynamics and correlation structure change along the batch, which were beyond the reach of the dynamic methods presented above.

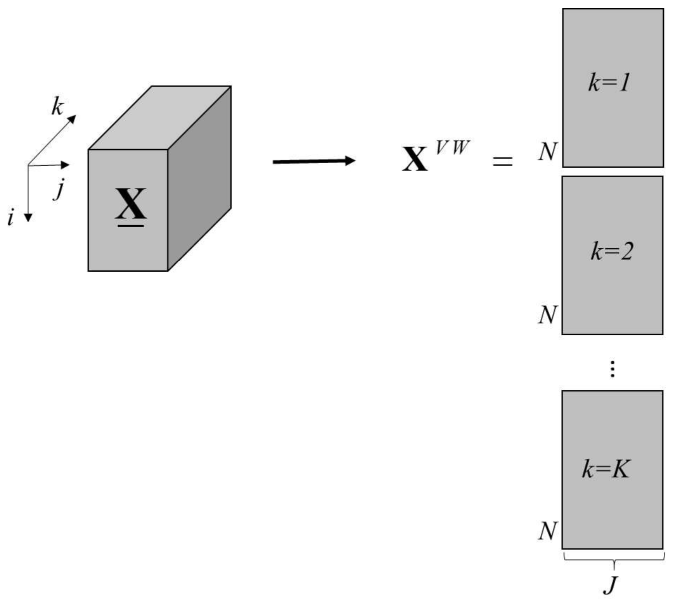
MPCA and MPLS have been the de-facto standards for BDA and were extensively explored in the literature (Martin, Morris et al. 1996, Lennox, Montague et al. 2001, Kourti 2002, García-Muñoz, Kourti et al. 2003, Kourti 2003, Ündey, Ertunç et al. 2003, Kourti 2005). By suitably scaling $\mathbf{X}^{BW}$ (e.g., auto-scaling), one can model deviations from the batch mean trajectory and allow all crossed and lagged-correlations to be estimated, greatly increasing the modeling flexibility. On the other hand, a large number of model parameters need to be estimated. These parameters include the means and standard deviations of all variables at every time point that are needed to auto-scale the data (*JK* means and *JK* standard deviations), as well as the degrees of freedom consumed in deriving the bilinear model (e.g., in the case of PLS models, the number of pseudo-degrees of freedom can be significantly larger than the number of latent variables selected (van der Voet 1999)). Another disadvantage is that, for online monitoring, future samples need to be estimated in order to compute MPCA scores for the entire batch. Common methods to predict future values include those known as “zero deviation” (assuming future measurements will not deviate from their mean values), “current deviation” (consider that future measurements present similar deviations to the mean trajectory as those observed for the last collected sample) and “projection methods” (that use missing data imputation techniques to estimate future values). In this context, the projection method (missing data technique) (García-Muñoz, Kourti et al. 2004) tend to exhibit better performance, as it is more accurate at predicting future unknown measurements and provides better estimates of the model scores, since this method benefits from the structural information contained in the MPCA model. In BDA, batch-wise unfolding is often preferred since models incorporate the non-stationary behavior of batches, namely through the subtraction of mean trajectory during pre-processing and the consideration of crossed-correlations between all variables at all times of the batch (a flexibility that may require special attention during the estimation stage). The latent variable model defined for the batch-wise unfolded matrix is similar to the one presented in eq. (2) and eq. (3), but instead of a feature matrix **X**, the model accounts for the variability in $\mathbf{X}^{BW}$. This carries very detailed time information since $\mathbf{X}^{BW}$ contains all variables at all sampling points. In other words, the analysis is conducted at the finest resolution possible.

BWU potentiates the generation of easily interpretable results, allowing the identification of important batch stages and process variables that are critical for quality. Results can also be examined at different levels. At a higher level, batches not conforming to normal operating conditions can be identified, whereas an intra-batch analysis can detect the precise time of occurrence of process faults by looking at appropriate monitoring statistics (the T^2 and instantaneous Q-statistic for each time point). Upon detecting process upsets, contribution plots can be applied to assist practitioners pinpointing variables deviating from NOC (Conlin, Martin et al. 2000, Westerhuis, Gurden et al. 2000) and to quantitatively assess how much each variable contribute to the observed statistics, signaling those with the highest contributions. A method named progressive PCA has also been proposed (Hong, Zhang et al. 2011) for fault identification. It works by repeatedly identifying variables with a high contribution, discarding them, and then building new PCA models. This process is repeated until no faulty observations are observed. The idea of this procedure is to track down the fault propagation path.

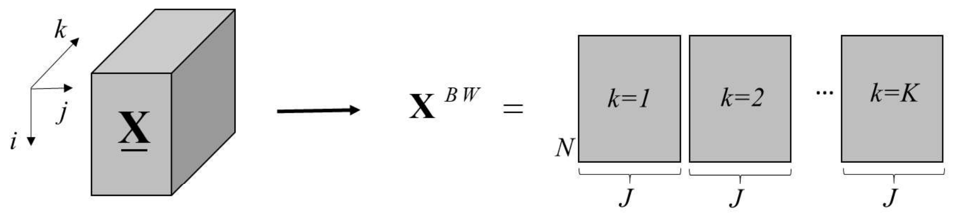
Another alternative for fault identification relies on using training datasets with known faulty conditions. In this case, fault identification is viewed as a classification problem, where a classifier is developed using process data and the corresponding fault labels.

Once the process is found to be operating outside NOC, the classifier is called to provide the type of fault causing such deviation. Good results have been reported with this approach (Wuyts, Gins et al. 2015) since it overcomes the smearing out problem associated with PCA; see also Van den Kerkhof, Vanlaer et al. (2013). However, it critically depends on the availability of a rich dataset with known fault types, whose signatures are consistent and well documented. This resource is very difficult to find in chemical processing industries, where even the same type of fault may impact the process differently depending on the time of fault occurrence, due to the existence of complex interactions between all the different sources of variability affecting the process (in this case, the same label corresponds to different processes signatures).

On the other hand, a model developed to correlate $\mathbf{X}^{BW}$ with quality variables $(\mathbf{y})$ can be applied to obtain online predictions of end-of-batch quality (Nomikos and MacGregor 1995) as well as investigating process upsets that are related to quality attributes. Confidence intervals were proposed to assess the uncertainty associated with the online predictions (Reiss, Wojsznis et al. 2010) and the model accuracy was further improved by finding sub-intervals that are more predictive, namely through multiway interval-PLS (MiPLS) (Stubbs, Zhang et al. 2013).



(a)



(b)

Figure 2.4. Unfolding the 3-way batch data matrix $(\underline{\mathbf{X}})$: (a) variable-wise, leads to $\mathbf{X}^{VW}$, with dimensions $NK \times J$; (b) batch-wise, leads to $\mathbf{X}^{BW}$, with dimensions, $N \times JK$.

Although the previous discussion addressed the relationships between dynamic methods and BWU modeling, there are methods in between, with varying degrees of complexity that are not depicted in Figure 2.1 for simplicity. An important group of methods that are not shown is the class of 3-way approaches, which explicitly address the tensorial nature of batch data by employing trilinear modeling. These methods do not require data unfolding before modeling, but still rely on synchronization. Two well-known representatives from this class are PARAFAC (Carroll and Chang 1970, Harshman 1970, Matero, Poutiainen et al. 2009) and Tucker3 (Tucker 1966). PARAFAC (Wise, Gallagher et al. 1999, Louwerse and Smilde 2000) decomposes the 3-way array $\underline{\mathbf{X}}$ into three modes (batch, variable, and time):

$$x_{i,\mathrm{j,k}} = \sum_{r=1}^{R} a_{i,r} b_{j,r} c_{k,r} + e_{i,\mathrm{j,k}} \tag{8}$$

where R is the number of retained components, $a_{i,r}$, $b_{j,r}$, and $c_{\mathrm{k},r}$ are the elements of loading matrices pertaining to each mode, and $e_{i,j,k}$ are the model residuals. The elements $a_{i,r}$ can be combined into a matrix **A** with dimensions $N \times R$, and similarly $b_{j,r}$, $c_{\mathrm{k},r}$, and $e_{i,j,k}$ can give rise to matrices **B**, **C**, and a 3-way array $\underline{\mathbf{E}}$. In PARAFAC, $\underline{\mathbf{E}}$ is conditioned to have small norm and **A**, **B**, and **C** are the elements of the trilinear model. One of PARAFAC's main advantage is the uniqueness of its solution (Bro 1997), meaning that the model cannot be rotated without decreasing its fitness ability, whereas other methods exhibit rotational ambiguity.

Tucker3 (Tucker 1966, Louwerse and Smilde 2000) is a 3-way method that offers more flexibility than PARAFAC. It decomposes $\underline{\mathbf{X}}$ into three orthogonal loading matrices (**A**, **B**, and **C**) and a core array $(\underline{\mathbf{G}})$, allowing each mode to have a different number of components:

$$x_{i,j,k} = \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} a_{i,p} b_{j,\mathrm{q}} c_{k,r} g_{p,q,r} + e_{i,\mathrm{j,k}} \quad (9)$$

where *P*, *Q*, and *R* are the number of components of the first, second, and third mode, respectively.

It has been shown that, for a similar number of components, an MPCA model can better fit a 3-way dataset than Tucker3, which in turn is more flexible than PARAFAC (Kiers 1991). This helps to position these methods in Figure 2.1. However, selecting the best one is highly dependent on the dataset and the goals pursued. PARAFAC and Tucker3 models present an additional complexity concerned with the need to select an appropriate scaling for the 3-way array. Whereas MPCA models typically employ auto-scaling as the default option to remove the non-stationary trend and ensure that every variable has comparable importance for model building, scaling the 3-way datasets for PARAFAC and Tucker3 is a significantly more complex task (Bro 1997). For instance, some authors (Westerhuis, Kourti et al. 1999, Wise, Gallagher et al. 1999) use auto-scaling to remove batch dynamics as in MPCA, whereas others have tested more intricate scaling options and showed that auto-scaling is not always appropriate (Gurden, Westerhuis et al. 2001). The consequence of not employing proper scaling is that some variables may be given too much importance or some relevant sources of structured variation can be missed (they are relegated to the error term), negatively impacting the performance of the method.

A taxonomy was proposed (Camacho, Pico et al. 2008) that classifies the aforementioned full-resolution methods as "single model" approaches due to the use of one data-driven model for the whole batch. The other class of approaches contains methods consisting of the application of multiple models across the batch duration – multi-model approaches. Time-evolving and adaptive methods (Ramaker, van Sprang et al. 2005) are examples of multi-model approaches. The former encompasses methods that consider all data available from the start of the batch until the current time point *k*, therefore, avoiding the need to predict future values. The less parsimonious case occurs when a model is developed for each time index *k* and one has effectively *K* models for BDA. Other variations are sometimes considered (Louwerse and Smilde 2000) where the batch duration is divided in, for instance, 10 time periods and one model is developed for each time period. This introduces delays for detecting process upsets or predicting quality parameters. On the other hand, adaptive methods consist of weighting differently the information arising from current measurements and that from the past. For instance, hierarchical PCA was adapted by Rännar, MacGregor et al. (1998) to online BDA and each $N \times J$ time slice corresponds to a block. The parameter controlling the rate at which the model adapts has to be tuned to the nature of the batch process, accounting for its

dynamic behavior. If the weight given to new information is very high, this approach will be equivalent to computing a separate model for each $N \times J$ time slice (this is called a local model (Ramaker, van Sprang et al. 2005)).
Besides PCA and PLS, other linear methods have been used for BDA after applying batch-wise unfolding. As an example, dynamic neighborhood preserving embedding (Hu and Yuan 2008) is a linear method that aims to preserve the local structure of the dataset, contrasting with PCA that preserves its global Euclidean structure. A major advantage of neighborhood preserving embedding is that it is more robust to outliers and may be more suitable to reveal the manifold structure of the data generating mechanism, provided that it exists. In a similar manner, multiway locality preserving projections (Hu and Yuan 2008) were applied for online process monitoring. Multiway independent component analysis (MICA) (Yoo, Lee et al. 2004) has been also used for monitoring batch processes and it is particularly advantageous when process variables exhibit non-Gaussian distributions. Combining both MICA and MPCA (Ge and Song 2007) was shown to be effective for extracting non-Gaussian and Gaussian components from batch data.

2.3.3 Multiresolution Methods

Multiresolution methods accommodate the existence of data available with different granularities or explore the introduction of granularity as an additional dimension to improve the performance of BDA methods. The second aspect is particularly interesting and has been largely overlooked in the literature. In fact, current full-resolution methods use data at their native resolution, tacitly assuming that such granularity is the most appropriate for analysis. However, the original resolution was established during the commissioning of the process' distributed control and data acquisition systems by third parties, long before the development of data-driven models take place, and with different goals in mind. Therefore, there is no solid basis to assume the native resolution to be the best for a given BDA task; on the contrary, it is very likely that the optimal configuration of variables' granularity is case dependent. As previously discussed, one of the disadvantages of time-resolved methods is the large number of model parameters obtained as a result of preserving the native resolution of process variables. To mitigate this drawback, multiresolution methods consider the possibility of selecting lower resolution representations of the variables to derive BDA models, consequently lowering the number of parameters. In this context, Gins, Van Impe et al. (2018) proposed a framework for multiresolution quality prediction (MRQP) that considers the optimal selection of coarser approximations of the original signals, in order to maximize predictive performance. More specifically, approximation matrices $\tilde{\mathbf{X}}_j^d$ are constructed for each variable *j* and approximation level *d*:

$$\tilde{\mathbf{X}}_j^d = \begin{bmatrix} \tilde{\mathbf{x}}_{1,j}^d \\ \vdots \\ \tilde{\mathbf{x}}_{N,j}^d \end{bmatrix} \tag{10}$$

The approximations of the signal $\left(\tilde{\mathbf{X}}_{i,j}^d\right)$ at different resolutions are obtained using the Haar wavelet transform. The Haar wavelet decomposes a signal into a set of approximation and detail coefficients. Whereas multiscale methods analyze both the detail and approximation coefficients, multiresolution approaches strictly make use of the approximation coefficients (Rato and Reis 2017). MRQP aims at predicting end-of-batch quality properties; therefore, all approximation matrices for all variables are concatenated and analyzed according to their relevance with respect to the target quality variable. In order to specify a suitable prediction resolution, sequential forward floating selection (SFFS) is applied to select variables at the proper resolution for model development. The selection procedure avoids different resolutions of the same variable to be selected. In other words, each variable only appears in the model at a given resolution or granularity, which can be different from other variables. Some variables may also not be selected at all, if they are found not to carry additional predictive power, at any resolution. In the original paper, 2-way PLS models were used and applied to simulated case studies and to an industrial dataset, illustrating the method's superiority (which was also established from a theoretical perspective). Nevertheless, the MRQP framework can be easily extended to other regression models, including 3-way methods, dynamic models, and others. A more recent proposal extends MRQP to multistage processes, offering more flexibility in how variables/resolution are selected (Rato and Reis 2018); in particular, the optimal selection may now depend upon the stage of the process, which is likely to be the case in many batch processes.

Considering that MRQP and its evolution are among the very few approaches available for multiresolution BDA, it is not surprising that they have not been extended to non-linear modeling. Multiresolution methods have the ability to tune the time granularity of analysis to the dynamic characteristics of a batch process, with a possible reduction in the number of parameters and an increase in model stability and accuracy. As an example, if a multiresolution analysis shows that it is sufficient to use minute data for a given task instead of data collected every second, this would imply that the number of columns in $\mathbf{X}^{BW}$ decreases by a factor of 60. When one considers non-linear modeling, the effect of decreasing the number of parameters is even more important, in order to avoid overfitting and improve method's stability. Developing non-linear multiresolution methods constitutes a third gap in BDA methods that future research should address in order to broaden their scope to a wider variety of situations.

2.4 Non-linear time-resolved methods

Non-linear time-resolved methods are the most complex methods for BDA. Besides having a large number of parameters and hyper-parameters, they contemplate non-linear transformations in order to model complex relationships. Non-linearity is often an inherent property of systems and phenomena taking place (e.g., the non-linear dependence of reaction rate from temperature, according to the well-known Arrhenius equation; inhibition kinetics in biological systems, etc.). However, in practice, most processes do not span an operation region large enough for the non-linearity to manifest itself in a clear way and sufficiently above the unstructured variability or noise level. Operation regimes are instead confined to smaller regions of operation due to quality and safety standards, where linear models often offer accurate and stable solutions to many BDA applications. However, in some instances, non-linear methods may be advantageous.

A class of non-linear methods commonly applied to BDA is composed of kernel methods (Cao, Liang et al. 2011). These methods project data into a higher dimensional space where the relationships between variables become linear. Instead of defining this non-linear relationship explicitly, kernel methods implicitly perform the projection by applying the kernel function to pairs of raw data points, which amounts to compute their inner products after non-linear transformation. This is the well-known "kernel trick". Typical kernel functions include the Gaussian kernel and the polynomial kernel. Kernel PCA was adopted by Lee, Yoo et al. (2004) for monitoring a continuous process and later on adapted to batch processes using batch-wise unfolding – a methodology called multiway kernel principal component analysis (Lee, Yoo et al. 2004). Similarly, dynamic kernel PCA (Choi and Lee 2004) has been modified to batch dynamic kernel PCA by Jia, Chu et al. (2010). More recently, new methods were proposed for improving the ability to identify the variables directly related to the process upset (Cho, Lee et al. 2005, Vitale, Noord et al. 2014, Vitale, de Noord et al. 2015). MICA was also combined with kernel methods (Zhang and Qin 2007, Tian, Zhang et al. 2009) for batch process monitoring of non-linear and non-Gaussian processes. Kernel PLS (Zhang, Teng et al. 2010, Wang, Wang et al. 2016), dynamic hierarchical kernel PLS (Zhang and Hu 2011, Zhang, Li et al. 2012), and kernel support vector regression (Desai, Badhe et al. 2006) have also been used for quality prediction and process monitoring in BDA.

Neural networks form another class of flexible non-linear approaches for BDA, which has been applied to both process monitoring and quality prediction. In (Dong and McAvoy 1996), the authors combine the principal curve algorithm to extract non-linear principal component scores, followed by a neural network to compress the dataset for monitoring purposes. Neural networks have been applied by Zhang, Morris et al. (1998) for quality prediction using bootstrap to generate an ensemble of neural networks whose combination finally conduct to an improvement of prediction performance.

Other linear and non-linear modeling frameworks have also been applied to BDA. For instance: support vector machines were used to model the non-linear relationship between process variables (Ge, Gao et al. 2011); bagging of support vector machines models (Ge and Song 2013) were adopted to build more robust and accurate models; support vector machines have also been combined with kernel independent component analysis (Zhang 2008) for process monitoring; hidden Markov models were used for process monitoring (Yu 2010) and compared with PCA-based approaches, as well as more complex approaches based on multi-hidden Markov model trees (Chen and Chen 2006); the combination of Markov models and dynamic MPCA was also contemplated (Chen and Jiang 2011).

As can be inferred from the previous discussion, a wide spectrum of non-linear methods is available for BDA. These methods require advanced training and are more challenging to implement by practitioners. They also have additional degrees of freedom on top of those existing for time-resolved linear methods. Therefore, they rank higher in the complexity scale and their use should be carefully considered taking into consideration this positioning and the added value they may bring to the analysis. This is not an easy task and researchers are challenged to provide guidelines on which ones to adopt. However, in spite of the difficulties associated with non-linear modeling, their potential utility has to be recognized with the recent advances leading to Industry 4.0. Non-linear methods tend to have very low bias and the challenge often comes from their inflated variance. The variance problem can be effectively tackled by collecting more data, which is precisely one of the pillars of Big Data. Even a process that only operates in a linear range can be successfully modeled by a non-linear model since the latter often has linear models as a particular case, and would eventually converge given that enough data are accumulated. Successful examples of the benefits of using large amounts of data for non-linear modeling can be found in the rapidly increasing literature of deep learning methods (LeCun, Bengio et al. 2015). In fact, it has been observed that whereas the performance of some linear and non-linear methods reaches a plateau after a certain amount of training data, deep learning has the capability of improving as larger datasets are gathered, surpassing other non-linear approaches in many applications.

2.5 Methodological Workflow for BDA

The availability of the numerous approaches presented in the previous sections suggests the need for a methodological workflow in order to navigate through the complexity scale. Such a workflow for BDA is presented in this last section; see also Figure 2.5. It is based on the Parsimony or Occam's razor principle and consists in starting from simple methodologies and moving to more complex approaches as long as the performance increments justifies the additional complexity.

The workflow starts with the problem definition step, where clear objectives are defined and the retrieval of datasets from historical databases is conducted. In terms of objectives, one should distinguish those that can be re-casted as a regression problem and those that involve a classification problem, in order to select suitable performance metrics or KPI (Key Performance Indicators). Classification metrics include the Area Under the receiver-operating Curve (Rato, Rendall et al. 2016) (*AUC*), True Positive Rate (*TRP*) and False Positive Rate (*FPR*), F-score among others. The performance in regression problems is typically assessed by the Root Mean Squared Error of Prediction (*RMSEP*). These measures of performance should be chosen according to the objectives of the problem at hand and in industrial processes, one often wants to constraint the number of false detection in order to avoid unnecessary control actions, whereas a missed detection may be not so critical because the fault usually persist long enough to be discovered by the monitoring system.

The next steps in the methodological framework are rather straightforward: if the available knowledge is enough to select an appropriate complexity level (see Figure 2.1), then the corresponding method is adopted and tested. If process knowledge is not enough or unavailable, the starting point should be the use of the PdF dictionary. The PdF dictionary is a suitable first choice since it contains a small number of features and it is easiest to implement. Thus, the models tend to be more parsimonious and may lead to similar or better results when compared to more complex approaches. After choosing the initial method complexity, one evolves one step in the complexity scale presented in Figure 2.1 and evaluates the improvement (or not) of the performance metric. This sequential procedure should stop once an acceptable performance is obtained or when the higher levels of complexity lead to no improvements in the results. In this scenario, one has adopted the least complex model that still achieves the desired accuracy or the one that maximizes it. In either way, one only makes use of as much complexity as needed to cope with the problem at hand, potentially avoiding time-consuming and highly technical solutions that sometimes do not offer proportional benefits.

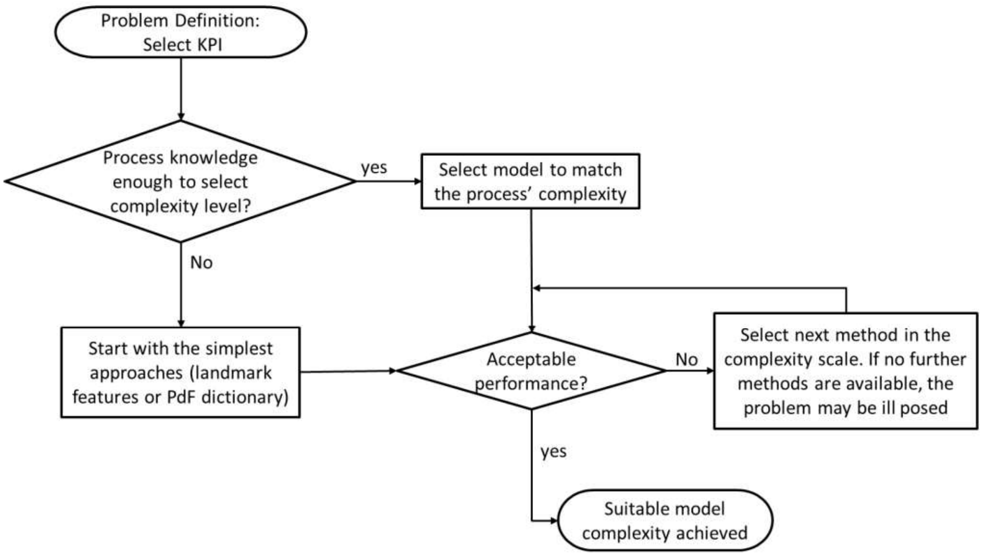

Problem Definition: Select KPI
Process knowledge enough to select complexity level?
yes
Select model to match the process' complexity
No
Start with the simplest approaches (landmark features or PdF dictionary)
Acceptable performance?
No
Select next method in the complexity scale. If no further methods are available, the problem may be ill posed
yes
Suitable model complexity achieved

Figure 2.5. Methodologic workflow for selecting a suitable approach for offline batch data analysis.

Chapter III
Feature-Oriented Methods for Batch Data Analysis

3 Feature-oriented Methods for Batch Data Analysis

In this chapter, a framework for extracting features from batch datasets is presented. The framework provides a unified setting for implementing several variants of feature-oriented methods proposed in the literature, including the new methodology based on process variables' profiles, named profile-driven features (PdF). It also integrates feature generation and feature analysis, in order to speed up the data exploration cycle, which is especially relevant for complex batch processes. FOBA 1.0 (feature-oriented batch analytics platform) is described in detail and applied to several case studies regarding different analysis goals: visualization, quality prediction, and end-of-batch process monitoring.

3.1 The Feature-Oriented Batch Analytics (FOBA) platform

In the scope of FOBA, each set of features (Landmark, PdF, SPA, etc.) correspond to a different "dictionary". A dictionary is a collection of objects whose properties lead to the computation of features for the variables profiles. The unifying and integrated platform for feature-oriented analysis of batch processes (FOBA) is depicted in Figure 3.1. As shown in this figure, the first stage corresponds to the generation of features from process data. For such, a dictionary must be selected. In the case portrayed, the selected dictionary is PdF (this is the default option; a combination of dictionaries is also possible, resulting in an inflated size of the features space). Following the application of the PdF dictionary, a set of features is obtained.

Feature generation can be implemented for the entire batch, or for each stage of the batch in a stage-wise analysis, as defined by the user. A stage-wise analysis is preferred since it provides features that better describe the localized patterns found in the batch evolution. Therefore, when *a priori* information related to batch stages is available, it should be employed to generate a collection of features per stage, which are concatenated afterwards to represent the behavior in all stages. Moreover, in this feature-generation stage, knowledge-based or other features that are expected to be important (e.g. the duration of a batch step) can also be specified and included in order to capture relevant process information that potentially increases data analysis performance in the next stage of FOBA. One should note that this ability to extend the feature space is not exclusive to PdF but can also be implemented with other feature-oriented methods (e.g. SPA, TIME).

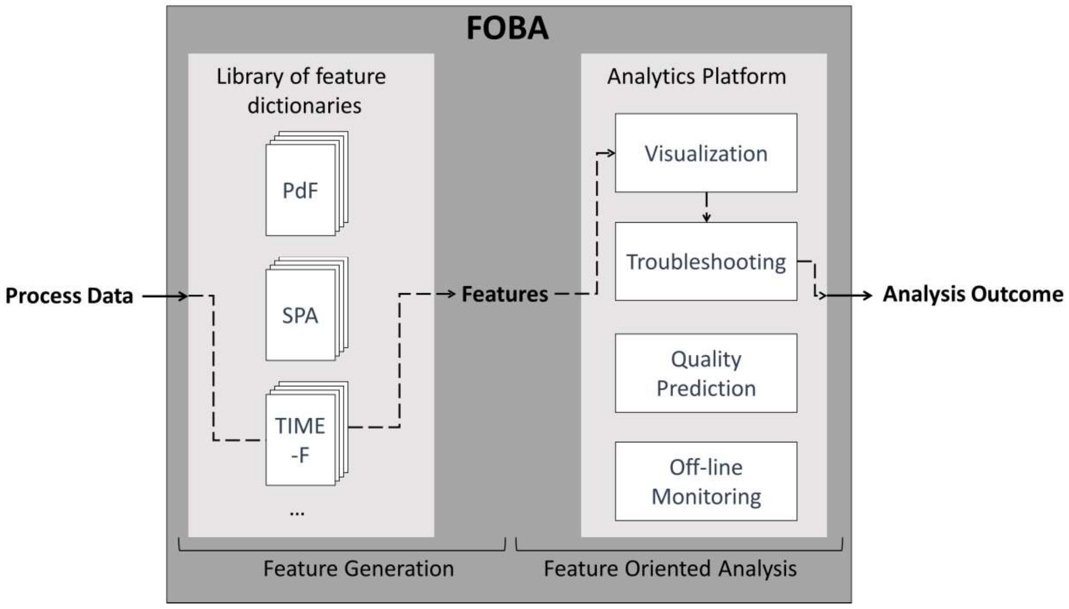


Figure 3.1. Schematic representation of the FOBA superstructure.

The entire set of features arising from all process variables (at all stages) will be analyzed in a second stage (Feature-Oriented Analysis). It is recommended to start with the visualization module, where a pre-selection and preliminary analysis of features is made, with resort to robust measures of association (Spearman correlation, mutual information) in order to remove noisy and irrelevant features, which in turn will improve the model interpretability and mitigate the risk of overfitting. A variety of graphical and descriptive analytical tools are applied in this module, in order to extract basic insights about the nature of variation along the batches and variables (e.g., stratified box-plots, analysis of scores from a PCA analysis; more information will be provided in Section 3.3, where FOBA is applied to several case studies).

Then, depending on the specific goal of the analysis, the workflow can proceed to one of the following modules: troubleshooting (root cause analysis), quality prediction and offline (or stage-wise) process monitoring, where specific tools are applied to conduct the required analysis.

3.2 Profile-driven Features (PdF)

In this section, the newly developed profile-drive features (PdF) dictionary is described in detail. PdF explicitly considers the specific non-stationary nature of the variables profiles in the computation of the features. Variable profiles that are widely different will be characterized by different sets of features: each set being more adequate to represent the variability exhibited by the respective type of profile. This is one of the differentiating aspects of this dictionary relatively to the other existing ones, namely SPA and TIME, where more general features are often computed for all variables, that may not fully account for their specific profile pattern. The PdF components are described in the following paragraphs (see also Figure 3.2).

Object-profiles. The dictionary is composed by a finite set of objects, $P = \{op_i\}_{i=1,\ldots,p}$. Each object corresponds to a given type of profile or pattern. A profile is a parameterized representation of a pattern that, upon training with process data, will lead to a realization of a curve. We call p_i the parametrized model structure associated with the object-profile op_i. Variables are assigned to the object-profile op_i if they exhibit a *similar* pattern (of course, the case of non-correspondence is also possible and actually rather common, as the dictionary has a wide coverage of possible profile patterns, not all of them necessarily appearing in every batch process); the assignment process will be addressed further ahead in this section. By "similar" it is meant that is possible to fit the profile structure to data obtaining, in the end, an estimated profile $\hat{p}_i$, $p_i \xrightarrow{data} \hat{p}_i$, with good fit. In other words, "similar" means that the profile is able to adequately describe the non-stationary behavior of the variable in question. Object-profiles considered in the dictionary tend to have very parsimonious model structures, with only a few parameters to be estimated from process data.

Estimation Engine. Each object-profile has an estimation engine. This estimation engine is a *method* (using object-oriented programming nomenclature) that takes data regarding a given variable under consideration and computes the estimated parameters of the profile assigned to such variable by maximizing model fitting (in the sense of minimizing the residual sums of squares). Thus, the estimation engine provides an estimated trajectory $(\hat{p}_i)$ for that variable, which will be used later on by the computation engine.

List of Profile Specific Features. Associated to each object-profile is a set of features that are pertinent to compute. These features are specific of the profile under consideration.

Features Computation Engine. This engine computes the values of the features for the object-profiles associated to all variables, using process data and the estimated profile provided by the estimation engine.

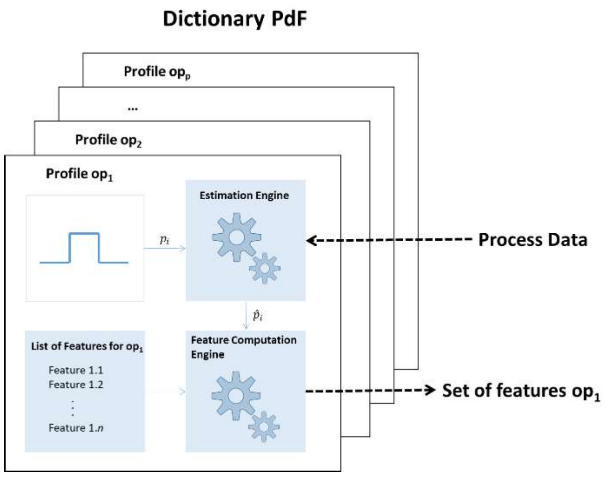


Figure 3.2. Schematic representation of the PdF dictionary and its main structural components.

Some examples of profile-objects are represented in Table 3.1.

The process of assigning variables to the characteristic profile (object) can be made in one of three possible ways, called "assignment modes" (AM):

– *User supervised (AM-US)*. The user provides the mapping between variables and the corresponding characteristic profile. This is the method adopted in all case studies since it can be done alongside a first exploratory stage for acquiring familiarity with the main variability patterns found in the dataset. However, this approach may be cumbersome or impractical in industrial processes containing a large number of variables. In those scenarios, the automated or semi-automated methods are preferred to facilitate the assignment task;

– *Automated (AM-AU)*. Profiles are automatically assigned using an algorithm based on the average quality of fit of each type of profile to the variables' data. For instance, measures such as R^2 or the sum of squared errors can be used to measure the similarity/dissimilarity between raw and estimated trajectories. However, one must take also into account the complexity of the object-profiles during automatic assignment. For instance, a linear object-profile will always fit equally well or better than a constant object-profile. Thus, the adjusted R^2 and other alternatives that penalize complexity might be more suitable (such as the AIC and BIC criteria). In other words, the mechanism for achieving an automatic assignment mode consists in using a measure of model fitting to guide the mapping between process variables and object-profiles. However, the AM-AU is still not fully developed and should be included in future work since further studies are required to identify and compare different measures of model fitting in the context of PdF. Thus, this assignment mode was not used in any of the case studies presented here because, as stated above, we value a first stage of data visualization, which can be easily combined with the AM-US option. BDA usually starts with data visualization and it is rather convenient to assign each process variable to one of the available object-profile, instead of utilizing the automated assignment mode;

– *Semi-automated (AM-SA)*. In this case, AM-AU is first conducted and the best candidates for each variable are preliminarily registered. The user then selects one among the suggested profiles for each variable. This alternative speeds up the assignment process relatively to AM-US, still allowing the user to have control over the process, which may also be opportune for some cases where variables profiles do not show a clear pattern.

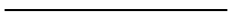
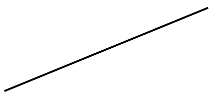
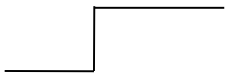
The proposed dictionary of profile-driven features presents the following positive characteristics:

– Avoids the need for preprocessing, namely trajectory mean shifting and scaling (which can be seen as an over-parametrized way of modeling the non-stationary behavior of individual variables);
– Does not require unfolding and complex batch alignment/synchronization – a task that is usually difficult or requires advanced training;
– Significantly reduces the dimension of the predictor space when compared to time-resolved alternatives;
– Retain variables profiles specificity in the computation of the features;
– Can be applied in batches with unequal lengths;
– Fast to implement;
– Easily expandable to include more features for each object-profile.

As disadvantages, we may refer the lack of time resolution in the analysis (it is the whole batch or stage that is being analyzed, and not its behavior at a given instant), which may lead to some information loss. In particular, fine changes in the local process correlation structure may be difficult to detect if they do not affect significantly the profile-driven features. Another disadvantage is the fact that the dictionary has always a finite set of features, which could limit its application to more specific processes. However, the dictionary is easily expandable with more object-profiles and features or combined with other dictionaries, as described in the next section, which mitigate the impact of this limitation.

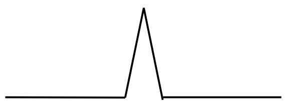
We would like to point out that, when PCA, PLS and other multivariate methods are applied to the feature space generated by PdF, it is also assumed that the distribution of features across different batches is *i.i.d.*. This assumption is common to most multivariate methods used in the analysis of batch process data.

Table 3.1. Examples of object-profiles available in the PdF dictionary.

Object-Profile		**Features**
Constant		Mean Variance [Residual statistics] Area
Linear		Slope Intercept SSR[*] Residual variance [Residual statistics] Area
Step		Step occurrence Mean before and after step Variance before and after step SSR[*] [Residual statistics] Area

[*]SSR is the sum of squared residuals between the estimated and measured profiles.
[Residual statistics] correspond to additional statistics that are not in the default set of features in the dictionary, but that could be included to improve performance in particular applications (e.g., high-order statistics to address non-Gaussian residuals).

Table 3.1 (cont.). Examples of object-profiles available in the PdF dictionary.

Object-Profile		**Features**
Pulse		Pulse Beginning Mean before, after and in the pulse Variance before, after and in the pulse SSR* [Residual statistics] Area
Impulse		Impulse beginning Maximum value of impulse Time of occurrence of maximum value Area

*SSR is the sum of squared residuals between the estimated and measured profiles.
[Residual statistics] correspond to additional statistics that are not in the default set of features in the dictionary, but that could be included to improve performance in particular applications (e.g., high-order statistics to address non-Gaussian residuals).

3.2.1 Ilustrative example of PdF

The workflow for the PdF dictionary consists of: (i) allocating variables to object-profiles $\left(x_j \overset{data/user}{\rightarrow} op_i\right)$; (ii) run the estimation engine and estimate the parameters of the model structure associated with the profiles $\left(p_i \overset{data}{\rightarrow} \hat{p}_i\right)$; and finally (iii) run the feature computation engine, from which the set of variables-specific features are obtained $\left(\hat{p}_i \overset{data}{\rightarrow} \{F_k\}\right)$. Consider, for instance, Figure 3.3.a which presents the totalized feed during 12 batches of a drying operation. The totalized feed corresponds to the total amount of material fed to the dryer at each time point during the batch (more details regarding this dataset are provided in the case study from Section 3.3.1). By visual inspection and adopting the user supervised assignment mode (AM-US), among the available object-profiles one may classify this variable as a linear object-profile. In this simple case, the estimated profile is obtained by computing the slope and the intercept which minimize the squared residuals (Figure 3.3.b presents an example of the estimated profile for one batch). The specific features characterizing the linear object-profile are the slope, intercept, area, SSR and the variance of the residuals, as defined in Table 3.1. These features are computed from the estimated profile and capture the essential characteristics of its evolution.

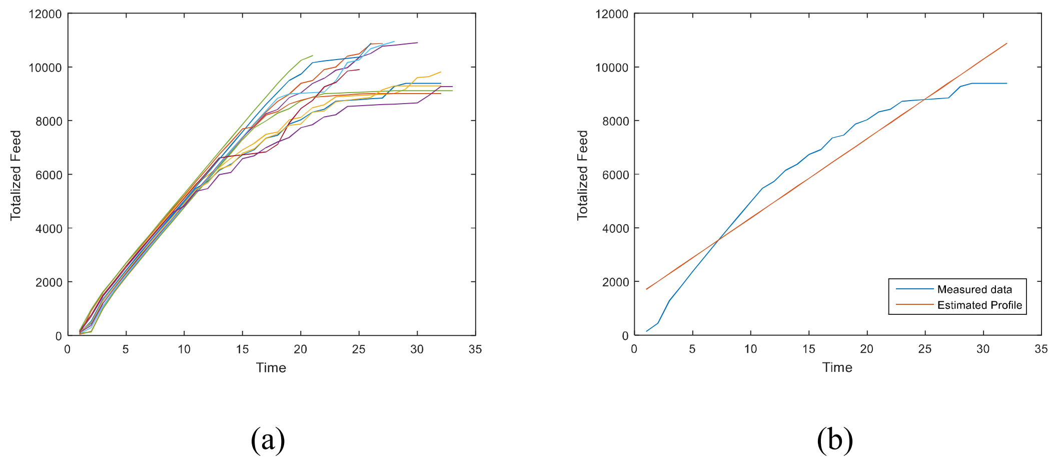


Figure 3.3. Example of the application of PdF. (a) The totalized feed is classified as a linear object-profile because it is the most similar object-profile among all those belonging to the current version of the PdF dictionary. (b) As an example, the estimated profile for the first batch is presented.

3.3 Case studies

In this section, the FOBA framework coupled with the PdF dictionary will be used to conduct a variety of tasks concerning offline batch process data analysis, with the purpose of demonstrating the potential advantages of adopting the proposed methodology and to illustrate their application in practice. In particular, the first task concerns data visualization of a real industrial crystallization process, where the operation of two dryers is compared. The second task concerns quality prediction for a widely used fed-batch simulated process for penicillin production, PENSIM (Birol, Ündey et al. 2002). Lastly, end-of-batch process monitoring is also conducted for the PENSIM simulator, allowing the identification of normal and abnormal batches. The proposed dictionary of features will be compared to benchmark approaches such as SPA and the more complex time-resolved two-way approaches (BWU with data alignment with resort to Dynamic Time Warping, DTW). The SPA features considered here are similar to those used for monitoring a semiconductor batch process (mean, variance, covariance, skewness, and kurtosis) in Ref. (He and Wang 2011); other features such as auto-correlation and lagged cross-correlation were not used. The use of lagged variables could potentially improve performance but only ad-hoc rules are available (Wang and He 2010) for selecting the proper lag. Thus, we tested SPA without these features.

3.3.1 Industrial Data Exploration

Data exploration or visualization is the recommended first task in any offline batch data analysis, where the main goal is to quickly extract useful insights of the nature of process variability. This task extensively explores graphical methods given their strong synergy with human visual and pattern recognition capabilities. Exploratory data analysis can be cumbersome and time consuming for batch data, given their intrinsic three-dimensional structure, which may get even more complex, due to the presence of multiple stages, misaligned process variables, batches with different lengths, etc. The FOBA framework avoids the alignment step because features are computed directly from the estimated object-profiles $(\hat{p}_i)$, providing a representation whose distribution can be used for assessing the batch-to-batch variability (some features also address the intra-batch variability, such as those involving the residuals of the estimated profiles). In this context, a case study is considered here with the aim of identifying differences in the operation of two industrial dryers working in parallel, in an industrial crystallization process (examples of the trajectory of the 20 process variables measured during the batch are provided in Figure 3.4 for one of the dryers). Existing process knowledge points to the existence of possible differences in the dryers given their production outcomes. This happens even though they share the same batch stages and recipe, which motivate a more detailed analysis of their operation, looking for possible sources for the different behavior. The proposed PdF dictionary, described in Section 2.1, was applied in a stage-wise fashion (i.e., the PdF features were computed for each stage in the crystallization process) and the trajectories of process variables were assigned manually (AM-US) to one of the object-profiles in the dictionary, as specified in Table 3.2.

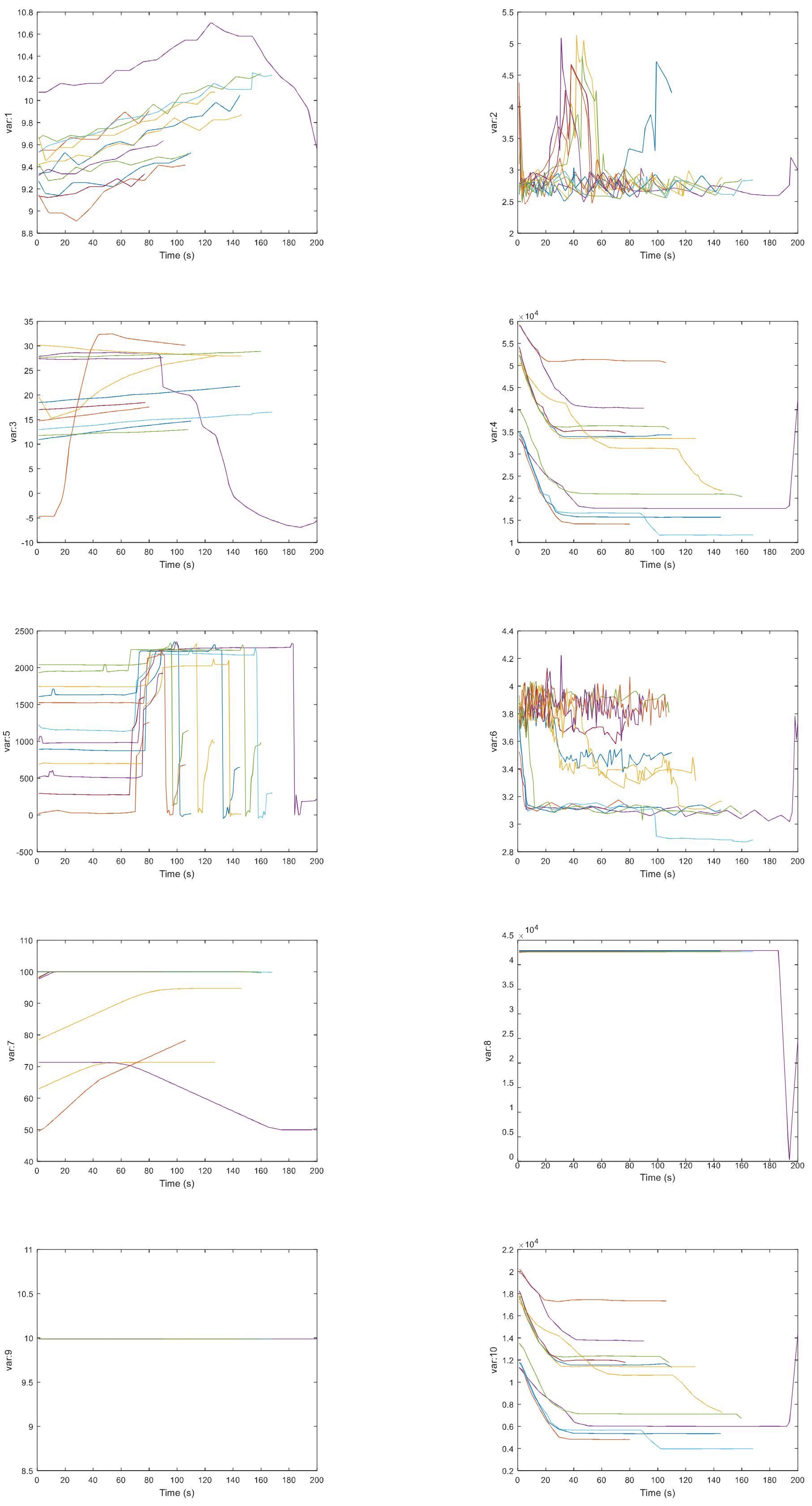


Figure 3.4. Example of the trajectory of 20 process variables measured during the drying operation.

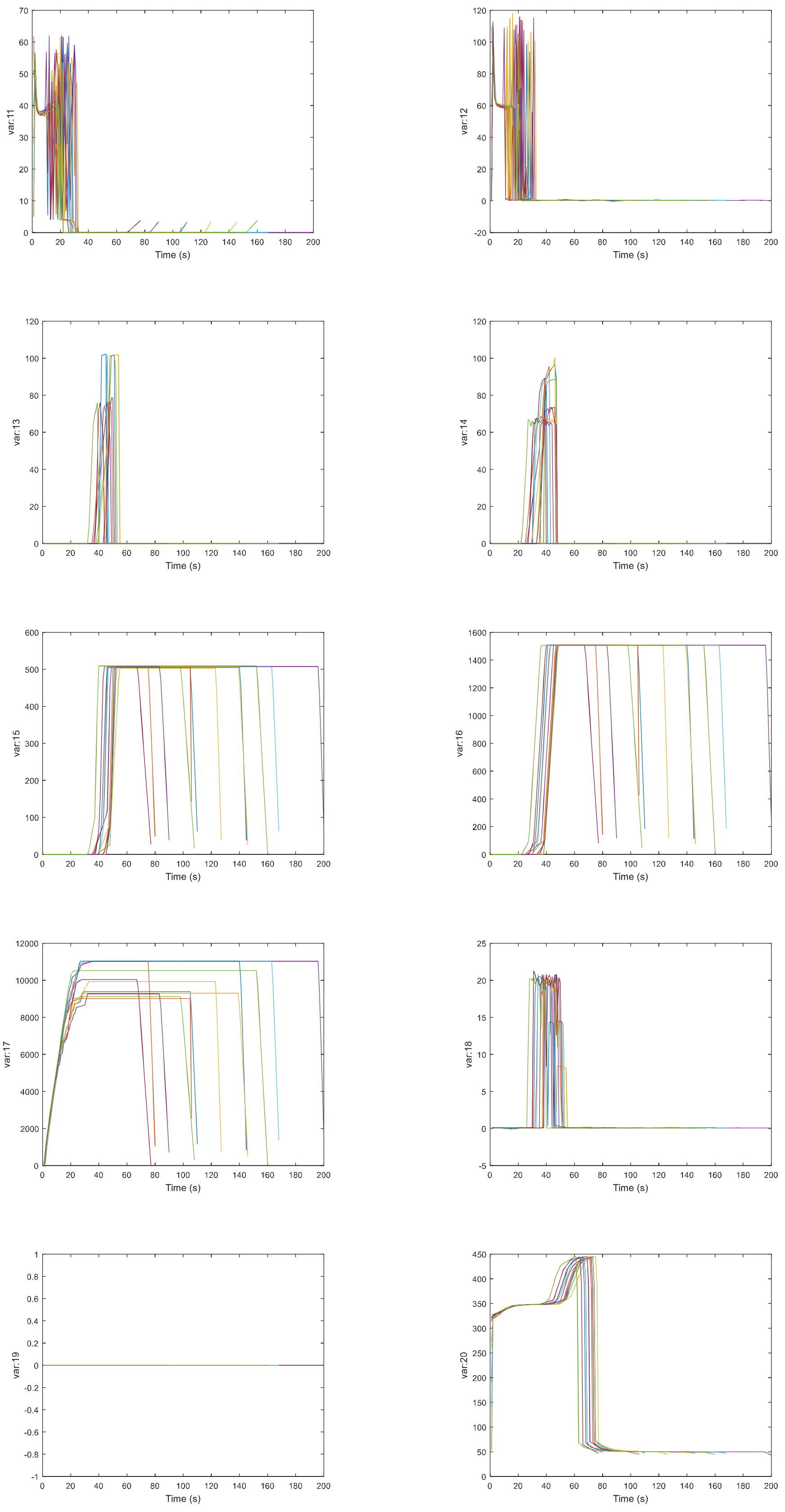


Figure 3.4(cont.). Example of the trajectory of 20 process variables measured during the drying operation.

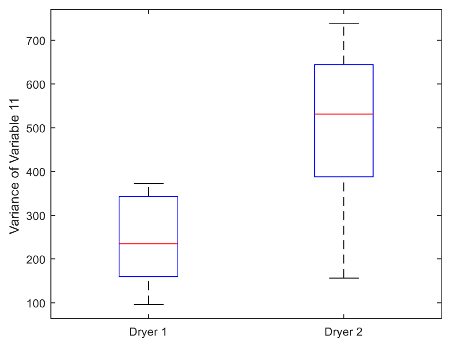
Table 3.2. User specified object-profiles for process variables at different batch stages of an industrial crystallization unit.

Variable Index	Batch Stage				
	Product feed	Spin	Wash 1	wash 2	Spin Dry
1	Constant	-	-	-	-
2	Constant	-	-	-	-
3	Constant	-	-	-	-
4	Linear	-	-	-	-
5	Constant	-	-	-	-
6	Constant	-	-	-	-
7	Linear	-	-	-	-
8	Constant	-	-	-	-
9	Constant	-	-	-	-
10	Linear	-	-	-	-
11	Constant	-	-	-	-
12	Constant	-	-	-	-
13	-	-	Linear	Linear	-
14	-	Linear	Constant	-	-
15	-	-	Linear	Linear	Constant
16	-	Linear	Linear	Constant	Constant
17	Linear	Constant	-	-	-
18	-	-	Constant	Constant	-
19	-	-	-	-	-
20	Pulse	Constant	Constant	Linear	Linear

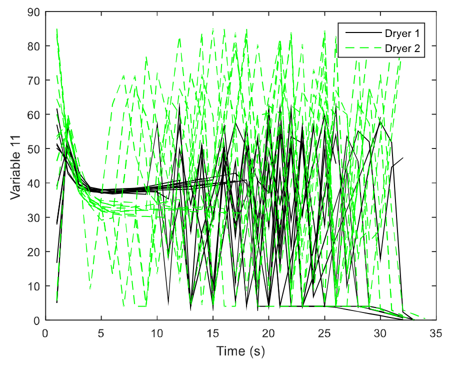
Table 3.2 shows that most variables analyzed were classified as simple object-profiles from the PdF dictionary. This stems from the fact that the majority of process variables had simple dynamic profiles, suggesting that the use of features can effectively capture most of the information regarding their evolution, although time-resolution is lost. Thus, despite the fact that the whole batch behavior for some variables shown in Figure 3.4 present rather intricate profiles which are not similar to the profiles available in the PdF dictionary, the patterns in a given batch stage are much simpler and can be easily classified into one the available object-profile. As stated above, the assignment into one of the object-profiles followed the manual assignment mode (AM-US), which was implemented by plotting each variable at each batch stage and then selecting the object-

profile that is more similar to the observed trajectory. Analyzing Table 3.2, one can also note that not all variables were analyzed at all batch stages because prior knowledge regarding the dryers' operation revealed that some variables are not relevant at some stages. Moreover, it is worth clarifying that the last batch stages were discarded and are not shown in Table 3.2 because they correspond to discharging operations that are not expected to impact the production quality. Thus, although Figure 3.4 presents the full batch evolution, the later part of the batches were not considered. The flexibility for discarding irrelevant variables at some batch stages (or even entire batch stages) is shared by all FOBA approaches – it can be performed in a supervised way (as in the present case) or in an unsupervised way (using feature selection approaches, e.g., based on parametric or non-parametric measures of association). The specification defined in Table 3.2 might seem cumbersome to conduct manually since one can potentially analyze 20 variables at 5 batch stages for a total of 100 profiles. However, the assignment task can be carried out rather quickly alongside a first step of data visualization by inspection of the process variables trajectories. After the assignment task, the estimation engine estimates the profile parameters of the selected object-profile using a least squares approach and the computation engine generates the features associated with the object-profile, thus completing the PdF workflow. As a practical example, variable #20 from the product feed step was assigned to the pulse object-profile (see Table 1 for the pulse object-profile), due to their intrinsic similarity. In order to obtain the estimated trajectory, the product feed step was split into three contiguous and non-overlapping regions with minimum sum of variances. Once the regions were identified, the estimated trajectory consists of the mean value for each region, which are then used to compute the features associated to the pulse object-profile.

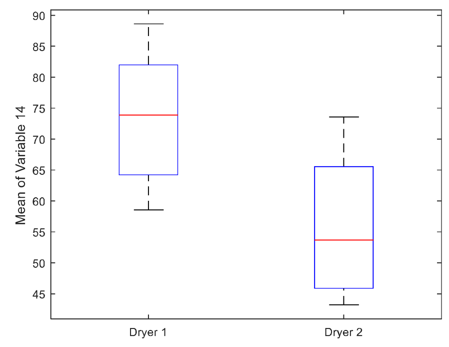
Returning to the problem of identifying differences between the dryers' operation, a dataset composed of 12 batches was available for each dryer and the assignment shown in Table 3.2 resulted in a total of 115 features for each batch. Thus, one has now available 2 matrices, $\mathbf{X}_1$ for dryer 1 and $\mathbf{X}_2$ for dryer 2, with dimensions 12×115 for each dryer. In order to find differences in the dryers' operation, a comparison of the distribution of features from both dryers was conducted: each column of $\mathbf{X}_1$ was compared to the corresponding column in $\mathbf{X}_2$ and a *t*-test (at a 5% significance level) was used to identify features whose distribution was statistically different across dryers. From the 115 features, 21 were found to be statistically different across dryers. Some of the identified features and process variables are presented in Figure 3.5.



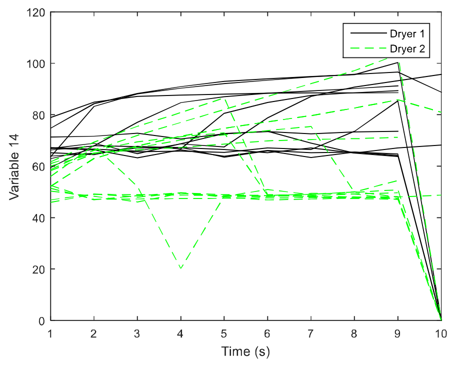
(a)



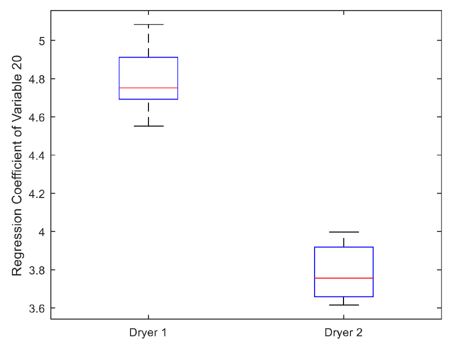
(b)



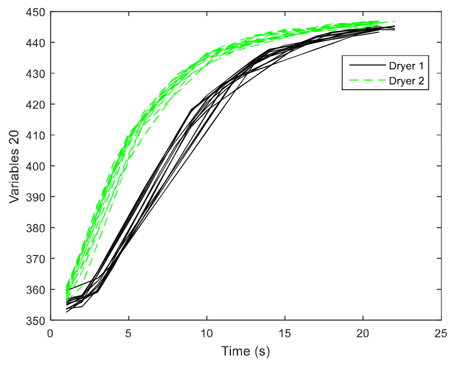
(c)



(d)



(e)



(f)

Figure 3.5. Examples of identified differences between dryer 1 and dryer 2 obtained with the PdF dictionary of features: (a) the variance of variable 11 at the product feed stage and (b) the plot of its evolution; (c) the average value of variable 14 at water wash stage and (d) the plot of its evolution; (e) the regression coefficient or slope for variable 20 at the spin dry stage and (f) the plot of its evolution.

Figure 3.5 shows that the identified variables have rather distinct trajectories and the PdF dictionary is able to effectively identify features with different distributions across dryers. Moreover, more than one feature can be identified for the same variable, highlighting multiple differences regarding their evolution. For instance, Figure 3.5.a shows that the variance of variable 11 at the product feed stage is higher for dryer 2 but another identified feature (not shown in Figure 3.5) was the average value, which is also higher for dryer 2. The average value and the variance cover independent aspects of the evolution of variable 11 and the interpretation of these features is straightforward since they are directly related to the object-profile assigned to variable 11. Variable 11 is the opening percentage of the feed valve and the fact that these differences were spotted, correlated to differences in the quality of the batch. Figure 3.5.c also shows that the average value of variable 14 is higher at the water wash stage, which is confirmed by looking at the variables' trajectory (Figure 3.5.d). Variable 14 is the opening percentage of the control valve for the recycling water flow and suggest that, on average, dryer 1 is using more recycled water than dryer 2. Lastly, Figure 3.5.e suggest that the slope of variable 20 at the spin dry stage is statistically different across dryers, a fact that is again confirmed by observing the trajectory (Figure 3.5.f). Variable 20 is the rotational speed of the dryers' and a clear difference is observed between dryer 1 and 2.
The different variables' behaviors quickly identified during the data visualization stage based on PdF features and FOBA, constitute useful information about process variation that can be used by process experts to interpret and raise conjectures about the operation and production quality of the two driers.

3.3.2 Quality prediction

Quality prediction is another important task in batch data analysis where the aim is predicting quality parameters based on the variables process trajectories. For this task, the PENSIM (Birol, Ündey et al. 2002, Van Impe and Gins 2015) simulator was adopted in order to test the potential of using the proposed PdF dictionary of features in this application context. The PENSIM simulator has been widely used in the literature as a testing environment for assessing and comparing methodologies for batch analysis. It consists of a detailed model of a fed-batch reactor for penicillin production. The simulator includes typical characteristics found in practice, namely noise and other sources of natural process variability, non-stationarity and PID controllers to regulate pH and the reactor temperature. The natural parameter for characterizing batch quality is the penicillin concentration at the end of the batch and in order to predict it, three sets of predictors were considered: features obtained from the PdF dictionary with manual assignment of object-profiles (AM-US), SPA features (namely, the mean, variance, covariance, skewness, and kurtosis) and time-resolved BWU process data, properly aligned with DTW. The relationship between predictors and penicillin concentration is modeled by PLS and a total of 70 batches were simulated for analysis. From the 70 batches, 50 were randomly selected for model training while the remaining 20 batches were used as a test set. Monte Carlo iterations of 5-fold cross-validation were used to determine the optimal number of latent variables (LV) and the median coefficient of determination of cross-validation (R^2_{cv}) is presented in Table 3.3 for the training set. Table 3.3 also presents the coefficient of determination for predictions on the test set (R^2_{test}).

Table 3.3. Prediction performances obtained with the PdF dictionary and benchmark methods using PLS as the regression method.

Dictionary	R^2_{cv}	R^2_{test}
PdF	0.82 (2 LV)	0.90
SPA	0.75 (2 LV)	0.82
BWU	0.85 (1 LV)	0.89

The prediction performances presented in Table 3.3 point to rather interesting results. They show that the PdF and BWU dictionaries were the top performing approaches under testing conditions, followed by SPA. The best performance obtained with PdF dictionary can be explained in terms of its ability to parsimoniously matching model and system complexity since, in this case study, PdF was able to compress the entire batch trajectory in a small number of features that preserved information regarding the penicillin concentration. Although the PLS model with PdF features uses 2 latent variables compared to 1 latent variable with BWU data, the reduction in the dimension of the predictor matrix is considerable: when PdF is adopted, the predictor matrix contains 67 columns/features describing each batch, whereas BWU data has a very wide predictor matrix with 19216 columns (16 variables and 1201 time points). Thus, in terms of the complexity scale, there is no clear advantage in this case to move to a higher complexity method (BWU), since no clear improvement in prediction performance will be obtained.

3.3.3 End-of-Batch Process Monitoring

In this section, we illustrate the use of the proposed PdF dictionary for end-of-batch process monitoring. Typical industrial databases contain hundreds or thousands of measurements made during batch operations and this information can in turn be used for quality assessment and improvement. The exploitation of such datasets provides knowledge regarding normal operating conditions, characterizing the typical measurement levels and correlation structures. Furthermore, the identification of abnormal batches can be conducted and the possible source of faults pinpointed. For this task, the PENSIM (Birol, Ündey et al. 2002, Van Impe and Gins 2015) simulator is again considered with the aim of identifying and characterizing abnormal batches. The set of 70 NOC batches described in section 3.2 were used to build a PCA model and additional 80 faulty batches were simulated and monitored. In the simulation of the faulty batches, 20 batches are affected by a drop in the aeration rate, 20 are affected by a change in the feed temperature, 20 contain a drift in the reactor temperature sensor while the remaining 20 batches are contaminated. The number of principal components was chosen by analyzing the explained variability and 3 principal components were selected according to the "broken-stick" criteria, explaining 72% of the features' total variability. Note that, similarly to the quality prediction case study (Section 3.2), the user specified assignment mode was used (AM-US) to map each variable to one of the object-profiles in Table 3.1. Furthermore, batch-wise features were computed since a good agreement between the batch profiles and the object-profiles was observed. The NOC and faulty batches were monitored using T^2 and Q statistics and the results are presented in Figure 3.6.a and Figure 3.6.b, where it can be seen that the PdF dictionary is able to effectively identify abnormal batches since the T^2 and/or Q statistics are clearly above their theoretical 95% upper control limits for the faulty batches. As benchmarks, Figure 3.6.c and Figure 3.6.d present the monitoring results obtained with the SPA dictionary (using as features the mean, variance, covariance, skewness, and kurtosis) while Figure 3.6.e and Figure 3.6.f presents results obtained with BWU data aligned with DTW. Although both benchmark methods are also able to detect most faults, a few of the monitoring statistics for fault 4 (batch indexes 131-150) are more or less at the level of the NOC data, which means that in a real monitoring scenario, these faults will be difficult to detect by these benchmark methods. BWU uses a very wide matrix with a high potential for overfitting due to the high number of model parameters and this fact is probably deteriorating its ability to detect fault 4. When comparing the ability of PdF and SPA to detect fault 4, one can observe a small advantage obtained by adopting the PdF dictionary, which results from the use of specific and targeted information about the time-varying profiles that captures the relevant dynamic patterns in the evolution of process variables in a more effective way. This example shows that methods belonging to the low spectrum of the complexity scale (PdF and SPA) are suitable for conducting off-line process monitoring and can have superior performance when compared to more complex approaches. The methodological differences between PdF and SPA suggest that their optimality is dependent on the case study under analysis, which is a typical situation for data-driven approaches. For instance, when monitoring a semiconductor batch process (He and Wang 2011), SPA conducted to better results compared to a k-nearest neighbor classifier and BWU, because the variability around the constant profiles of these process variables can be very well described by the features of this method. A similar result is expected for PdF using a constant profile. The theoretical justification for this is the following: apart from the mean, SPA uses mean-corrected statistics, such as the variance, covariance, skewness, and kurtosis. Therefore, the results of SPA should match those of PdF with a "constant" profile assignment, as long as the same statistics are used for characterizing the residuals around the level of this constant profile (which is also estimated by the mean). However, by default PdF proposes the use of a reduced set of statistics for characterizing the residuals, in order to avoid the generation of large amounts of features, whereas SPA makes also use of higher-order moments (e.g., skewness and kurtosis), which can be advantageous in certain applications, such as in the semiconductor industry.

Besides fault detection, another critical aspect of end-of-batch process monitoring is the ability to identify the source of the fault so that process engineers may obtain insights into its root causes. In order to illustrate the ability of the PdF dictionary to identify the faults' root causes, Figure 3.7 presents the average contributions of each feature to the Q statistic for the first 2 faults. The average is taken over all faulty batches affected by the same fault. As can be seen in Figure 3.7, the Q statistic is very specific since only 2 features are highlighted for each fault. In fault 1 (Figure 3.7.a), the process was disturbed by a drop in the aeration rate and upon inspection, features 50 and 51 correspond, respectively, to the mean and variance of the aeration rate. In fault 2 (Figure 3.7.b), the process was perturbed by a change in the feed temperature and features 47 and 48 correspond, respectively, to the mean and variance of the temperature of the feed flow. Similar results were obtained for faults 3 and 4 but are not presented here. The specificity obtained with the PdF dictionary is a useful characteristic since it clearly pinpoints the source of the fault, avoiding ambiguity in fault identification.

The results obtained with FOBA and, in particular, with the PdF and SPA dictionaries show that even a model with a small number of parameters can be suitable for fault detection and identification: the matrix obtained with the PdF dictionary contains 67 features for each batch while SPA had 153 features. These low number of features can be contrasted with the much larger number obtained with a BWU procedure (19216 columns, see Section 3.2), where the potential for overfitting is very high because all auto- and cross-correlations are described. The development of PdF- and SPA-based models required low user input, effort, and technical specialization since the features are easily computed from raw data and the complex synchronization step can be avoided, which in practice means a smaller barrier for model development. Thus, as long as the results obtained fit the purpose of analysis, one does not necessarily need to move to higher complexity methods, as the gains in doing so are not guaranteed *a priori*. Therefore, it is our opinion that more complex methods should not be considered as default approaches but their use should be justified by the improvements obtained over the simpler ones. In practice, process knowledge may be used to identify a suitable approach and a high complexity method may be the correct starting point if the process is known to contain many and complex auto- and cross-correlation features. However, for offline quality improvement activities, a more complex model may be difficult to estimate and interpret and although simpler models may found limits in capturing all relevant process characteristics, their simplicity can be an advantage for extracting the fundamental trends and regularities in the dataset.

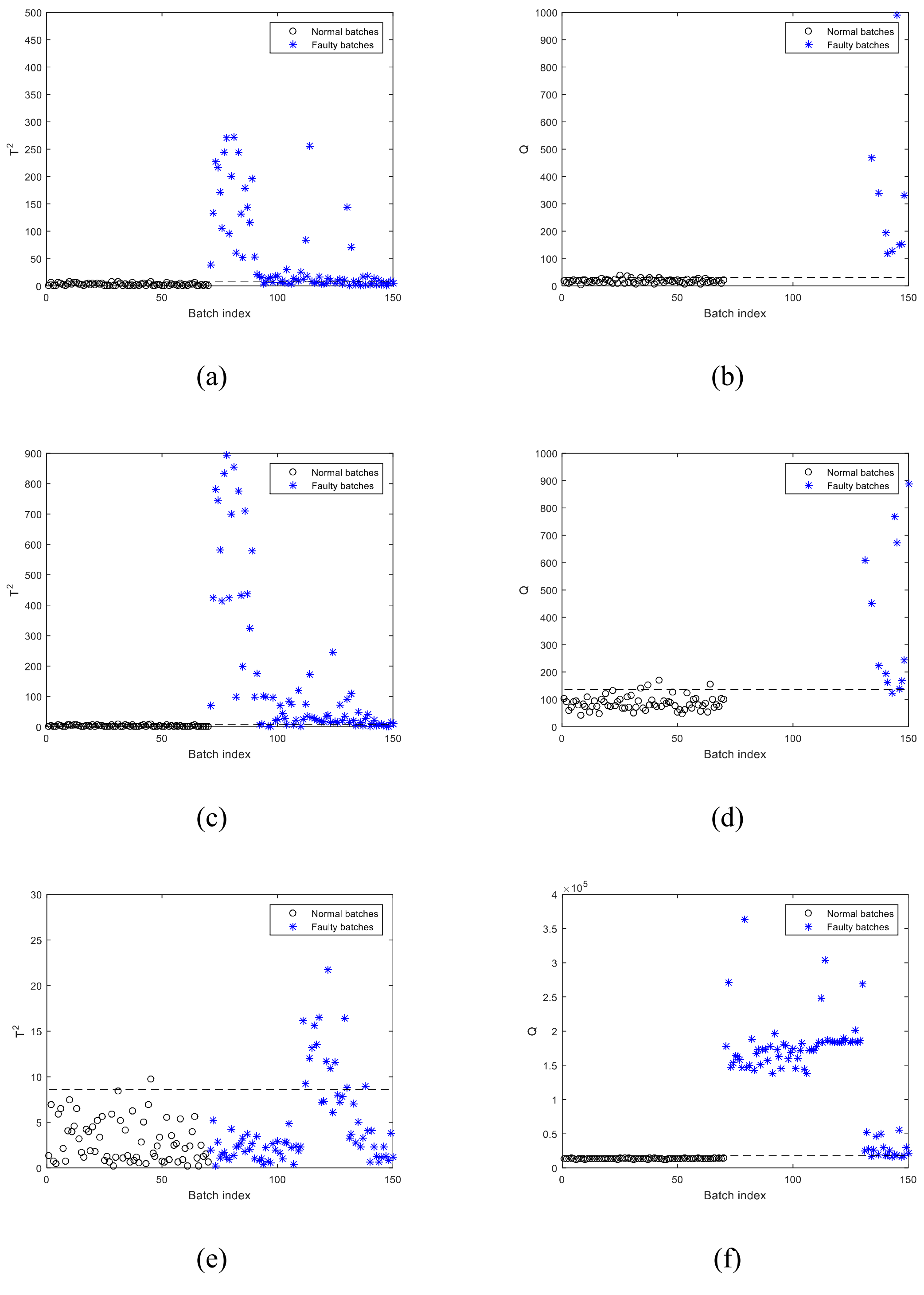
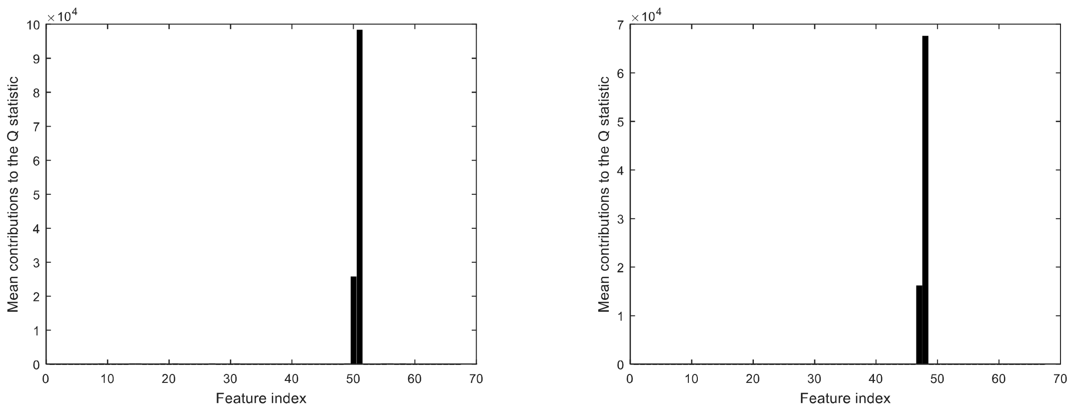


Figure 3.6. Monitoring statistics for NOC and abnormal simulated batches: (a) T^2 and (b) Q statistics for PCA model with the PdF dictionary; (c) T^2 and (d) Q statistics for PCA model with the SPA dictionary of features; (e) T^2 and (f) Q statistics for PCA model with the BWU data aligned with DTW. Batch indexes 1-70 correspond to NOC data, 71-90 correspond to fault 1, 91-110 correspond to fault 2, 111-130 correspond to fault 3 and 131-150 represent fault 4. The value of the Q statistic for some batches is very high and are not presented for ease of visualization. The dashed line is the 95% upper control limit.



(a) (b)

Figure 3.7. The average contribution of each feature to the Q statistic for (a) fault 1 and (b) fault 2.

Chapter IV
Time-Resolved Feature-Oriented Methods

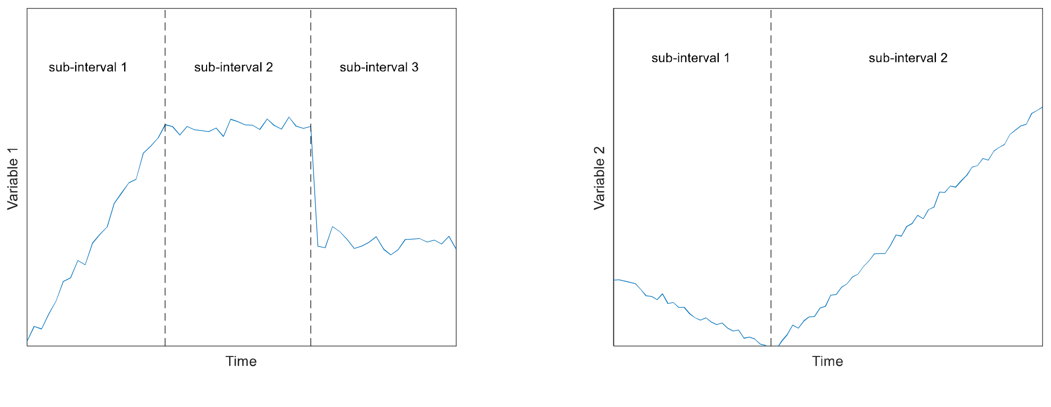
4 Time-Resolved Feature-Oriented Methods

The discussion in the previous chapter focused mainly on applying FOBA in a batch-wise fashion, i.e., features were extracted for the whole batch and analyzed by suitable chemometric and statistical methods. The outcomes of such analyses revealed features and variables that are important from the perspective of the entire batch; however, they could not pinpoint batch periods that are more critical. Following the taxonomy adopted in this thesis (see Section 2.1), FOBA methods are said to have limited time-resolution because they use a time grid with low granularity. The time-resolution of feature-oriented methods can be improved when process knowledge regarding different batch stages is available, which allows extracting features in a stage-wise fashion. However, in the more general case, this knowledge may not exist or even when available, the batch stages can still be subdivided to provide even more detailed information regarding process operation. Therefore, in this chapter, a proposal is put forward to improve the time-resolution of FOBA methods, that is capable of providing localized information regarding critical periods of a batch operation. The first section of this chapter describes an improved version of FOBA, named FOBA 2.0 (from now on, the previous version of FOBA will be also referred as FOBA 1.0). This novel approach contemplates three alternatives for improving the time-resolution capabilities of feature-oriented methods. Two of these alternatives make use of an algorithm for finding change-points in time-series data. This algorithm is described in the second section of this chapter. The third section describes a case study where these alternatives were applied and assessed, whereas the last section presents the results and their respective discussion.

4.1 FOBA 2.0

The defining characteristic of time-resolved methods is that they transpose the time dimension to the modeling phase and, therefore, the model can be interpreted in order to identify important stages. The BDA literature already contains some approaches that can be utilized to identify batch stages where the process variables' mean levels and correlation structure are stable (Lu, Gao et al. 2004, Camacho, Picó et al. 2008, Yao and Gao 2009). However, these approaches implicitly require batch synchronization in order to transpose all time-points to the modeling phase, ensuring that all batches have equal duration. Since one of the major advantages of feature-oriented methods is that they do not require batch synchronization, it would not be suitable to adopt the aforementioned approaches to identify batch stages.

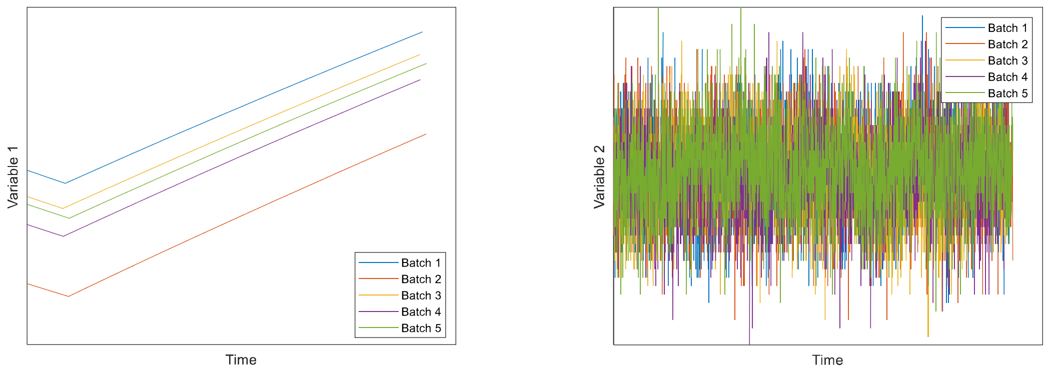
In FOBA 2.0, the batch is divided into sub-intervals followed by the extraction of features for each sub-interval. All features arising from all sub-intervals are concatenated and subsequently analyzed. The majority of feature-oriented methods are characterized by a variable-wise analysis first (where each variable is characterized by certain features), followed by a multivariate analysis of the features (where the multivariate relationship between variables is modeled). Following this principle, the identification of batch sub-intervals is also conducted variable-wise in FOBA 2.0. In other words, the procedure is flexible so that each variable may have a different number of sub-intervals that best match the variables dynamic patterns (this is depicted in Figure 4.1). Therefore, the novel component in FOBA 2.0 compared to the first version is that a splitting procedure is implemented to specify batch sub-intervals. One should note that these batch sub-intervals could be similar to what is typically associated with a batch operational stage (e.g., load reactants, set reaction conditions, etc.). This can indeed be an output of FOBA 2.0, however, the main goal is to improve the insights that can be obtained from the data-driven model (by increasing its time-resolution) and not specifically the identification of operational stages. FOBA 2.0 is able to pinpoint batch periods (e.g. the beginning, the mid-period, or the end of the batch) that are more relevant even when they do not match traditional operational stages.



(a) (b)

Figure 4.1. Illustrating the flexibility of FOBA 2.0 to identify different sub-intervals for two process variables.

In the context of FOBA 2.0, the profile of each variable over a batch is considered as a time series and the goal is to detect when the characteristics of this signal changes. In the literature of time series analysis, this task is referred to as change-point detection (Scott and Knott 1974, Fu 2011, Truong, Oudre et al. 2018) and is concerned with the identification of points where the statistical properties of a signal change abruptly. The point where these properties change is known as a change-point and an algorithm that has been successfully applied for detecting change-points is the pruned exact linear time (PELT) method (Killick, Fearnhead et al. 2012). The main advantages of PELT are its efficiency and accuracy. It was therefore integrated in FOBA 2.0 for the task of identifying sub-intervals for each variable. PELT is described in more detail in the next section, but for the remaining of this section, it suffices to acknowledge that the algorithm takes, as inputs, the profile of each variable over a batch and outputs the number of change-points detected and their location. Furthermore, PELT's output can be used to infer whether a process variable has a consistent dynamic profile or not. In this context, a consistent dynamic pattern means that a process variable presents dynamic characteristics that are stable across batches and, therefore, it is meaningful to consistently extract localized features for it. An example of such a variable is presented in Figure 4.2.a, where one can see that the same pattern is observed across all the batches. In contrast, some variables are noisier and do not have a stable dynamic pattern. In this case, batch-wise features are more appropriate (Figure 4.2.b). With PELT, one can distinguish between these two types of variables (variables with consistent profiles and those lacking dynamic consistency) by analyzing the distribution of the number of change-points detected for all the batches. Variables presenting a consistent dynamic pattern will typically have a small number of change-points and the variability of the number of change-points across batches is small. Variables that are noisier will often present a large number of change-points and the variability of this number across batches is large. This assessment step will be demonstrated in more detail in Section 4.3 when applying FOBA 2.0 to a simulated case study.



(a) (b)

Figure 4.2. Examples of the trajectories of variables presenting: (a) a consistent dynamic pattern where it is meaningful to extract localized features and (b) a noisier behavior where batch-wise features are more adequate.

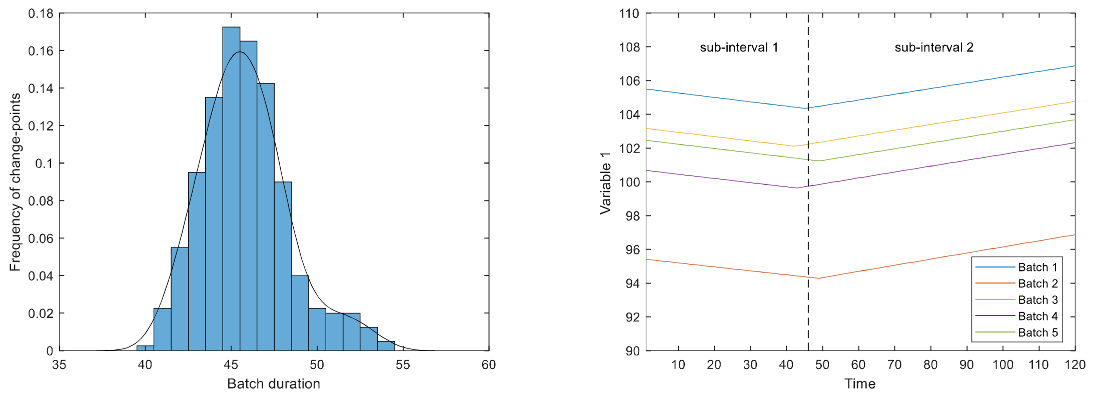
In FOBA 2.0, a new approach is proposed for establishing the batch sub-intervals in a way that does not require synchronization. This capability increases the modeling complexity of FOBA 2.0 methods because the number of features generated is higher (they are required to characterize the identified sub-intervals). Furthermore, the implementation complexity is also higher because one needs to specify the batch sub-intervals to be considered (either using PELT or another procedure). Multiple alternatives, with varying degrees of complexity, were envisioned for implementing FOBA 2.0. They were developed and tested, and will be described next:

i) *Equally spaced sub-intervals (ESI)*. In this approach, each batch is split into a pre-specified number of sub-intervals (e.g., 4 sub-intervals). Thus, all variables will share the same number of sub-intervals and for a given batch, the length of the sub-intervals is constant. On the other hand, the length of each sub-interval (e.g., the first sub-interval) will vary across batches because batches are not synchronized. The number of sub-intervals is a hyper-parameter that can be tuned by cross-validation or imposed by the user. A large number of sub-intervals would lead to noisy features whose computation only considers a few data points. Conversely, a small number of sub-intervals decreases the time-resolution information that can be obtained. In our experiments, changes in this parameter within reasonable values did not have a large effect on the results. The equally spaced approach is the simpler alternative that can be adopted to extract time-resolved features;

ii) *Equally spaced sub-intervals for variables with a consistent dynamic pattern (ESID)*. This approach adds a layer of complexity compared to the first alternative because it requires the preliminary identification of variables with a consistent dynamic pattern. As previously discussed, PELT is utilized for discriminating between consistent and noisy variables. Batch-wise features are computed for variables that are noisier whereas, for variables presenting a consistent dynamic pattern, the mode (computed over all batches) for the number of change-points detected by PELT is considered for establishing the splitting points to use. In particular, each consistent variable is split into equally spaced sub-intervals according to the number of change-points detected (*#sub-intervals* = mode(*#change-points*) + *1*). One can note two major differences between approaches ESI and ESID. The first is that only consistent variables will be split in ESID, whereas in ESI, all variables are split. The second difference is that with ESID, the number of sub-intervals (and, as a consequence, their length) differs depending on the variable under analysis. Variables with more intricate dynamics will be split into more sub-intervals whereas fewer sub-intervals will be considered for variables with simpler dynamics;

iii) *Unequally spaced sub-intervals for variables with a consistent dynamic pattern (USID).* This approach is an extension of ii) and the difference is that the sub-intervals for each consistent dynamic variable may be unequally spaced. Instead of only considering the number of change-points as being outputted by PELT (as the approach described in ESID), their location is also used to identify when the variable's characteristics change, signaling the start of a new sub-interval. From the three approaches studies, this is the more flexible for defining the sub-intervals ; see Figure 4.1.

The improved flexibility of approach iii) compared to alternatives i) and ii) can have, as a side effect, some robustness problems. These issues arise especially when the number of change-points for a given batch is different from the mode number of change-points, implying that a different pattern is present in the variable's profile. This information can identify a possibly faulty batch but it does not help locating the period where the fault occurs, failing to improve the time-resolution of the method. In order to increase the robustness of the USID approach, an additional step is employed that consists in fitting a probability function to the distribution of change-points observed in the model development stage. The modes of this distribution function are adopted to specify fixed change-points for all batches. In other words, the change-points for subsequent batches are assumed to occur at the same time instants as the modes obtained from the probability distribution function. This additional step is exemplified in Figure 4.3.a where the distribution of change-points for a process variable is shown. The mode(s) of the probability density function can be utilized to compute fixed change-points that will be later on employed to define unequally spaced sub-intervals, as depicted in Figure 4.3.b. One should note that the splitting point used in Figure 4.3.b is not optimal since the inflection point for some batches will not match the mode of the probability density function. This mismatch will carry over to the computation of the features and increase their variability; however, it greatly improves the robustness of the USID approach.



(a) (b)

Figure 4.3. Specifying the length of the sub-intervals in approach iii: (a) the distribution of change-points over all batches is approximated by a probability density function, and (b) the mode of this distribution is a used as the change-point for all batches.

The aforementioned alternatives achieve the goal of increasing the time-resolution of feature-oriented methods by splitting the batch into sub-intervals that may or may not be equally spaced. However, the fact that the major batch milestones are not synchronized implies that the variability of the features computed in each sub-interval will be inflated. In applications where the profiles tend to be less synchronized, a high percentage of the features' variability will reflect this type of variation, instead of describing the dynamic characteristics of the batch. Therefore, the sensitivity of these methods may deteriorate when the batch-to-batch variability is large. Nevertheless, their simplicity implies that they may be easily tested without much fine-tuning. Even when applying FOBA 2.0 to scenarios where it is less suitable, the time and expertise requirements are quite low, and the outcomes extracted may already improve the knowledge regarding process operation.
Figure 4.4 presents a general workflow for feature-oriented methods that starts with the definition of the resolution level for the information to be extracted from the dataset collected during process operation. If a low-level resolution is enough, FOBA 1.0 can be employed to extract batch-wise features (or stage-wise features if stages are known *a priori*). When finer resolution is aimed, time-resolved features should be adopted and

FOBA 2.0 selected, as it is able to extract features with such characteristics in a robust way.

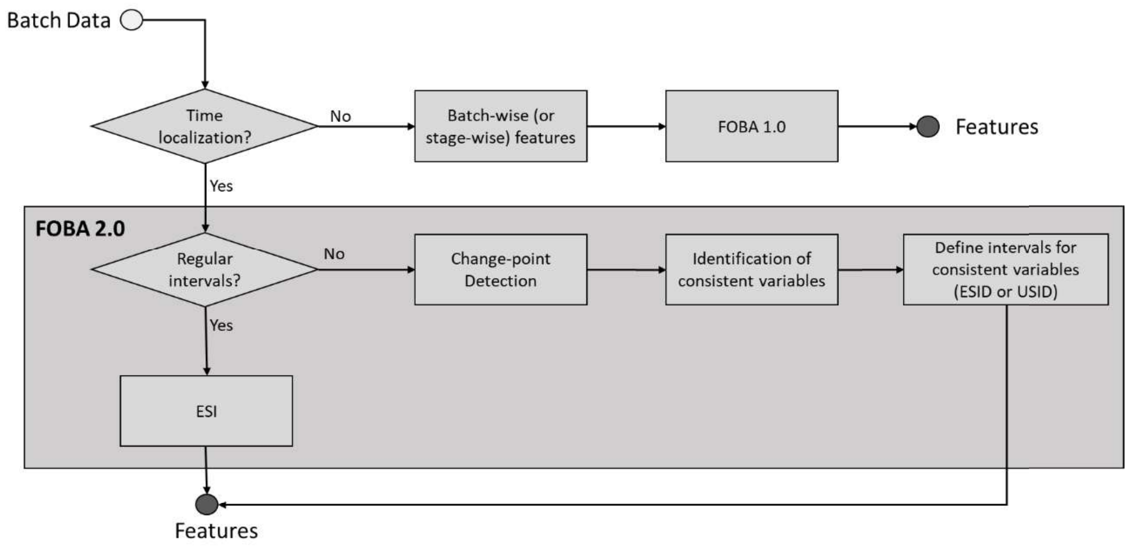

Batch Data
Time localization?
No
Batch-wise (or stage-wise) features
FOBA 1.0
Features
Yes
FOBA 2.0
Regular intervals?
No
Change-point Detection
Identification of consistent variables
Define intervals for consistent variables (ESID or USID)
Yes
ESI
Features

Figure 4.4. A general workflow for employing feature-oriented methods (FOBA and FOBA 2.0).

4.2 Change-point detection

Algorithms for change-point detection have been used to segment time series data in a wide range of applications ranging from electroencephalography recordings (Lavielle 2005), genetics (Chen and Gupta 2011), financial data (Chen and Gupta 1997), manufacturing (Keogh, Chu et al. 2001), among others. A review on change-point detection algorithms was recently published (Truong, Oudre et al. 2018), describing different classes of available methodologies as well as the most relevant approaches within each class. In general, these approaches assume that the measured signal is piecewise stationary and the aim of change-point detection methods is to identify the points where (abrupt) changes do occur. In more detail, the profile of a variable is assumed to change at unknown instants $t_1^* < t_2^* < ... < t_C^*$, where C corresponds to the number of change-points in the profile. The goal of change-point detection is to provide estimates of these change-points, $\hat{t}_l$, which split the variable's profile into homogenous regions. The output of change-point detection methods is a set of estimated change-points

$\hat{t} = \{\hat{t}_1, \hat{t}_2, ..., \hat{t}_{\hat{C}}\}$, where $\hat{C}$ is the number of estimated change-points. Some algorithms assume that *C* is known *a priori* whereas others can also provide an estimate for *C*, i.e., $\hat{C}$. The latter option is best suited in the context of BDA since the number of change-points in the batch is unknown. In order to identify the location and the number of change-points, the following optimization problem is solved:

$$\hat{\mathbf{t}} = \arg\min_{\mathbf{t}} \sum_{l=1}^{\hat{C}+1} \varsigma\left(x_{t_{l-1}+1:t_l}\right) + pen(\mathbf{t}) \tag{11}$$

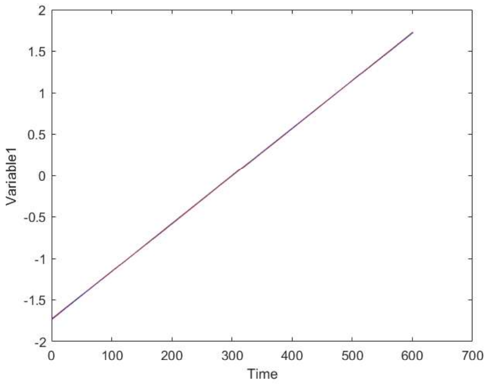
where $\varsigma\left(x_{t_{l-1}+1:t_l}\right)$ is the cost incurred for the interval $\left[t_{l-1}+1, t_l\right]$ and $pen(\mathbf{t})$ is a penalty term for large number of splits. It is assumed that $t_{\hat{C}+1} = K$, i.e., the last change-point correspond to the length of the signal and, therefore, the input signal is split into $\hat{C}+1$ intervals. In order to minimize eq. (11), a tradeoff is achieved between the cost due to the lack of fitting (which decreases as more change-points are added) and the penalty associated with the number of change-points (which increases with the number of change-points). Various cost functions and penalties can be considered, whereas several search methods can be employed to minimize eq. (11) and provide either an exact or approximate solutions. Exact search methods are computationally more expensive, whereas approximate methods are faster but may provide a sub-optimal minimum. In this context, the pruned exact linear time (PELT) method is an effective alternative that is both accurate (the solution is optimal for many cost functions) and efficient (the computational cost is, under mild conditions, linear in the number of observations). The penalty term utilized in PELT corresponds to a penalty level β multiplied by the number of change-points $\left(pen(\mathbf{t}) = \beta \times \hat{C}\right)$, where small values of β lead to partitions with many change-points and large values imply that only very significant changes are signaled. PELT improves on a method named optimal partition (Auger and Lawrence 1989) that expresses eq. (11) in a recursive fashion. Let $F(s)$ be the minimization of eq. (11) for the data $x_{1:s}$, then it can be shown that $F(s)$ can be written as a function of the minimal cost for the data $x_{1:t}$, denoted $F(t)$, for $t < s$:

$$F(s) = \arg\min_{t} \left\{F(t) + \varsigma\left(x_{(t+1):K}\right) + \beta\right\} \tag{12}$$

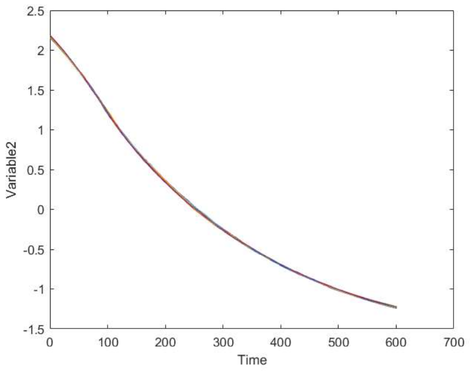
Eq. (12) provides a recursive function to compute the minimum cost and can be solved for $s = 1, 2, ..., K$. The overall computational cost of finding $F(K)$ is quadratic in *K* but PELT improves on this cost by using a pruning rule that can efficiently exclude candidate change-points. The theorem supporting PELT's pruning rule can be consulted in the original paper (Killick, Fearnhead et al. 2012), and the output of the algorithm is the number and location of change-points, which in context of FOBA 2.0, can be utilized to identify consistent variables and also specify the number of sub-intervals to consider per variable.

4.3 Case Study

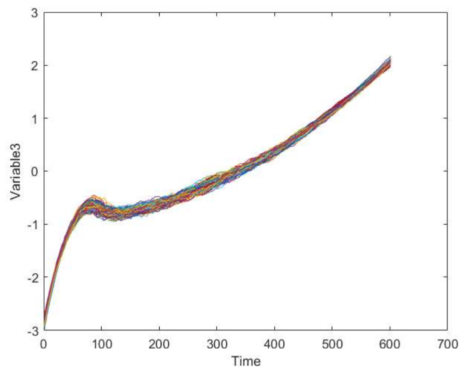
The case study considered here for testing the variants in FOBA 2.0 consists of a semi-batch reactor equipped with a cooling jacket, where an exothermic second order reaction takes place $(A+B \rightarrow C)$. The reactor is initially charged with reactant A and is continuously fed with a constant flow of reactant B. The temperature inside the reactor is controlled with resort to a PID controller, manipulating the flow of cooling fluid circulating in the jacket in order to maintain the reactor temperature at approximately 25°C. In order to mimic real process variability, a variety of noise patterns were simulated: Gaussian noise affects all reaction kinetics and heat transfer parameters, while most temperatures and flow rates are subject to auto-regressive drifting patterns. This case study will be referred to as the SEMIEX process. During process operation, 6 variables are measured, and 200 batches are available for model building. This system is rather simple but allows for a better understanding of the FOBA 2.0 alternatives and for illustrating the steps employed to uncover important batch stages. Figure 4.5 presents the trajectories of the 6 process variables for all the batches (variables are auto-scaled).

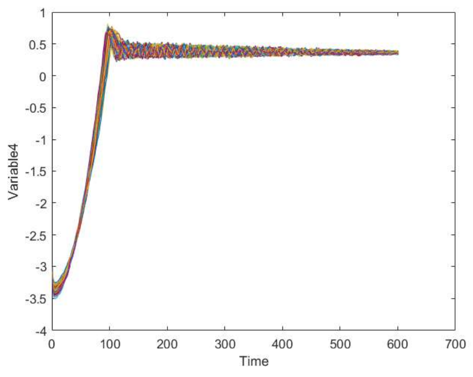


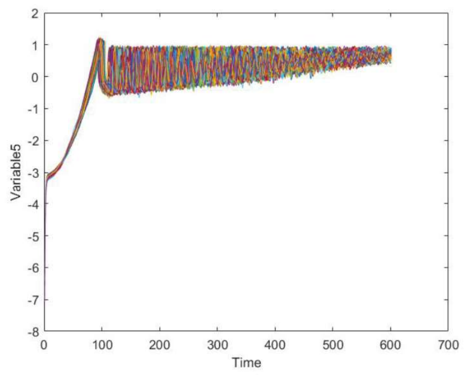
(a)



(b)


(c)


(d)


(e)

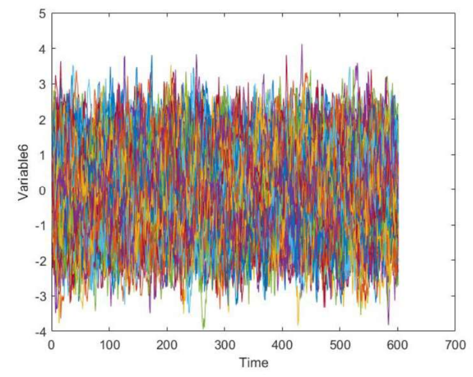
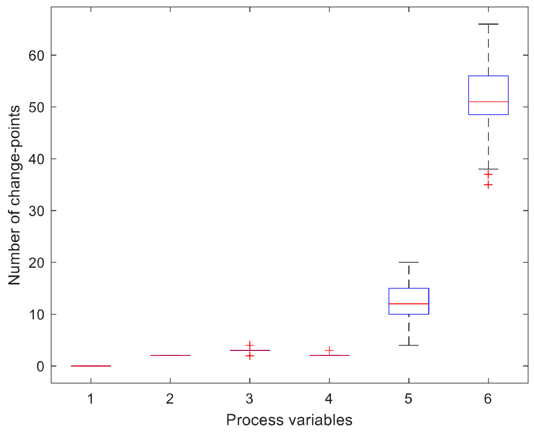

(f)

Figure 4.5. The trajectory of the 6 process variables in the SEMIEX case study after auto-scaling. The variables correspond to (a) volume of the reaction mixture, (b) the concentration of reactant A, (c) the concentration of reactant B, (d) the temperature of the reactor, (e) the temperature of the cooling fluid, and (f) the flow of reactant B.

As previously discussed, alternatives ESID and USID within FOBA 2.0 require the identification of variables presenting a consistent pattern. This is achieved by analyzing the distribution of change-points detected over all the batches for each variable. Figure 4.6 presents the number of change-points detected for all process variables and all batches and one can note a clear difference between variables #1-4 and variables #5 and #6. These differences can be explained when considering the profile of the variables (Figure 4.5). For instance, variable #1 has no change-points, which is in accordance with its profile (Figure 4.5.a) since it is a linear trend without significant changes. Variable #2 (Figure 4.5.b) also has a consistent dynamic pattern and two change-points are detected corresponding to changes in the profile slope (the first around time 200 and the other around time 400). On the other hand, variable #6 has a noisy pattern consisting of random noise around a mean level (Figure 4.5.f) and, therefore, many change-points are detected and their variability across batches is significant. Following the analysis of Figure 4.6, one can conclude that variables #1-4 can be split in order to implement alternatives ESID and USID of FOBA 2.0, whereas batch-wise features should be computed for variables #5 and variable #6. A particular case occurs in the SEMIEX simulator because variable #1 is a simple linear trend with no change-points. Therefore, this variable is not split and batch-wise features are also computed.



(a)

Figure 4.6. Distribution of change-points detected by PERL for each variable over all batches.

4.4 Results and Discussion

To test the proposed FOBA 2.0 alternatives, partial least squares (PLS) models were built. These models use, as predictors, features obtained from the process variables and the response is the concentration of product C. The models are trained using 150 randomly selected batches and their performance is assessed in an independent test set with 50 batches. The coefficient of correlation for the test set $\left(R^2_{test}\right)$ is presented in Figure 4.7 for various numbers of latent variables utilized for training the model. This solution for representing the performance of the three alternatives is utilized because, during the model training stage, the RMSE obtained by cross-validation always decreased as the number of latent variables increases. This representation demonstrates the performance of the methods and avoids the ambiguity that may arise from the selection of the number of latent variables. Figure 4.7 shows that there are small differences between the SPA alternatives (within FOBA 2.0) and the standard SPA approach, with a slight advantage to ESI. This may arise from the fact that this is a simple system and the prediction of the product's concentration is rather straightforward, given that all the process variables are measured. Nevertheless, it shows that FOBA 2.0 alternatives can be as effective or even more, as the standard version. The main advantage of these alternatives is that they can easily identify periods of the batch that are more critical for the quality parameter. The variable importance in projection (VIP) is a typical feature importance metric utilized to assess and compare the relevancy of the different features (Chong and Jun 2005). Figure 4.8 presents the feature importance for SPA and FOBA 2.0 with the ESI alternative. For SPA (Figure 4.8.a), only overall importance can be assessed and one can note that there are two highly important features. When analyzing Figure 4.8.b, one can have a deeper understanding of the critical periods. Although the profile of variable importance within each stage is similar, one can see that the first stage is more important. The variable that has the highest importance in every stage is variable #2 (concentration of reactant A). Since this reactant is loaded at the beginning of the batch, it would be expected that the initial amount of A would largely determine the concentration of product C. Thus, one can conclude that the ESI alternative was quite effective for improving the time-resolution of SPA and avoids some of the complexity associated with approaches ESID and USID. Therefore, it is the recommended method to be applied in FOBA 2.0. Nevertheless, as each case study presents different data characteristics and challenges, it is expected that alternatives ESID and USID may prove to be effective for more complex systems. Furthermore, although only SPA was considered for this case study, FOBA 2.0 can be extended to other feature-oriented methods, such as TIME-F: one of the three FOBA 2.0 alternatives can be utilized to identify sub-intervals for each process variable and TIME-F features can be computed for each sub-interval.

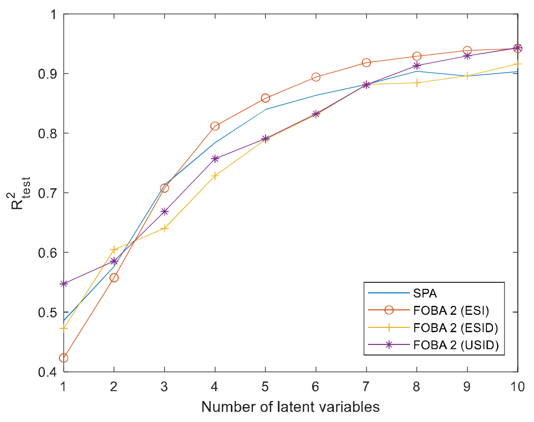
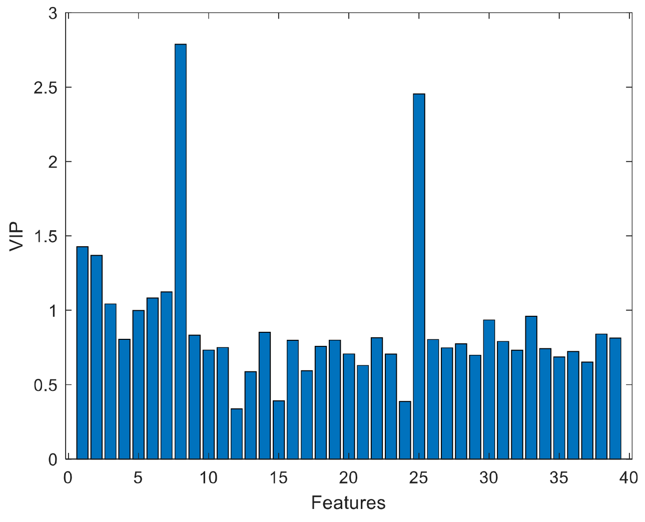
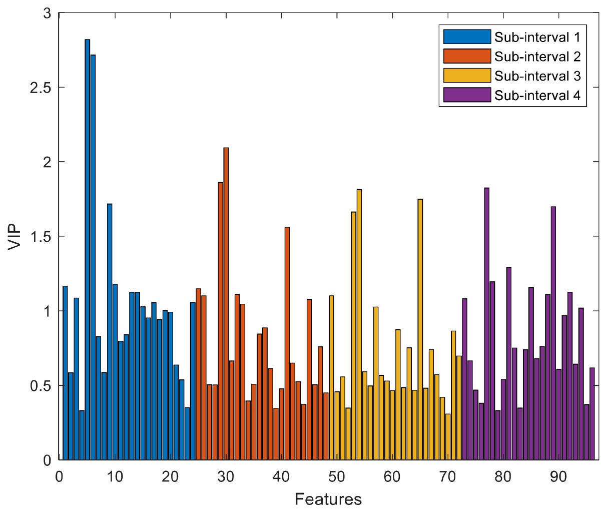


Figure 4.7. The performance in the test set for SPA and FOBA 2.0 alternatives.



(a)



(b)

Figure 4.8. Variable importance in projection for (a) SPA with batch-wise features and (b) FOBA 2.0 with the ESI alternative.

Chapter V
Wide Spectrum Feature Selection

5 Wide Spectrum Feature Selection

The previous two chapters presented frameworks for batch data that were able to extract features from batch trajectories. Once features become available, they can be utilized in a variety of tasks (e.g., visualization, troubleshooting, quality prediction, offline monitoring, etc.), as previously discussed. One important task in BDA is quality prediction, which requires developing a predictive model that relates features and a quality parameter of interest. However, the volume of industrial datasets often impairs the inclusion of all features (also known as predictor variables) in the model development stage. Furthermore, using all available features for data-driven modeling is not recommended, as most of them are expected to be irrelevant and their inclusion in the model may compromise robustness and accuracy. Therefore, screening and selecting the most promising features is a recommended step that must be addressed in modern predictive analytics and will be topic of this chapter. This chapter is divided into five sections. The first section provides an introduction to feature selection and enumerates many of the classes and methods available in the literature. The analysis of the literature motivates the development of a new methodology named wide spectrum feature selection for regression (WiSe) that is especially suitable for the context of BDA. WiSe is described in detail in the second section of this chapter. The last three sections of this chapter are dedicated to demonstrating the effectiveness of the WiSe methodology: the third section describes the datasets used to test WiSe, the fourth section describes how the methodology was compared against other benchmark alternatives, and the last section presents the comparison results and their discussion.

5.1 Introduction to Feature Selection

A pervasive characteristic of data collected from modern chemical processing industries (CPI) is the high number of measured variables/features (features and variables will be used interchangeably in this chapter since methods for feature selection can also be employed for variable selection), often ranging between hundreds to thousands or even tens of thousands. This corresponds to the Volume component of the big data 4 V’s (Qin 2014), the other components being Velocity, Variety, and Veracity. In this context, applying feature selection methods (FSM) is almost mandatory in order to remove noisy and irrelevant features that are not informative for the data-driven task considered and may compromise their predictive performance and robustness (Seasholtz and Kowalski 1993, Höskuldsson 1996, Broadhurst, Goodacre et al. 1997, Höskuldsson 2001, Brás, Lopes et al. 2008). A variety of FSM have been proposed in the literature and their main advantages include the lower dimensionality of the resulting dataset, easier identification of important variables and lower risk of overfitting. These FSM methods are often grouped into three classes (Guyon and Elisseeff 2003, Saeys, Inza et al. 2007, Bolón-Canedo, Sánchez-Maroño et al. 2013, Chandrashekar and Sahin 2014):

- *Filter methods.* Use an association metric between features and response variables in order to detect important and irrelevant features;
- *Wrapping methods.* Rank predictors by their importance in the context of a developed classification or regression model, possibly assessing the prediction improvements obtained by including/excluding different (groups of) features;
- *Embedded methods.* Follow a scheme for automatically removing predictor variables during model training by, for instance, introducing a penalty for the magnitude of regression coefficients.

The literature on feature selection is vast (Ruiz, Pérez et al. 2009, Liu and Motoda 2012), containing general review papers (Guyon and Elisseeff 2003, Bolón-Canedo, Sánchez-Maroño et al. 2013, Chandrashekar and Sahin 2014, Vergara and Estévez 2014, Li, Cheng et al. 2016), and reviews on specific areas such as bioinformatics (Saeys, Inza et al. 2007), microarray datasets (Lazar, Taminau et al. 2012, Bolón-Canedo, Sánchez-Marono et al. 2014), chemometrics and industrial applications (Anzanello and Fogliatto 2014). Among these contributions, some regard filter methods (mostly for classification purposes). For example, Relief (Kira and Rendell 1992, Robnik-Šikonja and Kononenko 2003) is a well-known representative algorithm belonging to this class. Relief is utilized in classification tasks and works by randomly selecting a sample and its two nearest neighbors, one from the same class and another from a different class. A weight is attributed to each feature according to its ability to separate the classes, and these weights are iteratively updated when a different sample is selected. Relief was later modified to be more robust to noise and missing data (Kononenko 1994) (ReliefF) and was extended to regression problems (Robnik-Šikonja and Kononenko 1997) (RReliefF). Other types of filters are based on information theory (Estévez, Tesmer et al. 2009, Vergara and Estévez 2014, Muhammad Aliyu 2015) (e.g. mutual information and symmetrical uncertainty) such as the fast correlation-based filter (Yu and Liu 2003, Yu and Liu 2004) (FCBF) and the max-relevance and min-redundancy filter (Peng, Long et al. 2005) (mRMR). These filters search the feature space for a set of relevant but non-redundant features and have appeared in many comparison studies with classification datasets (Hall and Smith 1999, Estévez, Tesmer et al. 2009, Bolón-Canedo, Sánchez-Maroño et al. 2013). More recently, another set of efficient filters have been proposed (Ferreira and Figueiredo 2012) for both supervised and unsupervised problems.

Wrapping methods (or simply wrappers) form a class of very successful FSM for CPI datasets. Due to the collinearity between features, partial least squares (Geladi and Kowalski 1986, Burnham, MacGregor et al. 1999, Wold, Sjöström et al. 2001) (PLS) is the *de facto* standard method in CPI together with its extensions with feature selection capabilities. Variable importance in projection PLS (Wold, Sjöström et al. 2001) (VIP-PLS), beta-coefficient PLS (Chong and Jun 2005) (beta-PLS), recursive PLS (Rinnan, Andersson et al. 2014) (rPLS), and PLS with genetic algorithms (Leardi and Gonzalez 1998) (PLS-GA) are examples of wrappers that select features in the context of a PLS model. Another popular type of wrapper methods is the one composed by methods derived from ordinary least squares (OLS) (Draper and Smith 1998) followed by the elimination of features that do not bring a statistically significant additional explanation power to the model. OLS can be combined with forward, backward or stepwise strategies, or even exhaustive enumeration strategies (Andersen and Bro 2010, Montgomery, Peck et al. 2012) for including and removing features from an OLS model.

In the class of embedded methods, least absolute shrinkage and selection operator (Tibshirani 1996) (LASSO) has been used to shrink model coefficients to zero, effectively removing features. Elastic net (Zou and Hastie 2005) (EN), which combines both ridge regression (Hoerl and Kennard 1970) and LASSO penalties, also has the desirable property of automatically eliminating features during model estimation.

Each of the aforementioned classes of FSM has advantages and drawbacks. However, in scenarios where a large number of variables are measured, filters tend to be the preferred approach (Yu and Liu 2003, Liu and Yu 2005, Ferreira and Figueiredo 2012, Lazar, Taminau et al. 2012, Bolón-Canedo, Sánchez-Maroño et al. 2013) as the other classes of methods depend on the incorporation of all features in the models under analysis, which is a cumbersome and computationally intensive task in high-dimensional settings. Filters are faster, easy to apply and automate, require less computational resources, and are more general since they are not biased by the modeling framework selected. Therefore, they are suitable to be applied on CPI datasets that are characterized by having a large number of predictor variables to be initially considered.

The analysis of the aforementioned literature shows that many filters for classification problems have been proposed and tested in the past, whereas filters for regression problems remain mostly unexplored, with only a few examples available (Robnik-Šikonja and Kononenko 1997, Muhammad Aliyu 2015). Adopting filters in regression settings is expected to bring similar benefits to those already obtained for classification problems. Therefore, in this chapter, we propose, test and compare an approach that contains algorithmic and methodological elements suitable for regression problems found in CPI: the wide spectrum feature selection for regression (WiSe). WiSe is a two-stage feature selection approach that combines a filter stage and a model development stage. In the first stage, a filter or a set of filters are adopted to identify and select important features, while noisy and irrelevant predictors are removed. These filters are based on measures of association such as Pearson's correlation coefficient (Gibbons and Chakraborti 2011, Hauke and Kossowski 2011), Spearman's rank coefficient (Gautheir 2001, Gibbons and Chakraborti 2011), and mutual information (Jaynes 1957, Shannon 2001). Each metric assesses a type of association between features and response variable such as linear (Pearson's correlation), monotonic (Spearman's coefficient), and non-linear (mutual information). Features selected in the first stage proceed to the second stage where a regression model is developed and, by employing a wrapping or embedded method, a smaller set of features is finally obtained. This two-stage approach leads to the development of more parsimonious models, which is a critical attribute for modeling CPI datasets. Another benefit closely linked to parsimony is prediction accuracy, which improves when irrelevant/noisy variables are removed. Two-stage approaches have already been proposed for classification problems (Peng, Wu et al. 2010, El Akadi, Amine et al. 2011), but to the best of the author's knowledge, this is the first time this strategy is used in a regression setting with a comprehensive coverage of filter methods.

5.2 Wide Spectrum Feature Selection for Regression (WiSe)

The wide spectrum feature selection for regression (WiSe) methodology was devised with the purpose to secure the derivation of robust regression models when analyzing high-dimensional datasets, by employing a judicious combination of FSM. WiSe is a two-stage procedure, as depicted in Figure 5.1. In the first stage, a filter layer is applied for an efficient feature screening, where noisy/irrelevant predictors are removed from the initial feature matrix, $\mathbf{X}$ with dimensions $N \times m_f$, where N stands for the number of observations (or batches in the context of FOBA) and m_f for the number of variables/features. The resulting matrix from this first stage, $\mathbf{X_F}$ with dimensions $N \times f$, $f \le m_f$, is then utilized for model building. The regression method considered can be of a wrapper or embedded type and further feature selection can take place, leading to a final matrix, $\mathbf{X}_\mathrm{p}$ with dimensions $N \times p$, $p \le f$. Details of the approaches considered in each stage of WiSe are given in the following subsections.

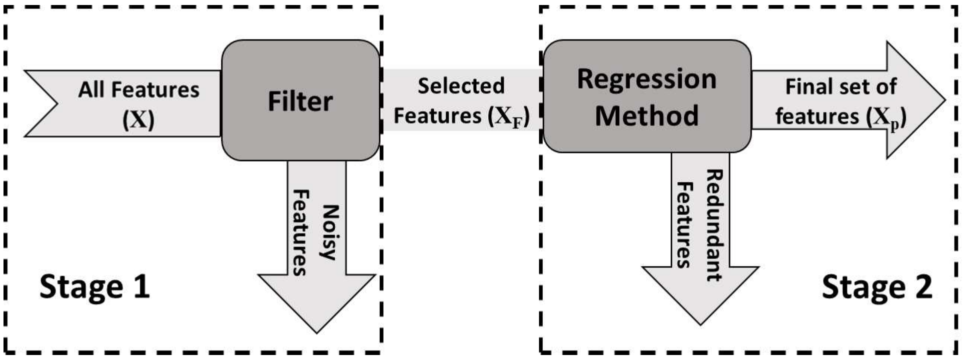


Figure 5.1. The proposed FSM: the first step uses a filter layer to reduce the number of features, followed by a second stage where, in the context of a regression method, additional features can be removed by application of a wrapper/embedded methodology.

5.2.1 Filtering Stage

Filters are screening methods based on measures of feature importance that are independent of a particular regression model. They simply relate features and response variables and measure the strength of the corresponding linear or non-linear associations. The metric of importance can be univariate or multivariate. As the name suggests, univariate filters assess the relevance of each feature individually, whereas multivariate filters account for the possible collinearity between predictors when obtaining a final set of features. In WiSe, we employ univariate filters. The reason is that they tend to be more efficient and scalable with data dimensionality, since they do not assess feature subsets. Furthermore, a common goal of multivariate filters (e.g. mRMR (Peng, Long et al. 2005)) is to obtain a final set of relevant but non-redundant features, penalizing and excluding features that are highly correlated with the response. This would be inappropriate in CPI contexts, where datasets closely follow a latent variable model with many collinear features. A better estimate of this latent model is obtained when all correlated features are considered for model building, during the second stage of the process. Therefore, excluding correlated features in the first stage may have a negative impact on the second stage of regression model building. Another reason for not excluding correlated features is that correlation between features does not necessarily imply the absence of complementary predictive information (Guyon and Elisseeff 2003). All these aspects concern the second stage of model building and should be handled in that phase; the main purpose of the first stage is to remove noisy and irrelevant features, for which we have found out that univariate filters do present an efficient, robust and scalable solution.

In order to measure the correlation between features and response variable, a set of metrics have been carefully selected to be contemplated in the first WiSe stage. After exploring the spectrum of bivariate association measures for continuous variables, three metrics were selected and tested: Pearson's correlation coefficient (Gibbons and Chakraborti 2011, Hauke and Kossowski 2011), Spearman's rank coefficient (Gautheir 2001, Gibbons and Chakraborti 2011), and symmetrical uncertainty (SU) (Jaynes 1957, Shannon 2001).

Pearson's correlation coefficient (Gibbons and Chakraborti 2011, Hauke and Kossowski 2011) measures the linear association between variables and its value ranges between $[-1,1]$. A value close to 1 indicates two positively correlated variables, values close to -1 indicate negative correlation, and values close to 0 suggest the absence of a linear correlation. Eq. (13) presents the formula for computing Pearson's correlation coefficient:

$$\rho_j = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(x_{i,j} - \bar{x}_j)}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}\sqrt{\sum_{i=1}^{n}(x_{i,j} - \bar{x}_j)^2}} \tag{13}$$

Where ρ_j is the Pearson correlation coefficient for the j^{th} feature, y_i is the response value for the i^{th} sample, $\bar{y}$ is the average response over all samples, $x_{i,j}$ is the value of the i^{th} sample for the j^{th} feature, and $\bar{x}_j$ is the average value of the j^{th} feature.

In many situations, the relationship between features and the response variable is non-linear and the Pearson correlation is no longer suitable for a reliable measure of the corresponding strength of the association. Alternatively, Spearman rank coefficient (Gautheir 2001, Gibbons and Chakraborti 2011) can adequately describe the association for monotonic non-linear relationships and its values also fall in the range $[-1,1]$. The Spearman rank coefficient is presented in eq. (14):

$$s_j = \frac{\sum_{i=1}^{n}\left(r_i^y - \bar{r}\right)\left(r_{i,j}^x - \bar{r}\right)}{\sqrt{\sum_{i=1}^{n}\left(r_i^y - \bar{r}\right)^2}\sqrt{\sum_{i=1}^{n}\left(r_{i,j}^x - \bar{r}\right)^2}} \tag{14}$$

Where s_j is the Spearman rank correlation for the *j*th feature, r_i^y is the rank of the response for the *i*th sample, $\bar{r}$ is the average rank, and $r_{i,j}$ is the rank of the *j*th feature in the *i*th sample. One can note that the Spearman rank correlation (eq. (14)) is similar to the Pearson correlation coefficient (eq. (13)), however, the former is computed based on the ranks while the latter is based on the observed values for features and response.

The last metric of correlation between features and the response variable is based on Shannon & Weaver's information theory (Shannon 2001). More specifically, mutual information assesses the decrease in uncertainty in the response variable due to the availability of a feature to explain its variation. More formally, entropy is a measure of a variable's uncertainty and, for instance, the entropy of the response is given by eq. (15):

$$H(y) = -\int_{-\infty}^{+\infty} p(y)\log\left[p(y)\right]dy \tag{15}$$

where $p(y)$ is the marginal density function of the response. The mutual information between the response and a given feature measures the decrease in uncertainty in the response provided by the knowledge of feature *j*:

$$\begin{aligned} I\left(y,x_j\right) &= H(y) - H\left(y \mid x_j\right) \\ &= \int\int p\left(x_j,y\right)\log\left(\frac{p\left(x_j,y\right)}{p\left(x_j\right)p(y)}\right)dxdy \end{aligned} \tag{16}$$

where $I\left(y,x_j\right)$ is the mutual information between the response (y) and feature x_j, and $p\left(y,x_j\right)$ is their joint probability density function. Mutual information is maximum when knowledge of a feature completely specifies the behavior of the response variable, i.e., when there is no remaining uncertainty on the value of the response $\left(H\left(y \mid x_j\right) = 0\right)$. On the other hand, it will be zero when the response and the feature are statistically independent $\left(H\left(y \mid x_j\right) = H(y)\right)$. Mutual information is able to measure any type of dependency between variables and is a suitable metric to be used in the first filtering stage of WiSe in order to be able to detect general non-linear relationships. However, a drawback of mutual information is that it is biased towards features with larger number of values. Symmetrical uncertainty (SU) (Press, Teukolsky et al. 1996), is a normalized metric with values between $[0,1]$, that avoids such bias:

$$SU(y,x_j)=2\times\frac{I\left(y,x_j\right)}{H\left(y\right)+H\left(x_j\right)} \tag{17}$$

All the aforementioned metrics rank features according to their degree of association with the response. However, in order to implement a filter approach, one must also specify a threshold for deciding which features to retain and which can be safely discarded. A commonly used approach selects the top *T* features, and optimizes *T* by cross-validation (the filter needs to be combined with a regression method for this optimization step). However, in this work we follow a different approach. It consists in first obtaining a measure of the statistical significance associated with the association for each feature, after which they are ranked according to an appropriate relevancy metric (*p*-value). Finally, the rank ordered features are selected by controlling the False Discovery Rate (FDR) of the process. More details are provided in the following paragraphs.

The relevance of each feature is first assessed by parallel analysis, as described in Table 5.1. The first step is the selection and computation of the correlation metric (Pearson's correlation, Spearman's rank correlation or SU) between the feature and the response variable (lines 1 and 2), and the definition of the number of trials for the parallel analysis loop. Then, a loop is performed within which the response variable is randomly permuted in each run, followed by the computation of the relevance metric (line 6). The metric computed after each random permutation constitutes a random realization of the reference distribution where no relationship exists under the same circumstances (same number of observations, scale, etc.). At the end of the loop, a reference distribution is obtained for the null hypothesis of no association, from which one can estimate an empirical *p*-value as the percentage of iterations where random permutations resulted in a correlation coefficient larger than the one observed for the originally ordering of the response variable. It is expected that relevant features will present very low *p*-values (*p*-value << 0.05), whereas irrelevant features would be well within the noise distribution (high *p*-values, i.e., *p*-value > 0.05).

Selecting a threshold for the *p*-value corresponds to specifying the probability of incurring in false alarms, i.e., it limits the probability of mistakenly selecting an irrelevant feature. Setting the same threshold for all features may, therefore, lead to an excessive number of false alarms, due to the multiple comparisons that are conducted in parallel (Colquhoun 2014, Glickman, Rao et al. 2014) given the very high number of features. A common alternative to control the false alarm rate is to use the Bonferroni correction, where the probability of false alarms is divided by the number of comparisons, decreasing the number of features that are declared to be relevant. However, the Bonferroni correction is often criticized for having very lower power, failing to detect real differences (i.e., relevant features). Therefore, in this work, we adopt the false discovery rate (FDR) (Benjamini and Hochberg 1995, Glickman, Rao et al. 2014) as a base procedure for feature selection. The FDR is defined as the proportion (*d*) of all discoveries that are false, i.e., it corresponds to the percentage of features that are deemed significant by the application of the feature selection method, but are in fact irrelevant to the prediction problem. We set *d* to be 0.2, which is a conservative number compared to the more standard values of 0.05 or 0.01. Nevertheless, this larger threshold has the effect of minimizing the number of relevant variables that are removed, by allowing more of the irrelevant features to be retained. The justification is the following: since further variable selection will be conducted in the second stage of WiSe, it is critical that during the first stage only really noisy variables are removed, while those that can be potentially relevant are left for a subsequent screening stage (where some may be discarded using a computationally more expensive selection methodology, such as a wrapper or an embedded approach). In other words, the first stage is focused on minimizing the false negative rate (missed detections), whereas the second stage maximizes the true positive rate (true detections).

Applying the FDR method consists in arranging the features' *p*-values in increasing order (p_1, p_2, …, p_m). Then, one determines the largest index *k* for which $p_i \leq d \times i / m_f$; features with *p*-values $p_1, p_2, \ldots, p_k$ are deemed relevant and are retained for the second WiSe stage; the remaining ones are discarded.

The use of combined univariate filters is also a good alternative to the procedure described above, since it allows detecting both linear and non-linear relationships between features and the response variable. Two filters can be combined by simply considering the union of the individually selected features. However, an alternative is to use the minimum *p*-value among those obtained by the use of the two association metrics, to characterize the importance of a feature. In this way, it is possible to assess the strength of any type of relationship between features and the response variable, be them linear, monotonic non-linear or non-linear. The threshold applied to the set of minimum *p*-values is also based on the FDR methodology, with a target rate set to 0.2.

Table 5.1. Parallel analysis algorithm for computing the relevance of each feature (exemplified for the case of Pearson correlation).

```
1. metric = @Pearson
2. coefficient = metric(x, y)
3. n_iterations = 100
4. for i=1:n_iterations
5.     random_coefficient(i) = metric(x, randomPermute(y))
6. p_value = sum(random_coefficient >= coefficient)/n_iterations
7. return p_value
```

5.2.2 Regression Stage

In the second stage of WiSe, features selected during the filtering stage are used to develop a regression model. By adopting a wrapper or embedded method, further reduction in the number of features can be achieved and a smaller final set of predictors is obtained. In the regression stage, we have tested the following regression methods: forward stepwise regression (FSR) (Andersen and Bro 2010, Montgomery and Runger 2010), least absolute shrinkage and selection operator (LASSO) (Tibshirani 1996), and partial least squares (PLS) (Wold, Ruhe et al. 1984, Geladi and Kowalski 1986, Wold, Sjöström et al. 2001). It is important to note that other regression methods could be adopted in this second stage, but these three methods provide a good coverage of available approaches for high-dimensional empirical modeling. FSR belongs to the class of variable selection methods and assumes effects sparsity, i.e. that only some features contain relevant information regarding the response. LASSO is representative of penalized regression methods, a class of approaches that introduce some bias in the estimation in exchange of a reduction in estimation variance, by constraining the magnitude of regression coefficients. PLS is a method that assumes the relationship between features and response variable to be governed by a few unmeasured sources of variation, also known as latent variables. PLS uses the available dataset to estimate the data latent structure. Details of these regression methods are presented below.

FSR (Andersen and Bro 2010, Montgomery and Runger 2010) builds a regression model by sequentially including or excluding features based on the p-value of the partial F-test. The model coefficients are obtained by minimizing the least squares error between the response and the predictions obtained with selected features:

$$\hat{\mathbf{b}}_{FSR} = \underset{b=[b_0 \ldots b_p]^{\mathrm{T}}}{\arg\min} \left\{ \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2 \right\} \tag{18}$$

where $\hat{\mathbf{b}}_{FSR}$ is the estimated regression coefficients, y_i is the i^{th} observed response value and $\hat{y}_i$ is the corresponding model prediction ($\hat{y}_i = b_0 + \sum_{j=1}^{p} b_j x_{i,j}$).

LASSO (Tibshirani 1996) also minimizes the least squares error, but adds a L_1-norm penalty in order to decrease the magnitudes of the regression coefficients:

$$\hat{\mathbf{b}}_{LASSO} = \underset{b=[b_0 \ldots b_f]^{\mathrm{T}}}{\arg\min} \left\{ \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2 + \gamma \sum_{j=1}^{f} \left| b_j \right| \right\} \tag{19}$$

where $\hat{\mathbf{b}}_{LASSO}$ is the vector of regression coefficients estimated by the lasso model, and γ is an hyper-parameter that controls model complexity. Model complexity increase for low values of γ, as coefficients tend to be larger or different from zero under these circumstances. Therefore, the amount of penalty is optimized by cross-validation and depending on its value, some regression coefficients can be set to zero, meaning that the corresponding features are actually removed from the model.

PLS (Wold, Ruhe et al. 1984, Geladi and Kowalski 1986, Wold, Sjöström et al. 2001) estimates a latent variable subspace that explains the variability in both the predictors and response space. Although the model was previously presented in Section 2.1, it is repeated here for easier readability:

$$\mathbf{X} = \mathbf{TP}' + \mathbf{E} \tag{20}$$

$$\mathbf{y} = \mathbf{Tc} + \mathbf{f} \tag{21}$$

where **T** is an $n \times a$ orthogonal matrix of scores that represent unmeasured and independent directions of variability, a is the number of latent variables, **P** is a $p \times a$ loading matrix for the feature space, **c** is a $a \times 1$ vector relating the scores and the response variable. **E** and **f** are residual matrices containing unstructured variability. The number of latent variables controls the complexity of the model and must be tuned and it is selected following a 10-fold cross-validation procedure.

The regression stage, as described above, is commonly applied in many practical scenarios. However, in the context of big data and Industry 4.0, building a model that contains all the available features is computationally very expensive (including the successive refinement stages) and prone to overfitting. WiSe circumvents this drawback by employing a filter to remove clearly noisy and irrelevant features. Therefore, the benefits are two-fold: regression models can be developed more efficiently, and the initial filtering stage is not influenced *a priori* by the regression method, broadening the range of methods that can be tested in this second stage.

5.3 Datasets

This section provides details regarding the simulated and industrial datasets used to test the proposed approach. Three datasets were simulated, mimicking different types of processes: a sparse process, a latent variable process, and a scenario where the relationship between features and the response is non-linear. The conditions tested are summarized in Table 5.2. The level of sparsity is defined as the ratio between relevant and total number of features (i.e., relevant plus irrelevant features). In all the simulated cases tested, there are 20 relevant features; therefore, there are 2000 features with a sparsity level of 1%, 400 features for a sparsity level of 5%, and 200 features for a sparsity level of 10%. The additional irrelevant variables are drawn from a multivariate normal distribution with a low degree of mutual correlations $\left(\rho_{i,k} = 0.2\right)$ and concatenated to the set of relevant ones. Signal-to-noise ratio (SNR) is defined for the response variable as the quotient between the variance of the noiseless values and the variance of the noise, and three SNR levels are tested as shown in Table 5.2. In all simulation scenarios, 10000 samples were generated to test the performance of the different alternatives considered. The next subsections provide further details regarding the simulated processes as well as the real industrial dataset.

Table 5.2. Simulated scenarios: all combinations of datasets, sparsity, and Signal-to-Noise (SNR) levels were tested.

Simulated Dataset	**Sparsity Level (%)**	**SNR**
sparse process, latent variable process, non-linear	2, 5, 10	10, 15, 25

5.3.1 Simulated Dataset: Sparse Process

The first simulated dataset is obtained from a sparse process where only a few features are directly related to the response, while the majority are irrelevant and do not contain any predictive information. This dataset simulates a scenario found in big data and Industry 4.0 where the number of measured variables is increasing and becomes very large, but the set of variables most related to the response, i.e. the variability drivers, remains the same as before, and is rather small. One often observes that the ratio between relevant and irrelevant features tends to zero as the number of measurements increase, which in turn requires the use of FSM to effectively build regression models. The sparse model simulated in this study is presented in eq. (22):

$$\mathbf{y} = \tilde{\mathbf{X}}\mathbf{b} + \boldsymbol{\varepsilon} \tag{22}$$

where $\tilde{\mathbf{X}}$ is an $n \times (m+1)$ extended matrix of features containing an additional column of 1's to accommodate the intercept, $\mathbf{b}$ is a $(m+1) \times 1$ column vector of regression coefficients and $\boldsymbol{\varepsilon}$ is a $m \times 1$ column vector of residuals, which follows an *i.i.d.* $N(0, \sigma^2)$ distribution. Samples are drawn from a multivariate normal distribution with a low correlation coefficient between features $(\rho_{i,k} = 0.2, i \neq k)$ and the regression coefficients were divided into three levels of importance. The regression coefficients from the most important features were drawn from an uniform distribution, corresponding to one-third of the relevant features, $b_{1,\ldots,6} \sim \mathrm{U}(5,7)$; the second level of importance also contains one-third of the features and was drawn from another uniform distribution with lower magnitudes, $b_{7,\ldots,12} \sim \mathrm{U}(3,5)$; the remaining regression coefficients present even lower magnitudes, $b_{13,\ldots,20} \sim \mathrm{U}(2,3)$.

5.3.2 Simulated Dataset: Latent Variable Process

Latent variable datasets are commonly found in industrial contexts where many correlated process variables are measured. Mass and energy conservation laws, control loops and operation practices prevent variables to change independently; therefore, although the number of measurable variables is large, their variability is dictated by only a few latent variables, which are furthermore unobservable. The latent variable model was presented in eq. (20) and (21), and one can note that the variability of the features and the response variable is a consequence of variation in a low dimensional vector of scores (**T**) that correspond to the unobservable sources of variation. Nevertheless, many other features are measured that are neither related to the predictors nor to the response, and the first filtering stage of WiSe (see Section 2.1) is expected to remove them. The simulated latent variable model contains 4 latent variables $(a=4)$, and the scores (**T**) are obtained from a multivariate normal distribution with zero mean and no correlation between features. The model loadings are obtained from a singular value decomposition of a random matrix with dimensions $(N \times 4)$.

5.3.3 Simulated Dataset: Non-linear Relationships

The third simulated dataset represents the scenario where the relationship between features and response variables is non-linear. In CPI, one can note that sometimes outputs of interest are non-linearly related to the inputs (e.g. product concentration and temperature). The non-linear model considered here is a piece-wise model where the relationship between a feature and the response is dependent on the level of the feature. For each feature, three regions are considered, corresponding to a linear trend, a non-linear quadratic function and a region where the response is independent of the feature. An example is presented in Figure 5.2, showing a typical relationship between a feature and the response variable. This relationship mimics a scenario where a feature is only important after a threshold is attained and can simulate, for instance, what happens when a controller is no longer able to maintain a controlled variable within its normal operating range, which in turn, starts influencing quality.

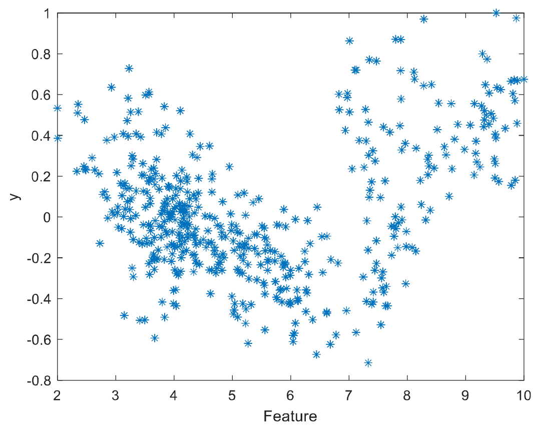


Figure 5.2. Example of the non-linear relationship between one feature and response variable in the simulated dataset.

5.3.4 Industrial Dataset

The industrial dataset was obtained from a refinery distillation system at Dow Chemical. This dataset contains 12,000 samples that are measured hourly for 61 features, spanning the period from the year 2011 to 2013. The process has four operating modes, which depend on the production priority. Figure 5.3 presents the first two principal components obtained from PCA, and the different operating regimes are identified. The original analysis (Lu, Castillo et al. 2014) concerned the development of a soft sensor that would be able to provide hourly estimates of the response variable. However, in this work, the dataset is analyzed in the context of identifying relevant features, followed by regression model building. This analysis is a first assessment of the potential of the features to predict the response and serve as a baseline benchmark, that could be later improved (Lu, Castillo et al. 2014).

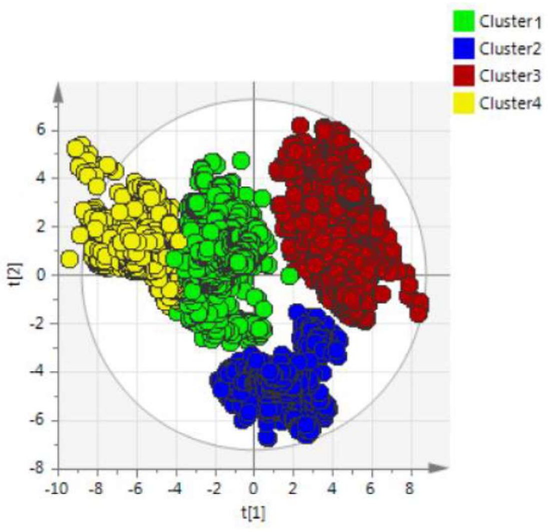


Figure 5.3. The first two principal components for the industrial dataset, highlighting the different operating regimes.

5.4 Comparison framework

For the purpose of comparing the performance of the different approaches tested in this work, a comparison framework was developed that take into consideration the problem specificities. The benchmarks correspond to the use of the second stage wrapper/embedded methods without the first stage of filtering (option "no filter"); this is the fairest way of assessing the advantage of WiSe, as the use of the classical methods without any feature selection at all (i.e., without any of the two stages) would put them in a clearly unfavorable position. The comparison framework was designed so that it is able to generate results that can be analyzed at different levels of aggregation. At a finer scale, the performance of the FSM methods is assessed in the scope of each specific testing scenario (a scenario constitutes a case study where the data generating mechanisms are fixed and only sampling variability is present). If one focuses on one scenario only, the performance results are easy to analyze and the identification of the most suitable method is rather straightforward. However, as the number of scenarios increases, it becomes cumbersome to extract tendencies. On the other hand, considering only the mean performance would highly bias the conclusions. Therefore, in order to aggregate results in a meaningful fashion, a key performance indicator (KPI) was developed that allows the analysis of general trends characterizing the methods' performance over a variety of scenarios. The KPI is based on the pairwise statistical differences between methods tested under the same circumstances. More specifically, all methods are pairwisely compared in each scenario with resort to a paired *t*-test. The paired *t*-test is used since performance is assessed in the same testing conditions and the number of comparisons made is limited, mitigating the risk of excessive false alarms. In each pairwise comparison, a method is deemed the "winner" when its performance is statistically superior to another method under comparison, receiving a score of 2 points; when the performances of two methods are not statistically different, both receive 1 point; and a method whose performance is worse in a pairwise comparison does not receive any points. The KPI for each FSM is obtained by summing all points collected from all pairwise comparisons where it participates. A high value of KPI indicates a method presenting a superior performance compared to others, whereas a small KPI indicates a method whose performance is comparatively poor in most scenarios. One should note that the definition of the KPI requires a measure of performance and for feature selection application, two metrics are often considered: a first metric based on the ability of a FSM to identify relevant features correctly, and a second metric based on the ability to predict a response variable accurately. Both of these performance metrics are described next.

5.4.1 Key Performance Indicator based on relevant features selected (KPI_{rf})

The key performance indicator based on relevant features selected (KPI_{rf}) measures the ability of a FSM to detect relevant features from a set of available predictors, i.e., to quantify the agreement between *a priori* knowledge regarding which features are important and those selected by a given FSM. In practice, and particularly in big data applications with many features, priori knowledge regarding the full set of relevant features is limited or incomplete. Therefore, KPI_{rf} is most suitable for simulated datasets where the data generating mechanisms are completely known from the simulation settings. To compute KPI_{rf}, one must first define the performance metric and the family of F_{β} scores is commonly utilized:

$$F_{\beta} = \frac{\left(1+\beta^{2}\right)\times TP}{\left(1+\beta^{2}\right)\times TP+\beta^{2}\times FN+FP} \tag{23}$$

where *TP* is the number of true positives (relevant features that are selected), *FN* is the number of false negatives (relevant features that are not selected), *FP* is the number of false positives (irrelevant features that are selected), and β is a parameter that controls the relative penalization of *FP* and *FN*. As described in Section 4.2, the aim of the WiSe approach is to first remove noisy features while keeping all the important variables. Therefore, the penalization for incurring in *FN* should be higher than for *FP*. In this context, the F_2 score achieves a good balance between *FP* and *FN* and it is the metric used for computing KPI_{rf}. Note that KPI_{rf} is utilized for the simulated datasets only since the F_2 score requires access to knowledge about the relevant predictors, which are completely known in a simulation case study.

5.4.2 Key Performance Indicator based on prediction performance (*KPI*$_{pp}$)

In industrial datasets, relevant predictors are unknown *a priori* and it is the goal of a FSM to identify them. However, validating the features selected by a FSM becomes a complex task due to unavailability of the data generating mechanisms that could be used as ground truth. An alternative is to consider prediction performance as the metric to compare different methods, and assume that models with better prediction performance (i.e. low prediction errors) also contain features that are more relevant. Therefore, the key performance indicator based on prediction performance (KPI_{pp}) is computed based on the root mean squared error in double cross-validation $\left(RMSE_{dcv}\right)$ metric:

$$RMSE_{dcv} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \tag{24}$$

Thus, besides the application of a FSM to remove noisy predictors, one must also use a regression model to predict the response variable based on available features. This constitutes the aforementioned two-stage approach defined in WiSe (see Section 4.2), where regression methods are utilized in the second stage of variable selection. $RMSE_{dcv}$ assess the prediction error in double cross-validation, a procedure that is more robust than single cross-validation. Double cross-validation contains an inner cycle where models are built and an outer cycle where their performance is assessed. The dataset is split into a training and validation sets. The training set is used for model building and to optimize hyper-parameters with 10-fold cross-validation. Once models are optimized, they are tested in the validation set, completing one iteration of double cross-validation. Since the results are dependent on the initial split into training and validation sets, the outer loop of double cross-validation consists in repeating this splitting step many times, resulting in a distribution of performance in the validation data and serve as the base metric for KPI_{pp}. Further details of double cross-validation can be consulted in the literature (Rendall, Pereira et al. 2016) and will be better detailed in the next chapter (Section 6.2.3). For comparing the methods, 50 iterations of double cross-validation were conducted.

In summary, the $RMSE_{dcv}$ is utilized as the metric to compute KPI_{pp} following the scheme based on pairwise comparisons using the *t*-test. In order to explore a broader range of interactions between filters and regression methods, FSR, LASSO, and PLS are utilized as representative methods.

5.5 Results and Discussion

This section presents the results obtained by applying the proposed filters to the simulated and industrial datasets. In the simulated datasets, the relevant features are known *a priori* and the KPI_{rf} is adopted to measure the effectiveness of FSM. Afterwards, KPI_{pp} is used to assess the benefits of selecting a good initial set of features for the regression task. In the industrial dataset, no *a priori* information is available regarding relevant features and only KPI_{pp} is adopted to compare the prediction performance of the different filters.

5.5.1 Simulated Datasets

The performance of the filters, regarding their ability to correctly detect relevant features, is presented in Figure 5.4. Since KPI_{rf} is stratified by dataset and sparsity level, its mean value $\left(\overline{KPI_{rf}}\right)$ over all noise levels is presented in Figure 5.4. The variability for different noise levels is small enough that the mean value is a good representative of the filters' performance.

The analysis of Figure 5.4 suggests that the results are dependent on the simulated conditions. More specifically, filters such as Pearson and Spearman have very comparable performances and obtain the highest amount of points in most scenarios. This stems from the fact that the majority of the simulations correspond to scenarios where the relationship between features and response variable follow a linear pattern; therefore, linear filters are expected to perform better overall. On the other hand, SU presented the best performance in the non-linear dataset and is only surpassed by other filters in very low sparsity levels.

Another interesting point that can be observed in Figure 5.4 is that combining filters is not the best alternative, even though this approach has the potential to detect both linear and non-linear relationships. The filter resulting from combining Pearson and Spearman filters obtains the best performance for the scenario with a non-linear dataset and very small sparsity level (1%). For other sparsity levels, filter combinations present performances similar to the remaining filters, although SU is the clear winner. Nevertheless, combining filters is a promising alternative that should be considered in practical scenarios because, in CPI, the type of relationship between features and response variable may not be known in full detail.

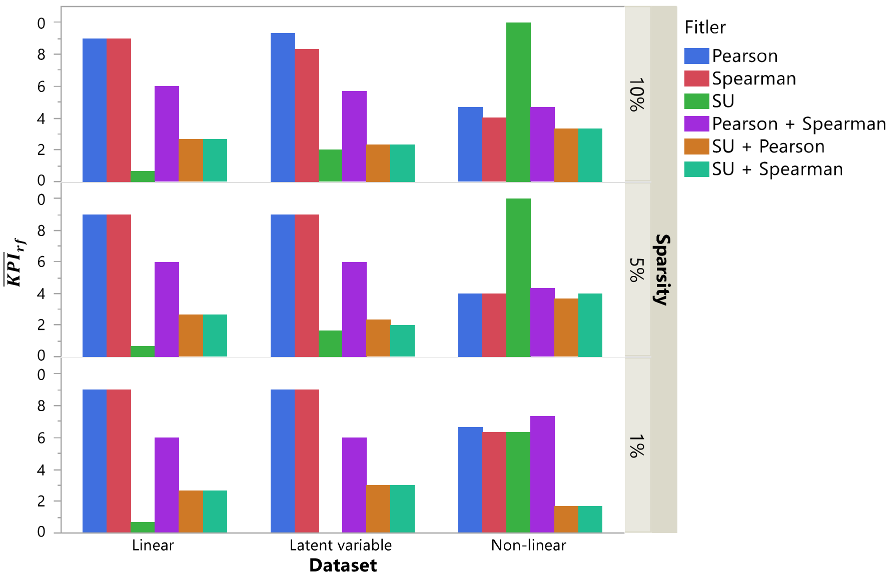


Figure 5.4. Performance of the filters for all scenarios tested.

The performance, as specified by KPI_{rf}, is computed with the F_2 score. This score improves when a filter outputs the correct class for each feature (true positives and true negatives) and is penalized by incorrect classification of the features (false positives and false negatives). The main objective of applying the filter prior to developing a regression model is to reduce the number of features, facilitating the iterative process of model development. Nevertheless, the filtering stage should minimize the number of relevant variables that are removed (i.e. false negatives). In order to assess how well the tested filters are able to identify relevant features, Figure 5.5 presents the number of true positives in each tested scenario. Analyzing Figure 5.5, one can verify that most filters are able to pick relevant variables and that, in each scenario, there is at least one of the filters is able to select 18 of the 20 relevant features. For instance, the combination of linear and non-linear filters is able to pick relevant variables in most scenarios in a very stable fashion, suggesting synergies are obtained with such combinations. Nevertheless, from the perspective of selecting important features, Figure 5.5 shows that most filters have a rather good general performance and implies that the differences observed in Figure 5.4 are largely dictated by the ability of the filters to eliminate irrelevant variables. In other words, most filters are detecting the truly relevant features and the observed differences in performance are due to the number of false positives. As the minimizing false positives is a secondary goal of the filters (see Section 2), these results (Figure 5.4 and Figure 5.5) demonstrate the validity of the WiSe approach in utilizing filters in order to remove irrelevant predictors in a regression setting. Any regression method developed over the resulting smaller set of features would be able to provide better predictions of the response variable, and better estimates of the model parameters. This is particularly useful for the latent variable dataset, where collinear features are not necessarily eliminated and can be used to estimate the underlying latent model. Since the performance of the filters is largely dependent on the number of false positives, Figure 5.6 presents the number of false positive of each filter in all simulated scenarios. Analyzing Figure 5.6, one can note that the general trend is that combining filters result in higher false positives compared to single filters. Using the minimum *p*-value to assess feature relevance will generate more false positives, but as previously observed, selects the most important features in a consistent manner. Another important point worth highlighting is the benefit of using FDR, which result in a smaller number of false positives. For instance, using the traditional approach of specifying a constant threshold *p*-value of 0.05 would result in approximately 100 irrelevant features being selected for the low sparsity level (2000 total features). The FDR decreases the false positives to about 45 features.

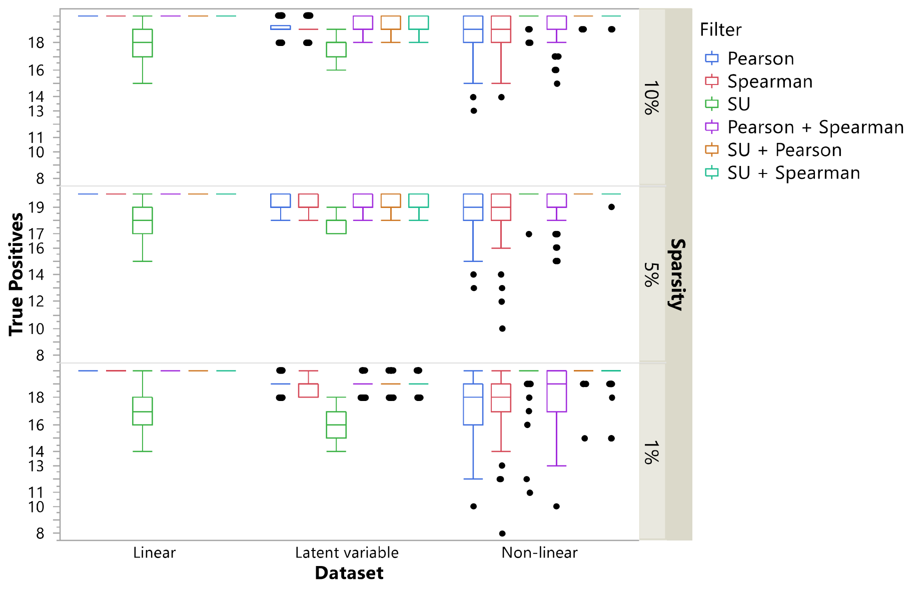


Figure 5.5. The number of relevant variables selected for all scenarios tested.

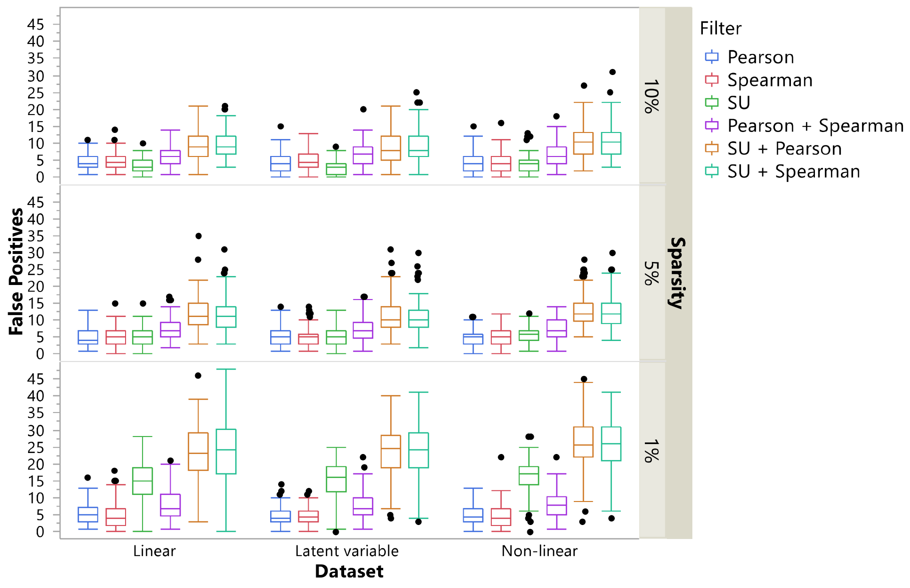


Figure 5.6. The number of irrelevant features selected by each filter in each scenario.

The previous results focused on the filters' ability to detect important features, and their performance was a compromise between true positives and false positives. The second dimension of analysis concerns assessing whether the first stage of WiSe can improve prediction performance or not. Therefore, Figure 5.7 presents the filters' performance in terms of KPI_{pp}, merging both stages of WiSe. Additionally, the case where no filters are utilized is presented for comparison. Figure 5.7 shows that the prior selection of potentially good features increases the prediction performance of all regression methods tested. Although the results are averaged over all dataset types and noise levels, there is a clear tendency for obtaining improved prediction performance when utilizing filters *a priori*. It is worth noting that FSR and LASSO conduct feature selection automatically, however, they benefit from building models with a smaller (filtered) initial set of features. The advantage of using WiSe is that model building step is facilitated, especially in the low sparsity level where 2000 features are available. The results in Figure 5.7 show that the Pearson and the Spearman filters tend to be the ones leading to better performance in most of the simulated scenarios. Nevertheless, other filters present acceptable performances and are better alternatives than using no filters. For instance, the combination of SU and Pearson seem to be suitable for the 10% sparsity level and is the third best method in that scenario. Overall, the results of this simulated study show that there is no filter that is superior in all case studies. Instead, the best filter is case dependent and one should test at least some filters in order to be confident in the set of selected features. Pearson, Spearman and SU filters by themselves presented generally superior performance. Combining SU and Pearson is also recommended in order to have a broader coverage of FSM.

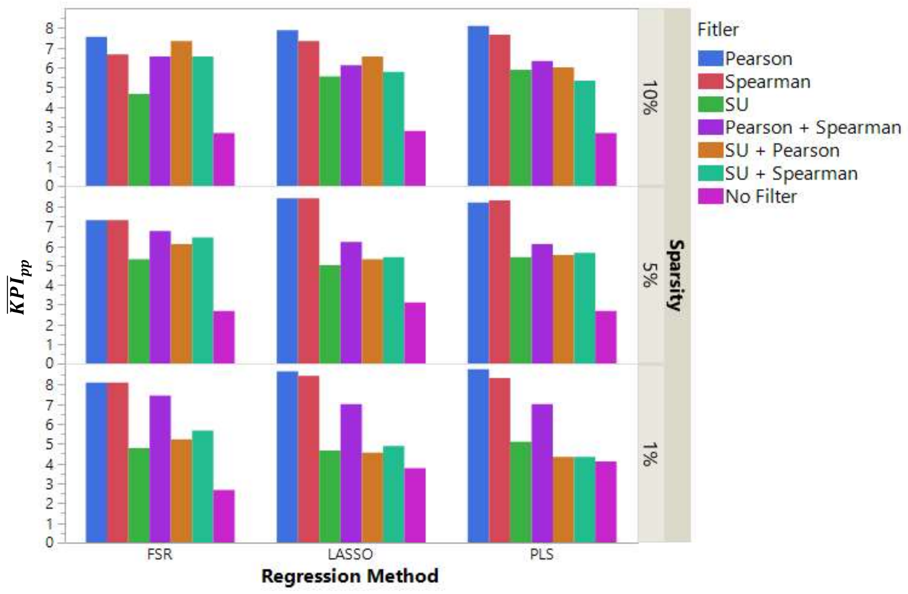


Figure 5.7. Prediction performance of filters combined with regression methods.

5.5.2 Industrial Dataset

The industrial dataset illustrates the advantages of WiSe in a real-world application. As the relevant features are not known *a priori*, we resort to prediction performance in order to assess the effectiveness of the filters. The implicit assumption is that filters leading to lower prediction errors are also selecting the correct features for model building. The KPI_{pp} is presented in Figure 5.8 for all filters and, additionally, the benchmark situation where no feature selection is applied. Figure 5.8 shows that selecting the appropriate filter is critical in terms of obtaining good performance. For the industrial dataset, it is observed that SU provides the best solution for predicting the test set. The combinations of SU and Spearman, and SU and Pearson are also interesting and provide good results when a PLS model is employed. Interestingly, utilizing no filters is better than some filters when considering FSR and LASSO models. This can be justified by the small number of variables available, mitigating the importance of removing irrelevant features in the first stage. Another reason is that FSR and LASSO are able to remove variables automatically during model building, whereas the standard PLS model does not apply any scheme for feature selection. Therefore, by not applying any feature selection methods in the PLS model, a worse performance is obtained. Nevertheless, SU shows the best results and is the recommended approach for this dataset. Besides prediction performance, the percentage of removed features is also an interesting metric to assess the decrease in the dimension of the feature space. Thus, Figure 5.9 shows the percentage of features removed at the first WiSe stage. One should note that when no filters are applied, all features proceed to the regression stage. The main point is that the application of filters reduces the number of features: single filters are less conservative and remove around 11% of the original features while combining filters remove only 3% of the features. Although these are small percentages compared to those obtained in the simulated study, it shows that even for smaller datasets, combining filters and wrapper methods is beneficial.

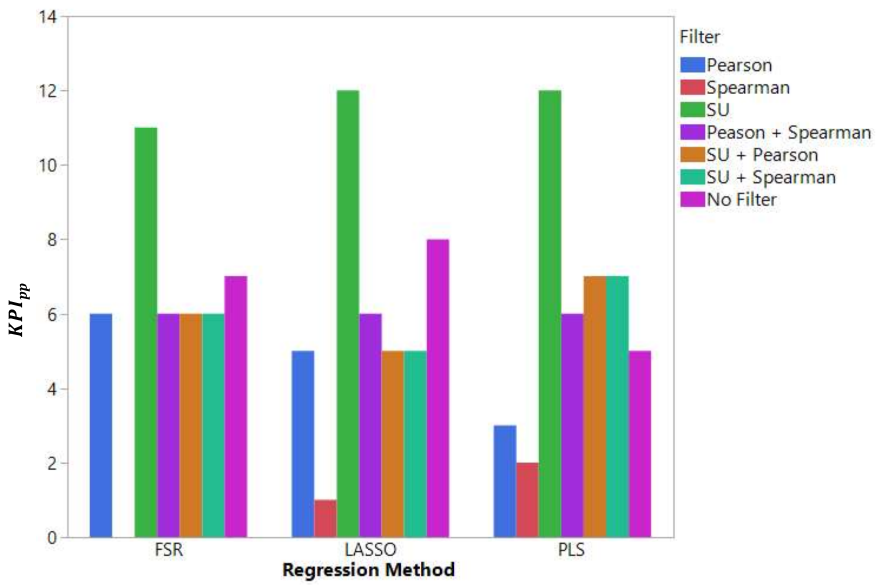


Figure 5.8. Test performance of different filters in an industrial dataset. Various interactions between filter and regression models are also considered.

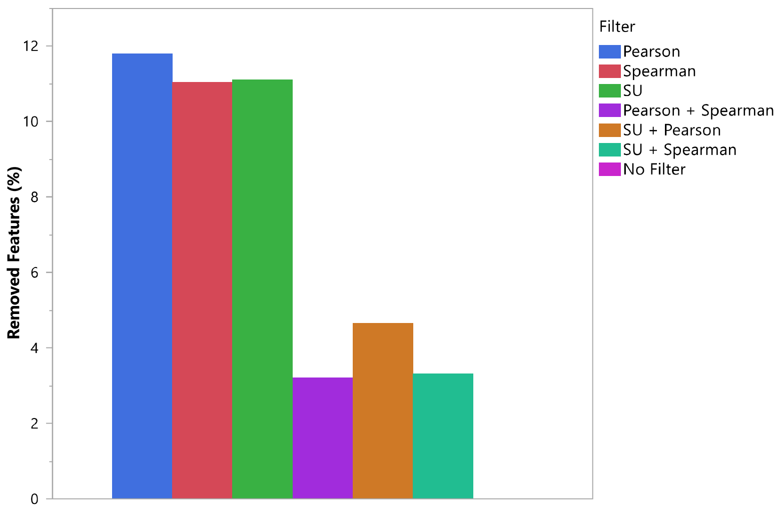


Figure 5.9. The percentage of features that each filter eliminates at the first filtering stage.

The simulated datasets and the industrial case study represent datasets typically found in CPI. Filters are efficient screening methods, allowing the removal of noisy and irrelevant features, whereas the relevant ones are retained for further analysis. This constitutes the first step in the WiSe approach and aims to reduce the dimensionality of the feature space. In the second stage, WiSe focuses on building regression models where further feature selection may occur. The results obtained in this work show the advantage of using WiSe for variable selection in regression problems and suggest its potential to be employed in big data scenarios where the number of features is large, but the relevant ones tend to be scarce. In these very sparse datasets, the application of WiSe should be even more advantageous, facilitating model development by identifying a set of features that tend to be adequate for model building. The results obtained showed that single filters (Pearson, Spearman, and SU), as well as the combination of SU and Pearson, constitute an initial good set of filters that should be tested in regression problems.

Chapter VI
Predictive Analytics Comparison Framework

6 Predictive Analytics Comparison Framework

In the modern era of big data and Manufacturing 4.0, there is a growing interest in using advanced analytical platforms to implement predictive modeling methodologies that can take advantage of the wealthy of data available. Typically, practitioners have their own favorite methods to address the modeling task, as a result of their technical background, past experience or software available, among other possible reasons. However, the importance of this task in the future justifies and requires more informed decisions about the predictive solution to adopt. Therefore, a wider variety of methods should be considered and assessed before opting for one of them. In this context, this chapter presents the predictive analytics comparison framework (PAC). PAC contains the ingredients necessary to be employed for assessing and comparing the predictive performance of regression methods in a robust and efficient manner, and can provide insights into the characteristics of the collected dataset and the data-generating mechanisms. As Chapter III and IV were concerned with feature generation and Chapter V dealt with selecting informative features, this chapter emphasizes the predictive modeling stage where the selected features are employed for predicting quality parameters of interest. This chapter is divided into five sections that describe PAC in detail. The first section provides a general overview of PAC and its importance in modern industrial settings, where practitioners face the challenging task of selecting a suitable regression methodology for their application. The second section describes PAC's role in data-driven model building and details its main components. The third section presents four case studies where PAC was applied, demonstrating its effectiveness and suitability. These results are further discussed in the fourth section, while the fifth section provides some final remarks regarding the PAC's role in real-world applications.

6.1 PAC – Overview

With the emergence of Manufacturing 4.0, advanced predictive analytics, and regression methods in particular (Draper and Smith 1998, Hastie, Tibshirani et al. 2001, Chatterjee and Hadi 2015), have been attracting considerable interest in many areas of science and in different application contexts, such as market analysis (Bollen, Mao et al. 2011, Chen, Chiang et al. 2012, Erevelles, Fukawa et al. 2016), manufacturing (Lee, Lapira et al.

2013, Li, Tao et al. 2015), food and beverage (Nørgaard, Saudland et al. 2000, Cozzolino, Kwiatkowski et al. 2008, Rendall, Pereira et al. 2016), pharmaceutical (Hoffman, Cho et al. 1999, Moţ, Soponar et al. 2010, Shams, Ajorlou et al. 2015), petrochemical and chemical (Braga, dos Santos Junior et al. 2014, Pinheiro, Rendall et al. 2016), etc. This interest is largely motivated by the growing availability of data collected from fast and informative process sensors as well as large databases that facilitate their storage, integration and retrieval. In this context, research on predictive methods has been driven by the need to develop suitable techniques equipped with the necessary methodological, algorithmic and computational components that allow them to cope with the prevalent characteristics observed in the collected datasets, such as high-dimensionality (Johnstone and Titterington 2009, Martens 2015), collinearity (Naes and Mevik 2001, Chong and Jun 2005), sparsity (Rasmussen and Bro 2012), non-linearity (Marini, Bucci et al. 2008), non-stationarity (Aguado, Ferrer et al. 2006), missing data (Walczak and Massart 2001, Arteaga and Ferrer 2002), among others. This effort led to the proliferation of a large number of methods and variants, spread through the vast technical literature, making it very difficult for practitioners to decide which methods best suit their particular application scenarios. Prior knowledge could be useful for selecting a suitable set of regression methods to adopt, but most often the particularities of each case study and the lack of more detailed information make it impossible to rule out other methods from the pool of candidates. Therefore, the prevalent and most realistic, honest, and unbiased perspective one often is forced to accept is the lack of absolute certainty about the best class of predictive methodologies to use (not to speak, the best method to use). We call this a "data-rich/knowledge poor" scenario, given the availability of data resources and, simultaneously, the lack of consistent information on how to derive the best predictive models from them.

In this context, comparison studies are unavoidable and represent a reliable way to test and select regression approaches that are eligible for predicting the response variable of interest. However, these studies take considerable time to carry out and require resources of knowledge, software and time that many users do not have at their disposal or simply cannot afford. Even for the few cases where they were conducted, they still present limitations in the number of methods tested, usually less than 5 (Kim 2008), leaving some classes absent from analysis (Mahesh, Jayas et al. 2015, Sharif, Makowski et al. 2017), as well as in the way the comparison is done (namely in the accuracy and robustness of the approach and metrics used). Therefore, the predictive analytics comparison framework (PAC) is here developed for problem-specific methods screening. PAC is able to speed up the selection process, while securing a rigorous and robust assessment of the methods under analysis and a proper use of the data available. This framework was developed and tested in different contexts (chemical industry, biofuels, drink and food, shipping industry, etc.), leading to consistent results that improved the base predictive solution adopted, with a very short implementation cycle. PAC was designed to help practitioners identifying the approaches with higher performance potential for their particular applications, using a structured, rigorous and informative methodology: the methodology is structured, because it is composed of four integrated components (see below); it is rigorous, because the comparison is conducted with a state of the art double cross-validation method that generates information for conducting formal statistical hypothesis tests which finally lead to a sound assessment of the methods' relative performances; finally, it is informative, because PAC will not only provide a report with results about the hierarchy of methods that best suit the application under analysis, but also present which predictors are more relevant in each class of methods, contributing to enrich the knowledge about the problem under analysis, among other interpretational information on the structure of data.

More specifically, PAC is composed by four components (Figure 6.1): i) analytics domain; ii) data domain; iii) comparison engine; iv) results report. The analytics block encompasses a rich variety of predictive approaches to be scanned in each application context under analysis. It establishes the analytics domain of assessment or comparison. The variety of methods considered was carefully considered, and are segmented in four classes: variable selection, penalized regression, latent variable, and tree-based ensemble methods. Each class of methods has different *a priori* assumptions regarding the data generating mechanism and their suitability depends on data characteristics such as the level of sparsity (in a sparse problem, only a few variables have predictive value), collinearity (existence of associations among regressors), modularity (presence of block-wise structure in the regressors) and the underlying relationship between predictors and response variable (if it is linear or some non-linearity is present).

The data domain regards the dataset that will be used to conduct the comparison study and that determines the inference-basis of the study. It should be carefully considered because the results will be critically dependent on what is inserted into this component. In predictive problems, attention should be paid to the existence of clustered data, multiple processes/ phenomena superimposed, transcription errors, outliers, signal to noise ratio in the response, etc. – this module is subject to the well-known GIGO principle of computer science ("garbage in garbage out"), which impacts the entire PAC framework.

The comparison engine performs a robust assessment of the predictive capabilities of each representative in the analytics domain, using the data domain as the inference basis.

The computations are conducted in such a way as to potentiate an optimized generation of performance metrics in the results block. The performance of each regression method is assessed by the root mean squared error of double cross-validation $\left(RMSE^{dcv}\right)$. Pairwise comparisons are conducted with resort to formal statistical hypothesis testing, in order to incorporate the variability of results in the analysis.

The last block of the PAC framework is the results report, where the final Key Performance Indicators (KPI) for the methods under analysis are provided, as well as additional information for interpreting the model, according to the nature of each class of methods (e.g., suitable measures of predictors' importance). Furthermore, one can also make inferences regarding the structure of the dataset, namely its sparsity and collinearity levels based on the profile of important predictor variables and the relative performance of the different methods.

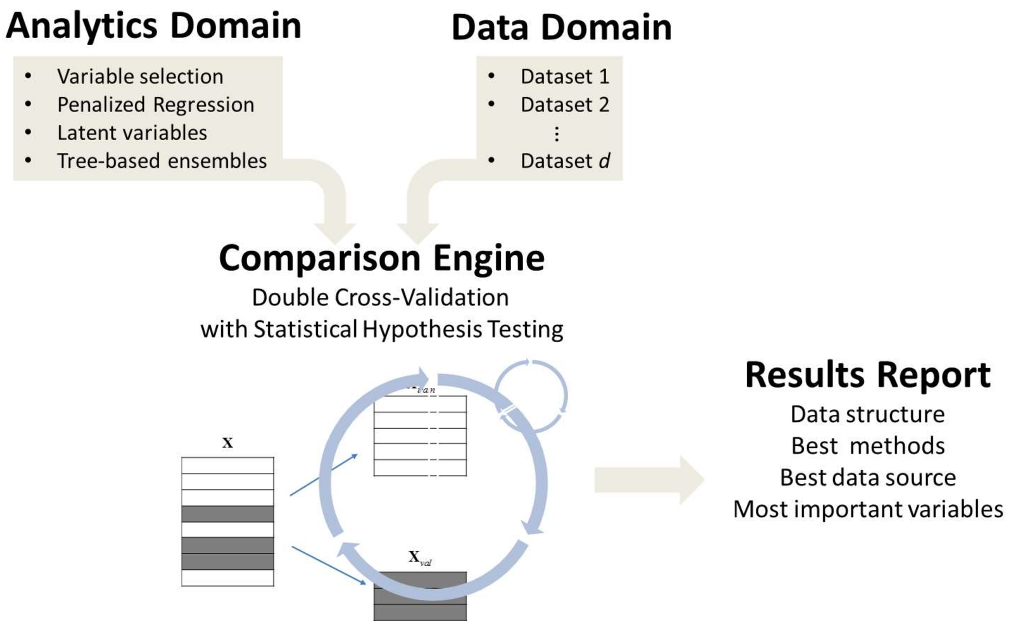


Figure 6.1. The PAC framework and its modules.

6.2 PAC – Detailed Description

The major components of PAC and the sequence of their application as programmed in the computational code are summarized in Figure 6.1. Each one of these modules will be described in the following four subsections, where details are provided on the methods used and how they operate. However, the rest of this subsection is dedicated to better frame the scope of PAC and its role in the analysis workflow for model screening and development.

PAC was developed in the computation platform Matlab, with the purpose to establish a systematic and robust framework for model screening and development. It aims at increasing the efficiency of the general workflow for model development, as depicted in Figure 6.2. In this workflow, the user first defines the relevant dataset for analysis and initiates the process. Then, an exploratory data analysis stage should be conducted where practitioners become familiar with the main variability patterns in data, assess the distribution of the samples and identify issues to be clarified or handled case by case. Following the exploratory data analysis stage, pre-processing is an optional step that, for some applications, may have a significant impact on the quality of the results, such as in multivariate calibration or soft sensors using spectral data (Martens and Naes 1989). Therefore, this step should be carried out whenever it may bring added value to the specific application under consideration. The existence of missing observations can also be considered during the pre-processing stage, where one has to decide between using some missing data imputation method (Walczak and Massart 2001, Arteaga and Ferrer 2002, Little and Rubin 2002) or simply discard samples containing missing observations. After pre-processing, PAC is applied to speed up model screening and development and to quickly extract insights about the class(es) of methods with better predictive capabilities, as well as further information on the relative importance of the predictors. New cycles of PAC can be iteratively conducted using the information collected from previous runs. For instance, the identification of irrelevant predictors and influential outliers may lead to a subsequent analysis in which these predictors and outliers are removed. When the user is satisfied with the results obtained, a fine tune post-optimization of the selected method(s) can be conducted, if necessary. For example, one may consider in this stage a more exhaustive source of the hyper-parameters to use in the best model, or to develop combined or aggregated predictive models. Post-optimization of the best performing methods often improves prediction performance and this may be relevant for some applications, while for others the interpretation insights acquired with PAC are enough to conclude the model development process.

In its current state, PAC is designed to handle problems where only one response variable is available. However, the proposed framework can also be applied to the case of multiple response variables by considering each response separately. This approach can be suboptimal w.r.t. to interpretation in the presence of correlated response variables, but not in terms of prediction accuracy; however, even in these contexts, PAC still provides useful insights as shown in a case study for predicting physicochemical properties of oils from infrared spectra (Pinheiro, Rendall et al. 2016). The iterative cycle suggested in Figure 6.2 is guided by improvements in prediction accuracy, assessing, in a rigorous and unbiased fashion, the performance of each method. Therefore, model interpretability is not the primary criterion considered and is limited by the tools available for exploring variables' importance for the top method(s). In some applications, model interpretability might be more relevant than prediction performance and only methods that provide suitable tools for assessing predictors importance should be considered in this case. The interested reader is referred to the bibliography cited for each regression method in order to assess their potential in terms of model interpretability, as well as to the work of Kvalheim, Arneberg et al. (2014).

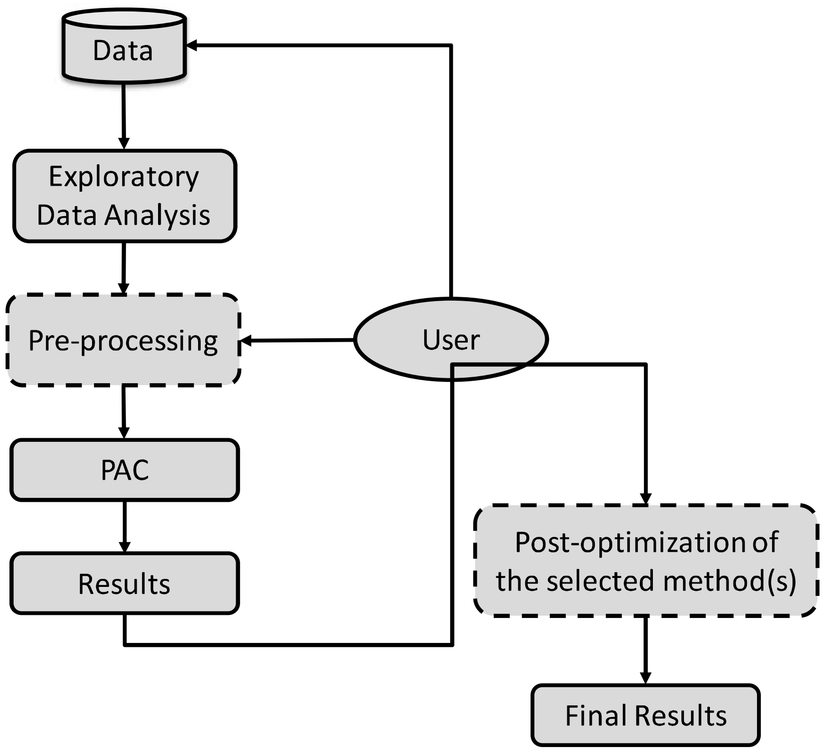

Data
Exploratory Data Analysis
Pre-processing
User
PAC
Results
Post-optimization of the selected method(s)
Final Results

Figure 6.2. The role of PAC in the scope of the general workflow for model screening and development.

6.2.1 Analytics Domain

This component comprises the set of regression approaches to be considered and compared w.r.t. the dataset inserted in the data domain component. The methods belonging to the analytics domain were carefully considered and result from a critical analysis of the technical literature. The literature on regression methods is extensive and has been steadily increasing with new methods being proposed to handle a wide range of dataset characteristics likely to be found in Manufacturing 4.0 applications. However, the number of methods selected to compose the Analytics Domain was enforced not to be very large, in order to keep the entire approach manageable by practitioners (this implies and justifies the absence of some regression approaches from this domain). Still, 13 methods were selected –a high number taking for reference other studies available in the literature– which are grouped into four classes according to their mutual methodological and algorithmic affinities: variable selection methods (Section 6.2.1.1), penalized regression methods (Section 6.2.1.2), latent variables methods (Section 6.2.1.3) and tree-based ensembles (Section 6.2.1.4). Each class contains methods that share similar prior assumptions regarding the data generating mechanisms. Other methods (and perhaps classes) can always be added to PAC in the future if found to be particularly pertinent, but those considered at the present stage already provide a good coverage of the analytics space, allowing users to successfully address a wide variety of applications.

Several of the regression methods considered provide different solutions to limitations presented by the ordinary least squares method (OLS) (Draper and Smith 1998, Martens and Mevik 2001, Naes and Mevik 2001, Reis and Saraiva 2004, Reis and Saraiva 2005, Montgomery, Peck et al. 2012). As they can also be traced back to OLS by changing one or more parts of its formulation, a very brief reference to this technique is warranted. Let us consider the base linear regression model in matrix format:

$$\mathbf{y} = \tilde{\mathbf{X}}\mathbf{b} + \boldsymbol{\varepsilon} \tag{25}$$

where $\mathbf{y}$ is $N \times 1$ column vector of response values, $\tilde{\mathbf{X}}$ is the $N \times (p+1)$ extended matrix of regressors containing an additional column of 1's to accommodate the intercept, $\mathbf{b}$ is the $(p+1)\times 1$ column vector of regression coefficients and $\boldsymbol{\varepsilon}$ is $N \times 1$ column vector of residuals, which assumedly follow an *i.i.d.* homogeneous distribution, not necessarily Gaussian. The OLS regression coefficients are estimated by minimizing the sum of squared residuals, as described in eq. (26):

$$\hat{\mathbf{b}}_{OLS} = \underset{b=[b_0 \dots b_p]^{\mathrm{T}}}{\arg\min} \left\{ \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2 \right\} \tag{26}$$

where $\hat{\mathbf{b}}_{OLS}$ is the estimated vector of regression coefficients, $\hat{y}_i$ is the i^{th} observed response value and $\hat{y}_i$ is the corresponding model prediction, given by $\hat{y}_i = b_0 + \sum_{j=1}^{p} b_j x_{i,j}$.

For the OLS case, there is an analytical solution, and the estimation problem can be recasted into one of solving a linear algebraic system of equations: $\left(\tilde{\mathbf{X}}^{\mathrm{T}}\tilde{\mathbf{X}}\right)\mathbf{b}_{OLS} = \tilde{\mathbf{X}}^{\mathrm{T}}\mathbf{Y}$. This classic methodology experiments strong limitations in high-dimensional collinear systems (where the inverse of $\left(\tilde{\mathbf{X}}^{\mathrm{T}}\tilde{\mathbf{X}}\right)$ becomes highly unstable or impossible to obtain), and therefore will not be part of the analytics domain. Several of the methods presented in the following sections represent modifications of this base formulation to circumvent some of its limitations.

6.2.1.1 Variable Selection

Variable selection methods assume the existence of a sparse structure in the regressors and/or the presence of redundant information. In other words, some predictor variables contain irrelevant information regarding the response (sparse structure) or some level of mutual association (collinearity) that makes them dispensable when others are present. Therefore, in order to build a suitable regression model, only a subset of the original regressors should be considered, and all those variables that do not carry additional predictive value, discarded (either because they are irrelevant to predict the response, or they are associated to others already in the model, and therefore do not bring any additional predictive power to the model). The inclusion of irrelevant variables in the model would needlessly increase model complexity and the estimation variance of its parameters, without improving prediction ability.

A wide variety of variable selection methods can be found in the literature. Each one of them proposes a different or modified scheme to select the relevant variables from the set of predictors. The PAC framework considers three different variable selection methods, as representatives of this class: forward stepwise regression (FSR) (Andersen and Bro 2010, Montgomery and Runger 2010), OLS implemented with a genetic algorithm for variable selection (GA) (Leardi, Boggia et al. 1992, Leardi 2007) and best subsets (BS). FSR (Andersen and Bro 2010, Montgomery and Runger 2010) combines both forward and backward selection protocols. It starts with an empty set of variables in the model and sequentially includes or removes variables according to the p-value of a partial F-test, based on which either the statistical significance of variables to be included in the model, or the non-significance of the variables to be excluded from the model is successively assessed. At each step of the algorithm, predictor variables are incrementally included in the model if their p-value for the partial F-test is smaller than a threshold (p_{in}). For each inclusion, a backward step is performed: the predictor with the highest p-value is removed if its p-value is higher than a threshold (p_{out}). The algorithm proceeds until no variables can be added in the inclusion step or removed in the backward step. The final model considers only the final set of selected predictors and their regression coefficients $\left(\hat{\mathbf{b}}_{\mathbf{FSR}}\right)$ are estimated by minimizing the sum of squared residuals, using OLS.

Genetic algorithms (GA) (Leardi, Boggia et al. 1992, Leardi 2007) mimic evolution theory and were inspired by the principle of the survival of the fittest. The measure of fitness adopted in this work is the cross-validation error. Each individual in the population of potential solutions contains binary information indicating which predictors are being selected for the model. The algorithm starts with an initial population of n_{ind} individuals and in each generation, models with smaller cross-validation errors are retained while the worst models are discarded. The top individuals are combined in order to produce offsprings: genes from two individuals are randomly split and interchanged in order to generate two new individuals (this procedure is called single cross-over). Following the reproduction step, each gene may mutate with a very small probability. Thus, after the elimination of poorly performing individuals, offsprings generation and the mutation step, a new population is obtained. This population with individuals presenting higher fitting scores can again evolve, producing new individuals with even smaller cross-validation errors. The main motivation behind GA is to conduct a more uniform search over the space of predictors, since methods such as FSR and others based on gradient searches, are more prone to be trapped in local optima. The randomness associated with the cross-over and mutation steps enables GA to move beyond these local optima and potentially achieve better prediction performances. In this work, the top individual in the last generation is considered for model building and its coefficients $\left(\hat{\mathbf{b}}_{\mathbf{GA}}\right)$ are obtained by minimizing the sum of squared residuals between the predictions (using the selected predictors) and the response (the OLS estimate).

The last method considered in the class of variable selection methods is best subsets (BS). As the name implies, all subsets of predictor variables are explored and the one which minimizes a given criterion is selected. The criterion adopted is Mallow's C_P (Draper and Smith 1998). In practice, the number of all subsets (2^p) is often too large to explore and a strategy must be devised in order to decrease the number of combinations to analyze. Thus, in this work we explore a modified version of BS: initially, FSR is applied and the number of selected predictors (p_{FSR}) is saved. Then, all subsets with p_{FSR} variables and others close to it are explored (a 10% deviation from p_{FSR} was used so that the number of variables in the subsets varies between $0.9p_{FSR}$ to $1.1p_{FSR}$).

6.2.1.2 Penalized Regression

The class of penalized regression methods is characterized by the addition of a penalty term for the size of the estimated regression vector. This penalty introduces a bias in the estimated model but stabilizes the least squares estimates under collinearity conditions. Four methods from this class were included in PAC: ridge regression (RR) (Hoerl and Kennard 1970), least absolute shrinkage and selection operator (LASSO) (Tibshirani 1996), elastic nets (EN) (Hastie, Tibshirani et al. 2001, Zou and Hastie 2005, Hesterberg, Choi et al. 2008), and support vector regression (SVR) (Smola and Schölkopf 2004, Canu 2005, Ahmed, Atiya et al. 2010). EN is a general method and its coefficients are obtained by solving the optimization problem formulated in eq. (27):

$$\hat{\mathbf{b}}_{\mathbf{EN}} = \underset{b=[b_0 \dots b_p]^{\mathrm{T}}}{\arg\min} \left\{ \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2 + \gamma \left(\alpha \sum_{j=1}^{p} \left| b_j \right| + \frac{1-\alpha}{2} \sum_{j=1}^{p} b_j^2 \right) \right\} \tag{27}$$

The hyper-parameter α $(\alpha \in [0,1])$ weights the relative contributions of the different types of penalization to the magnitude of the coefficients, namely the L_1-norm and the L_2-norm penalization, while γ controls the bias-variance tradeoff, by weighting the contribution of the classical least squares term with the penalization term for the regression coefficient size. A large γ constraint the value of the regression coefficients to be small and the model will be more biased. On the other hand, when smaller γ values are used, the model is more flexible (less bias) but presents higher variance (for γ=0, the OLS solution is obtained). Suitable values of γ and α were chosen through 10-fold cross-validation in order to control the bias-variance tradeoff and impose an adequate penalization, with the ultimate purpose of improving prediction accuracy.

RR and LASSO are particular cases of the general EN methodology, when the hyper-parameter α is set to 0 or 1, respectively. RR $(\alpha = 0)$ imposes a squared penalty (L_2-norm) on the magnitude of the regression coefficients, constraining all coefficients to be small. Thus, the solution obtained $(\hat{\mathbf{b}}_{\mathbf{RR}})$ is often characterized by having many small magnitude coefficients of different predictors, a situation that may be reasonable in collinear problems, but not in sparse scenarios. In contrast to RR, LASSO $(\alpha = 1)$

imposes a L_1-norm penalty on the regression coefficients, allowing some regression coefficients to be large, while others are small or effectively shrunk to zero. By combining both LASSO and RR penalties, EN can overcome some of the disadvantages of LASSO, namely its tendency to select only one variable from a group of highly collinear predictors. Thus, a grouping effect is often observed with EN regression coefficients: some groups of correlated predictors tend to be all actively contributing to predict the response, or they are removed altogether from the model. This may be suitable for problems with ordered predictors, such as in spectroscopic calibration problems (where it is known that only some spectral bands carry relevant information regarding the response and it may be desirable that all wavelengths in those bands are selected for predicting the response) or in multiblock data structures, where regressors can be organized in blocks according to some relevant criterion, such as belonging to the same unit or stage in the process or linked to the same function in the system.

The last penalized regression method included in the comparison framework is SVR (Smola and Schölkopf 2004, Canu 2005, Ahmed, Atiya et al. 2010). SVR attempts to minimize the squared magnitudes of regression coefficients in order to decrease the variance of the model. If only this penalty term was considered, then the solution would be trivial and no predictors would be selected. Thus, additional constrains are imposed, namely that the prediction errors are below a given threshold (τ). The optimization formulation considered in SVR is given by eq. (28):

$$\hat{\mathbf{b}}_{\mathbf{SVR}} = \underset{b=\left[b_0 \dots b_p\right]^{\mathrm{T}}}{\arg\min} \left\{ \sum_{j=1}^{p} b_j^2 + C \sum_{i=1}^{n} \tau_i \right\} \tag{28}$$

The first term in eq. (28) corresponds to a L_2-norm penalty on the magnitude of the coefficients and the second term corresponds to the τ-insensitive loss function $\left(\tau_i = \max\left\{0, \left|y_i - \hat{y}_i\right| - \tau\right\}\right)$. The support vectors correspond to samples that have errors above the threshold and their number is a measure of model complexity. The threshold (τ) controls the bias-variance tradeoff and was chosen by 10-fold cross-validation (setting $C = \max(\mathbf{y})$ usually works quite well (Ahmed, Atiya et al. 2010)).

6.2.1.3 Latent variable methods

The class of latent variable methods (Burnham, Viveros et al. 1996, Burnham, MacGregor et al. 1999) assume that the data generating mechanism follows a latent variable model (Burnham, MacGregor et al. 1999, Burnham, MacGregor et al. 2001), where only a few underlying and unmeasured variables are driving the observed variability in both the predictors (**X**) and response (**y**) spaces. The latent variables are estimated using linear combinations of the original variables. This is often a suitable strategy to handle highly collinear problems since collinear predictors can be in this way simultaneously considered to estimate the response. The latent variable model consists of eq. (29) and eq. (30):

$$\mathbf{X} = \mathbf{TP}' + \mathbf{E} \tag{29}$$

$$\mathbf{y} = \mathbf{Tc} + \mathbf{f} \tag{30}$$

where **T** is a $N \times a$ (often $a \ll p$) orthogonal matrix of scores that represents unmeasured and independent directions of variability, **P** is a $p \times a$ loading matrix that, combined with **T**, provides the structured variability in the predictor set (**X**). **c** is a $a \times 1$ vector relating the scores and the response variable. **E** and **f** are residual matrices with suitable dimensions.

Two approaches were contemplated in PAC for this class of methods: principal component regression (PCR) (Wold, Esbensen et al. 1987, Jolliffe 2002, Jackson 2005) and partial least squares (PLS) (Wold, Ruhe et al. 1984, Geladi and Kowalski 1986, Wold, Sjöström et al. 2001). PCR (Wold, Esbensen et al. 1987, Jolliffe 2002, Jackson 2005) is implemented by first applying principal component analysis (PCA) to the predictor space **X** (i.e., considering only eq. (29)). PCA extracts the eigenvectors of the predictors' covariance matrix and sorts them by decreasing magnitude of their associated eigenvalues. In particular, the eigenvector associated with the highest eigenvalue corresponds to the direction of maximum variance in the predictors' space and subsequent smaller eigenvalues correspond to directions explaining decreasing percentages of variability in the **X**-space. Thus, one can ignore eigenvectors whose eigenvalues are close to zero since they represent directions where no significant variability is observed. The principal components (**T** in eq. (29)) are orthogonal and their covariance matrix is easily inverted. PCR consist of applying OLS in order to relate the scores and the response variable. In PAC, the number of principal components is selected by 10-fold cross-validation.

The other method from the latent variable class is PLS regression (Wold, Ruhe et al. 1984, Geladi and Kowalski 1986, Wold, Sjöström et al. 2001). PLS was first developed by Herman Wold around 1975 and extracts latent variables that most explain the observed variability in the response, while providing good coverage of the input space. This is one of the main differences from PCR, where the latent variables are those that best explain the variance in the predictor set. The PLS model is presented in eq. (29) and eq. (30), and in PAC the number of latent variables is selected by 10-fold cross-validation. An interesting property of PLS is that it subsumes OLS as a limiting case, namely when all latent variables are included in the PLS model. This property also holds for PCR.

6.2.1.4 Ensemble methods

The last class of methods included in the comparison framework regards ensembles of regression trees (Breiman, Friedman et al. 1984, Dietterich 2000, Hastie, Tibshirani et al. 2001, Strobl, Malley et al. 2009). Regression trees approximate the relationship between predictors and response variable by a piece-wise constant function. The algorithm starts by identifying a split point (*v*) and split variables (*j*) that most decrease the sum of squared errors between the predicted and observed response, obtaining two regions: $R_1(j,v)=\{\mathbf{X}\,|\,x_{i,j}<v\}$ and $R_2(j,v)=\{\mathbf{X}\,|\,x_{i,j}>v\}$. The predicted response in each region is the mean value of the samples belonging to the region. Each region is further subdivided according to the decrease of the sum of squared residuals, until a stopping criterion is met (in this case until a minimum of 5 observations in each region are obtained). Trees tend to present high variance, and small changes in the training set often result in new trees with different split points and different split variables. Furthermore, since each region only contains a small number of samples, a rather poor estimation of the mean value is likely to be obtained. In order to improve prediction performance and decrease variance, trees are combined in ensembles. Three ensemble methods were incorporated in PAC: bagging of regression trees (BaRT), random forests (RF), and boosting of regression trees (BoRT). BaRT and RF are very similar, as both of them use bootstrap to generate new datasets, based on which different regression trees are built. The prediction for a given sample is obtained by averaging the predictions from all trees in the ensemble. The main difference between the two methods is that each tree in RF only considers a randomly selected subset of predictors, whereas in BaRT all predictors are used for all trees. The advantage obtained with RF is that the trees in the ensemble are less correlated, which in turn tends to increase prediction performance.
The predictions of BaRT are computed using eq. (31).

$$\hat{y}_i=\frac{1}{T_{BaRT}}\sum_{t=1}^{T}f_t\left(x_{i,j}\right) \tag{31}$$

where T_{BaRT} is the number of trees in the ensemble and $f_t\left(x_{i,j}\right)$ are the predictions from the t^{th} regression tree. A similar equation describes the prediction made by RF with T_{RF} trees in the ensemble.

The last method included in the comparison framework is boosting of regression trees (BoRT) (Freund and Schapire 1996, Freund, Schapire et al. 1999, Elith, Leathwick et al. 2008, Cao, Xu et al. 2010). Boosting is implemented by fitting a regression tree to a dataset, followed by the computation of prediction errors. Subsequent trees in the ensemble use the prediction errors from the previous tree as the response variable so that poorly modeled samples are given high weights. The reasoning for this is to increase model fitting, for which samples with high residuals should be better modeled. However, each model only captures a small fraction of the relationship between **X** and **y**, mitigating the risk of overfitting. In more detail, BoRT works by fitting a regression tree between the predictors (**X**) and prediction residuals, as described in eq. (32):

$$\mathbf{y}_t^{res} = \mathbf{y}_{t-1}^{res} - u \times f_{t-1}(\mathbf{X}) \tag{32}$$

where $\mathbf{y}_t^{res}$ is the residual vector for the *t*th regression tree $\left(\mathbf{y}_0^{res} = \mathbf{y}\right)$, $f_t(\mathbf{X})$ are the predictions from the *t*th regression tree which is constructed using the *t*th residuals and predictor variables, and *u* is a shrinkage parameter that prevents overfitting $(0 < u < 1)$. The hyper-parameter *u* is related to the number of trees (lower *u* values will require more regression trees and vice-versa) (Elith, Leathwick et al. 2008), so a low value for the shrinkage parameter is specified (*u*=0.02) and the number of trees is chosen by 10-fold cross-validation. The final BoRT model is presented in eq. (33).

$$\hat{\mathbf{y}}_i = \sum_{t=1}^{T} u \times f_t(x_{i,j}) \tag{33}$$

6.2.2 Data Domain

This component of PAC regards the specification of the dataset that will be used to make the structured comparison among the methods belonging to the Analytics Domain. It will provide the source of factual evidence for the relative performance rankings, and the inference basis that supports the analysis. Therefore, the findings reported by PAC will be circumscribed to the situations reflected in this dataset, such as operational conditions covered, predictors used and how they were varied (or not), etc. It is this component that makes PAC a problem-specific framework – the other three components remain mostly unaltered across applications.

When dealing with some particular problems, pre-processing may be a component that impacts the quality of the results and therefore should be addressed properly (e.g. in spectroscopic multivariate calibration). In PAC, predictor variables are scaled to have zero mean and unit variance (also known as auto-scaling) by default. This setting may be easily changed for some classes of methods. Further pre-processing should be considered in the optional stage presented in Figure 6.2.

The application of PAC also assumes that outlying observations have been removed in a first stage of exploratory data analysis, which is the typical starting point of every data-driven analysis. Prediction outliers can also be identified after the application of PAC, since the results report would indicate that particular samples have high residuals. Moreover, PAC assumes that a solution to handle missing data has already been applied during the pre-processing stage, either by removing observations containing missing data or imputing their values by suitable methods (Walczak and Massart 2001, Arteaga and Ferrer 2002, Little and Rubin 2002).

6.2.3 Comparison Engine

This component operates over the methods defined in the Analytics Domain, using the dataset inserted in the Data Domain. The comparison engine combines a double cross-validation scheme to generate raw information on the methods' predictive ability, with statistical hypothesis testing in order to assess their relative performance. The performance of each regression method is characterized by the root mean squared error of double cross-validation $\left(RMSE^{dcv}\right)$. All methods are trained and tuned using the same training sets and tested using the same unseen validation sets across the implementation of the cross-validation runs. Therefore, pairwise comparisons can be made through statistical hypothesis tests, for establishing their relative performance in a robust and accurate way. The complete procedure is described in detail in the following paragraphs.

Multiple cross-validation runs are used to assess the prediction performance of each regression method, which is evaluated through $RMSE_{d,m}^{dcv}$, where d is the run index and m is the regression method's index. Double cross-validation starts by randomly partitioning the dataset into a training set and a validation set (usually, 20% of the samples are reserved for the validation set), followed by model training using the training set. During model training, 10-fold cross-validation is used to select suitable hyper-parameters for each regression method (this is the inner cycle of the double cross-validation scheme) and a regression model is built with the training data and the selected hyper-parameter(s) (see Table 6.1 for the ranges and strategies considered in the optimization of the hyper-parameters in the inner cycle). The estimated model is then used to predict the samples in the validation set and the prediction errors, which are finally used to compute $RMSE_{d,m}^{dcv}$ for the current run. Then, in the next iteration of double cross-validation (in the outer cycle of double cross-validation), this procedure is repeated, i.e., the dataset is again randomly partitioned in a training set for model building/estimation and a validation set to assess the performance of the trained models. Typically, 50 iterations of the outer cycle of the double cross-validation scheme are conducted, leading to 50 values of $RMSE_{d,m}^{dcv}$ for each method (m).

The distribution of $RMSE_{d,m}^{dcv}$ characterizes the performance of each regression method. Thus, statistical hypothesis testing can be applied to assess, at a given significance level (5% in this work), whether the performance achieved with one regression method is smaller, equal or higher, when compared to another regression method. In particular, the paired *t*-test is used, since in each run of double cross-validation, all regression methods in the comparison framework make use of the same training and validation samples.

Table 6.1. The range of values for hyper-parameters and the strategies used to select them during model training.

Method	Hyper-parameter (s)	Possible value(s)	Selection strategy
MLR	-	-	-
FSR	p_{enter}	0.05	-
	p_{rem}	0.1	
GA	n_{ind}	30	10-fold cross-validation
	n_{gen}	100	
BS	p_{BS}	[0.9×p_{FSR}, 1.1×p_{FSR}]	Mallow's Cp
RR	α	0	10-fold cross-validation
	γ	0.02; 0.02; 0.2; 2; 20	
LASSO	α	1	10-fold cross-validation
	γ	0.001; 0.01; 0.1; 1; 10	
EN	α	0.001; 0.01; 0.1; 1	10-fold cross-validation
	γ	0.02; 0.02; 0.2; 2; 20	
SVR	T	0.001; 0.005; 0.01; 0.05; 0.1	10-fold cross-validation
PCR	a_{PCR}	[1:min(20, n, p)]	10-fold cross-validation
PLS	a_{PLS}	[1:min(20, n, p)]	10-fold cross-validation
BRT	T_{BRT}	50; 100; 500; 1000; 5000	10-fold cross-validation
RF	T_{RF}	50; 100; 500; 1000; 5000	10-fold cross-validation
BT	T_{BT}	50; 100; 500; 1000; 5000	10-fold cross-validation

6.2.4 Results Report

The last component of the PAC framework is dedicated to report the outcomes of the structured comparison analysis to the user. The results report was designed taking into consideration what type of information users would be mostly looking for when using PAC. Based on this analysis, a hierarchy of priorities was established. A first concern is to have access to the ranking of relative performances of the methods contemplated in the analytical domain, in the context of the dataset provided in the data domain. At a second level in the hierarchy of priorities, users become interested about some interpretational dimensions of the predictive models, namely which predictors are more important and which can be discarded, as well as the nature of the relationships found.

The ranking of methods according to their relative performance (high priority information) was established as follows. As the goal is to have a global perspective of the methods that perform better when comparing to all others, a robust procedure was devised that delivers exactly that, while taking into account the natural variability of predictions arising from models built from different training datasets. The procedure starts by considering the outcomes of the paired *t*-tests for each pair of methods under analysis and attribute points according to the following scoring system: if the predictive performance of the regression method “A” is better than that for method “B”, in a statistically significant way (smaller $\overline{RMSE}^{dcv}$), then method “A” receives 2 points and method “B”, 0; if method “A” has a statistically significant poorer performance than method “B” (larger $\overline{RMSE}^{dcv}$), it receives 0 points and method “B”, 2; if the difference of performances between the two methods is not statistically significant, each method receives 1 point. Summing up all scores obtained by each method in all pairwise comparisons, allow the user to access the relative capabilities of each method in the scope of all those considered in the analytics domain. Therefore, the total score for each regression approach will be the key performance indicator (KPI) used to establish the ranking of relative performance of all the methods under comparison. This KPI has the advantage of stemming from formal hypothesis tests that take into account the natural variability of results in order to access the relative performance of each pair of methods, bringing robustness and accuracy to the analysis, together with the ability to combine all pairwise outcomes in a global index that is directly linked to the priority goal of the user. A high value for the KPI indicates that the associated regression method consistently performed better than the other approaches, i.e., obtained statistically smaller $\overline{RMSE}^{dcv}$. Furthermore, since the analytics domain encompass 13 regression methods, the KPI varies between zero (a method with statistically higher $\overline{RMSE}^{dcv}$ compared to all other methods) and 24 (when a method presents statistically lower $\overline{RMSE}^{dcv}$ compared to the other 12 regression methods). Analyzing the KPI for all methods is often the preferred choice since it allows for an easy identification of the best method(s). However, a more detailed analysis can be conducted by decomposing the KPI into its contributions, in terms of winning and tied scores. This analysis would point out, for instance, whether a given method's performance was the result of several wins or if ties dominated the KPI value, in which case other methods present also comparable performances.

The proposed methodology was tested and found to be easily interpretable by users, leading to consistent results and pointing out the regression methods leading to improved prediction ability. Even though the pairwise analysis involve multiple simultaneous hypothesis tests, this will not bias the results due to the inflated Type I error, given the higher number of methods involved which make it unlikely that any potential false positives (i.e., the detection of a difference when in fact there is none) occur for the same method, always in the same favorable (our unfavorable) direction. Rather, the accumulated experience on using this approach (Reis and Saraiva 2004, Reis and Saraiva 2005, Reis, Rendall et al. 2015) reveals that it is both sensitive (because it represents a blocked comparison test and is based on a sample of considerable size, both factors leading to higher levels of statistical power) and robust (the size of the sample guarantees distributional convergence and the number of methods involved eliminates any significant bias towards a given method). Nevertheless, one must be aware that false alarms can occur and influence the results, particularly when comparing methods with similar KPI values. However, the multiple splits considered in the double cross-validation scheme make this phenomenon dispersed across all regressors and models, which in turn leads to an unbiased methodology but that is affected with some additional variance. Therefore, the expected effect is a reduction in the sensitivity to detect the best methods (because of the increasing dispersion of results) when their performance is not so superior regarding other methods. In practical terms, this implies that more ties will tend to occur among methods with similar performances, and only those methods performing significantly better will be pointed out by PAC. This increases the confidence in the selected methods put forward by this framework. Alternative procedures were also considered, such as multiple testing approaches and 2-way ANOVA. However, after a proper consideration of the pros and cons of these approaches, the proposed methodology was adopted. The corrections of the individual significance levels for each hypothesis test required to maintain the overall Type I error rate constant in multiple testing approaches, are overly conservative, making the comparison methodology much less sensitive even for rather visible differences in performance of the methods. On the other hand, the 2-way ANOVA approach is viable if one uses "regression methods" and "iterations" of double cross-validation as main factors, discarding their mutual interactions. However, this alternative would still require pairwise comparisons to assess the relative performance of the methods, which could be done through the least significant differences (LSD) method. Nevertheless, the aforementioned KPI was the preferred choice since it is simple, informative and its robustness was tested in a large number of case studies (namely the simulated scenarios where an oracle of the true model is known and consistent results were obtained).

The results report also presents additional information for model interpretation and to analyze the nature of the relationships found. These outcomes depend on the methods from which information is to be retrieved, which are usually the ones presenting higher KPI's. Several examples of this type of interpretational information will be provided in the Discussion subsection.

6.3 Case Studies

The application of PAC to different scenarios will be illustrated in this subsection. Two simulated and two real world situations are considered. The simulated scenarios allow for an assessment of PAC's operation under well-controlled circumstances, where the results can be interpreted on the grounds of known data generating mechanisms. They serve the purpose of testing the framework in situations where there are founded expectations on which methods should perform well, and can be used to analyze the consistency of PAC outcomes with respect to the known ground truth. The real datasets, on the other hand, bring out an interesting aspect that further corroborates the relevancy of using PAC: even when addressing the same prediction problem, the best class of approaches in the analytics domain is dependent on the type of data used in the data domain. Situations like this can only be properly resolved using a problem-specific comparison approach, such as PAC.

6.3.1 Datasets used in the Data Domain

6.3.1.1 Simulated Dataset 1: Sparse Process

Simulated datasets are a convenient alternative to test the performance of regression methods under well-controlled scenarios where the true model structure, the regression coefficients and the values of the response variable are known (with and without additive and homogeneous Gaussian noise). Thus, one can observe if the theoretical assumptions of different regression methods indeed work under such idealized scenarios and how their performance is affected by deviations from those scenarios. In order to cover two common situations found in practice, two types of processes were simulated: a sparse process and a latent variable process. The sparse process covers the case where only some regressors contain relevant predictive information regarding the response, while presenting low levels of mutual correlation (collinearity). Datasets with this type of structure are very common in big data applications in Manufacturing 4.0, where only some predictors among the large number of variables collected are really relevant for analysis. In these contexts, the ratio between the relevant "critical few" predictors and the "trivial many" irrelevant variables tends to decrease as the size of the dataset increases and the process variation sources remain the same. In other words, the additional measurements obtained in big data applications are often not related to the response variable and only increase the size of the predictive problem. Effect sparsity is also very common in "small data" applications, such as in statistical design of experiments (DOE), being considered one of the "principles" assumed in this methodology (the others are effect hierarchy and heredity) (Box, Hunter et al. 2005, Montgomery 2009, Wu and Hamada 2009) (the goal of DOE is devise an efficient experimental plan to identify the relevant predictors, or effects, among a set of linear –called main effects–, interactions and quadratic terms, and to build the predictive model for the response under analysis). This model is similar to the one considered applications of WiSe (see also Section 5.5.1); however, a smaller dimensionality of the dataset is simulated here to represent the outputs from the filtering stages of WiSe. The sparse model was presented in eq. (25). The variance of the error term was chosen so that the signal to noise ratio (*SNR*) is 20. In this case study, 100 samples were simulated and a total of 20 predictors ($N = 100$ and $p = 20$) were considered, where only 10 randomly selected predictors effectively contribute to the response $(b_j \neq 0)$. The remaining predictors do not influence the response and their regression coefficients are effectively zero. The 10 predictors that contribute to the response follow a multivariate normal distribution with unit variance, and present a weak mutual correlation between themselves $(\rho = 0.1)$. The regression coefficients for these 10 coefficients were drawn from a uniform distribution in the interval: $[-10, 10]$.

6.3.1.2 Simulated Dataset 2: Latent Variable Process

The second simulated scenario was obtained from a latent variable model and represents another typical situation found in practice, such as in spectroscopic applications or in large scale industrial processes operating under normal conditions (Burnham, Viveros et al. 1996, Burnham, MacGregor et al. 1999). In fact, data passively collected from industrial processes are typically high-dimensional and with a latent structure, as the variability of observable variables is driven by only a few underlying and unobservable sources of variation. Thus, this simulated dataset also presents a high degree of collinearity due to some underlying and unmeasured sources of variation. The latent variable model was presented in eq. (29) and eq. (30). In this case study, 500 samples and 100 predictor variables ($N = 500$ and $p = 100$) were simulated and the model contains 4 independent sources of variability $(a = 4)$.

6.3.1.3 Real Dataset 1: HPLC Data for Wine Age Prediction

This first real world dataset comes from the food & drink industry, containing chemical information collected during the aging process of one of the finest Portuguese fortified wines, the Madeira Wine. Samples with ages ranging from 1-20 years old were analyzed in a Waters Alliance liquid chromatograph (Milford, MA, USA) in order to quantify 23 phenolic compounds and the major furanic compounds for 52 wine samples ($N = 52$ and $p = 25$). Further details regarding the experimental apparatus can be found in the literature (Rendall, Pereira et al. 2016). The aim of this analysis is not only regarding prediction performance (of the aging time), but also the identification of important compounds that are related to the aging process and can act as markers of age and quality for this wine, which in turn help producers understanding better the complex chemistry of wine ageing and its dynamics.

6.3.1.4 Real Dataset 2: UV-Vis Spectra for Wine Age Prediction

This second real world dataset addresses the same prediction problem as the previous one, but instead of chemical information, one uses (untargeted) spectral information for predicting the aging time of Madeira wine. UV-Vis spectra were collected for this purpose. As UV-Vis is a fast and affordable solution to collect data from wine samples, it would be interesting to evaluate if it is able to lead to models with a good predictive performance for the wine age. Samples were analyzed in a Perkin-Elmer Lambda 2 spectrophotometer and their UV-Vis absorbance spectra were recorded (Pereira, Reis et al. 2010, Pereira, Reis et al. 2011). This dataset contains 52 samples and 109 wavelengths ($N = 52$ and $p = 109$) from 245 to 785 nm, at 5 nm intervals. See Ref. (Rendall, Pereira et al. 2016) for further details.

6.3.2 Results Report: KPIs obtained for each Dataset

This section presents the KPIs obtained with the application of PAC to each scenario described in the previous subsection. Since some of the regression methods in the comparison study are sensitive to the variance of the different predictor variables, variable scaling was employed. Auto-scaling was applied to the simulated and HPLC datasets by first removing the sample mean of each predictor variable and then diving by the sample standard deviation. On the other hand, as auto-scaling is in general not a good solution for pre-processing spectral data, the UV-Vis dataset was scaled using the standard normal variate technique (SNV) followed by mean centering. Variable scaling was applied consistently so that the scaling parameters obtained during model training were used to scale the validation set, in each run of double cross-validation.

6.3.2.1 KPIs: Sparse Process

The results for the dataset generated with a sparse model are presented in Figure 6.3, where the KPIs obtained by the different regression methods are shown. Analyzing Figure 6.3, one can note that regression methods whose structure and prior assumptions best match the data generating mechanism simulated, tend to perform better. Indeed, the best methods are from the class of variable selection and penalized regression, both of them having intrinsic capabilities for dealing with sparse and uncorrelated predictive scenarios. Nevertheless, latent variable methods still present rather good performances, whereas tree-based methods show the worse performance because their piece-wise constant approximation fails to describe well the linear and continuous behavior of the process. Further discussion of this dataset will be presented in Section 5.4, where the most important variables will be identified and compared to those used to generate the data.

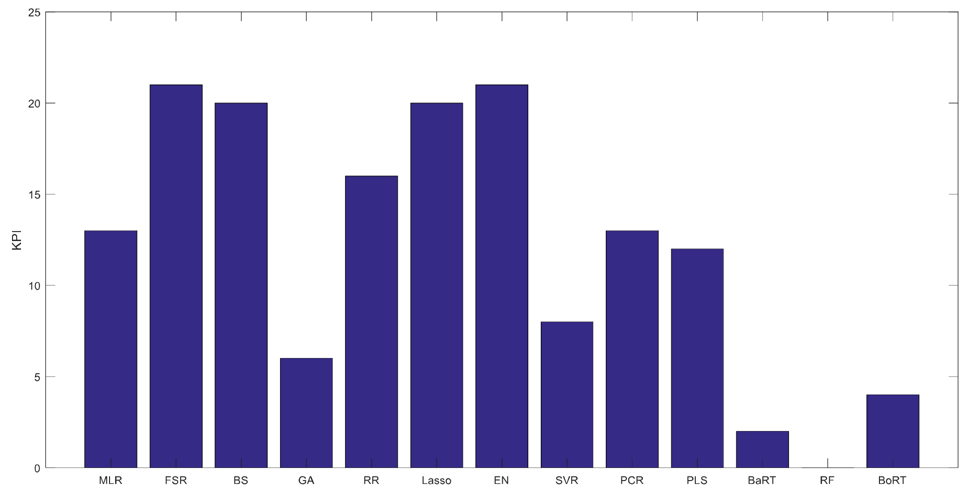


Figure 6.3. KPI obtained by each regression method when applied to a sparse dataset.

6.3.2.2 KPIs: Latent Variable Process

The dataset obtained from the latent variable model also leads to good results for regression methods matching their prior assumptions. Figure 6.4 presents the KPI for each regression method and it can be observed that methods from the latent variable class perform quite well, namely PCR and PLS. However, rather interestingly, other methods also perform very well. In particular, the class of penalized regression methods can also cope with the collinearity features present in this dataset. This may justify the use of this type of methods more frequently in chemometric applications, besides those from the latent variable class. Again, the tree-based ensemble class does not present good performances, which can be attributed to the poor matching to the data generating mechanisms: trees are built by splitting the predictors' space and approximating the response by a piece-wise constant function. However, the response is linearly related to the predictors. Furthermore, some predictors may be also ignored in tree-based models, if they are highly correlated. Although RF minimizes this drawback by only using a random subset of predictors, the best approach would be to combine the correlated predictors in order to estimate their common source of variability. As can be noted in Figure 6.4, BS was not included in the analytics domain for this case study because the number of predictor variables is rather large, which prevents this method to be applicable given the combinatorial explosion of the number of variable subsets to consider in these conditions.

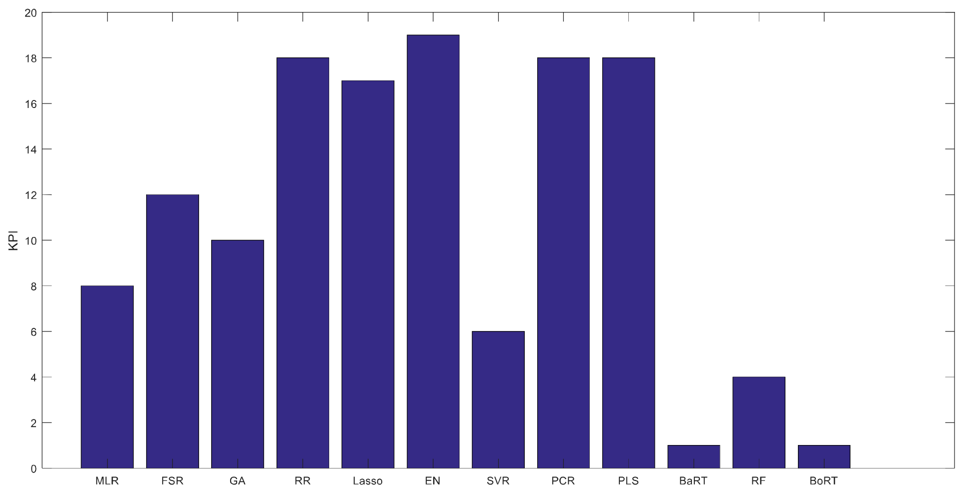


Figure 6.4. KPI obtained by each regression method when applied to a dataset generated using a latent variable model.

6.3.2.3 KPIs: HPLC Data for Wine Age Prediction

In the first real world dataset, HPLC is used to quantify phenolic and furanic compounds in wine samples with different aging times. These measurements are then used by PAC, as predictors of wine age. Figure 6.5 presents the KPI obtained for each regression method. It can be seen that BoRT has a significant advantage over the other regression methods and all methods from the ensemble class show good performances. This fact suggests the presence of a non-linear relationship between regressors and wine age, which can be properly approximated by the piece-wise nature of regression trees. The non-linear association is plausible since the concentration of the different wine constituents is expected to vary in a complex way, presenting rather distinct patterns related to the dynamic of wine aging (e.g. appearance of new compounds). Thus, one would not expect their concentration to increase/decrease indefinitely as assumed by most linear models. The $\overline{RMSE}^{dcv}$ obtained with BoRT is 0.54 years, which is a rather low (and therefore good) value when considering the range of variation of the response (1-20 years). Regarding other classes of regression methods, both penalized regression methods and latent variable methods present interesting predictive performances.

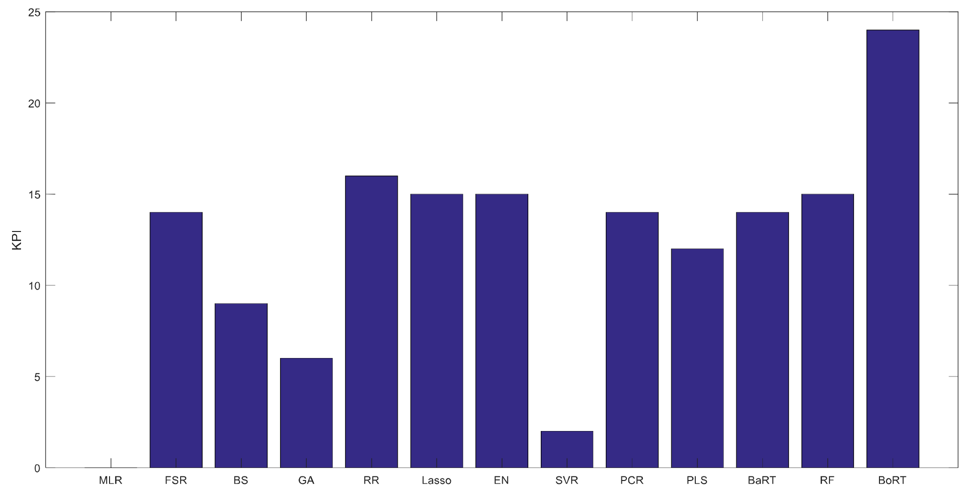


Figure 6.5. KPI obtained by each regression method when predicting wine age based on HPLC data.

6.3.2.4 KPIs: UV-Vis Spectra for Wine Age Prediction

This second case study consists of using UV-Vis spectra as regressors (instead of concentrations of phenolic and furanic components obtained from HPLC measurements) and again wine aging time as response variable. Figure 6.6 presents the KPIs obtained by each regression method. One can observe that penalized regression methods now achieve the best performances, with LASSO and EN as the top ranking methods. The $\overline{RMSE}^{dcv}$ obtained with LASSO is 2.1 years, which is four times higher than the error obtained in the HPLC dataset with BoRT, suggesting that the chemical information obtained with HPLC is more reliable for wine age prediction when compared to UV-Vis spectra. Methods from the tree-based ensemble class also presented good performances, particularly RF and BoRT, but not as good as in the previous dataset. In spite of the fact that non-linearity is still present, the highly collinear structure imposed by the UV-Vis spectra tends to dominate and methods able to cope with it, perform better. However, as these methods are intrinsically linear, the performance achieved is also lower, which can be further aggravated by the fact that some chemical information may not show up in the UV-Vis range of the spectrum. The latent variable class, often the preferred class for this type of datasets, did not obtain the best KPI.

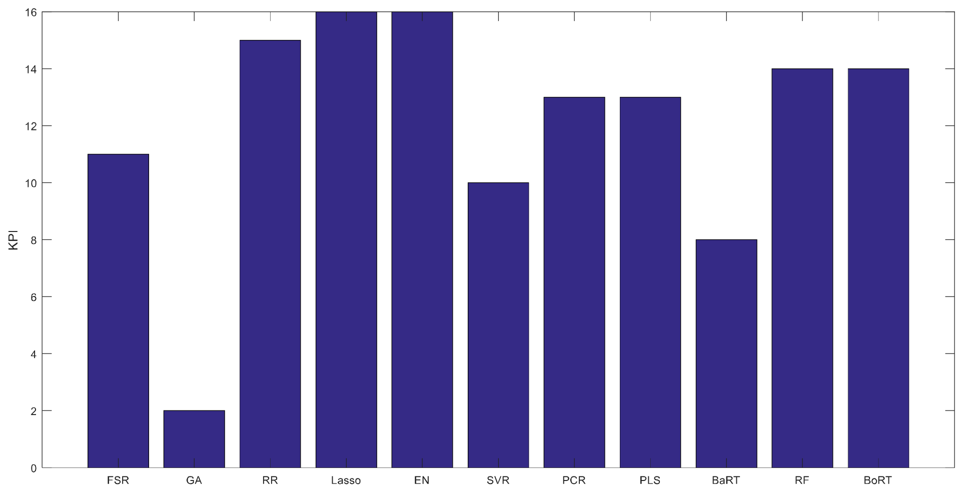


Figure 6.6. KPI obtained by each regression method when predicting wine age based on UV-vis spectra.

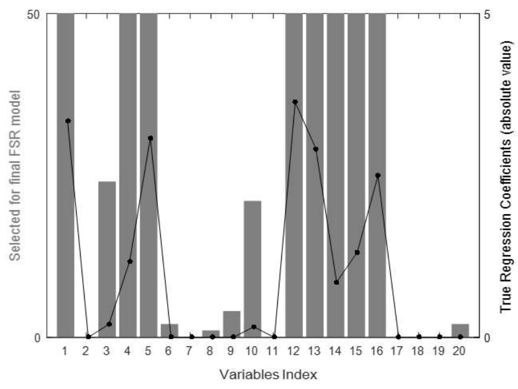
In summary, the analysis of these four case studies confirms that PAC leads to consistent results (see simulated case studies) and reveals non-obvious and unanticipated solutions (see real datasets). The inner workflow of PAC makes the analysis very efficient (the development time is short) and secures the quality of the outcomes (with the implemented comparison engine). The next section will show that PAC can also contribute to increase the knowledge of the data and processes under analysis.

6.4 Discussion

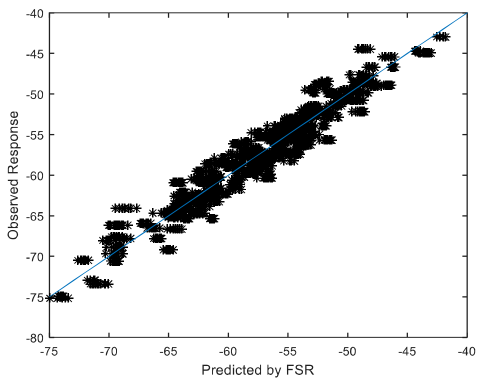
In this section, the results previously presented will be discussed in more detail. Further interpretational information will be provided for the regression models with higher KPIs, including the identification of the most important predictor variables.

6.4.1 Sparse Process

The KPIs for the dataset generated with a sparse model showed that better performances are obtained when the assumptions of the regression methods find good adherence to the dataset analyzed. In particular, FSR, a method from the variable selection class, obtained the best results since it correctly assumes that some variables should be discarded because they do not carry any predicting information regarding the response. This can be confirmed by observing the magnitudes of the true underlying OLS model (b in eq. (25)) and the number of times each predictor variable was included in the final FSR model (each predictor can be included at most 50 times, the number of double cross-validation iterations). Figure 6.7.a shows the results obtained during the FSR model training, where one can note that the most important predictors were always selected in the 50 iterations of double cross-validation. In fact, only two relevant predictors were not selected in some iterations (variable index: 3 and 10) due to the low magnitude of their regression coefficients. Nevertheless, FSR successfully selected the important predictors and the number of false positives is negligible. When plotting the predicted and observed response values over the 50 iterations of double cross-validation (Figure 6.7.b), the predictions fall close to the identity line and the errors are rather small. The median coefficient of determination over all iterations of double cross-validation is 0.92, which reinforces the fact that the models obtained accurately predict the response. Furthermore, the analysis of the other top ranked methods (BS, LASSO, and EN) shows similar structure in terms of predictor importance and they can effectively distinguish important and irrelevant predictors, presenting similar values of $RMSE^{dcv}$ (these results are not shown here for simplicity). Thus, the analysis of important predictors across different regression methods suggests a sparse structure where only 10 predictors are important, which is in accordance with the data-generating mechanisms. These results demonstrate the consistency of PAC in identifying the correct class of techniques that are more adequate to develop the predictive models.



(a)



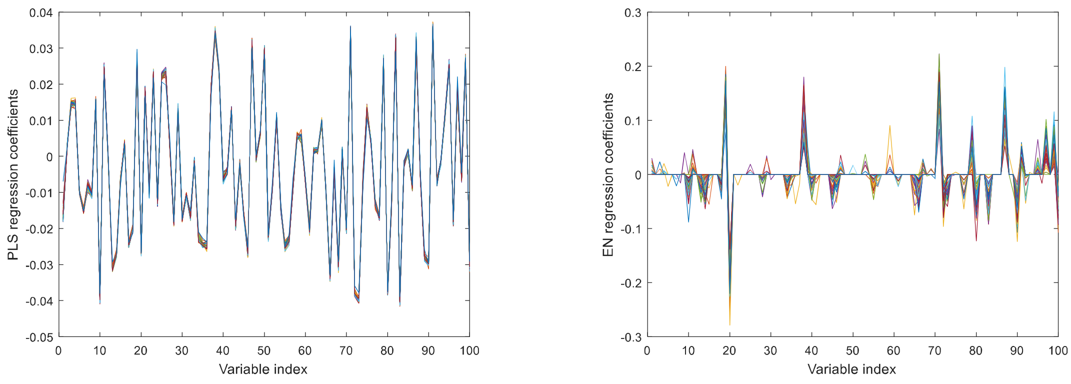
(b)

Figure 6.7. Results obtained when PAC was applied to data from the sparse process: (a) number of times each predictor variable was selected (bar) by FSR and the true values of the regression coefficients (connected dots); and (b) the predicted *vs.* observed response values in the validation sets from the 50 iterations of double cross-validation.

6.4.2 Latent Variable Process

The performance of the regression methods in the latent variable dataset also shows the advantages of matching the data generating mechanisms and the regression methods' prior assumptions: PLS and PCR were among the top methods. The class of penalized regression methods was also among the top methods and EN had the highest KPI value, showing that it is also a suitable method for this problem. In order to identify important predictor variables, Figure 6.8.a presents the regression coefficients for the PLS models obtained during all 50 iterations of double cross-validation and Figure 6.8.b shows the EN regression coefficients. Analyzing Figure 6.8, one can notice that the regression coefficients obtained from PLS had little variability over the 50 iterations of double cross-validation, i.e., the effects of different splits were negligible on the value of the regression coefficients. On the other hand, EN regression coefficients presented higher variability due to the strong correlation between predictor variables: in some iterations, a particular set of predictor variables has high importance while in other iterations, its importance decreases in favor of other variables that are correlated with them. Thus, for this dataset, PLS is actually the most suitable method due to the consistency of the estimated regression coefficients, although EN had the highest KPI. Further analysis of the KPI results showed that the difference in the distribution of $RMSE^{dcv}$ between the

PLS and EN is not statistically significant and the additional KPI point obtained by EN is due to its pairwise comparison with LASSO. More precisely, the difference between EN and LASSO is statistically significant (2 points are attributed to EN, see Section 2.4) whereas the difference between PLS and LASSO is not (only 1 point is attributed to PLS).



(a) (b)

Figure 6.8. Results obtained during model training for the Latent Variable simulated scenario: (a) PLS regression coefficients and (b) EN regression coefficients.

6.4.3 HPLC Data for Wine Age Prediction

When considering the first real world dataset, the results suggested a non-linear association between regressors (phenolic and furanic compounds) and wine age. As previously stated, the $\overline{RMSE}^{dcv}$ obtained with BoRT is small and further inspection revealed that the median coefficient of determination over all iterations of double cross-validation was 0.97, confirming the suitability of the combination of BoRT and HPLC to predict wine age. To assess the influence of the different predictors, Figure 6.9.a presents a measure of predictors' importance for BoRT, which accounts for the decrease in mean squared error due to splits on every predictor and normalized by the number of branch nodes (higher values of predictors' importance indicate a more relevant predictor). Variable #6 (an unknown compound) presents higher values of importance while variable #10 (epigallocatechin) is the second most important variable. To illustrate the non-linear relationship between epigallocatechin and wine age, Figure 6.9.b presents the predicted wine age when the concentration of epigallocatechin varies from its smallest to its largest value (observed in the dataset) and all other predictors are at their mean levels. The non-linear relationship is clear from Figure 6.9.b and a scatter plot of wine age *vs.* epigallocatechin also reveals a similar pattern: epigallocatechin values in the range 0.25-0.4 correspond to younger wines (typically 1-5 years old) and as the wine ages, there is a small increase in epigallocatechin concentration, obtaining values between 0.5-0.7 (these wines have aging times between 7 and 11). Finally, wines with 12 years old and older have zero or near zero concentration of epigallocatechin. Thus, since epigallocatechin is the second most important predictor and presents a non-linear relation with wine age, the ability to model this non-linear relationship is essential in order to build effective predictive models. The other classes of methods cannot account for this type of relationship and are therefore more limited in terms of prediction performance.

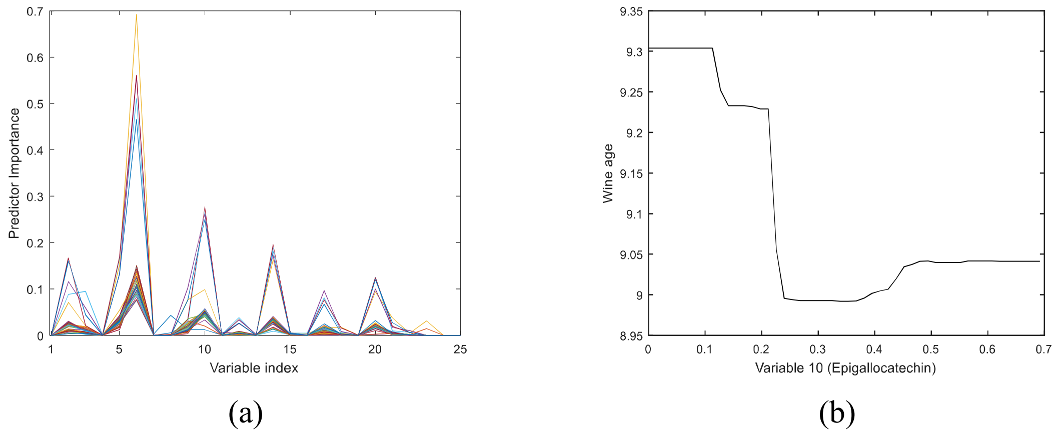
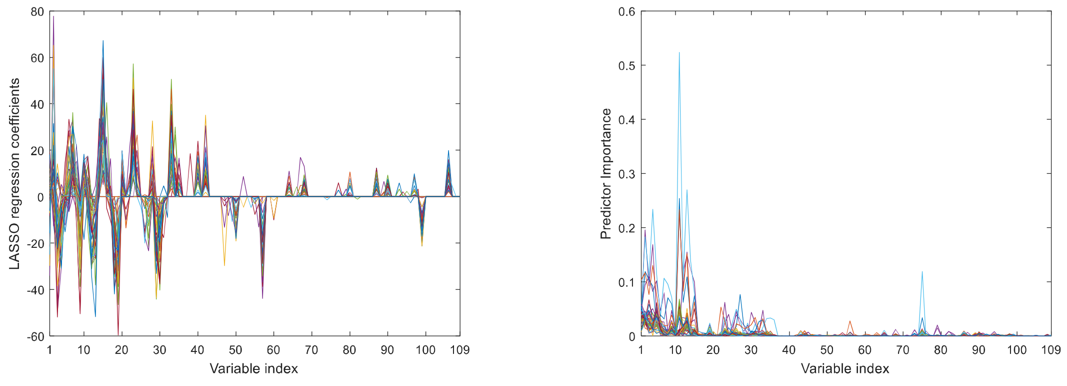


Figure 6.9. Results obtained when PAC was applied to estimate wine age with data from HPLC: (a) BoRT predictor importance and (b) the non-linear relationship between epigallocatechin and wine age.

6.4.4 UV-Vis Spectra for Wine Age Prediction

The last dataset regards the prediction of wine age using UV-vis spectra. The results show that the class of penalized regression methods obtained higher KPI values and are the recommended class for this dataset. In fact, the consistency of penalized regression methods across all the studied datasets suggests that it is a versatile class, which contains regression methods that adapt to many scenarios of sparsity and collinearity. In particular, LASSO's penalty is effective for variable selection, RR aggregates small contributions from many predictors and EN suitably combines both strategies in order to improve prediction accuracy. Thus, in scenarios where a more complete comparison study is not possible, the penalized regression class has the potential of being a suitable first choice. Furthermore, other types of penalties are also available for a wide range of applications (Eilers 2017). The median coefficient of determination over all iterations of double cross-validation was 0.86, which represents a rather good agreement between predicted and observed wine age values.

PLS and PCR are often the preferred methods for spectral datasets but this case study shows that they may not always be the best approach. In particular, one can observe that LASSO, EN, RF, and BoRT showed superior performances, which might suggest that the initial assumption of a latent variable structure may not be entirely valid and that the elimination of some variables might be advantageous. In order to identify important spectral regions, Figure 6.10.a presents LASSO's regressions coefficients and Figure 6.10.b presents the variables' importance for BoRT in the 50 iterations of double cross-validation. In Figure 6.10, one can observe an agreement regarding the relevant regions, and the spectra can be roughly divided into two regions: a relevant region covering the 245-540 nm (variable index: 1-60) and an irrelevant region containing larger wavelengths: 545-785 nm (variable index: 61-109). Thus, variable selection can effectively increase prediction accuracy and suggest that variants of PLS that also perform variable selection should be employed (such as interval PLS, VIP-PLS). Therefore, a potentially better approach may consist in selecting the relevant spectral region and then apply PLS over it. If only this region is considered (variable index 1-60), the median $RMSE^{dcv}$ of PLS decreases from 2.0 to 1.4 years, which is the minimum value observed when compared to all other regression methods.



(a) (b)

Figure 6.10. Predictor's importance obtained for the UV-vis dataset over 50 iterations of double cross-validation: (a) the regression coefficients for LASSO and (b) the BoRT predictor importance.

6.5 Final Remarks

PAC's analytics domain contains the main representatives of four important classes of methods and we believe this provides a good coverage of the relevant methods for the purposes of screening the classes with most potential for each application. The user can then decide to use the best methods in these classes (highest KPIs) or to explore other alternatives and variants not contemplated in the analytics domain. Also, sometimes there is the need to further tune the methods from the winning classes, in order to optimize prediction ability (Broadhurst, Goodacre et al. 1997, Alsberg, Kell et al. 1998, Brás, Lopes et al. 2008, Mehmood, Liland et al. 2012) or enhance interpretability (Reis 2013, Kvalheim, Arneberg et al. 2014). An example was provided for the UV-Vis dataset, where band selection could improve the performance of the latent variable models or even the use of a non-linear extension, such as Kernel-PLS (Rosipal and Trejo 2001). The inclusion of all the possible model variants in PAC is not a realistic endeavor not even one that suits the users' interests. Therefore, these refinement methods were not included, since their adoption relies on the subsequent analysis of results from the base formulation. The proper way to use PAC is therefore as an efficient screening tool that points to the best directions from the predictive capability standpoint: the models developed may be sufficient to solve the problem, but in other occasions, more refinement is required (post-optimization). Although post-optimization is not mandatory (see Section 6.2), it is very common in practice and it is highly recommended in order to improve both prediction performance and obtain more interpretable models due to the exclusion of irrelevant predictors. Another valid alternative is to build ensemble estimators using the methods with largest KPI (Hastie, Tibshirani et al. 2001). Again, this approach should be considered in the post-optimization stage of the workflow.

In terms of computational requirements, PAC was implemented on an eight-core CPU desktop and run in parallel since double cross-validation iterations can be computed independently. The case studies considered took 10-24h to complete, and have a high dependency on the number of variables. This is particularly noticeable for tree-based ensembles since they explore the predictors' space in order to build a considerable number of regression trees. Nevertheless, the range of results provided by PAC in a single run justifies its computational burden, allowing one to easily identify promising methods. If subsequent iterations of PAC are needed, one may exclude methods that presented poor results in previous runs, greatly decreasing the computational requirements. Furthermore, as computers with more cores are becoming commonly available, applying PAC will be much faster in those machines and may correspond to rather small and tolerable time requirements.

One type of methods not included in the Analytics Domain is the family of Artificial Neural Networks. These methods do not scale well with the dimensionality of the dataset and require preliminary dimension reduction techniques before being applied to large datasets. Nevertheless, the need for non-linear modeling can be detected (and handled) through the class of tree-based ensemble models, as illustrated in the case of the wine dataset using HPLC measurements. Then, further refinements or alternative approaches can be tried out at a second stage, using kernel methods or composed mappings of dimension reduction/non-linear modeling.

Chapter VII
Conclusions & Future Work

7 Conclusions & Future Work

This last chapter summarizes the main conclusions of this thesis and highlights the main outputs of each chapter, contextualizing these outcomes in a structured and logical sequence that starts with batch datasets and generates insights for improving process operation. Moreover, future research directions are enumerated that would enlarge the set of tools currently available to practitioners, providing tailored solutions to the challenge of analyzing batch datasets.

7.1 Conclusions

In this thesis, we have addressed some of the challenges in the field of batch data analysis (BDA) and developed frameworks for analyzing datasets collected during batch operation. These frameworks were integrated following a logical workflow that starts with batch data, converts it to features (using the feature-oriented batch analytics framework – FOBA 1.0 and FOBA 2.0), select the relevant ones (using the wide spectrum feature selection - WiSe), and finally use the important selected features for building predictive models (using the predictive analytics comparison framework - PAC). This set of tools represent effective and efficient solutions that may provide insights into the process operation, ultimately helping plant and process engineers to better control, troubleshoot and improve their processes. This thesis contains six main chapters, which are summarized below.

In Chapter I, the scope, motivations, and objectives of this thesis were described. In particular, the importance of batch processes and BDA was highlighted and some of the gaps in the BDA field were identified. These drawbacks motivated the specification of goals and the work conducted in this thesis aimed at achieving such objectives. The main contribution of this thesis are also summarized in Chapter I.

In Chapter II, the main classes of methods in the field of BDA were identified and assessed in terms of their implementation and modeling complexity. This analysis provided a mapping of the different families of methods into a complexity scale, clarifying their relationships and providing practitioners with a structured workflow to guide decisions on the methods to adopt and the sequence by which different alternatives should be considered, in accordance to the parsimony principle. BDA methods were grouped in three classes, namely, feature-oriented methods, linear time-resolved methods, and non-linear time-resolved methods. Feature-oriented methods are the more parsimonious and simple approaches. Alternatively, time-resolved methods provide effective solutions when more extensive descriptions of batch phenomena are required, such as modeling more intricate correlations and lagged-correlations between process variables. Although time-resolved methods require batch synchronization, they can pinpoint with great accuracy the time instants where process upsets occur and provide online estimates of quality parameters. Non-linear time-resolved methods occupy the top rank of the complexity scale. Their application should be carefully pondered due to the risk of overfitting and the added implementation complexity.

In Chapter III, a framework for feature-oriented batch analytics platform (FOBA) was presented, unifying available approaches for feature generation. The proposed framework integrates feature generation and feature analysis for batch data in the context of continuous improvement activities. We also presented a new dictionary of features, known as Profile-driven Features (PdF). The proposed PdF dictionary lies in between the current feature-oriented approaches (e.g. SPA, TIME) and the time-resolved approaches (2-way BWU, Tucker 3, PARAFAC, etc.) in the sense that it lacks time resolution but retains the specificity of the profiles through the use of profile-specific features. Chapter IV extended FOBA 1.0 to FOBA 2.0, improving the time-resolution of available feature-oriented methods. Three alternatives were envisioned and tested, and their performances were assessed in terms of providing local information that could pinpoint critical batch periods. They successfully identified periods that were more relevant for the quality of the batch and it is expected that their application on real world problems would generate similar benefits.

Chapter V focused on feature selection methods and a novel proposal was put forward, specifically for regression problems: the wide spectrum feature selection (WiSe). WiSe is a two-stage approach that combines filters for efficient removal of irrelevant features, and a regression method for model building and further feature selection. Filters are efficient screening methods and, in total, six filters were tested in the first WiSe stage: Pearson's correlation coefficient, Spearman's rank coefficient, symmetrical uncertainty (SU), and all paired combinations of these filters. Features selected in the first stage proceed to the second stage where forward stepwise regression (FSR), least absolute shrinkage and selector operator (LASSO), and partial least squares (PLS) are used for model building. FSR and LASSO have the ability to further decrease the number of features, resulting in an even smaller feature subset. From the six filters considered, Pearson' correlation coefficient, Spearman's rank coefficient, SU, and the combination of SU and Pearson stand out as a promising subset of filters that should be tested in regression problems.

Finally, in Chapter VI, a structured comparison framework for comparing advanced predictive analytics methods was proposed, described, applied and discussed. The proposed approach, called PAC, was designed to help practitioners developing predictive models in complex Manufacturing 4.0 scenarios (and not only), in an efficient, rigorous and robust way. PAC screens for the classes with most potential for each application, leaving to the user the decision of using the best methods found, or to explore further variants of them not contemplated in the Analytics Domain, but that makes sense for the particular case under analysis. In any case, the development time is reduced and the quality of the outcomes and rankings is secured (even for for non-specialized users), through the incorporation of a rich set of methods (13 overall) and the implementation of a robust comparison engine and an effective results reporting scheme.

7.2 Future Work

Novel and innovative developments in the BDA field are expected to continue driving improvements in plant operation, namely in process optimization, control, troubleshooting, and root-cause analysis. The following bullet points describe research directions that would benefit the field of BDA and would also have a positive impact on data-driven modeling in general:

- *Feature-oriented methods*. These methods compress batch evolution into a few features. However, they have not been adapted to online applications neither combined with non-linear methods. Developments in these directions would broaden the applicability of these methods to a wider range of problems. Furthermore, we intend to extend the application of FOBA 1.0 and FOBA 2.0 to more case studies, representing different analysis scopes: visualization, troubleshooting, quality prediction, and end-of-batch process monitoring;
- *Analysis of profiles*. Besides batch data, datasets in other applications (e.g. spectra) present characteristics similar to the batch profiles. FOBA 1.0 and FOBA 2.0 may be valuable alternatives for extracting relevant features from these profiles.

- *Multiresolution non-linear methods*. Non-linear methods can greatly benefit from using multiresolution frameworks to reduce the time-resolution of the data-driven analysis. The advantage is that the number of model parameters can be significantly reduced, mitigating the propensity of non-linear methods towards overfitting;
- *BDA software*. The literature on BDA is vast and increases each year with new proposals that attempt to further advance the state-of-the-art. However, the majority of the currently proposed methods are not available for practitioners and researchers to test, nor are they easily programmed and implemented. Therefore, time is "wasted" implementing methods that others have already developed, and the comparison of potential methods becomes a daunting task, usually beyond the time and technical resources available in the companies. Initiatives in line with reproducible research should be promoted and, ideally, a software platform developed where existing methods and new entries could be implemented and tested, greatly benefiting the increasing community of researchers and practitioners interested in BDA;
- *Developing training programs*. The complexity scale of BDA correlates well with the needs required at the different organization levels and the skills and expertise available at each level. Therefore, it provides a sound way to design and implement company-wide training programs, that maximize the impact of data analytics in the bottom line results;
- *Non-linear modeling*. Selecting among different non-linear methods is a challenging task since it is more difficult to find an underlying reasoning to segment them. Nevertheless, it is expected that industry 4.0 initiatives will foster their application due to the larger amounts of data that will be available, which attenuate some of the overfitting concerns of implementing non-linear approaches;
- *Fine-tuning PAC*. One area for future research is the use of PAC's outcome in a post-optimization stage, where, for instance, ensembles of different regression methods can be developed;
- *Sparsity and collinearity metrics*. Another research line consists in the exploitation of the accumulated information generated by PAC for establishing a knowledge base for direct methods' selection. For such, sparsity and collinearity metrics should be developed to characterize datasets and enable the generation of guidelines for the selection of suitable regression methods. Furthermore, an assessment of the computational complexity of the method should also be

conducted, especially for large datasets where some methods can take considerable time to run (especially the tree-based ensemble methods).

References

Aguado, D., A. Ferrer, A. Seco and J. Ferrer (2006). *Comparison of different predictive models for nutrient estimation in a sequencing batch reactor for wastewater treatment*. Chemometrics and intelligent laboratory systems 84(1): 75-81.

Ahmed, N. K., A. F. Atiya, N. E. Gayar and H. El-Shishiny (2010). *An empirical comparison of machine learning models for time series forecasting*. Econometric Reviews 29(5-6): 594-621.

Alsberg, B. K., D. B. Kell and R. Goodacre (1998). *Variable Selection in Discriminant Partial Least-Squares Analysis*. Anal. Chem. 70: 4126-4133.

Andersen, C. M. and R. Bro (2010). *Variable selection in regression—a tutorial*. Journal of Chemometrics 24(11-12): 728-737.

Anzanello, M. J. and F. S. Fogliatto (2014). *A review of recent variable selection methods in industrial and chemometrics applications*. European Journal of Industrial Engineering 8(5): 619-645.

Arteaga, F. and A. Ferrer (2002). *Dealing with missing data in MSPC: several methods, different interpretations, some examples*. Journal of chemometrics 16(8-10): 408-418.

Auger, I. E. and C. E. Lawrence (1989). *Algorithms for the optimal identification of segment neighborhoods*. Bulletin of mathematical biology 51(1): 39-54.

Benjamini, Y. and Y. Hochberg (1995). *Controlling the false discovery rate: a practical and powerful approach to multiple testing*. Journal of the royal statistical society. Series B (Methodological): 289-300.

Birol, G., C. Ündey and A. Cinar (2002). *A modular simulation package for fed-batch fermentation: penicillin production*. Computers & Chemical Engineering 26(11): 1553-1565.

Bollen, J., H. Mao and X. Zeng (2011). *Twitter mood predicts the stock market*. Journal of computational science 2(1): 1-8.

Bolón-Canedo, V., N. Sánchez-Maroño and A. Alonso-Betanzos (2013). *A review of feature selection methods on synthetic data*. Knowledge and information systems 34(3): 483-519.

Bolón-Canedo, V., N. Sánchez-Marono, A. Alonso-Betanzos, J. M. Benítez and F. Herrera (2014). *A review of microarray datasets and applied feature selection methods*. Information Sciences 282: 111-135.

Boqué, R. and A. K. Smilde (1999). *Monitoring and diagnosing batch processes with multiway covariates regression models*. AIChE Journal 45(7): 1504-1520.

Box, G. E. P., J. S. Hunter and W. G. Hunter (2005). *Statistics for Experimenters: Design, Innovation, and Discovery*. Hoboken, NJ (USA), Wiley.

Braga, J. W. B., A. A. dos Santos Junior and I. S. Martins (2014). *Determination of viscosity index in lubricant oils by infrared spectroscopy and PLSR*. Fuel 120: 171-178.

Brás, L. P., M. Lopes, A. P. Ferreira and J. C. Menezes (2008). *A bootstrap-based strategy for spectral interval selection in PLS regression*. Journal of Chemometrics 22(11-12): 695-700.

Breiman, L., J. Friedman, C. J. Stone and R. A. Olshen (1984). *Classification and regression trees*, CRC press.

Bro, R. (1997). *PARAFAC. Tutorial and applications*. Chemometrics and intelligent laboratory systems 38(2): 149-171.

Broadhurst, D., R. Goodacre, A. Jones, J. J. Rowland and D. B. Kell (1997). *Genetic Algorithms as a Method for Variable Selection in Multiple Linear Regression and Partial Least Squares Regression, with Applications to Pyrolysis Mass Spectrometry*. Analytica Chimica Acta 348: 71-86.

Burnham, A. J., J. F. MacGregor and R. Viveros (1999). *Latent variable multivariate regression modeling*. Chemometrics and Intelligent Laboratory Systems 48(2): 167-180.

Burnham, A. J., J. F. MacGregor and R. Viveros (1999). *Latent Variable Multivariate Regression Modeling*. Chemometrics and Intelligent Laboratory Systems 48: 167-180.

Burnham, A. J., J. F. MacGregor and R. Viveros (2001). *Interpretation of Regression Coefficients under a Latent Variable Regression Model. Journal Of Chemometrics*. 15**:** 265-284.

Burnham, A. J., R. Viveros and J. F. MacGregor (1996). *Frameworks for latent variable multivariate regression*. Journal of chemometrics 10(1): 31-45.

Camacho, J. and J. Picó (2006). *Multi-phase principal component analysis for batch processes modelling*. Chemometrics and Intelligent Laboratory Systems 81(2): 127-136.

Camacho, J., J. Pico and A. Ferrer (2008). *Bilinear modelling of batch processes. Part I: theoretical discussion*. Journal of Chemometrics 22(5): 299-308.

Camacho, J., J. Picó and A. Ferrer (2008). *Multi-phase analysis framework for handling batch process data*. Journal of Chemometrics 22(11-12): 632-643.

Camacho, J., J. Picó and A. Ferrer (2009). *The best approaches in the on-line monitoring of batch processes based on PCA: Does the modelling structure matter?* Analytica chimica acta 642(1): 59-68.

Canu, S., Grandvalet, Y., Guigue, V., Rakotomamonjy, A. (2005). *SVM and Kernel Methods Matlab Toolbox*. Perception Systemes et Information, INSA de Rouen, Rouen, France.

Cao, D.-S., Y.-Z. Liang, Q.-S. Xu, Q.-N. Hu, L.-X. Zhang and G.-H. Fu (2011). *Exploring nonlinear relationships in chemical data using kernel-based methods*. Chemometrics and Intelligent Laboratory Systems 107(1): 106-115.

Cao, D.-S., Q.-S. Xu, Y.-Z. Liang, L.-X. Zhang and H.-D. Li (2010). *The boosting: A new idea of building models*. Chemometrics and Intelligent Laboratory Systems 100(1): 1-11.

Carroll, J. D. and J.-J. Chang (1970). *Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition*. Psychometrika 35(3): 283-319.

Chandrashekar, G. and F. Sahin (2014). *A survey on feature selection methods*. Computers & Electrical Engineering 40(1): 16-28.

Chatterjee, S. and A. S. Hadi (2015). *Regression analysis by example*, John Wiley & Sons.

Chen, H., R. H. Chiang and V. C. Storey (2012). *Business intelligence and analytics: From big data to big impact*. MIS quarterly 36(4): 1165-1188.

Chen, J. and H.-H. Chen (2006). *On-line batch process monitoring using MHMT-based MPCA*. Chemical Engineering Science 61(10): 3223-3239.

Chen, J. and A. K. Gupta (1997). *Testing and Locating Variance Changepoints with Application to Stock Prices*. Journal of the American Statistical Association 92(438): 739-747.

Chen, J. and A. K. Gupta (2011). *Parametric statistical change point analysis: with applications to genetics, medicine, and finance*, Springer Science & Business Media.

Chen, J. and Y.-C. Jiang (2011). *Hidden semi-Markov probability models for monitoring two-dimensional batch operation*. Industrial & Engineering Chemistry Research 50(6): 3345-3355.

Chen, J. and K.-C. Liu (2002). *On-line batch process monitoring using dynamic PCA and dynamic PLS models*. Chemical Engineering Science 57(1): 63-75.

Chiang, L., B. Lu and I. Castillo (2017). *Big Data Analytics in Chemical Engineering*. Annual Review of Chemical and Biomolecular Engineering 8(1): 63-85.

Cho, J.-H., J.-M. Lee, S. Wook Choi, D. Lee and I.-B. Lee (2005). *Fault identification for process monitoring using kernel principal component analysis*. Chemical Engineering Science 60(1): 279-288.

Choi, S. W. and I.-B. Lee (2004). *Nonlinear dynamic process monitoring based on dynamic kernel PCA*. Chemical engineering science 59(24): 5897-5908.

Choi, S. W., J. Morris and I.-B. Lee (2008). *Dynamic model-based batch process monitoring*. Chemical Engineering Science 63(3): 622-636.

Chong, I.-G. and C.-H. Jun (2005). *Performance of some variable selection methods when multicollinearity is present*. Chemometrics and Intelligent Laboratory Systems 78(1–2): 103-112.

Cinar, A., S. J. Parulekar, C. Undey and G. Birol (2003). *Batch fermentation: modeling: monitoring, and control*, CRC press.

Colquhoun, D. (2014). *An investigation of the false discovery rate and the misinterpretation of p-values*. Royal Society open science 1(3): 140216.

Conlin, A., E. Martin and A. Morris (2000). *Confidence limits for contribution plots*. Journal of Chemometrics 14(5-6): 725-736.

Cozzolino, D., M. Kwiatkowski, R. Dambergs, W. Cynkar, L. Janik, G. Skouroumounis and M. Gishen (2008). *Analysis of elements in wine using near infrared spectroscopy and partial least squares regression*. Talanta 74(4): 711-716.

Das, A., J. Maiti and R. Banerjee (2012). *Process monitoring and fault detection strategies: a review*. International Journal of Quality & Reliability Management 29(7): 720-752.

Desai, K., Y. Badhe, S. S. Tambe and B. D. Kulkarni (2006). *Soft-sensor development for fed-batch bioreactors using support vector regression*. Biochemical Engineering Journal 27(3): 225-239.

Dietterich, T. G. (2000). Ensemble methods in machine learning. *Multiple classifier systems*, Springer**:** 1-15.

Dong, D. and T. J. McAvoy (1996). *Nonlinear principal component analysis—based on principal curves and neural networks*. Computers & Chemical Engineering 20(1): 65-78.

Draper, N. R. and H. Smith (1998). *Applied regression analysis*, New York: Wiley.

Dunia, R. and S. Joe Qin (1998). *Subspace approach to multidimensional fault identification and reconstruction*. AIChE Journal 44(8): 1813-1831.

Eilers, P. H. C. (2017). *Uncommon penalties for common problems*. Journal of Chemometrics: e2878-n/a.

El Akadi, A., A. Amine, A. El Ouardighi and D. Aboutajdine (2011). *A two-stage gene selection scheme utilizing MRMR filter and GA wrapper*. Knowledge and Information Systems 26(3): 487-500.

Elith, J., J. R. Leathwick and T. Hastie (2008). *A working guide to boosted regression trees*. Journal of Animal Ecology 77(4): 802-813.

Erevelles, S., N. Fukawa and L. Swayne (2016). *Big Data consumer analytics and the transformation of marketing*. Journal of Business Research 69(2): 897-904.

Estévez, P. A., M. Tesmer, C. A. Perez and J. M. Zurada (2009). *Normalized mutual information feature selection*. IEEE Transactions on Neural Networks 20(2): 189-201.

Ferreira, A. J. and M. A. Figueiredo (2012). *Efficient feature selection filters for high-dimensional data*. Pattern Recognition Letters 33(13): 1794-1804.

Fransson, M. and S. Folestad (2006). *Real-time alignment of batch process data using COW for on-line process monitoring*. Chemometrics and intelligent laboratory systems 84(1): 56-61.

Freund, Y., R. Schapire and N. Abe (1999). *A short introduction to boosting*. Journal-Japanese Society For Artificial Intelligence 14(771-780): 1612.

Freund, Y. and R. E. Schapire (1996). *Experiments with a new boosting algorithm*. ICML.

Fu, T.-c. (2011). *A review on time series data mining*. Engineering Applications of Artificial Intelligence 24(1): 164-181.

Gallagher, N. B., B. M. Wise and C. W. Stewart (1996). *Application of multi-way principal components analysis to nuclear waste storage tank monitoring*. Computers & chemical engineering 20: S739-S744.

García-Muñoz, S., T. Kourti and J. F. MacGregor (2004). *Model predictive monitoring for batch processes*. Industrial & engineering chemistry research 43(18): 5929-5941.

García-Muñoz, S., T. Kourti, J. F. MacGregor, A. G. Mateos and G. Murphy (2003). *Troubleshooting of an Industrial Batch Process Using Multivariate Methods*. Industrial & Engineering Chemistry Research 42(15): 3592-3601.

García-Muñoz, S. and J. MacGregor (2016). *Big data: success stories in the process industries*. Chem. Eng. Prog.: 36-40.

García-Muñoz, S., M. Polizzi, A. Prpich, C. Strain, A. Lalonde and V. Negron (2011). *Experiences in batch trajectory alignment for pharmaceutical process improvement through multivariate latent variable modelling*. Journal of Process Control 21(10): 1370-1377.

Gautheir, T. D. (2001). *Detecting trends using Spearman's rank correlation coefficient*. Environmental forensics 2(4): 359-362.

Ge, Z., F. Gao and Z. Song (2011). *Batch process monitoring based on support vector data description method*. Journal of Process Control 21(6): 949-959.

Ge, Z. and Z. Song (2007). *Process monitoring based on independent component analysis− principal component analysis (ICA− PCA) and similarity factors*. Industrial & Engineering Chemistry Research 46(7): 2054-2063.

Ge, Z. and Z. Song (2013). *Bagging support vector data description model for batch process monitoring*. Journal of Process Control 23(8): 1090-1096.

Ge, Z., Z. Song and F. Gao (2013). *Review of recent research on data-based process monitoring*. Industrial & engineering chemistry research 52(10): 3543-3562.

Geladi, P. and B. R. Kowalski (1986). *Partial least-squares regression: a tutorial.* Analytica Chimica Acta 185: 1-17.

Gibbons, J. D. and S. Chakraborti (2011). Nonparametric statistical inference. *International encyclopedia of statistical science*, Springer**:** 977-979.

Gins, G., P. Van den Kerkhof and J. F. M. Van Impe (2012). *Hybrid Derivative Dynamic Time Warping for Online Industrial Batch-End Quality Estimation*. Industrial & Engineering Chemistry Research 51(17): 6071-6084.

Gins, G., J. F. M. Van Impe and M. S. Reis (2018). *Finding the optimal time resolution for batch-end quality prediction: MRQP – A framework for multi-resolution quality prediction*. Chemometrics and Intelligent Laboratory Systems 172: 150-158.

Glickman, M. E., S. R. Rao and M. R. Schultz (2014). *False discovery rate control is a recommended alternative to Bonferroni-type adjustments in health studies*. Journal of clinical epidemiology 67(8): 850-857.

González-Martínez, J., A. Ferrer and J. Westerhuis (2011). *Real-time synchronization of batch trajectories for on-line multivariate statistical process control using Dynamic Time Warping*. Chemometrics and Intelligent Laboratory Systems 105(2): 195-206.

González-Martínez, J. M., O. E. De Noord and A. Ferrer (2014). *Multisynchro: a novel approach for batch synchronization in scenarios of multiple asynchronisms*. Journal of Chemometrics 28(5): 462-475.

Gonzalez-Martinez, J. M., R. Vitale, O. E. de Noord and A. Ferrer (2014). *Effect of synchronization on bilinear batch process modeling*. Industrial & Engineering Chemistry Research 53(11): 4339-4351.

González-Martínez, J. M., J. A. Westerhuis and A. Ferrer (2013). *Using warping information for batch process monitoring and fault classification*. Chemometrics and Intelligent Laboratory Systems 127: 210-217.

Gurden, S. P., J. A. Westerhuis, R. Bro and A. K. Smilde (2001). *A comparison of multiway regression and scaling methods*. Chemometrics and Intelligent Laboratory Systems 59(1): 121-136.

Guyon, I. and A. Elisseeff (2003). *An introduction to variable and feature selection*. Journal of machine learning research 3(Mar): 1157-1182.

Hall, M. A. and L. A. Smith (1999). *Feature selection for machine learning: comparing a correlation-based filter approach to the wrapper*. FLAIRS conference.

Harshman, R. A. (1970). *Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multimodal factor analysis*.

Hastie, T., R. Tibshirani and J. Friedman (2001). *The Elements of Statistical Learning*. NY, Springer.

Hauke, J. and T. Kossowski (2011). *Comparison of values of Pearson's and Spearman's correlation coefficients on the same sets of data*. Quaestiones geographicae 30(2): 87-93.

He, Q. P. and J. Wang (2011). *Statistics pattern analysis: A new process monitoring framework and its application to semiconductor batch processes*. AIChE Journal 57(1): 107-121.

Hesterberg, T., N. H. Choi, L. Meier and C. Fraley (2008). *Least angle and ℓ1 penalized regression: A review*. Statistics Surveys 2: 61-93.

Hoerl, A. E. and R. W. Kennard (1970). *Ridge Regression: Biased Estimation for Nonorthogonal Problems*. Technometrics 12(1): 55-67.

Hoffman, B., S. J. Cho, W. Zheng, S. Wyrick, D. E. Nichols, R. B. Mailman and A. Tropsha (1999). *Quantitative Structure− Activity Relationship Modeling of Dopamine D1 Antagonists Using Comparative Molecular Field Analysis, Genetic Algorithms− Partial Least-Squares, and K Nearest Neighbor Methods*. Journal of medicinal chemistry 42(17): 3217-3226.

Hong, J. J., J. Zhang and J. Morris (2011). *Fault localization in batch processes through progressive principal component analysis modeling*. Industrial & Engineering Chemistry Research 50(13): 8153-8162.

Höskuldsson, A. (1996). *Prediction Methods in Science and Technology*, Thor Publishing.

Höskuldsson, A. (2001). *Variable and Subset Selection in PLS Regression*. Chemometrics and Intelligent Laboratory Systems 55: 23-38.

Hu, K. and J. Yuan (2008). *Multivariate statistical process control based on multiway locality preserving projections*. Journal of Process Control 18(7-8): 797-807.

Hu, K. and J. Yuan (2008). *Statistical monitoring of fed-batch process using dynamic multiway neighborhood preserving embedding*. Chemometrics and Intelligent Laboratory Systems 90(2): 195-203.

Jackson, J. E. (2005). *A user's guide to principal components*, John Wiley & Sons.

Jaynes, E. T. (1957). *Information theory and statistical mechanics*. Physical review 106(4): 620.

Jia, M., F. Chu, F. Wang and W. Wang (2010). *On-line batch process monitoring using batch dynamic kernel principal component analysis*. Chemometrics and Intelligent Laboratory Systems 101(2): 110-122.

Johnstone, I. M. and D. M. Titterington (2009). *Statistical challenges of high-dimensional data*, The Royal Society.

Jolliffe, I. (2002). *Principal component analysis*, Wiley Online Library.

Kassidas, A., J. F. MacGregor and P. A. Taylor (1998). *Synchronization of batch trajectories using dynamic time warping*. AIChE Journal 44(4): 864-875.

Keogh, E., S. Chu, D. Hart and M. Pazzani (2001). *An online algorithm for segmenting time series*. Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on, IEEE.

Keogh, E. J. and M. J. Pazzani (2001). *Derivative dynamic time warping*. Proceedings of the 2001 SIAM International Conference on Data Mining, SIAM.

Kiers, H. A. (1991). *Hierarchical relations among three-way methods*. Psychometrika 56(3): 449-470.

Killick, R., P. Fearnhead and I. A. Eckley (2012). *Optimal detection of changepoints with a linear computational cost*. Journal of the American Statistical Association 107(500): 1590-1598.

Kim, Y. S. (2008). *Comparison of the decision tree, artificial neural network, and linear regression methods based on the number and types of independent variables and sample size*. Expert Systems with Applications 34(2): 1227-1234.

Kira, K. and L. A. Rendell (1992). *The feature selection problem: Traditional methods and a new algorithm*. Aaai.

Kononenko, I. (1994). *Estimating attributes: analysis and extensions of RELIEF*. European conference on machine learning, Springer.

Kourti, T. (2002). *Process analysis and abnormal situation detection: from theory to practice*. IEEE Control Systems 22(5): 10-25.

Kourti, T. (2003). *Abnormal situation detection, three-way data and projection methods; robust data archiving and modeling for industrial applications*. Annual Reviews in control 27(2): 131-139.

Kourti, T. (2003). *Multivariate dynamic data modeling for analysis and statistical process control of batch processes, start-ups and grade transitions*. Journal of Chemometrics 17(1): 93-109.

Kourti, T. (2005). *Application of latent variable methods to process control and multivariate statistical process control in industry*. International Journal of Adaptive Control and Signal Processing 19(4): 213-246.

Kourti, T., J. Lee and J. F. Macgregor (1996). *Experiences with industrial applications of projection methods for multivariate statistical process control*. Computers & Chemical Engineering 20: S745-S750.

Kourti, T. and J. F. MacGregor (1995). *Process analysis, monitoring and diagnosis, using multivariate projection methods*. Chemometrics and intelligent laboratory systems 28(1): 3-21.

Kourti, T., P. Nomikos and J. F. MacGregor (1995). *Analysis, monitoring and fault diagnosis of batch processes using multiblock and multiway PLS*. Journal of process control 5(4): 277-284.

Ku, W., R. H. Storer and C. Georgakis (1995). *Disturbance detection and isolation by dynamic principal component analysis*. Chemometrics and Intelligent Laboratory Systems 30(1): 179-196.

Kvalheim, O. M., R. Arneberg, O. Bleie, T. Rajalahti, A. K. Smilde and J. A. Westerhuis (2014). *Variable importance in latent variable regression models*. Journal of Chemometrics 28(8): 615-622.

Lavielle, M. (2005). *Using penalized contrasts for the change-point problem*. Signal Processing 85(8): 1501-1510.

Lazar, C., J. Taminau, S. Meganck, D. Steenhoff, A. Coletta, C. Molter, V. de Schaetzen, R. Duque, H. Bersini and A. Nowe (2012). *A survey on filter techniques for feature selection in gene expression microarray analysis*. IEEE/ACM Transactions on Computational Biology and Bioinformatics 9(4): 1106-1119.

Leardi, R. (2007). *Genetic algorithms in chemistry*. Journal of Chromatography A 1158(1): 226-233.

Leardi, R., R. Boggia and M. Terrile (1992). *Genetic algorithms as a strategy for feature selection*. Journal of chemometrics 6(5): 267-281.

Leardi, R. and A. L. Gonzalez (1998). *Genetic algorithms applied to feature selection in PLS regression: how and when to use them*. Chemometrics and intelligent laboratory systems 41(2): 195-207.

LeCun, Y., Y. Bengio and G. Hinton (2015). *Deep learning*. Nature 521(7553): 436-444.

Lee, J.-M., C. Yoo, S. W. Choi, P. A. Vanrolleghem and I.-B. Lee (2004). *Nonlinear process monitoring using kernel principal component analysis*. Chemical Engineering Science 59(1): 223-234.

Lee, J.-M., C. Yoo and I.-B. Lee (2004). *Fault detection of batch processes using multiway kernel principal component analysis*. Computers & chemical engineering 28(9): 1837-1847.

Lee, J., E. Lapira, B. Bagheri and H.-a. Kao (2013). *Recent advances and trends in predictive manufacturing systems in big data environment*. Manufacturing Letters 1(1): 38-41.

Lennox, B., G. Montague, H. Hiden, G. Kornfeld and P. Goulding (2001). *Process monitoring of an industrial fed-batch fermentation*. Biotechnology and Bioengineering 74(2): 125-135.

Li, J., K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang and H. Liu (2016). *Feature selection: A data perspective*. arXiv preprint arXiv:1601.07996.

Li, J., F. Tao, Y. Cheng and L. Zhao (2015). *Big Data in product lifecycle management*. The International Journal of Advanced Manufacturing Technology 81(1): 667-684.

Little, R. J. A. and D. B. Rubin (2002). *Statistical Analysis with Missing Data*. Hoboken (NJ), Wiley.

Liu, H. and H. Motoda (2012). *Feature selection for knowledge discovery and data mining*, Springer Science & Business Media.

Liu, H. and L. Yu (2005). *Toward integrating feature selection algorithms for classification and clustering*. IEEE Transactions on knowledge and data engineering 17(4): 491-502.

Louwerse, D. and A. Smilde (2000). *Multivariate statistical process control of batch processes based on three-way models*. Chemical Engineering Science 55(7): 1225-1235.

Lu, B., I. Castillo, L. Chiang and T. F. Edgar (2014). *Industrial PLS model variable selection using moving window variable importance in projection*. Chemometrics and Intelligent Laboratory Systems 135: 90-109.

Lu, N. and F. Gao (2005). *Stage-based process analysis and quality prediction for batch processes*. Industrial & engineering chemistry research 44(10): 3547-3555.

Lu, N., F. Gao and F. Wang (2004). *Sub-PCA modeling and on-line monitoring strategy for batch processes*. AIChE Journal 50(1): 255-259.

MacGregor, J. and A. Cinar (2012). *Monitoring, fault diagnosis, fault-tolerant control and optimization: Data driven methods*. Computers & Chemical Engineering 47(Supplement C): 111-120.

Mahesh, S., D. Jayas, J. Paliwal and N. White (2015). *Comparison of partial least squares regression (PLSR) and principal components regression (PCR) methods for protein and hardness predictions using the near-infrared (NIR) hyperspectral images of bulk samples of Canadian wheat*. Food and bioprocess technology 8(1): 31-40.

Marini, F., R. Bucci, A. L. Magrì and A. D. Magrì (2008). *Artificial neural networks in chemometrics: History, examples and perspectives*. Microchemical Journal 88(2): 178-185.

Marjanovic, O., B. Lennox, D. Sandoz, K. Smith and M. Crofts (2006). *Real-time monitoring of an industrial batch process*. Computers & chemical engineering 30(10): 1476-1481.

Martens, H. (2015). *Quantitative Big Data: where chemometrics can contribute*. Journal of Chemometrics 29(11): 563-581.

Martens, H. and B.-H. Mevik (2001). *Understanding the Collinearity Problem in Regression and Discriminant Analysis*. Journal of Chemometrics 15: 413-426.

Martens, H. and T. Naes (1989). *Multivariate Calibration*. Chichester, Wiley.

Martin, E., A. Morris, M. Papazoglou and C. Kiparissides (1996). *Batch process monitoring for consistent production*. Computers & chemical engineering 20: S599-S604.

Matero, S., S. Poutiainen, J. Leskinen, S.-P. Reinikainen, J. Ketolainen, K. Järvinen and A. Poso (2009). *Monitoring the wetting phase of fluidized bed granulation process using multi-way methods: The separation of successful from unsuccessful batches*. Chemometrics and Intelligent Laboratory Systems 96(1): 88-93.

Mehmood, T., K. H. Liland, L. Snipen and S. Sæbø (2012). *A review of variable selection methods in Partial Least Squares Regression*. Chemometrics and Intelligent Laboratory Systems 118: 62-69.

Miletic, I., S. Quinn, M. Dudzic, V. Vaculik and M. Champagne (2004). *An industrial perspective on implementing on-line applications of multivariate statistics*. Journal of Process Control 14(8): 821-836.

Montgomery, D. C. (2009). *Design and Analysis of Experiments*. Asia, John Wiley & Sons.

Montgomery, D. C., E. A. Peck and G. G. Vining (2012). *Introduction to linear regression analysis*, John Wiley & Sons.

Montgomery, D. C. and G. C. Runger (2010). *Applied statistics and probability for engineers*, John Wiley & Sons.

Moţ, A. C., F. Soponar, A. Medvedovici and C. Sârbu (2010). *Simultaneous Spectrophotometric Determination of Aspirin, Paracetamol, Caffeine, and Chlorphenamine from Pharmaceutical Formulations Using Multivariate Regression Methods*. Analytical Letters 43(5): 804-813.

Muhammad Aliyu, S. (2015). *Feature Selection with Mutual Information for Regression Problems*.

Naes, T. and B.-H. Mevik (2001). *Understanding the Collinearity Problem in Regression and Discriminant Analysis* Journal of Chemometrics 15: 413-436.

Nomikos, P. and J. F. MacGregor (1994). *Monitoring batch processes using multiway principal component analysis*. AIChE Journal 40(8): 1361-1375.

Nomikos, P. and J. F. MacGregor (1995). *Multi-way partial least squares in monitoring batch processes*. Chemometrics and intelligent laboratory systems 30(1): 97-108.

Nomikos, P. and J. F. MacGregor (1995). *Multivariate SPC charts for monitoring batch processes*. Technometrics 37(1): 41-59.

Nørgaard, L., A. Saudland, J. Wagner, J. P. Nielsen, L. Munck and S. B. Engelsen (2000). *Interval partial least-squares regression (iPLS): a comparative chemometric study with an example from near-infrared spectroscopy*. Applied Spectroscopy 54(3): 413-419.

Pal, S. K. and P. Mitra (2004). *Pattern recognition algorithms for data mining*, CRC press.

Peng, H., F. Long and C. Ding (2005). *Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy*. IEEE Transactions on pattern analysis and machine intelligence 27(8): 1226-1238.

Peng, Y., Z. Wu and J. Jiang (2010). *A novel feature selection approach for biomedical data classification*. Journal of Biomedical Informatics 43(1): 15-23.

Pereira, A. C., M. S. Reis, P. M. Saraiva and J. C. Marques (2010). *Analysis and assessment of Madeira wine ageing over an extended time period through GC–MS and chemometric analysis*. Analytica Chimica Acta 659: 93-101.

Pereira, A. C., M. S. Reis, P. M. Saraiva and J. C. Marques (2011). *Madeira wine ageing prediction based on different analytical techniques: UV–vis, GC-MS, HPLC-DAD*. Chemometrics and Intelligent Laboratory Systems 105(1): 43-55.

Pinheiro, C. T., R. R. Rendall, M. J. Quina, M. S. Reis and L. M. Gando-Ferreira (2016). *Assessment and Prediction of Lubricant Oil Properties Using Infrared Spectroscopy and Advanced Predictive Analytics*. Energy & Fuels.

Press, W. H., S. A. Teukolsky, W. T. Vetterling and B. P. Flannery (1996). *Numerical recipes in C*, Cambridge university press Cambridge.

Qin, S. J. (2014). *Process data analytics in the era of big data*. AIChE Journal 60(9): 3092-3100.

Ramaker, H.-J., E. N. van Sprang, J. A. Westerhuis and A. K. Smilde (2005). *Fault detection properties of global, local and time evolving models for batch process monitoring*. Journal of Process control 15(7): 799-805.

Rännar, S., J. F. MacGregor and S. Wold (1998). *Adaptive batch monitoring using hierarchical PCA*. Chemometrics and intelligent laboratory systems 41(1): 73-81.

Rasmussen, M. A. and R. Bro (2012). *A tutorial on the Lasso approach to sparse modeling*. Chemometrics and Intelligent Laboratory Systems 119: 21-31.

Rato, T. J., J. Blue, J. Pinaton and M. S. Reis (2017). *Translation-Invariant Multiscale Energy-Based PCA for Monitoring Batch Processes in Semiconductor Manufacturing*. IEEE Transactions on Automation Science and Engineering 14(2): 894-904.

Rato, T. J. and M. S. Reis (2013). *Fault detection in the Tennessee Eastman benchmark process using dynamic principal components analysis based on decorrelated residuals (DPCA-DR)*. Chemometrics and Intelligent Laboratory Systems 125: 101-108.

Rato, T. J. and M. S. Reis (2017). *Multiresolution Soft Sensors (MR-SS): A New Class of Model Structures for Handling Multiresolution Data*. Industrial & Engineering Chemistry Research 56(13): 3640-3654.

Rato, T. J. and M. S. Reis (2018). *Optimal Selection of Time Resolution for Batch Data Analysis. Part I: Predictive Modelling*. AIChE Journal (Accepted).

Rato, T. J., R. Rendall, V. Gomes, S.-T. Chin, L. H. Chiang, P. M. Saraiva and M. S. Reis (2016). *A Systematic Methodology for Comparing Batch Process Monitoring Methods: Part I—Assessing Detection Strength*. Industrial & Engineering Chemistry Research 55(18): 5342-5358.

Rato, T. J., R. Rendall, V. Gomes, P. M. Saraiva and M. S. Reis (2018). *A Systematic Methodology for Comparing Batch Process Monitoring Methods: Part II—Assessing Detection Speed*. Industrial & Engineering Chemistry Research 57(15): 5338-5350.

Reis, M. S. (2013). *Applications of a new empirical modelling framework for balancing model interpretation and prediction accuracy through the incorporation of clusters of functionally related variables*. Chemometrics and Intelligent Laboratory Systems 127: 7-16.

Reis, M. S., R. D. Braatz and L. H. Chiang (2016). *Challenges and Future Research Directions*. Chemical Engineering Progress 112(3): 46-50.

Reis, M. S. and R. S. Kenett (2018). *Assessing the Value of Information of Data-Centric Activities in the Chemical Processing Industry 4.0*. AIChE Journal (In Press).

Reis, M. S., R. Rendall, S.-T. Chin and L. H. Chiang (2015). *Challenges in the specification and integration of measurement uncertainty in the development of data-driven models for the chemical processing industry*. Industrial & Engineering Chemistry Research 54: 9159-9177.

Reis, M. S. and P. M. Saraiva (2004). *A Comparative Study of Linear Regression Methods in Noisy Environments*. Journal of Chemometrics 18(12): 526-536.

Reis, M. S. and P. M. Saraiva (2005). *Integration of Data Uncertainty in Linear Regression and Process Optimization*. AIChE Journal 51(11): 3007-3019.

Reiss, R., W. Wojsznis and R. Wojewodka (2010). *Partial least squares confidence interval calculation for industrial end-of-batch quality prediction*. Chemometrics and Intelligent Laboratory Systems 100(2): 75-82.

Rendall, R., B. Lu, I. Castillo, S.-T. Chin, L. H. Chiang and M. S. Reis (2017). *A Unifying and Integrated Framework for Feature Oriented Analysis of Batch Processes*. Industrial & Engineering Chemistry Research 56(30): 8590-8605.

Rendall, R., A. C. Pereira and M. S. Reis (2016). *Advanced predictive methods for wine age prediction: Part I–A comparison study of single-block regression approaches based on variable selection, penalized regression, latent variables and tree-based ensemble methods*. Talanta.

Rinnan, Å., M. Andersson, C. Ridder and S. B. Engelsen (2014). *Recursive weighted partial least squares (rPLS): an efficient variable selection method using PLS*. Journal of Chemometrics 28(5): 439-447.

Robnik-Šikonja, M. and I. Kononenko (1997). *An adaptation of Relief for attribute estimation in regression*. Machine Learning: Proceedings of the Fourteenth International Conference (ICML'97).

Robnik-Šikonja, M. and I. Kononenko (2003). *Theoretical and empirical analysis of ReliefF and RReliefF*. Machine learning 53(1-2): 23-69.

Rosipal, R. and L. J. Trejo (2001). *Kernel Partial Least Squares Regression in Reproducing Kernel Hilbert Space*. Journal of Machine Learning Research 2: 97-123.

Ruiz, F. E., P. S. Pérez and B. I. Bonev (2009). *Information theory in computer vision and pattern recognition*, Springer Science & Business Media.

Saeys, Y., I. Inza and P. Larrañaga (2007). *A review of feature selection techniques in bioinformatics*. bioinformatics 23(19): 2507-2517.

Scott, A. J. and M. Knott (1974). *A Cluster Analysis Method for Grouping Means in the Analysis of Variance*. Biometrics 30(3): 507-512.

Seasholtz, M. B. and B. Kowalski (1993). *The Parsimony Principle Applied to Multivariate Calibration*. Analytica Chimica Acta 277: 165-177.

Shams, I., S. Ajorlou and K. Yang (2015). *A predictive analytics approach to reducing 30-day avoidable readmissions among patients with heart failure, acute myocardial infarction, pneumonia, or COPD*. Health Care Management Science 18(1): 19-34.

Shannon, C. E. (2001). *A mathematical theory of communication*. ACM SIGMOBILE Mobile Computing and Communications Review 5(1): 3-55.

Sharif, B., D. Makowski, F. Plauborg and J. E. Olesen (2017). *Comparison of regression techniques to predict response of oilseed rape yield to variation in climatic conditions in Denmark*. European Journal of Agronomy 82: 11-20.

Smola, A. J. and B. Schölkopf (2004). *A tutorial on support vector regression*. Statistics and computing 14(3): 199-222.

Strobl, C., J. Malley and G. Tutz (2009). *An introduction to recursive partitioning: rationale, application, and characteristics of classification and regression trees, bagging, and random forests*. Psychological methods 14(4): 323.

Stubbs, S., J. Zhang and J. Morris (2013). *Multiway Interval Partial Least Squares for Batch Process Performance Monitoring*. Industrial & Engineering Chemistry Research 52(35): 12399-12407.

Tian, X., X. Zhang, X. Deng and S. Chen (2009). *Multiway kernel independent component analysis based on feature samples for batch process monitoring*. Neurocomputing 72(7): 1584-1596.

Tibshirani, R. (1996). *Regression Shrinkage and Selection via the Lasso*. Journal of the Royal Statistical Society. Series B (Methodological) 58(1): 267-288.

Truong, C., L. Oudre and N. Vayatis (2018). *A review of change point detection methods*. arXiv preprint arXiv:1801.00718.

Tucker, L. R. (1966). *Some mathematical notes on three-mode factor analysis*. Psychometrika 31(3): 279-311.

Undey, C. and A. Cinar (2002). *Statistical monitoring of multistage, multiphase batch processes*. IEEE Control Systems Magazine 22(5): 40-52.

Ündey, C., S. Ertunç and A. Çinar (2003). *Online batch/fed-batch process performance monitoring, quality prediction, and variable-contribution analysis for diagnosis*. Industrial & engineering chemistry research 42(20): 4645-4658.

Ündey, C., E. Tatara and A. Çınar (2004). *Intelligent real-time performance monitoring and quality prediction for batch/fed-batch cultivations*. Journal of Biotechnology 108(1): 61-77.

Van den Kerkhof, P., J. Vanlaer, G. Gins and J. F. M. Van Impe (2013). *Analysis of smearing-out in contribution plot based fault isolation for Statistical Process Control*. Chemical Engineering Science 104: 285-293.

van der Voet, H. (1999). *Pseudo-degrees of freedom for complex predictive models: the example of partial least squares*. Journal of Chemometrics 13(3-4): 195-208.

Van Impe, J. and G. Gins (2015). *An extensive reference dataset for fault detection and identification in batch processes*. Chemometrics and Intelligent Laboratory Systems 148: 20-31.

Vergara, J. R. and P. A. Estévez (2014). *A review of feature selection methods based on mutual information*. Neural computing and applications 24(1): 175-186.

Vitale, R., O. E. de Noord and A. Ferrer (2015). *Pseudo-sample based contribution plots: innovative tools for fault diagnosis in kernel-based batch process monitoring*. Chemometrics and Intelligent Laboratory Systems 149: 40-52.

Vitale, R., O. Noord and A. Ferrer (2014). *A kernel-based approach for fault diagnosis in batch processes*. Journal of Chemometrics 28(8): S697-S707.

Vitale, R., J. A. Westerhuis, T. Næs, A. K. Smilde, O. E. de Noord and A. Ferrer (2017). *Selecting the number of factors in principal component analysis by permutation testing—Numerical and practical aspects*. Journal of Chemometrics 31(12): e2937.

Walczak, B. and D. L. Massart (2001). *Dealing with Missing Data*. Chemometrics and Intelligent Laboratory Systems 58: Part I: 15-27, Part II: 29-42.

Walczak, B. and D. L. Massart (2001). *Dealing with missing data: Part II*. Chemometrics and Intelligent Laboratory Systems 58(1): 29-42.

Wang, J. and Q. P. He (2010). *Multivariate statistical process monitoring based on statistics pattern analysis*. Industrial & Engineering Chemistry Research 49: 7858-7869.

Wang, J. and Q. P. He (2010). *Multivariate statistical process monitoring based on statistics pattern analysis*. Industrial & Engineering Chemistry Research 49(17): 7858-7869.

Wang, X., P. Wang, X. Gao and Y. Qi (2016). *On-line quality prediction of batch processes using a new kernel multiway partial least squares method*. Chemometrics and Intelligent Laboratory Systems 158: 138-145.

Westerhuis, J. A., S. P. Gurden and A. K. Smilde (2000). *Generalized contribution plots in multivariate statistical process monitoring*. Chemometrics and Intelligent Laboratory Systems 51(1): 95-114.

Westerhuis, J. A., T. Kourti and J. F. MacGregor (1999). *Comparing alternative approaches for multivariate statistical analysis of batch process data*. Journal of Chemometrics 13(3-4): 397-413.

Wise, B. M., N. B. Gallagher, S. W. Butler, D. D. White Jr and G. G. Barna (1999). *A comparison 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: A Journal of the Chemometrics Society 13(3-4): 379-396.

Wold, S., K. Esbensen and P. Geladi (1987). *Principal component analysis*. Chemometrics and Intelligent Laboratory Systems 2(1-3): 37-52.

Wold, S., N. Kettaneh-Wold, J. F. MacGregor and K. G. Dunn (2009). 2.10 - Batch Process Modeling and MSPC. *Comprehensive Chemometrics*. S. D. B. T. Walczak. Oxford, Elsevier**:** 163-197.

Wold, S., N. Kettaneh, H. Fridén and A. Holmberg (1998). *Modelling and diagnostics of batch processes and analogous kinetic experiments*. Chemometrics and intelligent laboratory systems 44(1): 331-340.

Wold, S., A. Ruhe, H. Wold and I. Dunn, WJ (1984). *The collinearity problem in linear regression. The partial least squares (PLS) approach to generalized inverses*. SIAM Journal on Scientific and Statistical Computing 5(3): 735-743.

Wold, S., M. Sjöström and L. Eriksson (2001). *PLS-regression: a basic tool of chemometrics*. Chemometrics and intelligent laboratory systems 58(2): 109-130.

Wu, C. F. J. and M. S. Hamada (2009). *Experiments - Planning, Analysis, and Optimization*. Hoboken, New Jersey, Wiley.

Wuyts, S., G. Gins, P. Van den Kerkhof and J. Van Impe (2015). *Fault Identification in Batch Processes Using Process Data or Contribution Plots: A Comparative Study*. Advanced Control of Chemical Processes.

Yao, Y. and F. Gao (2009). *A survey on multistage/multiphase statistical modeling methods for batch processes*. Annual Reviews in Control 33(2): 172-183.

Yoo, C. K., J.-M. Lee, P. A. Vanrolleghem and I.-B. Lee (2004). *On-line monitoring of batch processes using multiway independent component analysis*. Chemometrics and Intelligent Laboratory Systems 71(2): 151-163.

Yu, J. (2010). *Hidden Markov models combining local and global information for nonlinear and multimodal process monitoring*. Journal of Process Control 20(3): 344-359.

Yu, L. and H. Liu (2003). *Feature selection for high-dimensional data: A fast correlation-based filter solution*. ICML.

Yu, L. and H. Liu (2004). *Efficient feature selection via analysis of relevance and redundancy*. Journal of machine learning research 5(Oct): 1205-1224.

Zhang, J., A. Morris, E. Martin and C. Kiparissides (1998). *Prediction of polymer quality in batch polymerisation reactors using robust neural networks*. Chemical Engineering Journal 69(2): 135-143.

Zhang, Y. (2008). *Fault detection and diagnosis of nonlinear processes using improved kernel independent component analysis (KICA) and support vector machine (SVM)*. Industrial & Engineering Chemistry Research 47(18): 6961-6971.

Zhang, Y. and Z. Hu (2011). *On-line batch process monitoring using hierarchical kernel partial least squares*. Chemical Engineering Research and Design 89(10): 2078-2084.

Zhang, Y., S. Li, Z. Hu and C. Song (2012). *Dynamical process monitoring using dynamical hierarchical kernel partial least squares*. Chemometrics and Intelligent Laboratory Systems 118: 150-158.

Zhang, Y., B. Lu and T. F. Edgar (2013). *Batch trajectory synchronization with robust derivative dynamic time warping*. Industrial & Engineering Chemistry Research 52(35): 12319-12328.

Zhang, Y. and S. J. Qin (2007). *Fault detection of nonlinear processes using multiway kernel independent component analysis*. Industrial & engineering chemistry research 46(23): 7780-7787.

Zhang, Y., Y. Teng and Y. Zhang (2010). *Complex process quality prediction using modified kernel partial least squares*. Chemical Engineering Science 65(6): 2153-2158.

Zou, H. and T. Hastie (2005). *Regularization and variable selection via the elastic net*. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 67(2): 301-320.