UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

A contribution to chemical process operation support: new machine learning and surrogate models-based approaches for process optimization, supervision and control

Ahmed Shokry Abdelaleem Taha Zied

A CONTRIBUTION TO CHEMICAL PROCESS OPERATION SUPPORT: NEW MACHINE LEARNING AND SURROGATE MODELS-BASED APPROACHES FOR PROCESS OPTIMIZATION, SUPERVISION AND CONTROL

By: **Ahmed Shokry Abdelaleem Taha Zied**

Directed by Prof. Antonio Espuña

A Dissertation Submitted for the Degree of Doctor of Philosophy in Chemical Process Engineering

Center for Process Systems and Environmental Engineering - CEPIMA

Department of Chemical Engineering

Escola d'Enginyeria de Barcelona Est - EEBE

Universitat Politècnica de Catalunya - UPC

February 2021

Resumen

En la Industria de Proceso, como en otros ámbitos, la toma de decisiones se basa en la valoración de las consecuencias de dichas decisiones a través de modelos (implícitos o explícitos). La escala y complejidad de los modelos necesarios dependen de la complejidad del proceso, del nivel jerárquico al que se toman las decisiones (p. ej.: gestión de la cadena de suministro, planificación de proceso, programación de operaciones, control,…) y del horizonte de tiempo considerado. El uso de modelos basados en principios básicos (First Principle Models – FPM) habitualmente permite predecir con precisión el comportamiento de un sistema y llevar así a decisiones fundamentadas y explicables. Sin embargo, su uso se ve obstaculizado por problemas prácticos, dado que en ocasiones requiere cálculos iterativos aún sin tener garantizada su convergencia a una solución factible. Estos problemas son más frecuentes a medida que se desciende en la jerarquía de toma de decisiones (p. ej.: control supervisor), especialmente si la resolución (optimización) del sistema implica muchos cálculos de simulación utilizando un FPM complejo (p. ej.: altamente no lineal, involucrando variables enteras, etc.). Una forma de superar estas dificultades consiste en aplicar técnicas basadas en "modelos subrogados" o sustitutos, construidos a partir de datos recopilados del proceso real, de datos previamente simulados (utilizando un FPM), o de una combinación de ambos. Aunque estos modelos se utilizan en muchas áreas, en el ámbito de la ingeniería química habitualmente se emplean solo para el diseño de procesos y en sistemas de optimización de estado estacionario.

Esta tesis presenta un marco para el uso eficaz y eficiente de modelos subrogados, construidos mediante técnicas de aprendizaje automático, en la toma de decisiones en diferentes fases de la operación, el control y la optimización de un proceso. En este contexto, el Capítulo 3 presenta una metodología para la optimización de la operación en estado estacionario de procesos no lineales. El Capítulo 4 propone la utilización de metodologías basadas en el aprendizaje automático en problemas de optimización de operaciones sujetas a incertidumbre (optimización multiparamétrica). El Capítulo 5 extiende este planteamiento a la construcción de sistemas de control predictivo (MPC) explícito de procesos no lineales. El Capítulo 6 propone una metodología para la construcción sistemática de modelos subrogados en sistemas dinámicos no-lineales multi-variable, metodología que se aplica en el capítulo 7 a la

de optimización de procesos dinámicos (control óptimo de sistemas no lineales en estado no estacionario). Esta misma metodología se integra en el Capítulo 8 con técnicas de clasificación para su aplicación a la detección y diagnosis de fallos (Fault Detection and Diagnosis - FDD) de sistemas dinámicos multivariable. Finalmente, en el Capítulo 9 se presenta la aplicación de estas metodologías para el entrenamiento de sensores virtuales ("soft-sensors") y su aplicación a procesos de producción por lotes que trabajan con condiciones iniciales cambiantes. Cada una de estas aplicaciones, y los prototipos resultantes, se han plateado después de una cuidadosa revisión de las aportaciones más recientes en estos campos, que ha permitido identificar las dificultades para la implementación de las técnicas existentes en sistemas prácticos de soporte a la toma de decisiones, y la forma de superar estas dificultades mediante la utilización de modelos alternativos, que se resumen en el Capítulo 1.

La eficacia de las metodologías desarrolladas se ilustra a través del análisis de su aplicación a diferentes casos, tanto propuestos en esta Tesis como de referencia en los diferentes ámbitos de aplicación. Estos resultados han merecido su publicación en diferentes revistas científicas de primer nivel, así como su difusión a través de congresos internacionales, incluidas dos conferencias de invitadas.

Abstract

In the chemical process industry, the decision-making hierarchy is inherently model-based. The scale and complexity of the considered models (e.g., enterprise, plant or unit model) depend on the decision-making level (e.g., supply-chain management, planning, scheduling, operation) and the allowable time slot (weeks, hours, seconds) within which model simulation runs must be performed and their output is analyzed to support the decision making. The use of high-fidelity models, which include detailed physics-based description of the process, is attracting wide interests of the process engineers. Since, these First Principle Model (FPMs) are able to accurately predict the real behavior of the process, leading to realistic optimal decisions. However, their use is hindered by practical challenges as the high computational time required for their simulation and the unguaranteed reliability of their consistent convergence. The challenges become prohibitive at lower levels of the decision-making hierarchy (i.e., operation), where decisions are required online within time slots of minutes or seconds entailing lots of simulation runs using such complex and highly nonlinear FPMs. Surrogate modelling techniques are potential solution for these challenges, which relies on developing simplified, but accurate, data-driven or machine learning models using data generated by FPM simulations, or collected from a real process. Although, there are progressive developments of surrogate-based methods in the chemical engineering area, they are concentrated in process design and steady-state optimization areas.

This Thesis presents a framework for the proper and effective use of surrogate models and machine learning techniques in different phases of the process operation. The objective is to provide efficient methodologies, each supports the decision making in a specific phase of the process operation, namely; steady-state operation optimization, Model Predictive Control (MPC), multivariate system identification and multistep-ahead predictions, dynamic optimization, Fault Detection and Diagnosis (FDD) and soft-sensing. Each developed methodology is designated according to careful State-Of-Art (SOA) review that identifies the gaps and missing requirements to be covered. The SOA, identified gaps and the contributions of each methodology are summarized in Chapter 1 and detailed in the introduction of each of the following chapters.

In this context, Chapter 3 presents a surrogate-based methodology for steady-state operation optimization of complex nonlinear chemical processes modelled by black-box functions. Chapter 4 proposes machine learning-based methodologies for multiparametric solution of complex operation optimization problems subjected to uncertainty. Chapter 5 presents a data-based multiparametric MPC methodology that enables simple implementations of explicit MPC for nonlinear chemical processes. Chapter 6 proposes a data-driven methodology for multivariate dynamic modelling of nonlinear chemical processes and for multistep-ahead prediction. Chapter 7 suggests a dynamic optimization methodology for solving optimal control problems of complex nonlinear processes based on data-driven dynamic models. Chapter 8 shows a hybrid methodology to improve FDD of chemical processes run under time-varying inputs based on multivariate data-driven dynamic models and classification techniques. Chapter 9 presents data-driven soft-sensing methodologies for batch processes operated under changeable initial conditions. The effectiveness of the developed methodologies is proved by comparing their performances to those of classical solution procedures existing in the SOA, via their applications to different benchmark examples and case studies. The promising results and their sound analysis allowed to publish many papers in top-ranked journals and proceedings, and to present them at several top-ranked international conferences including **two Keynote** presentations.

Acknowledgment

I would like to thank my supervisor, Prof. Antonio Espuña, for his support, guidance and encouragement during the thesis journey. Thank you for sharing your board knowledge and deep expertise in the Chemical Process Engineering field with unique patience and flexibility. I am also thankful to Prof. Moisès Graells for his relevant help and profitable collaboration in the Fault Detection and Diagnosis line in this Thesis. Thanks to all the CEPIMA research group members and, specially, to Dr. Kefah Hjaila, Dr. Sergio Medina-González and Dr. Hamed Ardakani for their fruitful collaborations and joyful company.

I am grateful to Prof. Enrico Zio and Prof. Piero Baraldi, pioneer experts in Engineering Reliability and Prognostic and Health-Management, for allowing me a great chance to be in contact with these vital fields. The skills, knowledge and competencies I acquired from you are positively and significantly impacted my professional career and attitude. The support, trust and kindness that I have received from you are unforgettable. I also wish to show my gratitude to Dr. Federico Antonello and Dr. Luca Pinciroli for their productive collaborations and kind company. I do owe my gratitude to Prof. Eric Moulines, a world-class expert in Statistics and Data-Science, for giving me a valuable opportunity to be in involved in this fundamental area and their applications to distinct engineering fields. I really appreciate your trust, professional guidance, wise recommendations and gentle support.

My deep and sincere gratitude to my family. I am forever indebted to my father, Shokry Abdelaleem Zied. Thank you for being my outstanding and realistic example for honesty, sincerity, tolerance and love. I am grateful for, and I miss a lot, your inspiring conversations through which you were sharing with me your enlightened foresight and wise comprehension of life. Dad, you took a part of my soul and passed away, leaving in my heart a wound that never heals. My deepest gratitude goes to my mother, Sohier Eldallal, for her unparalleled love, unconditional support, sacrifices, continuous encouragement and endless belief in me during hard times. Mom, no words can express how I love you and how I am grateful to you.

Last, but foremost, I am absolutely and sincerely grateful to my wife, my friend and my life-partner, Asmae Ben-Brahim, who has completely supported and motivated me

with her love, sacrifices, patience and encouragement during relatively hard and tiring periods. Thank you Asmae, for standing by me through all my travails, my absences and my impatience. This journey would not have been possible if you were not with me. You gave me strength, discussed ideas and prevented several wrong turns. I love you and I'll always do.

Ahmed Shokry Abdelaleem Zied

March, 2021

To my beloved,

Shokry Abdelaleem Zied,

Sohier El Dallal, and

Asmae Ben-Brahim

Table of Contents

List of Figures

List of Tables

List of Publications

Chapter 3

1) *Ahmed Shokry, Antonio Espuña.* Applying Metamodels and Sequential Sampling for Constrained Optimization of Process Operations. ***Lecture Notes in Computer Science***: ***Artificial Intelligence and Soft Computing****, 8468, 396-407, 2014, DOI:10.1007/978-3-319-07176-3_35.*
2) *Ahmed Shokry, Aaron D. Bokhariski, Antonio Espuña.* Using Surrogate Models for Process Design and Optimization. ***Uncertainty Modeling in Knowledge Engineering and Decision Making****, 483-488, 2012, DOI:10.1142/9789814417747_0078.*

Chapter 4

3) *Ahmed Shokry, Sergio Medina, Piero Baraldi, Enrico Zio, Eric Moulines, Antonio Espuña.* A Machine Learning-based Methodology for Multi-Parametric Solution of Chemical Processes Operation Optimization under Uncertainty. Submitted to ***Chemical Engineering Journal,*** 2020.
4) *Ahmed Shokry, Sergio Medina, Antonio Espuña.* Mixed-Integer MultiParametric Approach based on Machine Learning Techniques. ***Computer Aided Chemical Engineering****, 40, 451-456, 2017, DOI:10.1016/B978-0-444-63965-3.50077-5.*
5) *Ahmed Shokry, Antonio Espuña.* Dynamic Optimization of Batch Processes under Uncertainty via Meta-MultiParametric Approach. ***Computer Aided Chemical Engineering*** *40, 2215-2220, 2017, DOI:10.1016/B978-0-444-63965-3.50371-8.*
6) *Ahmed Shokry, Antonio Espuña.* Optimization Under Uncertainty Based on Multiparametric Kriging Metamodels. ***Operations Research Proceedings****, 2016, Springer, DOI:10.1007/978-3-319-42902-1_78.*
7) *Sergio Medina-González, Ahmed Shokry, Javier Silvente, Gicela Lupera Calahorrano, Antonio Espuña*. Optimal management of bio-based energy supply chains under parametric uncertainty through a data-driven decision-support framework. ***Computers & Industrial Engineering****, 139, January 2020, 105561, DOI: 10.1016/j.cie.2018.12.008.*
8) *Gicela Lupera Calahorrano, Ahmed Shokry, Georgios M. Kopanos, Antonio Espuña.* Mixed-integer Multiparametric Meta-Modeling: a Machine Learning Tool Applied to Reactive Scheduling. ***Computer Aided Chemical Engineering****, 43, 163-168, 2018, DOI:10.1016/B978-0-444-64235-6.50030-9.*
9) *Gicela Lupera Calahorrano, Ahmed Shokry, Sergio Medina-González, Eduardo Vyhmeister,Antonio Espuña.* Ordinary Kriging: a Machine Learning Tool Applied to Mixed-integer Multiparametric Approach. ***Computer Aided Chemical Engineering****, 43, 531-536, 2018, DOI: 10.1016/B978-0-444-64235-6.50094-2.*
10) *Gicela J. Lupera Calahorrano, Ahmed Shokry, Gerard Campanya, Antonio Espuña.* Application of the Meta-MultiParametric Methodology to the Control of Emissions in the Industry under Continuous and Discrete Uncertain Parameters. ***Chemical Engineering Research and Design****, 155 B, 365-373, 2016, DOI: 10.1016/j.cherd.2016.09.006.*
11) *Sergio Medina, Ahmed Shokry, Javier Silvente, Antonio Espuña*. A meta-multiparametric framework: Application to the operation of bio-based energy supply chains. ***Computer Aided Chemical Engineering****, 37, 1955–1960, 2015, DOI: 10.1016/B978-0-444-63576-1.50020-0.*

Chapter 5

12) *Ahmed Shokry, Canan Dombayci, Antonio Espuña.* Multiparametric Metamodels for Model Predictive Control of Chemical Processes. ***Computer Aided Chemical Engineering****,38, 937–942, 2016, DOI:10.1016/B978-0-444-63428-3.50161-2.*

Chapter 6

13) *Ahmed Shokry, Piero Baraldi, Enrico Zio, Antonio Espuña.* Dynamic Surrogate Modelling for Multistep-Ahead Prediction of Multivariate Nonlinear Chemical Processes. ***Industrial & Engineering Chemistry Research, Accepted manuscript****, 59, 15634–15655, 2020. doi.org/10.1021/acs.iecr.0c00729*

14) *Ahmed Shokry, Antonio Espuña*. The Ordinary Kriging Metamodel in Multivariate Dynamic Modeling and Multistep Ahead Prediction. ***Computer Aided Chemical Engineering*** *43, 265-270, 2018, DOI: 10.1016/B978-0-444-64235-6.50047-4.*

Chapter 7

15) *Ahmed Shokry, Antonio Espuña.* Sequential Dynamic Optimization of Complex Nonlinear Processes based on Kriging Surrogate Models. ***Procedia Technology****, 15, 376–387, 2014.DOI: 10.1016/j.protcy.2014.09.092.*

Chapter 8

16) *Ahmed Shokry, Mohammad Hamed Ardakani, Gerard Escudero, Moisès Graells, Antonio Espuña.* Dynamic Kriging based Fault Detection and Diagnosis Approach for Nonlinear Noisy Dynamic Processes. ***Computers & Chemical Engineering****, 106, 758-776, 2017, DOI: 10.1016/j.compchemeng.2017.03.016.*

17) *Ahmed Shokry, Mohammad Hamed Ardakani, Gerard Escudero, Moisès Graells, Antonio Espuña.* Kriging based Fault Detection and Diagnosis Approach for Nonlinear Noisy Dynamic Processes. ***Computer Aided Chemical Engineering****, 38, 55–60, 2016, DOI: 10.1016/B978-0-444-63428-3.50014-X.*

18) *Mohammad Hamed Ardakani, Ahmed Shokry, Gerard Escudero, Moisès Graells, Antonio Espuña*. Online Quantification of the Concept Drift Using Incremental Learned Classifier and Non-automatic Clustering. ***Computer Aided Chemical Engineering****, 43, 1069-1074, 2018, DOI: 10.1016/B978-0-444-64235-6.50187-X.*

19) Mohammad Hamed Ardakani, Ahmed Shokry, Gerard Escudero, Moisès Graells, Antonio Espuña. Unsupervised Automatic Updating of Classification Models of Fault Diagnosis for Novelty Detection. ***Computer Aided Chemical Engineering****, 43, 1123-1128, 2018, DOI: 10.1016/B978-0-444-64235-6.50196-0.*

20) *Mohammad Hamed Ardakani, Mahdieh Askarian, Ahmed Shokry, Gerard Escudero, Moisès Graells, Antonio Espuña.* Optimal Feature Selection for Designing a Fault Diagnosis System. ***Computer Aided Chemical Engineering****, 38, 1111-1116, 2016, DOI: 10.1016/B978-0-444-63428-3.50190-9.*

21) *Mohammad Hamed Ardakani, Ahmed Shokry, Ghazal Saki, Gerard Escudero, Moisès Graells, Antonio Espuña.* Imputation of Missing Data with Ordinary Kriging for Enhancing Fault Detection and Diagnosis. ***Computer Aided Chemical Engineering****, 38, 1377-1382, 2016, DOI: 10.1016/B978-0-444-63428-3.50234-4.*

Chapter 9

22) *Ahmed Shokry, Patricia Vicente, Gerard Escudero, Montserrat Perez Moya, Moisès Graells, Antonio Espuña*. Data-driven Soft-Sensors for Online Monitoring of Batch Processes with Different Initial Conditions. ***Computers and Chemical Engineering****,118, 159-179, 2018, 10.1016/j.compchemeng.2018.07.014.*

23) *Ahmed Shokry, Montserrat Pérez-Moya, Moisès Graells, Antonio Espuña.* Data-Driven Dynamic Modeling of Batch Processes Having Different Initial Conditions and Missing Measurements. ***Computer Aided Chemical Engineering*** *40, 433-438, 2017, DOI: 10.1016/B978-0-444-63965-3.50074-X.*

24) *Ahmed Shokry, Francesca Audino, Patricia Vicente, Gerard Escudero, Montserrat Perez Moya, Moisès Graells, Antonio Espuña*. Modeling and Simulation of Complex Nonlinear Dynamic Processes Using Data Based Models: Application to Photo-Fenton Process. ***Computer Aided Chemical Engineering****, 37, 191–196, 2015, DOI: doi.org/10.1016/B978-0-444-63578-5.50027-X.*

List of Presentations at International Conferences†

Chapter 3

1) *Victor M. Martín Barrios, Ahmed Shokry, Moisès Graells*. Operational Cost Minimization of Distillation Columns Using Metamodeling Techniques. ***Poster presentation, the 22nd International Congress of Chemical and Process Engineering CHISA****, Prague, Czech Republic, 2016.*

2) *Ahmed Shokry, Antonio Espuña.* Applying Metamodels and Sequential Sampling for Constrained Optimization of Process Operations. ***Oral presentation, the 13th International Conference on Artificial Intelligence and Soft Computing ICAISC****, Zakopane, Poland, 2014.*

3) *Ahmed Shokry, K. Hejila, A. Espuna.* Adaptive evolutionary optimization of complex processes using a kriging based genetic algorithm. ***Oral presentation, the 13th Mediterranean Congress of Chemical Engineering 13MCCE****, Barcelona, Spain,2014.*

4) *Ahmed Shokry, Aaron D. Bokhariski, Antonio Espuña*. Using Surrogate Models for Process Design and Optimization. ***Oral presentation, the 10th International FLINS Conference on Uncertainty Modeling in Knowledge Engineering and Decision Making****, Istanbul, Turkey, 2012.*

Chapter 4

5) *Gicela Lupera Calahorrano, Ahmed Shokry, Georgios M. Kopanos, Antonio Espuña.* Mixed-integer Multiparametric Meta-Modeling: A Machine Learning Tool Applied to Reactive Scheduling. *Clustering.* ***Oral presentation, the 28th European Symposium on Computer Aided Process Engineering-ESCAPE28****, Graz, Austria, June, 2018.*

6) *Gicela Lupera Calahorrano, Ahmed Shokry, Sergio Medina-González, Eduardo Vyhmeister,Antonio Espuña.* Ordinary Kriging: A Machine Learning Tool Applied to Mixed-integer Multiparametric Approach. ***Oral presentation, the 28th European Symposium on Computer Aided Process Engineering-ESCAPE28****, Graz, Austria, June, 2018.*

7) *Gicela Lupera Calahorrano, Ahmed Shokry, Georgios M. Kopanos, Antonio Espuña.* Reactive scheduling using Meta- mixed-integer Multiparametric Model. ***Poster presentation, The 10th World Congress of Chemical Engineering****, Barcelona, Spain, 2017.*

8) *Ahmed Shokry, Sergio Medina, Antonio Espuña.* Mixed-Integer MultiParametric Approach based on Machine Learning Techniques. ***Oral presentation, the 27th European Symposium on Computer Aided Process Engineering****, Barcelona, Spain, 2017.*

9) *Ahmed Shokry, Antonio Espuña.* Dynamic Optimization of Batch Processes under Uncertainty via Meta-MultiParametric Approach. ***Poster presentation, the 27th European Symposium on Computer Aided Process Engineerin****g, Barcelona, Spain, 2017.*

10) *Ahmed Shokry, Antonio Espuña.* A Data-Based Multiparametric Programming Methodology Using Ordinary Kriging Metamodels. ***Oral presentation, the international conference on Operations Research -OR2015****, Vienna, Austria, 2015.*

11) *Gicela Lupera Calahorrano, Ahmed Shokry, Gerard Campanya, Antonio Espuña.* Emissions Control in Industry Using a Multiparametric Metamodel. ***Oral presentation, the 10th European Congress of Chemical Engineering ECCE10****, Nice, France, 2015.*

25) *Sergio Medina, Ahmed Shokry, Javier Silvente, Antonio Espuña*. A meta-multiparametric framework: Application to the operation of bio-based energy supply chains. ***Oral presentation**, **the 25th European Symposium on Computer Aided Process Engineering-ESCAPE25** and **the 12th Process Systems Engineering PSE**, Copenhagen, Denmark, June 2015.*

Chapter 5

12) *Ahmed Shokry, Canan Dombayci, Antonio Espuña.* Multiparametric Metamodels for Model Predictive Control of Chemical Processes. ***Poster presentation**, **the 26th European Symposium on Computer Aided Process Engineering-ESCAPE26**, Portorož, Slovenia, June 2016.*

Chapter 6

13) *Ahmed Shokry, Antonio Espuña. Utilización de Meta-Modelos para el Control y la Optimización de Sistemas Dinámicos. **Keynote presentation**, **XXXVI Jornadas Nacionales de Ingeniería Química**, Zaragoza, Spain, 4-6 September, 2019.*

14) *Ahmed Shokry, Antonio Espuña*. The Ordinary Kriging Metamodel in Multivariate Dynamic Modeling and Multistep Ahead Prediction. ***Keynote presentation**, **the 28th European Symposium on Computer Aided Process Engineering-ESCAPE28**, Graz, Austria, June, 2018.*

15) *Ahmed Shokry, Antonio Espuña.* Multistep Ahead Prediction Using Ordinary Kriging Applied to Modeling and Simulation of Complex Nonlinear Dynamic Processes. ***Poster presentation**, **the 10th European Congress of Chemical Engineering**, Nice, France, 2015.*

Chapter 7

16) *Ahmed Shokry, Antonio Espuña*. Sequential Dynamic Optimization of Complex Nonlinear Processes based on Kriging Surrogate Models. ***Oral presentation, the 2nd International Conference on System-Integrated Intelligence**: Challenges for Product and Production Engineering, Bremen, Germany, July, 2014.*

Chapter 8

17) *Ahmed Shokry, Mohammad Hamed Ardakani, Gerard Escudero, Moisès Graells, Antonio Espuña.* Kriging based Fault Detection and Diagnosis Approach for Nonlinear Noisy Dynamic Processes. ***Poster presentation**, **the 26th European Symposium on Computer Aided Process Engineering-ESCAPE26**, Portorož, Slovenia, June 2016.*

18) *Mohammad Hamed Ardakani, Ahmed Shokry, Ghazal Saki, Gerard Escudero, Moisès Graells, Antonio Espuña.* Imputation of Missing Data with Ordinary Kriging for Enhancing Fault Detection and Diagnosis. ***Poster presentation**, **the 26th European Symposium on Computer Aided Process Engineering-ESCAPE26**, Portorož, Slovenia, June 2016.*

19) *Mohammad Hamed Ardakani, Ahmed Shokry, Gerard Escudero, Moisès Graells, Antonio Espuña*. A Framework for Unsupervised Fault Detection and Diagnosis Based on Clustering Assisted Kriging Observer. ***Oral presentation**, **the 3rd Conference on Control and Fault-Tolerant Systems SysTol3**, Barcelona, Spain, 2016.*

20) *Mohammad Hamed Ardakani, Ahmed Shokry, Gerard Escudero, Moisès Graells, Antonio Espuña.* Unsupervised Automatic Updating of Classification Models of Fault Diagnosis for Novelty Detection. ***Poster presentation**, **the 28th European Symposium on Computer Aided Process Engineering-ESCAPE28**, Graz, Austria, June, 2018.*

21) *Mohammad Hamed Ardakani, Ahmed Shokry, Gerard Escudero, Moisès Graells, Antonio Espuña.* Online Quantification of the Concept Drift Using Incremental Learned Classifier

and Non-automatic Clustering. ***Poster presentation***, ***the 28th European Symposium on Computer Aided Process Engineering-ESCAPE28***, *Graz, Austria, June, 2018.*

22) Mohammad Hamed Ardakani, Mahdieh Askarian, Ahmed, Shokry, Gerard Escudero, Moisès Graells, Antonio Espuñaa. Optimal Feature Selection for Designing a Fault Diagnosis System. ***Poster presentation***, **the *26th European Symposium on Computer-Aided Process Engineering-ESCAPE26****, Portorož Slovenia, June, 2016.*

Chapter 9

23) *Ahmed Shokry, Montserrat Pérez-Moya, Moisès Graells, Antonio Espuña.* Data-Driven Dynamic Modeling of Batch Processes Having Different Initial Conditions and Missing Measurements. ***Poster presentation***, ***the 27th European Symposium on Computer Aided Process Engineering***, *Barcelona, Spain, 2017.*

24) Ahmed Shokry, Francesca Audino, Patricia Vicente, Gerard Escudero, Montserrat Perez Moya, Moisès Graells, Antonio Espuña. Modeling and Simulation of Complex Nonlinear Dynamic Processes Using Data Based Models: Application to Photo-Fenton Process. ***Poster presentation***, ***the 25th European Symposium on Computer Aided Process Engineering-ESCAPE25*** *and* ***the 12th Process Systems Engineering PSE****, Copenhagen, Denmark, June 2015.*

† The results of some of works which are included in the "List of Publications", are also presented at international conferences, therefore, they are also shown in the "List of Presentations at International Conferences"

Chapter 1: Introduction

Within the framework of Chemical Process Operations, computer-based simulation and optimization tools have become essential supports for any decision-making procedure. In many cases, these tools are based on First Principle Models (FPMs) of the process, which are used at the different operational levels to perform different functions. In order to address some of the main challenges that the use of these FPMs-based tools is currently facing, this thesis proposes alternative/complementary strategies based on the use of surrogate models and machine learning tools.

This first chapter presents the context of the thesis (Section 1.1) and the specific challenges that have been addressed (Section 1.2); then, it summaries the stat-of-art available solutions for addressing these challenges in order to identify the gaps and the missing needs that will be covered by the thesis work (Section 1.3). After that, the chapter highlights the thesis objectives (Section 1.4.1) and contributions (Section 1.4.2), as well as the thesis structure (Section 1.5).

1.1 CONTEXT: CHEMICAL PROCESS OPERATION OPTIMIZATION, ITS MAIN MODULES AND THEIR FUNCTIONS

Process operation optimization is an important layer in the general decision-making hierarchy of chemical plants management. It receives, as inputs, the outcomes and decisions coming from higher level layers (i.e., supply chain optimization, planning and scheduling) (Marchetti, et al., 2014). These outcomes and decisions mostly include forecasts of prices and demands, production rate targets over long time periods (weeks/days), assignment of resources to activities (raw material allocation, tasks to units allocation, maintenance interventions, staffing), sequencing of activities and determination of starting and ending times for the execution over short periods of time (Muller, et al., 2017; Seborg, et al., 2016). Then, the process operation optimization layer provides as output: *i*) the real-time optimal values of the process variables (i.e., pressures, flow rates, cooling temperatures, etc.) at the which the plant and its units must operate to achieve the required performance, considering quality, capacity, safety and environmental restrictions and requirements and, more importantly, reacting to sudden and unexpected variations of the process or external parameters (e.g., equipment efficiencies, raw material characteristics, demand etc.), *ii*) detailed and timely orders to the basic equipment control systems to implement actions to maintain the plant units functioning at these set-points (or reference trajectories) against expected disturbances (e.g., small

fluctuation in the feed temperature) and iii) timely information about the process functionality state, i.e., if it is functioning under normal or abnormal conations, and about the possible type of fault that impacts the process leading to these abnormal conditions.

Figure 1.1 shows a schematic representation of the main modules/activities required for such a task, their usual activation sequence and the scales of the process models considered in each module, where each module and its associated model scale are highlighted with the same color. The following parts in this section discuss these main modules and the functions performed by each of them.

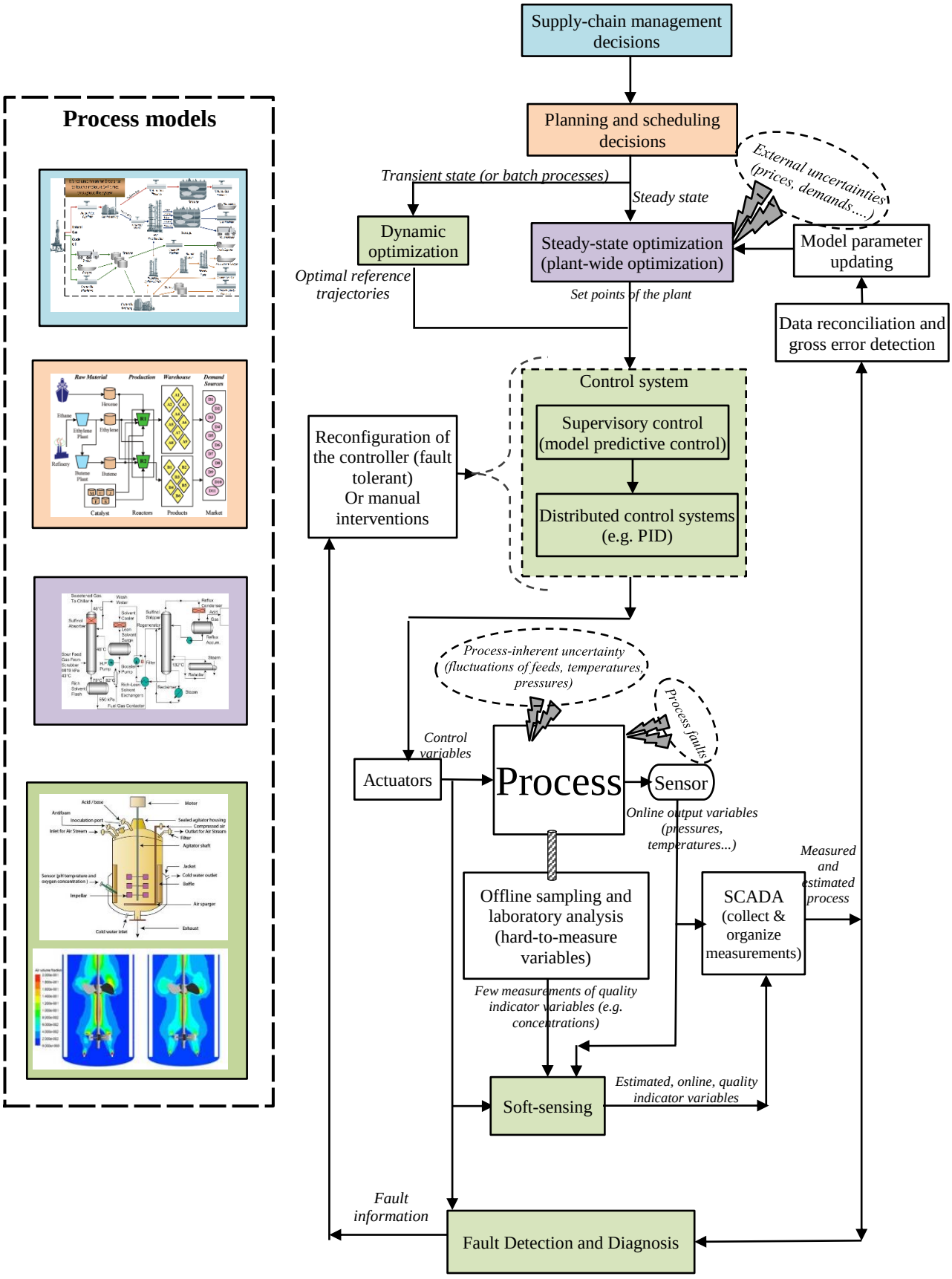


Figure 1.1. Process operation modules (right) and associated process model scales (left)).

1.1.1 Steady-state optimization and uncertainty handling

The first module to be considered in the usual process operation decision-making sequence is the steady-state optimization, which aims at obtaining the optimal values of the process variables (temperatures, pressures, feed compositions, flowrates, valve opening, etc.) at which the plant and its units must operate in order to maximize certain performance criteria (e.g., efficiency, profit and/or operational cost) and to satisfy all the constraints (demand, resources availability, equipment capacities, environmental restriction, etc.) and requirements (product quality, production yields, safety, etc.) (Vaccari & Pannocchia, 2017; Biegler, 2010). This goal is achieved by solving, in real time, an optimization problem based on a detailed and rigorous steady-state model of the process (Shao, et al., 2019). Depending on the model characteristics, such as its structure, transparency (e.g., white, grey, black-box), availability of derivative information, and on the formulations of the objective(s) and constraints of the optimization problem, different algorithms can be used, like derivative-free algorithms (e.g., Genetic Algorithms), where the explicit values of the objective(s) function are used to direct the optimization search, derivative-based algorithms (e.g., interior point algorithms), where the optimization search is directed based on the derivatives of the objective(s) with respect to the decision variables (Salback, 2004; Caballero & Grossmann, 2008), etc.

On another hand, the presence of uncertainty sources in the system at different levels is unavoidable (Acevedo & Pistikopoulos, 1997; Li, 2010; Jiao, et al., 2012), including model-inherent uncertainty, related to the lack of knowledge about the exact values of model physical parameters (e.g., kinetic rates, heat transfer coefficients) (Flemming, et al., 2007; Norbert, et al., 2017; Diangelakis, et al., 2017), process-inherent uncertainty, associated to fluctuations of the operating practices (e.g., feed stream concentrations, temperatures, pressures, recipes, processing time, equipment availability, equipment efficiencies) (Mesfin & Shuhaimi, 2010; Papathanasiou, et al., 2019), as well as external uncertainty (e.g.: resources characteristics, prices and demands).

The first type of uncertainty (i.e., model-inherent) usually occurs in a slow and continuous/evolving manner, leading to the increase of the mismatch between the model predictions and the real process behavior along the time. To minimize the process-model mismatch, the values of the model parameters must be updated in a systematic manner at prescheduled periods of time (typically hour(s)) using reconciled estimates of the measured steady-state data of the plant variables (Fadda, 2017; Biegler, 2010). These reconciled estimates are obtained by applying data reconciliation and gross error detection techniques to the real data collected by the sensors in order to reduce, respectively, the effect of random

errors and sensor faults (bias, drifting, miscalibration, total failure, etc.) (Chaudhary, 2009). Other technologies allow to directly estimate the new values of the model parameters within the data reconciliation and gross error detection tasks (Chaudhary, 2009).

In contrast, the latter two types of uncertainty sources (i.e., process-inherent and external uncertainty) may occur in a sudden and unexpected way. Hence, many methods have been developed for handling these two types of uncertainty in optimization problems, most of them can be categorized into two main approaches: proactive and reactive (Medina-González, et al., 2020). The proactive approach aims at providing conservative optimal decisions minimizing the consequences of the uncertainty and variability on the performance measure(s) of the system (i.e., objective function(s)) (Jiao, et al., 2012). Stochastic programming and robust optimization are among the most popular methods in the proactive approach (Grossmann, et al., 2016). In stochastic programming methods, the uncertain parameters are dealt as stochastic variables with "a-priori" known probability distribution functions, whose parameters are estimated from historical data. In this context, the goal becomes to identify the optimal decision variables that maximize/minimize the expected value of the objective function(s) and achieve feasibility over the distribution of the uncertain parameters (Li, 2010). Robust optimization methods deal with unknown but bounded uncertain parameters and aim at finding the optimal solution that ensures the feasibility over the entire range of realizations of the uncertain parameters (Norbert, et al., 2017)

On the other hand, the reactive approach is considered when it is necessary to, promptly, provide online update of the optimal values of the decision variables in response to real-time changes of the uncertain parameters value, which can be identified once unveiled. Since reactive approaches require providing the optimal solution for each specific realization of the uncertain parameters, they are preferred for the application in dynamic or online operation environments (Pistikopoulos, et al., 2007).

Among the reactive methods, Multi-Parametric Programming (MPP) offers outstanding capabilities (Pistikopoulos, 2008): *i*) its solution provides simple mathematical expressions mapping the optimal decisions (variables and objective) over the entire space of the uncertain parameters, *ii*) once the uncertainty is unveiled, the optimal decisions can be easily and immediately calculated by these simple functions avoiding huge computational cost required by repetitive optimization procedure and *iii*) MPP is not only able to handle the uncertainty related to the process conditions, but also to the optimization problem parameters (e.g., relative weights or importance of different objectives). Therefore, MPP very well fits to the requirements of dynamic production and operations environment (i.e., the thesis context) (Pistikopoulos, et al., 2007).

1.1.2 Dynamic optimization

Transient states can be experienced by continuous processes in situations like start-ups, shutdowns or transitions between different operational conditions that may be caused by many reasons. In these cases, as well as in batch processes, dynamic optimization (which is also called open-loop optimal control) is, instead, carried out, considering a dynamic model of the process (Banga, et al., 2005; Wang, et al., 2017). Dynamic optimization techniques allow the identification, in a fast and accurate way, of the optimal time-profiles of the process control variables that must be applied over a specific period of time (period of transition of a continuous process or period of a batch process) in order to drive the process to the required state at the end of this time period (Biegler, 2007). In case of continuous processes transitions, the required state is a steady-state, while in case of batch processes, it is typically the optimal batch performance at the end of batch time (e.g., to increase the production yield or to ensure product quality).

1.1.3 Model predictive control

After obtaining the optimal set-points of the plant, they are sent to the supervisory control module and, subsequently, to the distributed control module which are responsible of implementing them and holding the plant units operating at these set-points against expected process fluctuations, such as feed stream concentrations, temperatures and pressures (Mesfin & Shuhaimi, 2010; Papathanasiou, et al., 2019). In the case of batch processes/units or continuous processes in transient state, the optimal set-points become optimal reference trajectories, which the control system should track along pre-specified time horizons (i.e., batch time, transition time). Model Predictive Control (MPC) technologies are, nowadays, the backbone of the supervisory control modules in the chemical industries (Kouramas, et al., 2011; Katz, et al., 2020), because they offer very efficient capabilities in front of other technologies, such as proportional integral derivative controllers or linearized quadratic regulators. MPC is capable of efficiently handling multivariable control problems that involve complicated interactions and relations between the process variables and treating constraints, e.g., bounds on the maximum and/or minimum values of the control inputs or output variables (Chaudhary, 2009). Additionally, MPC allows to incorporate economical and even environmental terms in the objective function of the involved optimization problem, such as the cost associated to the profiles of the control inputs to be applied (Chaudhary, 2009; Katz, et al., 2020). In other words, the objective function considered in the MPC numerical optimization problem is not just the error between the current state of the process and the required state (the optimal set-points or reference trajectories).

1.1.4 Fault detection and diagnosis

Additional to the different sources of process/model uncertainties and fluctuations, whose undesired effects can be diminished by periodical parameter updating, repetitive steady-stead optimization and control schemes, the process can be also affected by faults or malfunctions (Venkatasubramanian, et al., 2003a; Venkatasubramanian, et al., 2003b). A fault is an unexpected change of the process behavior with respect to its expected normal conditions, which hampers the process normal operation causing unacceptable deterioration of its performance that may even lead to dangerous operating conditions (Patton, et al., 1995; Calado, et al., 2001). Faults can be classified into three types (Venkatasubramanian, et al., 2003a; Park, et al., 2020): *i*) sensor faults, which are, by terminology, related to the malfunction or failure of the sensors, such as drifting, miscalibration, biases, and freezing, *ii*) actuator faults that are associated to their inability to correctly interpret and convert the control signals, received from the controller, into appropriate forces (e.g., motor torque) needed to derive the system, such as control valve stuck-open and stuck-closed, and *iii*) process faults, related to malfunctions in the process/units, such tank leakage, equipment damage, sever unknown changes in feed streams characteristics, etc.

The Fault Detection and Diagnosis (FDD) module plays an essential role in guaranteeing safety and reliability of industrial processes operation, due to its ability of early detecting faults occurrence and discovering their root cause (Park, et al., 2020). This contributes to avert sudden shutdowns, breakdowns or even catastrophic events, and eventually to avoid large economic losses due to production stop and/or replacement of spare parts (Amozeghar & Khorasani, 2016). A FDD system performs two main functions: first, detecting the occurrence of fault, as opposite to the process normal behavior and, second, diagnosing the fault type or characteristics (Patton, et al., 1995; Narasimhan, et al., 2008).

1.1.5 Soft-sensing

In order to perform the numerical analysis in most of the previously mentioned modules of the process operations (e.g., MPC, FDD) and to obtain realistically effective/optimal decisions, the availability of continuous and real-time measurements of the process variables (control/input and state/output) is a must. These real-time measurements are used to continuously feed the model (e.g., values of the initial conditions of the real process state variables are required at each time step for the solution of the MPC problem, real time values of the process variables required to, continuously, feed the FDD system). But, for an important class of process variables, which are called Quality Indicator Variables (QIV), online and continuous measurements are not always attainable due to technological and/or economic limitations (Kadlec, et al., 2009; Lin, et al., 2007). On the contrary, in many cases QIV values are obtained through expensive and time-consuming offline sampling and laboratory analysis

(Zamprogna, et al., 2005; Desai, et al., 2006). As a result, large laboratory delay and human errors in the procedure may prevent reliable optimization, control, monitoring and supervision of the process (Liu et al., 2012).

Soft-sensing techniques have been proposed as a promising solution that has proven its effectiveness in these situations (Kadlec, et al., 2009). Soft-sensors are computational techniques that provide online and continuous "estimations" of the process QIV values by exploiting the measurements of other variables of the process that are reliably and continuously recorded online with minimum cost by means of the physical sensors network (e.g., temperature, pressure, flowrate) (Hoskins & Himmelblau, 1988).

1.2 IMPORTANCE AND CHALLENGES OF THE USE OF PROCESS FPMS IN CHEMICAL PROCESS OPERATION

This section explores the importance of the process models in the previously mentioned process operation support modules, and highlights the challenges that frequently face and/or hinders their usage.

1.2.1 Steady-state optimization and uncertainty handling

Regarding the steady-state optimization module, there is a growing trend of using detailed and high-fidelity mathematical models of the process based on "first principles" (FPMs) (Kajero, et al., 2017). However, the development of such analytical models for most chemical, petrochemical and pharmaceutical processes is a challenging task due to the required deep knowledge, effort and time. As a result, specialized simulation software tools have been developed to model and simulate such complex processes, most of them appearing in black box modular style, e.g., Aspen and gPROMs (Quirante, et al., 2018). Their ease of usage for modeling comes with many practical drawbacks and computational obstacles when they are used for optimization, especially for large-scale systems (Norbert, et al., 2017; Kelly & Zyngier, 2017). For example, the optimization of a full-scale petrochemical plant (crude oil and gas treatment facility, refinery, etc.) based on its FPM could demand several hours to converge and, in many cases, it does not converge to an optimal solution (Salback, 2004; Kajero, et al., 2017). The aforementioned drawbacks and obstacles include:

i) high nonlinearity due to the sophisticated phenomena typically involved in the FPM (thermodynamics, reactions kinetics, heat and mass transfer, etc.),

ii) expensive computational cost required for their simulation due to the complexity of the solution procedure –e.g., iterative schemes and/or integration techniques- used to converge them (Garud, et al., 2017), and also to the huge number of equations

contained, e.g., a full-scale refinery model could contain millions of equations (Henao & Maravelias, 2011),

iii) complex architectures, since most of them appear to the user in modular black box style involving intricate connections and recycles among the different units and, also, with no access to the embedded first principle equations (Caballero & Grossmann, 2008), and

iv) noisy calculations, which are introduced by these simulators (e.g., caused by the termination criteria) and hinders the efficient use of derivative-based optimizers, because of the bad estimates of the derivatives and, consequently, the poor optimization results (Quirante, et al., 2018).

These obstacles and challenges can be easily magnified when optimization under uncertainty must be addressed in order to handle process-inherent and/or external uncertainty sources. In more detail, if stochastic programming or robust optimization (i.e., a proactive approach, see Section 1.1.1) are considered, additional challenges will include *i*) the large computational cost associated to the analysis of a large number of uncertainty scenarios, which significantly grows with the number of uncertain parameters, *ii*) the need of complete knowledge of the characteristics of the uncertain parameters to identify their types and probability distributions, which is unrealistic especially in dynamic environments and iii) the limitation that the provided solution becomes suboptimal for most of the realizations of uncertainties during the operation/production (Li, 2010; Pistikopoulos, 2008). On the other hand, the application of the most flexible and reliable reactive approach (i.e., MPP, which is preferred in dynamic or online operation environments as the ones targeted by this Thesis, see Section 1.1.1) requires a well-contracted white-box model of the process (Pistikopoulos, et al., 2007). So, it cannot be applied when considering complex steady-state FPMs characterized by the aforementioned challenging attributes (high nonlinearity, black boxes, large number of equations, noisy, etc.).

1.2.2 Model predictive control

In the MPC scheme, an online dynamic optimization problem (i.e., open loop optimal control) is solved at each sampling period, based on a dynamic model of the process. Firstly, the dynamic model is fed/updated by the current real measurements of the state/output variables collected from the process, which represent the initial conditions of the model at this sampling period (Pistikopoulos, 2008). Secondly, the dynamic optimization problem is solved to find the optimal profile of the control input variables over the entire prediction horizon (an order of magnitude of sampling periods) (Rivotti, et al., 2012). Then, only the values of the calculated optimal control profile corresponding to the first sampling period are implemented

in the plant, and at its end, the state/output variables are measured and their values are used to set up the next open loop optimal control problem, and so on (Tenny & Rawlings, 2004). However, MPC technology faces a major challenge associated to the high computational effort required to repeatedly solve the online open loop control problem at each sampling period. And the solution of an open loop control problem requires the repetitive evaluation of the process dynamic model, which may become computationally unaffordable (Katz, et al., 2020). The situation becomes more challenging when a complex and highly nonlinear dynamic FPM of the process is to be considered, due to the complexity of the solution procedure –e.g., iterative schemes and/or integration techniques- required to solve such FPMs (Davis & Ierapetritou, 2008).

Discretization techniques that transform dynamic FPMs from continuous-time (e.g., differential equations-based FPMs) to discrete-time representations and linearization techniques that linearize the nonlinear behavior, are used to reduce the complexity of such differential models allowing their smooth usage in MPC (Nagy, 2007). Even with the use of these auxiliary simplification methods (which typically implies additional effort, time and also deep mathematical knowledge and, also, leads to a decrease in the resulting model prediction accuracy in favor of its simplicity), the application of MPC to such linearized discrete state-space FPMs can fail when dealing with large-scale and/or fast dynamic processes (Katz, et al., 2018).

1.2.3 Dynamic optimization

As previously mentioned, dynamic optimization techniques, which are also referred to as open loop optimal control techniques, must be performed when dealing with continuous processes in transient state or batch processes (Diehl, et al., 2006; Wang, et al., 2017). Addressing a dynamic optimization problem requires an accurate dynamic FPM of the process/units, typically in the form of differential equations, which is able to predict the evolution of the proceed output or state variables in response to any given time-profile of the control input variables (Banga, et al., 2005). The problem typically involves a multifaceted objective, which is usually based on the final state of the system, but also on its evolution. Two types of methods are considered in the state-of-the-art for solving dynamic optimization problems (Carrasco & Banga, 1997; Banga, et al., 2005). Indirect methods use the analytical necessary conditions from the calculus of variations to formulate a boundary value problem, which is usually very difficult to solve and requires a deep a priori knowledge of the nature of the problem (initialization, constraints structure, etc.), so they are usually inapplicable to the industrial practice (Srinivasana, et al., 2003). Alternatively, direct methods discretize the considered time domain, so as to transform the original infinite continuous optimal control problem into a finite constrained NonLinear Programming (NLP) problem, which is then

solved by appropriate numerical nonlinear optimization tools (e.g., Sequential Quadratic Programming (SQP), trust region search) (Banga, et al., 2005). In spite of their efficiency, practicality and popularity, direct methods can be hindered by the complexity of a dynamic FPM of the process, due to the demanding numerical techniques required for its solution (e.g., integration techniques) (Biegler, 2007).

1.2.4 Fault detection and diagnosis

Model-based FDD approaches have been widely used for chemical processes supervision (Venkatasubramanian, et al., 2003a), within which many FDD methods have been built on the basis of the dynamic state-space FPM of the process. Model-based FDD methods rely on what is named "analytical redundancy" (Patton, et al., 1994; Qin, 2012), through monitoring the extent of matching between the actual process measured features (e.g. state/outputs variables, coefficients or parameters) and the corresponding features calculated by means of a dynamic analytical model of the process, representing the normal or fault-free features. This results in error or residual signals between the model-estimated features and the actual process-measured features (Patan & Parisini, 2005; Isermann, 2005). The values of these errors indicate the extent of the process malfunctioning and, thus, they are used to detect and diagnose faults, by comparing them to threshold values for the errors, or using a more elaborated statistical analysis (Patton, et al., 1995; Narasimhan, et al., 2008; Caccavale, et al., 2010; Elhsoumi, et al., 2011). Amongst model-based methods, observer-based, parity space-based and parameter estimation-based methods are the most common. Model-based methods show great advantages when dealing with dynamic processes, where the monitored inputs and outputs variables are fed into a processor (i.e., diagnostic observer) that represents the knowledge about the process dynamics in order to generate a fault indicator /residual (Patton, et al., 1994; Elhsoumi, et al., 2011). However, they are associated with many shortcomings that complicate their implementation (Venkatasubramanian, et al., 2003a). First of all, the difficulties to create an accurate dynamic FPM of the process should be considered (Ardakani, et al., 2016a; Ardakani, et al., 2016c; Banu & Umab, 2011). Second, most of these methods are based on linear state-space models, whose effectiveness is reduced when applied to highly nonlinear complex processes, because they result in poor linear approximations (Venkatasubramanian, et al., 2003a; Serdio, et al., 2014). Finally, applications addressing large-scale processes would result in a high number of observers, which end up with solutions requiring an unaffordable computational effort if they must be used on-line (Venkatasubramanian, et al., 2003a).

1.2.5 Soft-sensing

Finally, the early and traditional approach for soft-sensing in chemical processes rely on the use of dynamic FPMs that includes a detailed process description based on phenomenological knowledge (Lin, et al., 2007; Jin, et al., 2014). These FPMs are used to predict/monitor the process behavior, either solely or using the information provided by physical sensors (e.g., for continuously adjusting their parameters). However, as previously mentioned, accurate and reliable FPMs of chemical processes are often unobtainable, especially for complex highly nonlinear ones because of the required deep knowledge about the process behavior (Jain, et al., 2007; Jin, et al., 2015).

Furthermore, the available dynamic FPMs of the process/units are often developed under the assumption of favorable (i.e., ideal) working conditions, which are typically not encountered at industrial scale, which is characterized by uncontrolled disturbances, different operating conditions, continuously varying parameters (e.g. heat transfer coefficients) and, possibly, different units/reactors geometries, etc. (Qin, 2012; Kajero, et al., 2017). Also, since the dynamic FPMs of the process/units typically do not consider the physical characteristics of mechanical and electrical components, connections and piping, which remarkably influence the real process, the accuracy of the FPMs-based soft-sensors predictions are reduced (Kadlec, et al., 2009; Jin, et al., 2014; Ali, et al., 2015).

1.3 CHALLENGES TREATMENT METHODS, AND EXISTING GAPS

This Section summarizes the State-Of-Art (SOA) methods and techniques used to minimize the drawbacks and challenges of the use of complex FPMs in each of the aforementioned process operation modules. Also, the section identifies some of the existing gaps with respect to the yet unresolved challenges of using complex FPMs in such applications or regarding other cases in which process real measurements are available without having a reliable FPM. Driven by these gaps, this Section also highlights the potential contributions of the thesis.

1.3.1 Steady-state optimization and uncertainty handling

In order to tackle the challenges associated to the use of complex FPMs in chemical processes operation optimization, the use of Surrogate Based Optimization (SBO) approaches have been proposed and received a big deal of attention (Quirante, et al., 2018). Roughly speaking, the basic idea of SBO is to use the original complex FPM for generating input-output data points (“computer experiments”) that are used to develop accurate, but simple and fast-running, data-driven models (“surrogate models”), which are used instead of the complex FPM

in optimization problem (Ochoa-Estopier & Jobson, 2015). In most of the SBO methods proposed in the chemical process engineering area, two surrogate model types have been common choices, which are the Artificial Neural Networks (ANNs) and kriging models (Kajero, et al., 2017). ANNs offer universal and powerful approximation capabilities due to their flexible structure that can be adapted to capture complex nonlinear behaviors. On the other hand, kriging is able to provide high prediction accuracy with relatively smaller number of training data points, beside its outstanding capability of estimating an error or variance, which represents the uncertainty about the kriging model prediction. Nevertheless, in the SBO literature (Jones, et al., 1998; Jones, 2001; Zuhal, et al., 2019), it has been demonstrated that non-interpolating surrogate models (i.e., regression models, such as ANN) are unreliable in optimization, because they do not appropriately capture the shape of the function to be approximated, and it is usually better to use surfaces that interpolate the data with linear combinations of basic functions (e.g., kriging).

In the chemical process engineering area, two main classes of SBO methods can be identified. The first class is based on partitioning the simulation model into different units or subgroups of units, for each of which a surrogate model is developed. The different surrogates are aggregated/linked to constitute the final approximate model of the process, based on which different optimization schemes have been designated (Salback, 2004; Henao & Maravelias, 2011; Quirante & Caballero, 2016; Quirante, et al., 2018). In most of these cases, the surrogate models must be retrained in each iteration with completely new datasets generated by the FPM simulation. This is because of the continuous modification of the surrogate models input domains during the optimization search as a consequence of shrinking the search area, each iteration, around the current/candidate optimal solution (i.e., refining the optimization search), in order to guarantee the accordance between the output domain of each surrogate model and the input domain of the subsequent/connected surrogate. The advantages of this class of SBO methods include *i*) the capability of handling large-scale systems by splitting them into small units/sections (i.e., surrogate models) and *ii*) the possibility to construct hybrid process models, which combines units or sections of the plant based on their simple and fast FPMs (e.g., splitters, pumps) with surrogate models of other complex units or sections (e.g., distillation columns, reactors). Whereas their limitations are that they iteratively discard the previous training datasets and generate new sets for fitting new surrogate models, which can be computationally prohibitive in an online environment. Also, they do not consider the surrogate models uncertainty during the optimization search, and even when kriging surrogate models are used, they do not exploit the potential capabilities provided by their estimated variance.

On the contrary, the second class of SBO methods (Palmer & Realef, 2002; Kempf, et al., 2012; Chia, et al., 2012; Ochoa-Estopier & Jobson, 2015; Ochoa-Estopier, et al., 2018) is

based on the development of a global surrogate model approximating the entire modular simulator or the flow-sheet of the process. In more details, the input and output variables of these global surrogate models are selected over the entire process flow-sheet as the variables of interest for the optimization problem formulation (i.e., variables representing the optimization decisions (input) and variables constituting the objectives and constraints (outputs)). During the optimization iterations, these global surrogate models are retrained with an updated dataset that includes the original training dataset and, in addition, very few points that represent information about the optimal solution obtained in the previous iteration. The advantages of this class of SBO methods are: *i*) they take into account the surrogate models prediction uncertainty (i.e., the predictors error), which is an essential need in SBO (Jones, 2001; Zhang, et al., 2018), *ii*) they add efficient global exploration capabilities to the search mechanism by not only directing it to the minimum value of the objective predictor, but also to its maximum prediction error (Zuhal, et al., 2019), *iii*) the eventually obtained global surrogate model of the entire process/plant can be further exploited and used for different analysis (Kempf, et al., 2012) and *iv*) relatively few simulation runs of the original FPM are required for updating the surrogate models during the optimization search (Forrester & Keane, 2009), which makes this SBO class more suitable for online application. Nevertheless, this class has some drawbacks such as the difficulty to construct global surrogate models that accurately capture the behavior of large-scale processes and, more importantly, the difficulty of handling constraints.

In the SBO literature (Jones, et al., 1998; Jones, 2001; Zuhal, et al., 2019), it has been shown that even if an interpolating surrogate model is used (e.g., kriging), exploring the surrogate with an arbitrary optimizer can fail even to find local optima, because the surrogate model prediction uncertainty is not considered by the traditional optimizers (Zhang, et al., 2018; Zuhal, et al., 2019). Consequently, there is a need for SBO methodologies that do not only consider the surrogate model prediction, but also consider the uncertainty about this prediction.

On another side, the previously discussed sudden and uncertain variations of some process parameters poses more challenges to the steady-state operation optimization, and can harm the effectiveness of such SBO methods, because these surrogate models are trained by data generated from a FPM whose parameters are set at predefined specific values that lead to the best process-model match. So, any sudden change in the process parameters values makes the surrogate models are no longer valid and, consequently, the obtained optimal solution based on their analysis.

Finally, up to the author's knowledge, the literature, yet, doesn't include proposals or studies for reducing the challenges that face the applications of MPP approaches for handling

uncertainty in the operation optimization of steady-state processes for which the available model is complex, highly nonlinear and/or black box.

1.3.2 Model predictive control

In order to overcome the limitations and challenges of the high computational burden required for solving the MPC problem when an expensive dynamic FPM of the process is considered, explicit MPC methods (also called MultiParametric-MPC (MP-MPC)) have been proposed (Pistikopoulos, 2008; Tian, et al., 2020).

Explicit MPC aims at avoiding the online computations, by solving the MPC problem offline by means of a MPP formulation, which provides the solution in the form of very simple and "explicit" mathematical expressions able of calculating the optimal values of the control inputs the should be applied the next sampling step, as a function of the current values of the process state variables (Pistikopoulos, 2008). The obtained explicit functions take, in most cases, piecewise affine form, and act as explicit control laws that are employed online to calculate, in a very simple and computationally cheap way, the optimal values of the control inputs.

However, again, further to the complex mathematical knowledge required to develop the MPP analysis (Rivotti, et al., 2012), the availability of a dynamic discrete-time linear state-space model of the process is usually a necessity for the practical application of the explicit MPC (Pistikopoulos, et al., 2002; Kouramas, et al., 2011). This, again, may hinder the MP-MPC usage in cases where the available process dynamic FPM is highly nonlinear, high dimensional, with a complicated structure (e.g., sequential simulation models) and/or black box (Rivotti, et al., 2012; Medina-González, et al., 2020). Model approximation and order reduction techniques have been proposed (Rivotti, et al., 2012); however, this may oversimplify the processes behavior and, consequently, degrade the controller performance. Additionally, the effort dedicated to this model simplification step should be also considered.

1.3.3 Data-driven dynamic modeling for supporting control applications

In most control, monitoring and supervision systems (e.g., MPC, dynamic optimization, FDD, etc.), a reliable and accurate dynamic model of the process able to rapidly predict the future values of the process outputs is a must (Nelles, 2001; Ali, et al., 2015). As mentioned before (Section 1.2.2), discretization and linearization techniques may help to reduce the complexity of dynamic FPMs and to obtain simpler discrete-time state-space FPM, however, this may not resolve the computational challenges in cases of large-scale and/or fast dynamic processes (Nelles, 2001; Boukouvala, et al., 2011). In other cases, reliable dynamic FPMs for

complex processes are not available, due to the limited knowledge about the sophisticated behaviors and complex phenomena characterizing these processes (reaction kinetics, thermo-dynamic, etc.), while only real data collected from the process is available (Bradford, et al., 2018; Ali, et al., 2015).

In both cases, system identification or data-driven dynamic modeling methods can be used to construct empirical dynamic models for predicting the future values of the process outputs (Nelles, 2001; Baraldi, et al., 2013). The data used to build these empirical models can be either generated from complex FPM simulations or measured from the real process (Kajero, et al., 2017).

Many methods have been developed for linear dynamic system identification, but their application to nonlinear processes provides unsatisfactory results (Nagy, 2007). As a consequence, advanced data-driven nonlinear modelling techniques, such as ANNs (and their derivatives, e.g., radial basis-ANNs, recurrent-ANNs) and recently Gaussian Process (GP) models (Zhou, et al., 2015; Mattosa, et al., 2017), have been widely proposed to capture nonlinear dynamic relations between the nonlinear process inputs and outputs.

ANNs have become a popular choice due to their universal approximation abilities (Himmelblau, 2000; Poznyak, et al., 2019). A significant number of successful applications of ANNs to dynamic modelling are reported over a wide spectrum of fields (Nelles, 2001; Masters, 1993; Himmelblau, 2000). Especially in the chemical process engineering area, ANNs have been extensively used as Nonlinear AutoRegressive eXogenous (NARX) models for dynamic modelling and system identification of both univariate (single output) (Nagy, 2007; Sadeghassadi, et al., 2018; Poznyak, et al., 2019) and multivariate (multi-output) systems (Adebiyi & Corripio, 2003; Caccavale, et al., 2010; Li & Li, 2015; Lee, et al., 2018). But their usage has two main practical drawbacks: *i*) the large effort required to select a good network structure (Kajero, et al., 2017), and *ii*) the curse of dimensionality (Ažman & Kocijan, 2011).

Recently, GP models have shown promising performance in dynamic modelling and system identification in terms of high prediction accuracy and ease of their parameters tuning, besides, their abilities to reduce the previously mentioned limitations of ANNs (Deisenroth, et al., 2009; Ažman & Kocijan, 2011). This is due to their nonparametric nature: they do not approximate the system by fitting the parameters of a selected structure or functional shape but, instead, they search for relationships among the measured data through a correlation function/model (Boukouvala, et al., 2011). Therefore, the number of the metamodel parameters to be identified is significantly low compared to other parametric models (e.g., ANNs models) and, consequently, the size of the required set of training data is significantly

reduced (Ažman & Kocijan, 2011). Besides, GP models offer high tuning flexibility (Boukouvala, et al., 2011; Rasmussen & Williams, 2006).

In most of the literature studies, GP models have been proposed for univariate dynamic modeling of nonlinear chemical processes (Ažman & Kocijan, 2011; Zhou, et al., 2015), where they are employed as NARX models to predict the future value - over one step-ahead - of an output of interest as a function of the process current inputs and output values. The developed model is, then, used to perform multistep-ahead prediction via recursive calculation, where the predicted output at the current time is fed-back to the model as a part of its input for the next time step prediction. Very few works have extended the GP and kriging capabilities to multivariate dynamic modeling of chemical processes: Hernandez and Grover (2010) Boukouvala, et al. (2011) and Bradford, et al. (2018). However, these works share common limitations: *i*) they have been validated considering processes characterized by very smooth/steady dynamics, without any influencing control/external inputs (Hernandez & Grover, 2010) or with very simple changes in them (Boukouvala, et al., 2011), *ii*) they provided simple Markovian state-space models and they have not illustrated the ability of their methodologies to develop dynamic models with delayed/lagged inputs, *iii*) they presumed that a FPM is always available, which is combined with Design Of Computer Experiments (DOCE) methods to optimally select the training datasets, and *iv*) the robustness of their methodologies to handle different case studies, and their flexibility to integrate different metamodel types are not explored.

An efficient dynamic modelling methodology should be able to handle the challenges usually encountered in real processes, which are : *i*) the existence of many external inputs that control or influence the process causing significant changes in its outputs behavior, *ii*) the possibility of incorporating lags in the model inputs in order to capture possible delayed behavior of the process itself, and/or to compensate for missing repressors (Espinosa & Vandewalle, 1998a; Espinosa & Vandewalle, 1998b), *iii*) handling practical situations, in which real data collected from the process is the only source of information available (i.e., no FPM).

1.3.4 Dynamic optimization

As previously mentioned in Section 1.2.3, direct methods are, in practice, the most common techniques for solving dynamic optimization problems. Direct methods are classified according to the variables to be discretized (Wang, et al., 2017). Sequential approaches (also known as Control Vector Parameterization (CVP)) discretize only the control variables in the form of piecewise low order polynomials, and then a NLP optimization problem is carried out in the space of the discretized control variables, which requires the successive evaluation

(simulation runs) of the dynamic FPM of the process during its solution. On the contrary, simultaneous approaches discretize both control and state variables by approximating them by a family of polynomials on finite elements (Diehl, et al., 2006), so they avoid the inner evaluation of the differential FPM, although they result in a NLP problem of a very large-scale (due to the presence of state variables together with the control variables as optimization decisions (Banga, et al., 2005; Carrasco & Banga, 1997)). Besides, they require the introduction of extra constraints to enforce the continuity of the discretized state variables (Diehl, et al., 2006).

The sequential strategy is straightforward and relatively easy to construct and to apply, and results in a NLP optimization problem of a much reduced size (Carrasco & Banga, 1997; Banga, et al., 2005; Diehl, et al., 2006; Biegler, 2007). However, a major challenge that faces the sequential approach is the huge computational effort associated to a large number of evaluations of the nonlinear process model. Since each evaluation implies the integration of this differential model using expensive integration techniques (Diehl, et al., 2006; Biegler, 2007). This challenge is amplified in cases of complex, large-scale and/or highly nonlinear problems (Srinivasana, et al., 2003), and the computational cost may become unaffordable if a fast identification of the process control profiles is required, which is the case in many industrial applications (transitions between desired operating conditions, response to sudden disturbances or unexpected events, model based control, etc.) (Nagy, 2007).

With respect to the simultaneous strategy, it is not facing direct complications regarding the default/classical use of FPMs (i.e., simulations and the required computational time), because they discretize both the control and the state variables. However, they face obstacles associated to the very large-scale of the NLP problem resulting from this discretization, which includes a large number of equality and inequalities constraints and a potentially large number of degrees of freedom (Biegler, 2007).

Finally, it is worth highlighting that, in the chemical engineering area, the use of data-driven techniques has been rarely proposed in the literature to support dynamic optimization tasks.

1.3.5 Fault detection and diagnosis

In order to cope with the challenges associate to the use of FPMs for FDD of chemical processes (Section 1.2.4), knowledge-based and data-based FDD approaches have been proposed and, also, widely used as powerful alternatives (Calado, et al., 2001; Venkatasubramanian, et al., 2003b).

Knowledge-based approaches rely on the development of some diagnostic rules and the establishment of rule-based expert systems, which necessitate a deep knowledge about process structure and components under the normal (fault-free) and the different possible faulty situations and scenarios (Calado, et al., 2001). However, knowledge acquisition is generally a challenging task (Calado, et al., 2001).

The data-based FDD approaches rely on using data-driven Classification Techniques (CTs), e.g., Support Vector Machines (SVMs), Gaussian Naïve Bayes classifiers (GNBs), Decision Tree (DT), ANNs, etc. These approaches have shown a great flexibility and robustness for the FDD of nonlinear chemical processes without requiring any mathematical model of the process (Askarian, et al., 2016; Ardakani, et al., 2016c). These CTs are trained based on pattern recognition principles from process historical data, including information about normal and different faulty situations (Patton, et al., 1994). Then, the trained CT can be used for the process supervision in order to detect and diagnose possible faults from the process output measurements.

However, these CTs also suffer from serious limitations. The first one is that the classification of faults is based only on the measurements of the process outputs, disregarding any knowledge about the system dynamics (Caccavale, et al., 2010). As a result, they are mostly used for FDD of steady-state processes, where the process is expected to operate under constant conditions/controls (Patton, et al., 1994; Amozeghar & Khorasani, 2016), while it is usually considered that, in dynamic systems, CTs could easily produce false alarms by diagnosing the changes in the processes outputs as faults. This is due to the lack of information about the dynamics governing the relation between the process inputs and outputs. The second limitation is the sensitivity of the CTs to the measurement noise, which makes the errors that very often contaminate the measurements to create false diagnosis and alarms. These usual errors may be random (e.g., sensors white noise) or not (e.g., outliers / biases due to instruments malfunctioning, miss-calibration or poor sampling) (Patton, et al., 1994; Ardakani, et al., 2016b).

Therefore, some works have proposed the use of data-driven dynamic observers (mostly, based on ANNs) to mimic the system dynamic behavior, identifying the underlying dynamic mapping between the system inputs and outputs (Honggui, et al., 2014; Smarsly & Petryna, 2014; Serdio, et al., 2014; Tayarani-B. & Khorasani, 2015; Amozeghar & Khorasani, 2016). These approaches generate a residual vector between the data-driven observer estimated outputs and the process measured outputs which are then used to detect and isolate faults using a threshold value for each residual component or applying some statistical analysis.

Few works (Amozeghar & Khorasani, 2016) have combined these data-based predictors (and the generated residuals) with CTs to automate and improve the FDD task. However, in most of these works, CTs are trained to isolate each fault type when the residual component of a specific output exceeds a specific threshold value. This approach neglects the basic and most important characteristic of any CT, which is its ability to identify a certain pattern in the features (i.e., residuals), regardless of the specific values of the pattern. Furthermore, the identification of a specific threshold value for each residual component as a fault indicator is not a trivial task, as it requires a prior knowledge about the process nominal behavior besides its behavior under the effects of the fault, and may be even infeasible if scenarios with time-varying inputs are considered.

1.3.6 Soft-sensing

Data-driven soft-sensing methods have been proposed, also, to alleviate the complications encountered when using FPMs for soft-sensing in chemical processes. They are gaining wide interest in the process industry, because of their practicability, robustness and flexibility to be developed and applied to a wide range of processes, in addition to their independence from the need to a process mathematical model (Hoskins & Himmelblau, 1988). They are based on the construction of a data-driven model able to accurately approximate the relation between the QIV and other online variables (Bonne & Jorgensen, 2004; Facco, et al., 2009).

In the literature, data-driven soft-sensors have been vastly applied to continuous processes, in order to predict the process steady-state behavior, although they have shown limitations dealing with the transient states of the process (e.g., start-up and shut-down) (Facco, et al., 2009; Wang, et al., 2016). Comparatively, the development and application of data-based soft-sensors to batch processes, which are always in transient state, have been found to be relatively more complicated (Bonne & Jorgensen, 2004; Liu, et al., 2012).

In this scope, the combination of principal component regression and partial least-squares techniques is the most common method for building data-based soft-sensors for linear processes (Jin, et al., 2014; Zamprogna, et al., 2005).

With respect to nonlinear processes, ANNs-based approaches (Masters, 1993) have been often adopted, due to their universal approximation and efficient generalization performance (Yan, et al., 2004; Kadlec, et al., 2009). Several types of ANNs have been efficiently applied for soft-sensing, as feedforward ANNs, radial basis ANNs and fuzzy ANNs (Nelles, 2001; Nagy, 2007). These applications, however, reported the ANNs problems such as the required laborious effort for selecting the network structure and configuration (e.g.,

number of layers, number neurons in each layer, transfer function type, training method) (Azman & Kocijan, 2007; Davis & Ierapetritou, 2007).

The Support Vector Regressions (SVR) model has been also proposed for soft-sensing in batch processes (Yan, et al., 2004; Desai, et al., 2006; Kadlec, et al., 2009). SVR techniques have very good generalization properties and quickness of tuning (associated to the optimization problem solution time for the support vectors selection) (Jain, et al., 2007; Kadlec, et al., 2009). However, the effort and the time required to select the parameters of the SVR model –prior to the optimization-, as the penalty cost, the error margin and the variance become a major limitation (Forrester, et al., 2008).

Recently, GP models are attracting huge attention in the soft-sensing of batch processes area, and have been applied either to continuous (Grbić, et al., 2013; Wang, et al., 2016; Liu, et al., 2016) or to batch processes (Jin, et al., 2015), offering high prediction accuracy and tuning flexibility while requiring a relatively small set of the training data. But the computational effort and capabilities required for the GP model parameters tuning could be a serious shortage, especially for high dimensional cases and/or large training datasets. The kriging models (Krige, 1951; Kleijnen, 2017), which are considered as specific forms/applications of the GP models, have never been introduced to the area of the soft-sensing of batch chemical processes yet.

Most data-driven soft-sensing approaches for batch processes proposed in the literature have not considered the initial conditions of the batches in their design, since they have been tailored for batch processes operated under fixed initial conditions. These approaches have addressed the batch-to-batch data variability -due to a very slight change in the initial condition- from the uncertainty and noise perspectives: input-output training data from different batch runs are assumed to have random errors due to undesired disturbances, which are expected to be representative of a population of batches that are swarming around the mean behavior of the process or what is called the “reference batch” or the “golden batch” (Kadlec, et al., 2009). Then, the correct underlying process behavior can be identified, thanks to the regularization abilities of the employed machine learning techniques, which enable them to learn from this uncertain and perturbed data, and to filter out the assumed noise.

1.4 OBJECTIVES AND CONTRIBUTIONS

This Section states the general objectives of the Thesis (Section 1.4.1) and delineates the specific contributions (Section 1.4.2) that the thesis presents in order to realize/constitute these objectives.

1.4.1 Objectives

Directed by the challenges and criticalities facing the use of FPMs in chemical processes optimization, supervision and control (Section 1.2) and the defined gaps in the SOA methodologies for treating these challenges (Section 1.3), this section delineates the main objectives of the thesis.

- **Objective 1:** the implementation of different state-of-art techniques for DOCE and sequential sampling, data-driven models (also referred to as -depending on the usage context- machine learning models, metamodels or surrogate models) and model validation and assessment procedures.
- **Objective 2:** the development of a framework for data-based modeling of steady-state processes, which integrates the previously implemented techniques and methods (in Objective 1). This framework is aimed at the flexible and robust construction of accurate machine learning or surrogate models of different types, and also the comparison between them, to select the best surrogate model type for the case study to be addressed.
- **Objective 3:** the development of new methods for steady-state operation optimization of processes based on surrogate models, which enable the optimization of complex chemical processes that are difficult to be optimized through existing/classical optimization methods. These difficulties can be due to the complexity and high nonlinearly of the process model and/or the existence of uncertainty in some of the process model parameters.
- **Objective 4:** the development of an efficient and generic framework for data-driven dynamic modelling and emulation of multiinput-multioutput, complex and nonlinear chemical processes. The framework should be aimed at providing dynamic models able to accurately and speedily predict the future behavior of the process outputs over large time horizons.
- **Objective 5:** the integration of these data-driven dynamic models in efficient methodologies for the enhancement of the process monitoring (e.g., a soft-sensing methodology), control (e.g., a dynamic optimization methodology) and supervision (e.g., a FDD methodology).

1.4.2 Thesis contributions

This section defines the specific contributions that this thesis presents in order to cover the gaps and missing requirements highlighted in Section 1.3. Also, the relations between each contribution and the previously stated objectives are outlined.

Figure *1.2* illustrates the locations of each contribution with respect to the process operation modules and associated process model scales.

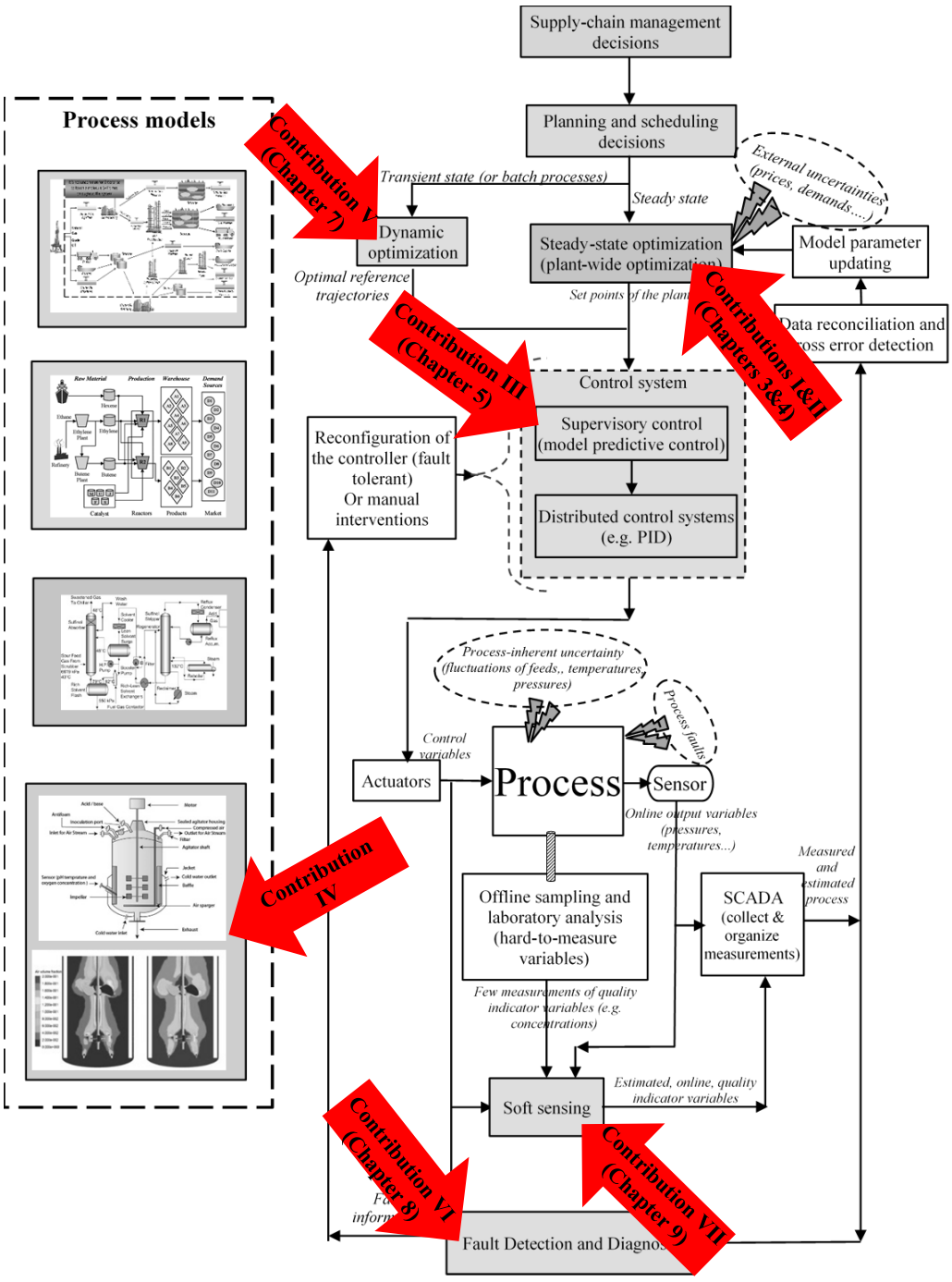


Figure 1.2. Localization of the thesis contributions (red arrows) within the process operation modules and associated process modes scales.

- ***Contribution I:*** *development of a SBO methodology for the constrained optimization of complex, nonlinear steady-state processes, in which the objective function and/or the constraints are represented by black-box models.*

The methodology expands the capabilities of the second class of SBO methods by efficiently handling constraints, and is aimed to assist in the hour-to-hour or day-to-day operation optimization of complex chemical processes, guaranteeing the reliability of the computations and the quick convergence to the optimal solution. This contribution is related to ***Objectives*** **1, 2** *and* **3**.

- ***Contribution II:*** *a novel, easy-applicable and generic data-driven methodology for the multiparametric solution of continuous and mixed integer optimization of chemical processes operation, influenced by traceable uncertainty sources, has been developed. The methodology is aimed at providing very accurate and fast-running data-based models* (*referred to as MultiParametric Metamodels* (*MPMs*)) *that approximate the multiparametric behavior of the optimal solution over the uncertain parameters space. The purpose is to overcome the obstacles that face classical MPP when applied to process operation optimization problems, where complex, highly nonlinear and/or black-box models are used. This contribution is also related to* ***Objectives*** **1, 2** *and* **3**.

- ***Contribution III:*** *it consists in the development of a novel Data-Based MultiParametric -Model Predictive Control (DBMP-MPC) methodology, which enables simple implementations of explicit MPC in situations when the available dynamic FPM model of the process is complex, highly nonlinear and/or black-box, and/or when the deep mathematical knowledge required to develop traditional MP-MPC techniques is not obtainable. This contribution is related to* ***Objectives*** **1, 2** *and* **5**.

- ***Contribution IV:*** *development of a data-driven methodology for multivariate dynamic modeling and multistep-ahead prediction of nonlinear chemical processes using machine learning models. The method overcomes the main limitations currently attributed to the existing approaches in terms of a) the ability to provide accurate data-driven dynamic models for general multi-input/multi-output processes that may involve complex dynamic behaviors (complex control input profiles, delayed behaviors, etc.), b) the ability to simulate the process future outputs over large time horizons, c) the capability to accommodate different types of data modeling techniques and d) the ability of handling different situations, either when a limited set of input-output data signals are available, or when the training data can be optimally generated using a FPM and DOCE techniques. The methodology also introduces the use of the kriging model for the multivariate dynamic modeling in the chemical*

process field in a robust and flexible manner. Finally, the methodology provides a novel DOCE procedure for dynamic modeling, considering the purpose of the simplification and complexity reduction of expensive dynamic FPMs. This contribution is related to ***Objectives*** **1, 2** *and* **4**.

- ***Contribution V:*** *development of a data-driven CVP methodology based on multivariate dynamic data-driven models (**contribution IV**) and a sequential dynamic optimization strategy. The methodology is aimed at enhancing the solution of open-loop optimal control problems in situations where a complex FPMs of the process is to be used and also to assist in situations where a reliable dynamic FPM of the process is not available. This contribution is related to* ***Objectives*** **1, 2** *and* **5**.

- ***Contribution VI****: involves the development of a novel hybrid FDD methodology that combines a dynamic observer based on data-driven multivariate dynamic models (**contribution IV**) and* CTs. *The purpose is to improve the data-driven FDD of nonlinear chemical processes operated under time-varying inputs and, subjected to different types, severities and styles (abrupt and incipient) of faults. This contribution is related to* ***Objectives*** **1, 2, 4** *and* **5**.

- ***Contribution VII:*** *includes, first, the development of a soft-sensing methodology for a special type of batch processes that is rarely explored in the area of soft-sensing: those batch processes that show a characterized variability in their initial settings or conditions (processes aiming to manage raw materials whose specifications or properties differ from one batch to another, or when different product qualities/quantities are to be generated). Hence, the objective is to develop a soft-sensor able to estimate the QIVs along the batch run under any set of initial conditions in the expected operating range. Second, development of an efficient soft-sensor for a real batch pilot plant for waste water treatment, which involves an Advanced Oxidation Process (AOPs) based on the photo-Fenton reaction. Due to the complexity and high nonlinearity of these processes, the best way to address their analytical or phenomenological modeling is still under debate in the scientific and research community; while many data-based modelling studies of these processes have been accomplished from the point of view of experimental design in laboratory scale, their monitoring and control have been never addressed from a soft-sensing perspective, i.e. at industrial or pilot plant scale. Third, exploring the advantages of the kriging technique –as a kind of GP metamodels- for soft-*

sensing in the chemical engineering area. This contribution is related to Objectives 1, 2, 4 and 5.

1.5 THESIS STRUCTURE

This section outlines the thesis structure, which is composed by additional nine Chapters. Excluding Chapter 2 (tools and techniques) and Chapter 10 (conclusions and future works), each of the remaining Chapters (from 3 to 9) addresses one of the contributions previously delineated in Section 1.4.2. Therefore, the Thesis structure is as follows:

- **Chapter 2** overviews the general basics of the tools and techniques used in this thesis for building and developing the novel methodologies. These techniques include DOCE, machine learning models for regression (i.e., surrogate model), machine learning models for pattern recognition (i.e., classifiers), clustering methods and optimization algorithms.
- **Chapter 3 (Contribution I)** reviews in detail the literature of SBO of chemical processes and presents a new SBO methodology for the steady-state operation optimization of complex nonlinear chemical processes, in which the objective function and/or the constraints are represented by black-box functions. The proposed approach consists in replacing the complex, nonlinear, black-box model of the processes built based on first principles with global kriging surrogate models. Then, an active optimization strategy involving a sequential sampling procedure, based on the Expected Improvement (EI) (for unconstrained optimization) or the Constrained Expected Improvement (CEI) (for constrained optimization) techniques, is used to explore the search space of the decision variables and to adapt, accordingly, the surrogate models, so as to reach a global solution for problem. The methodology is tested and compared with classical optimization procedures based on sequential quadratic programming. Both have been applied to three benchmark mathematical examples and to two case studies of operation optimization of chemical processes modeled by modular black-box simulators.
- **Chapter 4 (Contribution II)** presents a general overview on the existing methods for process operation optimization under uncertainty, and presents two novel machine learning-based methodologies for the multiparametric solution of such problems. The first method addresses continuous optimization problems, and aims at developing global MultiParametric Metamodels (MPMs), which are trained using input-output data (uncertain parameters-optimal variables and objective), to approximate the multiparametric behavior

of the optimal solutions over the entire space of the uncertain parameters. The second method targets general Mixed-Integer optimization problems. The method models the multiparametric behavior of the continuous variables by using clustering techniques in order to isolate or highlight those potential local regions of the uncertain parameters space over which the optimal solution behaves significantly different. Then individual MPMs are trained to approximate the optimal solution behavior of each continuous decision variable over each of the identified local regions. For integer decision variables, the method harnesses classification techniques to predict the optimal values of these integer variables also as a function of the uncertain parameters. In both methods, the input-output data are generated through running the optimization problem based on the original process FPM using state-of-art optimizers several times and considering different values of the uncertain parameters that are selected by DOCE techniques. The effectiveness and capabilities of the proposed methods have been proven through their applications to different benchmark examples from the MPP literature and to three cases studies of process and unit operations optimization.

- **Chapter 5 (Contribution III)** presents a Data-Based MultiParametric-Model Predictive Control methodology. The proposed methodology is based on the use of machine learning models which are trained offline using input-output data (initial state variables-optimal control variables) to obtain surrogate models, acting as control laws that approximate the values of the optimal control variables that must be applied along the future sampling period as a function of the current state variables values. Then, during the online application, the optimal control is calculated through simple interpolations using these surrogate models. The input-output training data are generated offline by solving the open loop optimal control problem several times, each using different combination of the initial state variables values selected by a DOCE technique. The method is tested with benchmark problems used in the MultiParametric-Model Predictive Control literature, involving a simple discrete state-space model and a differential FPM of a stirred tank reactor.

- **Chapter 6 (Contribution IV)** reviews in detail the SOA of data-driven dynamic modelling in the chemical engineering area, and presents a novel methodology for data-driven multivariate dynamic modelling and multistep-ahead prediction of nonlinear chemical processes using data-driven models. The proposed methodology utilizes machine learning techniques for building a

group of NARX models, each of them able to predict the evolution of one process output as a function of the other inputs and outputs of the process, over a suitable time lag. The set of multivariate dynamic models are, then, used to forecast the process outputs along larger time intervals (multistep-ahead prediction), through a recursive and inter-coordinated prediction scheme. The methodology also offers a new procedure for training data selection for dynamic modelling, based on the DOCE technique when a FPM of the process is available. The capabilities of the kriging technique are compared with those of one of the most popular techniques (i.e., ANNs). The application of the proposed methodology is illustrated through its application to three case-studies of nonlinear dynamic processes selected from the process industry presented in the literature, including a bioreactor, three-interconnected-tanks and an oil-shale pyrolysis batch reactor.

- **Chapter 7 (Contribution V)** reviews the current methodologies and techniques for the dynamic optimization of chemical processes based on dynamic FPMs. First, it introduces a novel data-driven methodology for the sequential dynamic optimization applicable to solve the open loop optimal control problem of complex highly nonlinear processes. The method is based on the construction of a set of multivariate dynamic surrogate models (**Chapter 6**), which are able to accurately and rapidly predict the process output behavior corresponding to any time-profile of the process control inputs. Second, a sequential dynamic optimization procedure is tuned to integrate this set of dynamic surrogate models representing a complex process FPM. The methodology is applied to three well-known problems from the process systems engineering area, including a plug-flow reactor, batch reactor, and a parallel reaction problem.

- **Chapter 8 (Contribution VI)** presents a detailed literature review on the different approaches and methods for FDD in the chemical engineering area and, then, proposes a novel hybrid data-based methodology for FDD. The main modules of the novel methodology are also described, which are: *i)* a dynamic observer based on multivariate dynamic surrogate models (**Chapter 6**) capable to estimate the expected normal outputs of the process, *ii)* static kriging models smoothing the real measurements of the process outputs to reduce the noise effects and *iii)* data-based classification techniques, which are trained with patterns of residuals created from the comparison between the estimated outputs by the observer and the smoothed real outputs of the process. Different classification techniques such as ANN, SVM, GNB and DT, have been

developed and compared. The performance of the method is illustrated through its application to the well-known three-tank benchmark case study, considering different dynamic operating conditions and faulty situations, including scenarios with modified fault severities and fault styles.

- **Chapter 9 (Contribution VII)** presents a new soft-sensing methodology based on machine learning models for the online prediction of QIV of batch processes operated under changeable initial conditions. The chapter reviews in detail the state-of-art of soft-sensing in the chemical process engineering area and, consequently, claims the contributed novelties. The chapter also compares, within the proposed methodology, the performance of the kriging technique to the most common data-based modelling techniques used for soft-sensing as SVR and ANN, in order to assess its capabilities. The effectiveness and the capabilities of the proposed method is proved by its application to two simulation benchmark case-studies, including a simple batch reactor and a fed-batch fermenter for Penicillin Production. The application is also extended to a real photochemical pilot plant case-study built to investigate water treatment processes based on the photo-Fenton reaction, working in a batch mode and considering paracetamol as reference contaminant.
- **Chapter 10** concludes the Thesis contributions and presents possible future research lines that can be built on the basis of the Thesis.

Chapter 2: Tools and Techniques

This chapter overviews the basics and general characteristics of the different tools and techniques used to build and develop the methodologies in this Thesis. These tools and techniques include design of computer experiment methods, machine learning models for regression (surrogate models or metamodels), machine learning models for classification, clustering techniques and optimization algorithms.

2.1 DESIGN OF COMPUTER EXPERIMENTS

In the area of physical experimentations and laboratory-based investigations, Design Of Experiments (DOE) techniques have been established (Fisher, 1971; Fisher, 1980) and extensively used to select specific combinations of input values (independent design variables) at which experiments must be run to obtain an optimal quantification of the effect of these input variables on the behavior of a certain observed output (dependent) variable. In this sense, different methods have been developed, as full-factorial, fractional-factorial designs for fitting linear regression models, central composite and Box-Behnken designs for fitting polynomial regression models (Fang, et al., 2005). The DOE considers three basic principles: randomization, blocking and replication, in order to avoid prediction bias, obtain homogenous response, and to minimize the experimental random error, respectively (Fang, et al., 2005).

The rapid growth of computer capabilities has motivated huge interests of the engineering research community to study/analyze products and processes using high fidelity and detailed simulation models describing these products or processes. However, serious obstacles hinder the smooth use of such high-fidelity simulation models, such as their complexity, high nonlinearity, sophisticated structure and/or the computational burden required for their convergence (Fang, et al., 2005; Ibrahim, et al., 2019). The rise of advanced machine learning techniques (e.g., Artificial Neural Networks (ANNs), kriging) has inspired the construction of simplified data-driven models trained using input-output data generated by the simulation of the complex FPM (Garud, et al., 2017). Then, these simplified data-driven models (metamodels or surrogate models) take the place of the complex FPM in the targeted application (e.g., optimization, sensitivity analysis, uncertainty quantifications, etc.), providing accurate predictions with simple usage and much lower computational cost. Consequently, this has induced the development of the Design Of Computer Experiment (DOCE) techniques (Jurecka, 2007), which aim at selecting the best combinations of the input variables values -within specific domain or bounds- that can be used for the simulation of the

complex FPM providing the most representative information/knowledge about the output behavior (Garud, et al., 2017). The set of combinations of the input variables values is called a "sampling plan", $[x_i]_n, x_i \in R^k$, where n is the number of sample points or instances and k is the number of input dimensions. The main objective is to collect as much information as possible about the output behavior over all the local sub-regions of the input space, assuming that most computer simulation models are deterministic. As a result, DOCE techniques consider samples selection criteria different from those of the DOE, which are, mainly, the space-fillingness and stratification of the sampling plan. While the space-fillingness criterion aims at spreading the sampling plan points over the whole input domain, the stratification ensures that the sampling plan points projection onto each input variable axis is uniform (Garud, et al., 2017; Forrester, et al., 2008). Both criteria ensure high uniformity of the sampling plan and better covering of all the local sub-regions of the input domain.

Many DOCE techniques have been developed for static surrogate modeling. The most common techniques include Latin hypercube sampling (Forrester, et al., 2008), low discrepancy sequences as the Hammersley technique (Ibrahim, et al., 2019) and space-filling designs as max-min techniques and Space-filling Latin Hypercube Sampling (SLHS) design (Joseph, 2016). Alternatively, sequential or adaptive sampling are special type of DOCE techniques that are commonly related to the use of kriging/GP models (Kajero, et al., 2017). In these sequential techniques, the total number of training points are not selected at once, but: the surrogate model is, initially, fitted with a relatively small number of training points, and it is, iteratively, adapted by adding new training points of interest (infill or update points) to the initial training dataset and, then, the surrogate model is refitted so as to enhance a desired criterion or index of its performance. This criterion is highly dependent on the eventual use of the surrogate model (for global approximation, surrogate-based optimization, reliability analysis, etc.). The iterative procedure stops when the surrogate model performance index reaches a desired level. For example, when the surrogate model is to be used just for global approximation, the most common performance criterion is the maximization of the surrogate model global accuracy, which implies the selection of infill points that maximize the estimated prediction error of the kriging/GP model (Jurecka, 2007; Forrester, et al., 2008). Notice that the term "global" refers to the globality with respect to the entire input domain.

Most of these DOCE techniques show both desired and limiting characteristics in terms of the uniformity of the generated sampling plan and the required computational cost. For example, Latin hypercube and low discrepancy sequence designs provide sampling plans of good uniformity with very low computational cost (Ibrahim, et al., 2019). Space-filling designs are able to provide sampling plans with very high uniformity, although the associated computational cost is relatively high (Joseph, 2016; Caballero & Grossmann, 2008). Because,

these techniques usually encompass a complex optimization problem, in which the locations of the input combinations (i.e., instances) within the k-dimensional input space of the model (i.e., the decision variables) are manipulated to maximize a certain space-filling criterion (i.e., objective function) (Forrester, et al., 2008). For instance, in the max-min space-filling designs, the objective function is to maximize the minimum distance between the sample points. The former criterion tends to distribute the sampling points uniformly across the k-dimensional input domain (Fang, et al., 2005). Sequential sampling designs have also shown very high uniformity and high efficiency, since they take maximum advantages of each point in the sampling plan, but the computational cost of such techniques is extremely high, since each iteration involves an optimization problem seeking for the point that optimally enhances the surrogate model accuracy, in addition to the subsequent fitting of the surrogate model with the updated training set (Jones, et al., 1998). For the previous reasons, the iterative DOCE procedures are favorable when dealing with very expensive FPM (e.g., computational fluid dynamic models) where the cost of one simulation run using the FPM is much higher than one iteration of the sequential sampling procedure. A more detailed analysis about the different DOCE techniques can be found in (Garud, et al., 2017; Fang, et al., 2005; Jurecka, 2007; Garud, et al., 2017)

After designing an efficient sampling plan$[x_i]_n, x_i \in R^k$, a computer experiment or a simulation run using the complex FPM is carried out at each point of the sampling plan, to obtain the response or output variable values $[y_i]_{n,}\ y_i \in R$.

In general, most of the Thesis chapters/methodologies consider the Hammersley design technique, due to its ability to provide sampling plans of good uniformity and stratification properties with very low computational cost (Garud, et al., 2017; Ibrahim, et al., 2019). In particular, Chapter 3 employees sequential sampling techniques for building SBO methods.

On another hand, number of sample points (n) required to train the surrogate model in order to capture the output behavior with satisfactory accuracy is case-dependent. Because the selection of n depends on the input dimensionality of the surrogate model (k), the volume of the input space and, also, on the intricacy and nonlinearity of the considered output behavior. In general, as n increases, the effort (time/cost) required not only for executing the experiments, but also for the surrogate model fitting increases. Then, the modeler should carefully balance the trade-offs between the required surrogate model accuracy, the computational cost and the eventual application benefits of the surrogate model.

2.2 MACHINE LEARNING FOR REGRESSION (SURROGATE MODELS)

In the Thesis different machine learning models for regression have been considered. The main objective is not the detailed comparison of the machine learning models performance, but to assess the robustness, applicability and flexibility of the proposed methodologies (in which these machine learning models are employed, as well as other tools) by handling different data-based modelling techniques and software.

2.2.1 Ordinary kriging

The Ordinary Kriging (OK) model has emerged in the field of geo-statistics (Krige, 1951; Cressi, 1993), and after the pioneer work of Sacks, et al. (Sacks, et al., 1989) and Jones (Jones, 2001), OK became popular for modeling and optimization of complex highly nonlinear static systems in various engineering areas. The OK is a nonparametric data-driven model that has shown potential capabilities to approximate highly nonlinear, multimodal and complex systems (Fang, et al., 2005; Queipo, et al., 2005). These capabilities stem from the ability of the OK to combine global modeling through estimating a general trend of the system to be approximated, and local modeling through a spatial correlation function. Besides, this model is able to estimate a prediction variance or error, which represents an uncertainty measure about its prediction (Forrester, et al., 2008).

Given a set of n input-output training data $[x_i, y_i]_n, x_i \in R^k, y_i \in R$, $i = 1,2,..n,$, the OK assumes the predictor $\hat{y}(x) = \mu_{ok} + Z(x)$, where the constant term μ_{ok} represents the main trend of the system to be approximated, and $Z(x)$ is a deviation/residual from that trend, which accounts for detailed complex behavior of the system that could not be captured via the main trend μ_{ok}. The residual $Z(x)$ is modeled as a stochastic Gaussian process with expected value of $E(Z(x)) = 0$, and a covariance between two residuals $cov(Z(x_i), Z(x_j))$ that only depends on their corresponding inputs locations x_i, x_j. Thus, it can be calculated as: $cov\left(Z(x_i), Z(x_j)\right)$=$\sigma_{ok}^2\, R(x_i, x_j)$, being σ_{ok}^2 the process variance, and $R(x_i, x_j)$ a correlation function calculated as $R(x_i, x_j) = exp\left(-\sum_{l=1}^{k} \xi_l \left|x_{i,l-} x_{j,l-}\right|^{p_l}\right) + \delta_{ij}\, \lambda,$ where, $\xi_l, l = 1, \dots k$ are the model hyper-parameters, p_l are smoothing parameters, δ_{ij} is the Kronecker delta and λ is a regularization constant that enables the kriging predictor to regress noisy data (Azman & Kocijan, 2007). The value of the parameter ξ_l represents a measure of the degree of correlation among the data along the l^{th}input dimension.

The maximization of the likelihood function (Eq.(2.1)) of the observed data $[Y]_{n\times 1}$ yields the closed form mathematical expressions for the optimal values of μ_{ok} and σ_{ok}^2 that are shown in Eq.(2.2) and Eq.(2.3), respectively, where $[X]_{n\times k}$ is the matrix of training inputs,

$[Y]_{n\times 1}$ is the corresponding vector of the training outputs, $[R]_{n\times n}$ is the correlation matrix between the training inputs and $[\mathbf{1}]_{n\times 1}$ is the identity vector- it is highlighted with bold font to differentiate it from the normal number 1 in the equations- (Caballero & Grossmann, 2008).

$$Lik\left(\mu_{ok}, \sigma_{ok}^2, X\right) = \frac{\mathbf{1}}{(2\,\pi\,\sigma_{ok}^2)^{\left(\frac{n}{2}\right)}|R|^{\left(\frac{1}{2}\right)}}\; exp\left(-\frac{(Y-\mathbf{1}\,\mu_{ok})^T\,R^{-1}(Y-\mathbf{1}\,\mu_{ok})}{2\,\sigma_{ok}^2}\right) \tag{2.1}$$

$$\mu_{ok} = \frac{\mathbf{1^T}\,R^{-1}Y}{\mathbf{1^T}\,R^{-1}\mathbf{1}} \tag{2.2}$$

$$\sigma_{ok}^2 = \frac{(Y-\mathbf{1}\,\mu_{ok})^T\,R^{-1}\,(Y-\mathbf{1}\,\mu_{ok})^T}{n} \tag{2.3}$$

The substitution of the optimal values of μ_{ok} and σ_{ok}^2 in the likelihood function leads to the maximization of the concentrated log-likelihood function, which is given by Eq.(2.4).

$$Max_{\,(\xi_l,p_l)}\left[-\frac{n}{2}ln\left(\sigma_{ok}^2\right)-\frac{1}{2}ln(|R|)\right] \tag{2.4}$$

The kriging predictor (Eq.(2.5)) and its estimated error (Eq.(2.6)) are obtained by deriving the augmented likelihood function of the original training data set and a new interpolating point (x^*, y^*). In Eq. (2.5), $[r]_{n\times 1}$ is the vector of correlations between the point to be predicted $x^*$and the original training data points, and calculated as $R\,(x_i, x^*)$ (Jones et al., 1998; Caballero & Grossmann, 2008; Forrester et al., 2008).

$$\hat{y}(x^*) = \mu_{ok} + r^T R^{-1}(Y-\mathbf{1}\mu_{ok}) \tag{2.5}$$

$$\hat{s}^2(x^*) = \sigma_{ok}^2(1+\lambda-r^T R^{-1}r+(1-\mathbf{1}^T R^{-1}r)^{-1}/(\mathbf{1}^T R^{-1}\mathbf{1})) \tag{2.6}$$

The fitting of an OK model is achieved by obtaining the optimal parameters $[\xi_l, p_l, \lambda]$ through the maximization of the concentrated log-likelihood function. In practice, this optimization problem is computationally challenging, because of the high computational cost associated to the repetitive calculation of the inverse of the correlation matrix $[R]_{n\times n}$ during the optimization iterations. This effort quickly grows with the size of the training data set and/or the model input dimensionality. Besides, the nature of the concentrated log-likelihood function itself is quite complicated, because it is flat near the optimum (Fang, et al., 2005). More details about these computational challenges and the numerical methods and optimization techniques to overcome or reduce these obstacles can be found in (Forrester, et al., 2008).

This Thesis considers the OK implementation developed by Forrester, et al. (2008), because of its high efficiency, generality and applicability. Besides, the "*fmincon*" algorithm included in the Matlab optimization toolbox is used for the maximization (nonlinear

optimization) of the concentrated likelihood function (Eq.(2.4)), considering different values of the initial solution, so as to avoid the entrapment in a local optimum (Matlab, 2018). The Cholesky factorization has been used to find the inverse of $[R]_{n\times n}$ matrix to avoid ill-conditioning, and the smoothness parameters p_l are often kept to the value of 2, which provide smooth infinitely differentiable correlation functions (Forrester et al., 2008).

The Thesis also considers (in some chapters) another different software implementation for the GP model, which is the GP-Regression (GPR) algorithm based on the function "*fitrgp*" included in the Matlab statistics and machine learning toolbox (Matlab, 2018).

2.2.2 Artificial neural networks

Artificial Neural Networks are a very well-known and widely-used efficient technique for nonlinear data-driven modelling. The technique is inspired from the biological neural networks of the brain nervous system (Masters, 1993; Himmelblau, 2000). An ANN is a lattice of nodes, termed as neurons, which are placed in this lattice through a certain number, $n_{\mathbb{L}}$, of layers, $\mathbb{L}_{\mathrm{I}}, \mathrm{I} = 1,2,..n_{\mathbb{L}}$, and are interlinked together to be capable of the nonlinear processing of the information. Figure 2.1 shows a schematic representation of one-hidden layer ANN. The weight value $\varpi_{i^{\mathbb{L}_{\mathrm{I}}},j^{\mathbb{L}_{\mathrm{I}+1}}}$ is assigned to the link connecting the i^{th} neuron in the layer, $\mathbb{L}_{\mathrm{I}}$, to the j^{th}neuron in the successive layer, $\mathbb{L}_{\mathrm{I}+1}$; additionally, a bias, $b_{i^{\mathbb{L}_{\mathrm{I}}}}$, is considered as an independent input to the i^{th} neuron in each layer, $\mathbb{L}_{\mathrm{I}}$. Considering one-hidden layer ANN (Figure 2.1), the output $a_{j^{\mathbb{L}2}}$ of thej^{th} neuron in the hidden layer is computed as the weighted sum of its inputs received from the neurons in the previous layer plus the bias, see Eq.(2.7), where $Q^{\mathbb{L}1}$ and $Q^{\mathbb{L}2}$ are the numbers of neurons in the input and hidden layers, respectively. The computed value is, then, processed by a transfer function, ƒ, and is sent to the output layer (Masters, 1993; Nagy, 2007) that calculates the ANN output, $\hat{y}$, as in Eq.(2.8). Notice that the formulations for the multi-layer ANN is straightforward.

The training of an ANN is accomplished relying on a set of input-output training patterns $[x_i, y_i]_n, x_i \in R^k, y_i \in R^k,\ i = 1,2,..n$, and through the solution of a nonlinear optimization problem, in which an objective or loss function is minimized by tuning the optimization variables values represented in the weights and the biases of the neurons (Masters, 1993; Nagy, 2007). With respect to ANNs models for regression, the loss/objective function of the training task is related to the sum of errors between the predicted outputs by the network, $\hat{y}_i$, and the target outputs, y_i, being the Mean Squared Error (MSE) (given by Eq.(2.9)) the most common loss function.

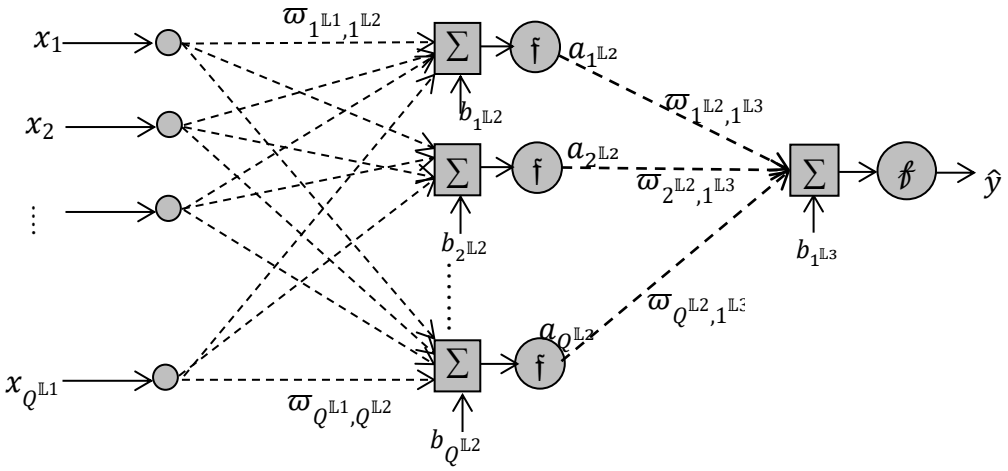


Figure 2.1. Schematic illustration of a feedforward ANN structure: an input layer, at least one hidden layer and an output layer.

$$a_{j^{\mathbb{L}2}} = \mathfrak{f}\left(b_{j^{\mathbb{L}2}} + \Sigma\, x_{ij}\, \varpi_{i^{\mathbb{L}1},j^{\mathbb{L}2}}\right), \quad i = 1,2,..Q^{\mathbb{L}1}, \qquad j = 1,2,..Q^{\mathbb{L}2} \tag{2.7}$$

$$\hat{y} = \mathfrak{f}\left(b_{1^{\mathbb{L}3}} + \Sigma a_{j^{\mathbb{L}2}}\, \varpi_{j^{\mathbb{L}2},1^{\mathbb{L}3}}\right) \quad j = 1,2,..Q^{\mathbb{L}2} \tag{2.8}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.9}$$

Amongst various kinds of ANNs, the feed-forward multi-layer perceptron is considered as the most popular kind used in engineering practices (Himmelblau, 2000), as it offers high efficiency, accuracy and straightforward applicability (Fang, et al., 2005; Nagy, 2007).

The "*feedforwardnet*" function of the Matlab ANN toolbox is used in this Thesis to build multilayer ANNs for regression (Matlab, 2018). A trial and error procedure is employed for selecting the suitable number of layers, number of neurons and the training algorithm achieving a compromise between the structure simplicity and the prediction accuracy. Two training algorithms have been considered in the Thesis, depending on the different application cases, which are the Levenberg-Marquardt backpropagation based on the Matlab function "*trainlm*" and the Bayesian regularization backpropagation based on the Matlab function "*trainbr*". The latter minimizes a combination of MSE and the network weights, which leads to very good generalization properties. The default "*sigmoid*" transfer function in the hidden layers, $\mathfrak{f}$, and "*linear*" transfer function in the output layer, $\mathfrak{f}$, are maintained.

2.2.3 Support vector regression

Given a set of input-output training data $[x_i, y_i]_n, x_i \in R^k, y_i \in R,\ i = 1,2,..n,$ a Support Vector Regression (SVR) model (Vapnik, 1995) maps the input data original

space into a high-dimensional feature space, often through a basis or kernel function $\Phi\ (x_i, x_j)$ that may be presented by different styles as linear, polynomial, Gaussian, etc. Then, the modeling problem becomes the determinations of the optimal (flattest) surface $\hat{y}(x) = b + \sum_{i=1}^{n} w_i \Phi\ (x_i, x_j)$ in this feature space, which fits the data, where $b = \mu_{svr}$ is a base or bias (Forrester & Keane, 2009). This can be done through the minimization of the weights vector norm $|w|^2, w \in R^n$. In order to ensure better generalization performance, SVR allows specifying margins or a tube around the training data with a radius $\pm\varepsilon$, within which prediction errors in the training data are accepted or tolerable (constraints of the optimization problem). Additionally, to tolerate outliers, the data that presents a prediction error bigger than $\pm\varepsilon$ is penalized using the so-called ε-sensitive loss function (Forrester, et al., 2008). Then the model fitting problem can be expressed as:

$$Min\ \frac{1}{2}|w|^2 + \frac{C_{svr}}{n}\sum_{i=1}^{n}(\xi_i^+ + \xi_i^-) \tag{2.10}$$

$$\text{S.T.} \quad \left.\begin{array}{l} y_i - \mu_{svr} - w\ x_i \leq \varepsilon + \xi_i^+ \\ \mu_{svr} + w\ x_i - y_i \leq \varepsilon + \xi_i^- \\ \xi_i^+\ ;\ \xi_i^-\ \geq 0 \end{array}\right\} \tag{2.11}$$

Where $\xi_i^+\ ;\ \xi_i^-$ are the slack variables that describe the size of the positive and negative violation or excess than the tube radius ε for each training data sample, and $C_{svr} > 0$ is a penalty factor that controls the trade-off between the model complexity (the flatness of $\hat{y}$) and the degree to which errors larger than ε are tolerated (Forrester & Keane, 2009). A schematic representation of the problem is illustrated in Figure 2.2.

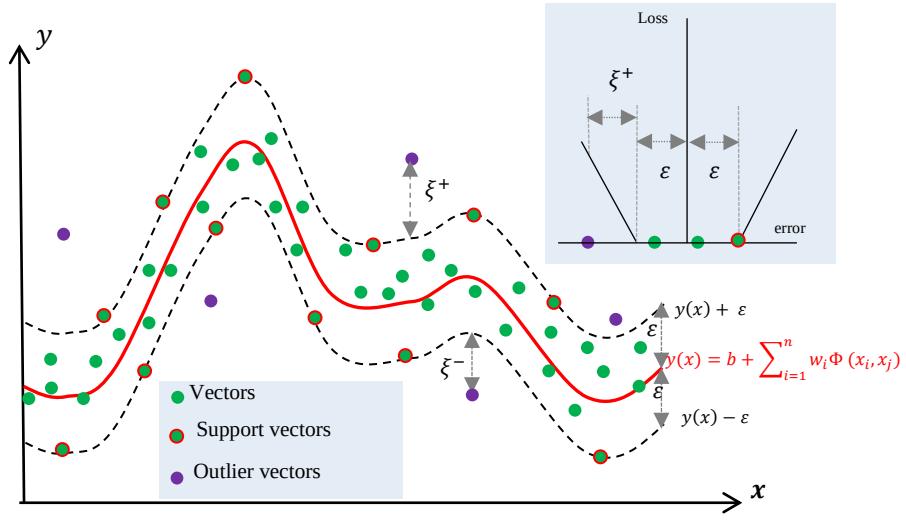


Figure 2.2. Representation of the SVR model.

The constrained optimization problem can be reformulated into a dual problem form by introducing Lagrange multipliers $\eta_i^+, \eta_i^-, \alpha_i^+, \alpha_i^-$ to the constraints in Eq.(2.11), in order to combine them with the objective forming at the end the Lagrangian function:

$$L = Min\ \frac{1}{2}|w|^2 + \frac{C_{svr}}{n}\sum_{i=1}^{n}(\xi_i^+ + \xi_i^-) - \sum_{i=1}^{n}\alpha_i^+(\varepsilon + \xi_i^+ - y_i + \mu_{svr} + w\,x_i) + \sum_{i=1}^{n}\alpha_i^-(\varepsilon + \xi_i^- + y_i - \mu_{svr} - w\,x_i) - \sum_{i=1}^{n}(\eta_i^+\,\xi_i^+ + \eta_i^-\,\xi_i^-) \quad (2.12)$$

The resulting objective L is then minimized with respect to w, μ_{svr} and the primal variables $\xi_i^{\pm}$, and also it is maximized with respect to the dual variables $\eta_i^{\pm}, \alpha_i^{\pm}$, where $\eta_i^{\pm}, \alpha_i^{\pm} \geq 0$. For the active constraints $(\alpha_i^+ + \alpha_i^-) \geq 0$, the corresponding y_i will become the support vectors, while for inactive constraints $(\alpha_i^+ + \alpha_i^-) = 0$, the corresponding y_i will be excluded from the prediction (Forrester & Keane, 2009).

The values of these Lagrange multipliers α_i^+, α_i^- are determined by solving the dual optimization problem. The training vectors (samples) with non-zero Lagrange multipliers are called support vectors, which represent/construct the margins or the borders of the tube. Finally, the optimal weights w and the constant bias μ_{svr} can be calculated from the relations in Eq.(2.13) and Eq.(2.14), and the final predictor is expressed by Eq.(2.15).

$$w = \sum_{i=1}^{n}(\alpha_i^+ - \alpha_i^-)\Phi(x_i) \quad (2.13)$$

$$b = \mu_{svr} = y_i - \sum_{j=1}^{n}(\alpha_j^+ - \alpha_j^-)\Phi(x_i, x_j) \quad (2.14)$$

$$\hat{y}(x^*) = b + \sum_{i=1}^{n}(\alpha_j^+ - \alpha_j^-)\,\Phi(x^*, x_i) \quad (2.15)$$

A drawback of the SVR is the huge time and effort required to select the kernel function type and the values of its parameters (e.g., the value of the parameter σ_{svr}, in a Gaussian kernel $\Phi(x_i, x_j) = exp(-\|x_i, x_j\|^2/2{\sigma_{svr}}^2)$), which are case dependent. The detailed mathematical description and derivations can be found in (Vapnik, 1995; Forrester, et al., 2008; Forrester & Keane, 2009). This Thesis uses the SVR algorithm based on the function "*fitrsvm*" included in the Matlab statistics and machine learning toolbox (Matlab, 2018).

2.3 MACHINE LEARNING FOR CLASSIFICATION

Classification Techniques (CTs) are supervised machine learning models that perform pattern recognition tasks (Vapnik, 1995; Zhang, 2000). Given a set of input-output data $[x_i, y_i]_n,\ x_i \in R^k, i = 1,2,..n,\ y_i \in \{l_1, \dots, l_j, \dots, l_{n_l}\},\ n \ggg n_l$, a classifier is trained to

assign to a new observation, x_i, a specific labels $y_i = l_j$ among a predefined set of n_l labels or categories (Zhang, 2000). Among many types of classification techniques available in the literature, this Thesis, consider ANN and Support Vectors machine (SVM) classifiers because of their widely reported high accuracy and application simplicity (García-Laencina, et al., 2010). In the Thesis, classifiers are employed to model categorical output variables consisting of specific classes, for example, binary decision variables in optimization problems, and different faults types affecting the process.

2.3.1 Support vectors machine

In the literature, the SVM technique always shows very good classification capability (Rocco & Zio, 2007 ; Tao, et al., 2018) in terms of i) providing high classification accuracy, ii) requiring small computational effort for its training due to the relatively small number of parameters, which are optimized by solving a simple quadratic optimization problem; iii) managing imbalanced datasets thanks to its Cost-Sensitive SVM settings. Given a set of input-output training data $[x_i, y_i]_n$, $x_i \in R^k, i = 1,2,..n$, $y_i \in \{+1, -1\}$, a binary SVM model is built by solving the problem of identifying the optimal linear decision boundary separating the training instances into two classes (i.e., +1 ad -1) (Christianini & Shawe, 2000), see Figure 2.3.

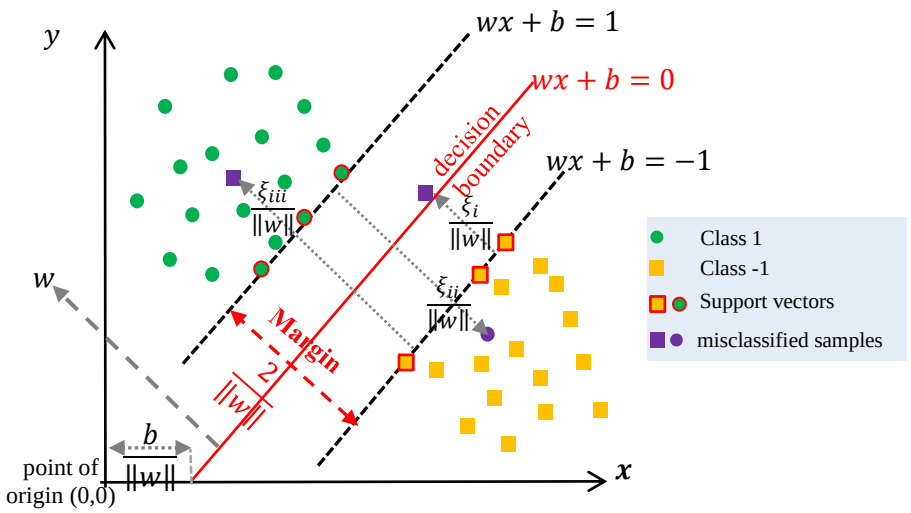


Figure 2.3. Representation of the SVM model.

This decision boundary is parametrized by (w, b), where w is the vector perpendicular to the boundary and b is the distance from the point of origin (0,0), such that $\langle w, x_i \rangle + b \geq 0$ when $y_i = +1$ and $\langle w, x_i \rangle + b \leq 0$ when $y_i = -1$, with $\langle \cdot, \cdot \rangle$ is the dot-product in R^k (Vapnik, 1995). The optimal decision boundary is the one with the maximum margin, which is the distance from the decision boundary to its closest positive and negative training instances

(which are the support vectors), and it geometrically equals to $\frac{2}{\|w\|}$ (Maldonado, et al., 2014). A perfect linear separation of all the n training patterns is not always possible, therefore, in order to tolerate misclassified samples, a slack variable ξ_i is introduced for each training pattern $i = 1, \dots, n$ (Christianini & Shawe, 2000). Then problem can be formulated as in Eq.(2.16), where $C > 0$ is a parameter that penalizes the misclassification. Then, the constrained optimization problem can be reformulated into a dual problem by introducing Lagrange multipliers, and then solved through quadratic optimization (Platt, 1999).

$$\left.\begin{array}{c} \min\limits_{w,b,\xi} \frac{1}{2}\|w\|^2 + \frac{C}{n}\sum\limits_{i=1}^{n}\xi_i \\ \text{S.T.: } y_i(w^T x_i + b) \geq 1 - \xi_i, i = 1, \dots, n \\ \xi_i \geq 0 \end{array}\right\} \quad (2.16)$$

The SVM algorithm can be generalized to non-linear classification, which is needed when data patterns cannot be separated by a linear hyperplane (Platt, 1999). This procedure is known as kernel trick, where the input data original space is mapped into a high-dimensional feature space through a kernel function that can be of different types, e.g., linear, polynomial, Gaussian, etc. (Xu, et al., 2018). On another side, the SVM classifier is binary in nature (i.e., able to manage two classes only, e.g., on/off), and its usage in multiclass classification problems requires elaborated employment strategies such as one-versus-all or one-versus-one (Xu, et al., 2018).

This Thesis uses the support vector classification algorithm based on the function "*fitcsvm*" included in the Matlab statistics and machine learning toolbox (Matlab, 2018).

2.3.2 ANN classifier

A feedforward ANN for classification has the same structure as that of regression (see Section 2.2.2, Figure 2.1), except for the output layer structure, output transfer function and the training loss function (Li, et al., 2019; Kline & Berardi, 2005). Generally, the structure of the output layer of an ANN classifier consists in a number of neurons equals to the number of the considered classes (i.e., one neuron is associated to one class). The output values calculated by the neurons in the output layer are, then, processed by a Soft-Max transfer function (Li, et al., 2019), which provides the final output of the classifier in the form of a multinomial probability vector of length equals to the number of classes. Each value in this vector ranges between [0,1], representing the probability that the current observation belongs to the specific class (Li & Wang, 2020), and the summation of all the probabilities in the vector equals to 1 (Kline & Berardi, 2005). Regarding the loss function for training an ANN classifier, the cross-

entropy type is typically considered, which is a measure of goodness of separability of the distributions of classes.

In this Thesis, the "*patternnet*" function of the Matlab ANN toolbox is used to construct multilayer ANNs for classification (Matlab, 2018). The ANN structure is also selected by a trial and error procedure, as in Section 2.2.2. The scaled conjugate gradient backpropagation algorithm based on the Matlab function "*trainscg*" is used to train the network, and the default "*tan-sigmoid*" transfer function in the hidden layers and "*softmax*" transfer function in the output layer, are used.

In order to assess the performance of the classifiers (ANN, or SVM), the $f1$-score criterion is considered, which is calculated as in Eq.(2.17) (based on a test dataset) and represents the harmonic mean of classifier precision and recall. The $f1$-score ranges from 0.0 (worst value) to 1.0 and (best value) (Tao, et al., 2018).

$$f1 - score = \frac{2 \times Precision \times Recall}{Percision + Recall} \tag{2.17}$$

Where, $Recall = TP/(TP + FN)$, $Preciosn = TP/(TP + FP)$ and TP, TN, FP, FN are the numbers of true positive, true negative, false positive and false negative predictions, respectively, for a test dataset, (Rocco & Zio, 2007).

2.4 CLUSTERING TECHNIQUES

Clustering is an unsupervised data-driven technique, which aims at partitioning unlabeled dataset into smaller groups or subsets called clusters, in such a way that objects in the same cluster are more similar (in certain respects) to each other than to those in other clusters (Kaufman & Rousseeuw, 1990). They are very useful in discovering hidden structures in the data. There are different categories of clustering methods, including connectivity-based (e.g., hierarchical clustering), centroid-based (e.g., K-means clustering), distribution-based (e.g., Gaussian mixture models), density-based clustering and many others (Lloyd, 1982). The selection of the specific clustering method is highly dependent on the required function of the clustering analysis, the behavior/nature of the data and the affordable computational time.

The K-means algorithm is one of the most used clustering techniques, because of its simplicity and low convergence time (Lloyd, 1982). It is an iterative data-partitioning algorithm that partitions a dataset of n observations into a user-specified number of K clusters. The algorithm provides as output, K centroids for the K clusters, besides a membership label for each data point associating it to one cluster, in such a way that the point is closest to the centroid of its cluster than to the centroids of the other clusters (Kaufman & Rousseeuw, 1990). The very basic steps of the algorithm are as follow:

i) Initialization: once the number $\mathbb{K}$ of clusters has been specified, $\mathbb{K}$ centroids are randomly (for instance) selected in the data space.

ii) Assignment: each sample in the dataset is assigned to a cluster, in such a way that the sample is closer to the centroid of this cluster than the centroids of the other clusters.

iii) Update: the positions of the centroids of the clusters are updated based on a proximity measure (i.e., objective function) calculated based on the samples that are already assigned to them.

iv) Repeat ii) and iii) until no change in positions of the centroids or in the distribution of the samples over the $\mathbb{K}$ clusters.

The most common proximity measure for updating the centroids is the minimization of the Sum of Squared Error (SSE), while the most popular distance measure is the squared Euclidian type (Kaufman & Rousseeuw, 1990). Assuming that, at the m^{th} iteration of the algorithm, the centroids of the $\mathbb{K}$ clusters are $[c_1^m, \dots, c_i^m, \dots, c_{\mathbb{K}}^m]$, the SSE can be calculated as:

$$SSE = \sum_{i=1}^{\mathbb{K}} \sum_{x_j \in c_i^m} \left\| x_j - c_i^m \right\|^2 \tag{2.18}$$

Hence, the centroids at the next iteration $m + 1^{th}$ can be obtained by minimizing the SSE measure, through setting its differentiations with respect to c_i^m to be equal to zero (Lloyd, 1982), see Eq.(2.19).

$$\frac{\partial\, SSE}{\partial\, c_i^m} = \frac{\partial \sum_{i=1}^{\mathbb{K}} \sum_{x_j \in c_i^m} \left\| x_j - c_i^m \right\|^2}{\partial\, c_i^m} = 0 \tag{2.19}$$

This leads to the closed form expression for calculating the centroids at the next iteration, as:

$$c_i^{m+1} = \frac{1}{n_{c_i^m}} \sum_{x_j \in c_i^m} x_j \tag{2.20}$$

The $\mathbb{K}$-means algorithm based on the function "*kmeans*" included in the Matlab statistics and machine learning toolbox (Matlab, 2018), is considered.

2.5 OPTIMIZATION ALGORITHMS

In all the layers of the general decision-making hierarchy of chemical plants management, most of the decisions are made through solving different types and scales of

model-based optimization problems (superstructure optimization, supply chain optimization, planning, scheduling, steady-steady operation optimization, control, etc.) (Biegler, 2010). A very general formulation of an optimization problem can be represented as follows:

$$\left.\begin{array}{c} \min f(x,y) \\ S.T.:\ g_i(x,y) \le 0, \quad i = 1,\dots,m, \\ h_i(x,y) = 0, \quad i = 1,\dots,m_{eq}, \\ x \in R^k,\ lb \le x \le ub, \\ y \in \{0,1\}^{\mathbb{K}} \end{array}\right\} \tag{2.21}$$

Where $f(x,y)$ is the objective function to be optimized (e.g., operational cost, profit, efficiency, etc.), which depends on a number of k continuous variables x (pressures, temperatures, quantity of raw materials) that can be modified within a specific domain contained by a lower and upper bound (lb and ub, respectively), as well as on $\mathbb{K}$ integer variables y (e.g., selection of units, technologies). The objective can be subject to a set of m_{eq} equality constraints (e.g., mass and energy balances, product quality) and to a set of m inequality constraints (e.g., equipment capacities, environmental and safety restrictions) (Fletcher, 1987).

The objective and constraints may be explicit, and this occurs in situations when using a simplified model of the process that can be expressed in algebraic equations (Caballero & Grossmann, 2008). This is typically encountered in the upper level decision making layers, (e.g., supply-chain management, planning) where "managemental-like" decisions are to be made. In these situations, a model of the enterprise, in which the units and production processes are roughly represent by linear or slightly nonlinear shortcut models (e.g., fabrication of 1 kg of product requires 2 kg of raw materials and 3 liters of fuel/energy) (Medina-González, et al., 2020). Here the evaluation of the model may be cheap, but a large number of variables and constraints exist (e.g., to express the mass and energy balances of the system/process in the form of explicit optimization constraints).

On the other hand, the objective and constraints may be also implicit, and this is usually encountered in the lower level decision making layers, (e.g., process operation and control) where complex and highly nonlinear simulation model are used to describe the process based on first principles (thermodynamics, reaction kinetics, heat and mass transfer, etc.) (Caballero & Grossmann, 2008). Here the number of variables and constraints are reduced but the evaluation of the models is expensive and consumes large computational time (Forrester, et al., 2008).

The algorithms required to solve an optimization problem are dependent on the problem type, which is identified according to many factors, which include, mainly, the type of relation between the objective and/or constraints with the decision variables (linear, quadratic,

nonlinear), the nature of the decision variable them-selfies (continuous, integer) and the availability of the derivative information (Fletcher, 1987; Biegler, 2010). The following subsections overview the main types of optimization problems considered in this Thesis and the algorithms used for their solutions.

2.5.1 Linear programming

A Linear Programming (LP) problem is considered when all the decision variables are strictly continuous and the involved objective function and all constraints are described by linear relationships (Dantzig & Thapa, 1997). The general formulation of a LP problem is represented as follows:

$$\left.\begin{array}{c} \min f = c^t x \\ S.T.: \ A\,x \leq b, \\ A_{eq}\,x = b_{eq}, \\ x \in R^k, \ \ lb \leq x \leq ub, \end{array}\right\} \tag{2.22}$$

Where c is a k-dimensional vector of cost coefficients, A is a $m \times k$ coefficients matrix of m inequality constraint, A_{eq} is a $m_{eq} \times k$ coefficients matrix of m_{eq} equality constraint, b is a m-dimensional vector of right-hand-side coefficients of the m inequality constraint, b_{eq} is a m_{eq}-dimensional vector of right-hand-side coefficients of the m_{eq} equality constraint, and lb, ub are k-dimensional vectors of lower and upper bounds for the decision variables, respectively.

The linear behavior of the functions (objectives and constraints) associated to this problem generates a feasible solution space defined by the intersections of a set of hyperplanes, hence, the optimal solution is at one of the vertices of the feasible polytope (Fletcher, 1987). This, enormously, facilitates its solution and multiple searching methods have been developed including, but not limited to, the simplex method and the Interior-Point Method (IPM) (Dantzig & Thapa, 1997). Both methods consist of an iterative searching approach that stops once the vertex with the best possible value is found. The main difference is that the simplex method moves from one vertex to the next through the boundaries of the feasible region, while the interior-point method searches inside the feasible space and never touches the boundaries before finding the optimal value (Medina-González, 2019).

In this Thesis the solver "*linprog*", included in the Matlab optimization toolbox, is used based on a dual simplex algorithm (Matlab, 2018).

2.5.2 Nonlinear programming

In cases when all the decision variables are continuous and at least one of the functions (objective and constraints) is nonlinear, the problem is considered as NonLinear Programing (NLP), for which the general formulation is as follows (Biegler, 2010):

$$\left.\begin{array}{c} \min f(x) \\ S.\,\mathrm{T.}:\ \ h_i(x) = 0, \qquad i = 1, \dots, m_{eq}, \\ g_i(x) \le 0, \quad i = 1, \dots, m, \\ x \in R^k, \ \ lb \le x \le ub \end{array}\right\} \tag{2.23}$$

For differentiable objective function and constraints, a local optimum can be defined by the optimality conditions known as the Kuhn-Tucker conditions (Biegler, 2010). There are many algorithms capable of solving NLP optimization problems, including Generalized Reduced Gradient algorithm, sequential quadratic programming (SQP) (Fletcher, 1987) and IPM (Wright, 1996). The SQP is the mostly used and effective nonlinear optimization algorithm (Medina-González, 2019), which can be derived by applying Newton's method to the Kuhn-Tucker conditions. It converts the non-linear optimization problem into a quadratic programming problem with linear constraints (Fletcher, 1987).

Solving NLP problems is challenging due to the possibility of the presence of non-convex feasible zones and multiple local optimal solutions (Biegler, 2010). In this Thesis, the solver "*fmincon*", included in the Matlab optimization toolbox, is used based on either SQP or IPM (Matlab, 2018).

2.5.3 Quadratic programming

A special case of NLP problems when a quadratic objective function is subjected to linear constraints is classified as Quadratic Programming (QP) problem (Fletcher, 1987). It can be stated in a general form as follows:

$$\left.\begin{array}{c} \min f = \dfrac{1}{2}\, x' Q x + c^t x \\ S.\,\mathrm{T.}:\ \ A\, x \le b, \\ A_{eq}\, x = b_{eq}, \\ x \in R^k, \ \ lb \le x \le ub \end{array}\right\} \tag{2.24}$$

Where Q is a $k \times k$ positive definite matrix. The most common methods used to solve QP problems include IPM and augmented Lagrangian (Papageorgiou & Fraga, 2007). In this Thesis, the solver "*quadprog*", included in the Matlab optimization toolbox, is used based on an interior-point-convex algorithm (Matlab, 2018).

2.5.4 Mixed-Integer programming

Real-life optimization problems, commonly, imply the consideration of binary variables (e.g., on/off settings or logical decisions), which turns LP or NLP problems into a Mixed-Integer LP (MILP) or Mixed-Integer NLP (MINLP) problems, respectively (Fletcher, 1987). Mixed-integer problems can be stated in a general form as in Eq.(2.21).

MILP is one of the most extensively explored formulations due to its flexibility and extensive modeling capability (Medina-González, 2019). The methods to solve MILP problems are based on enumerative algorithms that discard the less efficient alternatives. Among the algorithms used to solve MILP problems Branch and Bound can be highlighted, which is based on decomposing the original MILP problem into continuous LP sub-problems and solving them sequentially. The optimal solution is obtained by solving a subset of LP subproblems while searching within a decision tree of the discrete variables (Duran, 1986). On the other hand, MINLP problems are commonly solved through generalized benders decomposition and outer-approximation methods (Duran, 1986).

In this Thesis, the DICOPT solver integrated in the GAMS mathematical optimization environment is used to solve MINLP problems. The DICOPT solver combines solvers of the sub-problems nonlinear programming using CONOPT and Mixed Integer programming using CPLEX (Medina-González, 2019).

2.5.5 Derivative-free optimization

In many real-world optimization problems, the derivative information is unavailable, unreliable, or impractical to obtain, for example, when the objective function is noisy, non-smooth and/or undifferentiable (Rios & Sahinidis, 2013). This hinders the applications of most derivative-based optimization methods (like the previously overviewed techniques) (Forrester, et al., 2008). In these cases, derivative-free optimization algorithms represent a powerful alternative, where the values of the objective function are used to direct the optimization search (Rios & Sahinidis, 2013). Many derivative-free optimization algorithms are available in the literature, and most of them are population-based (Conn, et al., 2009). Evolutionary algorithms (e.g., genetic, differential evolution algorithms) and swarm-intelligence-based algorithms (e.g., particle-swarm, artificial bee-colony, bird flocking algorithms) are just examples of the most common derivative-free optimization algorithms (Slowik, 2020).

Particularly, the Genetic Algorithm (GA) has a large number of successful applications to real-world optimization problems over a wide range of engineering fields (Mitchell, 1996). The GA optimization search simulates the biological process, in which successive generations/populations of candidate solutions are trying to adapt to their environment through

genetic inheritance from parents to children and through survival of the fittest (Schmitt, 2001). The stochastic search of the GA starts with a randomly generated initial population of individuals (chromosomes), each one of them is made by a string of the decision variables (genes), being each gene encodes one of the decision variables values. The individuals of the current population are ranked according to a fitness function based on the objective function value (Mitchell, 1996). A new (i.e., next) population of individuals is then generated by applying the genetic operators, which include the crossover, selection and mutation. The selection operator is applied to select and copy surviving members (parents) from the current population to the new one. Individuals with higher fitness function values have a higher chance to be chosen than those with lower fitness function values. The crossover operator aims to interchange the information and genes between chromosomes via combining two or more parents to reproduce new children, then, one of these children may, hopefully, collect all good features that exist in his parents (Mitchell, 1996). The mutation operator is applied to alter one or more genes of a probabilistically selected chromosome. This Thesis considered the GA algorithm and its implementation in the Matlab optimization toolbox based on the solver "*ga*" (Matlab, 2018).

Chapter 3: SBO of Chemical Process Operations

This Chapter presents a methodology for operation optimization of complex nonlinear chemical processes, in which the objective function and/or the constraints are represented by black box functions. The proposed approach consists of replacing the complex, nonlinear, black box model of the processes built based on first principles with kriging metamodels. A sequential sampling strategy, based on the expected improvement (for unconstrained optimization problems) or the constrained expected improvement (for constrained optimization problems) techniques, is used to explore the search space of the decision variables and to adapt, accordingly, the metamodels, so as to reach a global solution for problem. The methodology has been tested and compared with classical optimization procedures based on sequential quadratic programming. Both have been applied to three mathematical examples and to two case studies of optimization of chemical process operation. The results show that the proposed methodology provides accurate solutions, significantly reduces the required computational time and guarantees high computational reliability, which make it very effective for hour-to-hour or day-to-day operation optimization of complex processes.

3.1 INTRODUCTION

In the decision-making hierarchy of chemical plants management, plant-wide optimization, or process operation optimization, is a principle layer that receives in inputs, the outcomes and decisions coming from the above layers (i.e., planning and scheduling) (Hauptman & Jovan, 2004; Roffel & Betlem, 2004). These inputs, basically, include forecasts of prices and demands, production rate targets over long time periods (weeks/days), assignment of resources to activities, (raw material allocation, tasks to units allocation, maintenance interventions, staffing), sequencing of activities and determination of starting and ending times for the execution over short periods of time (Muller, et al., 2017; Marchetti, et al., 2014).

The goal of process operations optimization is to obtain the optimal values of the process variables (temperatures, pressures, compositions, flow rates, etc.) at which the plant and its units must operate to maximize certain performance criteria (e.g., efficiency, profit, operational cost), while satisfying all the constraints (equipment capacities, environmental restrictions, etc.) and requirements (product quality, production yields, safety, etc.) (Vaccari & Pannocchia, 2017; Biegler, 2010). This is achieved by solving, in real time, an optimization

problem, which embeds a detailed and rigorous steady-state model of the process (Shao, et al., 2019). Depending on the model characteristics, such as its structure, transparency (e.g., white, gray, black-box), availability of derivative information, and on the formulations of the objective(s) and constraints of the optimization problem, different algorithms can be used, e.g., derivative-free algorithms (e.g., Genetic Algorithm), where the explicit values of the objective(s) function are used to direct the optimization search, or derivative-based algorithms (e.g., interior point algorithms), where the optimization search is directed based on the derivatives of the objective(s) with respect to the decision variables (Salback, 2004; Caballero & Grossmann, 2008).

In order to maximize the credibility of the optimal set-points when they are implemented in the plant, the process-model mismatch is minimized by updating the process model parameters (e.g., heat transfer coefficients, catalyst activities, distillation column tray efficiencies) before performing the operation optimization, relying on reconciled estimates of the measured steady-state measurements of the plant variables (Fadda, 2017; Biegler, 2010). The reconciled estimates are often obtained by applying data reconciliation and gross error detection techniques to the real data collected by the sensors in order to reduce the effect of random errors and sensor faults (bias, drifting, miss-calibration, total failure, etc.), respectively (Chaudhary, 2009). In other technologies, the model parameters are estimated or updated within the data reconciliation and gross error detection task (Chaudhary, 2009). Finally, after the optimal set-points are obtained, they are delivered to the supervisory control layer and, subsequently, to the distributed control layer in order to maintain the plant functioning at these reference points. As the plant operates, this computational scheme works periodically; usually in a frequency of hour(s) depending on the manufacturing system nature (Seborg, et al., 2016).

Recently, there is growing interest to use complex and high-fidelity mathematical models of the process in the operation optimization task. However, the development of such analytical models for many chemical, petrochemical and pharmaceutical processes is a challenging task due to the required deep knowledge, cost, efforts and time (Quirante, et al., 2018). As a result, specialized simulation software tools have been developed to model and simulate such complex processes most of them appear in black box modular style, e.g., Aspen and gPROMs (Caballero & Grossmann, 2008).

Although these First Principle Models (FPMs) are able to capture more detailed features and sophisticated characteristics of the process and, consequently, provide more accurate estimation of its behavior, they show many practical drawbacks and challenging characteristics (Flemming, et al., 2007; Norbert, et al., 2017; Quirante, et al., 2018), such as:

i) high nonlinearity and complexity, due to the sophisticated phenomena typically involved (thermodynamics, reactions kinetics, heat and mass transfer, etc.) and to the large number of equations contained in the analytical or FPMs. For example, a full-scale model of a refinery could contain more than a million of equations (Henao & Maravelias, 2011),

ii) complex model architectures, since they appear to the user in modular black-box style (e.g., Aspen Plus) involving intricate connections and recycles among the different units and, also, with no access to the embedded first principle equations (Caballero & Grossmann, 2008),

iii) large computational cost required for the model simulation due to the complexity of the utilized numerical solution procedures –e.g., iterative schemes and/or integration techniques- (Garud, et al., 2017).

iv) noisy calculations introduced by these simulators (e.g., caused by the termination criteria), which hinder the efficient use of derivative-based optimizers due to the bad estimates of the derivatives and, consequently, the poor optimization results (Quirante, et al., 2018).

These challenging characteristics inherent to the FPMs of chemical processes represent an obstacle to their use for the operation optimization, especially for large scale and/or fast dynamic processes (Kelly & Zyngier, 2017). For example, the optimization of a full-scale petrochemical plant (crude oil and gas treatment facilities, refineries, etc.) based on its FPM model requires several hours to obtain the optimal solution, and in many cases, the optimization process may fail to converge (Salback, 2004; Kajero, et al., 2017).

To overcome the above drawbacks and cope with the above challenges, Surrogate Based Optimization (SBO) approaches have been proposed and have received significant attention in the chemical process industry area (Quirante, et al., 2018). The basic idea of SBO is to use the original complex FPM for generating input-output data points by "Computer Experiments", and use them to develop accurate, but simple and fast-running, data-driven models ("metamodels" or "surrogate models"). Then, these data-driven models are used in replacement of the complex FPM in the addressed optimization problem (Ochoa-Estopier & Jobson, 2015). In this context, two types of machine learning or surrogate models have been common choices, which are Artificial Neural Networks (ANNs) and kriging (Kajero, et al., 2017). ANNs offer universal and powerful approximation capabilities due to their flexible structure (of neurons) that can be modified to capture complex nonlinear behaviors. On the other hand, kriging also provides high prediction accuracy with relatively small number of training data points, besides an outstanding characteristic represented in its estimated variance,

which represents the uncertainty about the kriging model prediction. Nevertheless, in the SBO literature (Jones, et al., 1998; Jones, 2001; Zuhal, et al., 2019), it has been demonstrated that non-interpolating surrogate models (i.e., regression models, such as ANN) are unreliable in optimization, because they do not appropriately capture the shape of the function to be approximated, and it is usually better to use surfaces that interpolate the data with linear combinations of basis functions (e.g., kriging).

In the chemical process engineering area, two main classes of SBO approaches can be identified. The first is based on the development of global surrogate models approximating the entire modular simulator or flow-sheet of the process (the FPM). In more details, the input and output variables of these global surrogate models are selected over the entire process flow-sheet as the variables of interest for the optimization problem formulation (i.e., variables representing the optimization decisions (input) and variables constituting the objectives and constraints (outputs)) (Palmer & Realef, 2002; Kempf, et al., 2012; Chia, et al., 2012; Ochoa-Estopier & Jobson, 2015; Ochoa-Estopier, et al., 2018; Davis & Ierapetritou, 2007). The second class of approaches is based on partitioning the simulation model into different units or subgroups of units, for each of which a dedicated surrogate model is developed; then, the different surrogates are aggregated to constitute the final approximate model of the process, based on which different optimization schemes can be designated (Salback, 2004; Henao & Maravelias, 2011; Quirante & Caballero, 2016; Quirante, et al., 2018; Caballero & Grossmann, 2008).

Regarding the first class, Slaback (2004) proposed a method for SBO of a refinery operation, where each unit-model based on first principles is substituted by a unit-model based on ANN. The refinery large-scale approximate model is obtained by coordinating/linking the ANN-based unit-models and, then, an iterative optimization procedure is performed: once the optimum operating conditions are determined, it is simulated by the FPM and then used to update the ANN models, and so on until no enhancement in two successive optimal solutions (i.e., iterations). Henao and Maravelias (2011) proposed a similar method, but for superstructure optimization, in which each unit in the modular simulator is replaced by an ANN model, then the superstructure is composed by aggregating all the ANN-based unit models. Then, an adaptive Mixed-Integer SBO optimization scheme is performed: in each iteration, the input domains of the surrogate models are updated according to the obtained optimal solution value and its feasibility (and, also, according to the overlapping between the output domain of each surrogate model and the input domain of the subsequent connected surrogate), which implies the generation of completely new training datasets and the refitting of all the surrogate models.

Caballero & Grossmann (2008), Quirante and Caballero (2016) and Quirante et al. (2018) developed hybrid SBO methods, in which kriging models replace only those units in the process modular simulator that introduce numerical noise and/or requiring large CPU time for their convergence (e.g., distillation columns, reactors), while other noiseless and computationally affordable units are maintained (e.g., splitters, pumps). The optimization is carried out considering the hybrid model (units based on the modular process simulator and others based on kriging models): a trust region-like search mechanism is employed, which contracts and/or moves a trust region around the optimal solution obtained in each iteration, which implies contracting or shifting the surrogate models input domains and, consequently, requires generating new training datasets and refitting the surrogate models. Their methods have been applied to different applications, including the operation optimization of sour water stripping plant and the design optimization of distillation systems.

The advantages of this class include: a) the capability of handling large-scale systems by splitting them to small units/sections (i.e., surrogate models) and b) the possibility to combine both accurate and fast FPM models of some units or sections in the plant with surrogate models of other problematic units or sections. Whereas their limitations are that it does not exploit the potentials capabilities provided by the kriging variance in order to account for the surrogate models uncertainty during the optimization search. Also, it iteratively discards the previous training datasets and generates new sets for fitting new surrogate models, which can be computationally prohibitive in an online environment.

Considering the second class of SBO approaches, the early work of Palmer & Realef (2002) has presented guidelines for the optimal development of kriging and polynomial regression surrogate models from a complex steady-state simulator of ammonia synthesis plant. Davis & Ierapetritou (Davis & Ierapetritou, 2007) developed a SBO method based on fitting an initial kriging model that represents a global picture of the objective behavior over the entire feasible search space in order to identify promising local regions at which other local regression surrogate models (with completely new training datasets) are fitted to refine the search. The global optimum is, then, selected as the best solution obtained among the identified set of local optima. Ochoa-Estopier and Jobson (2015) and Ochoa-Estopier, et al. (2018) developed a SBO method for operation optimization of crude oil distillation systems, in which ANN models took the place of the complex FPM. The nonlinear optimization problem is solved, based on the ANNs models by a simulated annealing algorithm. Chia et al. (2012) have proposed a multi-objective SBO method for optimizing the operation of a batch reactor, in which two objectives are considered; the minimization of the kriging prediction and the maximization of the kriging estimated error. The optimal solution is, iteratively added to the original set of the training data, and the surrogate model is refitted again, and so on. Few works

has been proposed based on the EGO algorithm "Efficient Global Optimization" (Jones, et al., 1998; Zuhal, et al., 2019), in which a global surrogate model representing the objective is initially fitted with few training points, and it is explored by a probabilistic search mechanism ("expected improvement") that considers both the kriging prediction and the uncertainty about this prediction. In each iteration, the surrogate model is updated with the obtained optimal solution (sampled from the FPM) and then refitted. Kempf et al. (2012) used an EGO-based method for the unconstrained design optimization of a nuclear reactor. The advantages of the second class are:

i) it considers the surrogate model prediction uncertainty (i.e., error), which is an essential need for any reliable SBO method (Jones, et al., 1998; Jones, 2001; Zuhal, et al., 2019). Since it has been proven that exploring the surrogate model with an arbitrary optimizer can fail even to find local optima, because the surrogate model prediction uncertainty is not considered by the traditional optimizers (Zhang, et al., 2018; Zuhal, et al., 2019).

ii) it adds efficient global exploration capabilities to the search mechanism by not only directing it to the minimum value of the predictor (i.e., the surrogate model representing the objective), but also to its maximum estimated error (Zuhal, et al., 2019),

iii) the eventually obtained global surrogate models of the entire process/plant can be used for different analysis (Kempf, et al., 2012),

iv) relatively few simulations run of the original FPM are required for updating the surrogate models during the optimization search (Forrester & Keane, 2009), which makes this SBO class more suitable for online application

Nevertheless, this class of SBO approaches shows some drawbacks as the difficulty to construct global surrogate models that accurately capture the behavior of large-scale processes or plants, and, more importantly, the difficulty of handling constraints.

This Chapter considers the second class of approaches for SBO of chemical processes and extends its capabilities by developing a methodology for the constrained SBO of complex nonlinear chemical processes. The proposed methodology consists of replacing the complex nonlinear black box model based on first principles with a set of kriging surrogate models representing the objective function and the constraints. A sequential sampling strategy, based on the constrained expected improvement techniques, is used to explore the search space of the decision variables (i.e., the surrogate models input) and to update the set of surrogate

models, so as to reach a global solution for the problem. The methodology has been compared with classical optimization procedures. Both have been applied to three mathematical examples and to two case studies of optimization of chemical process operation.

3.2 PROPOSED FRAMEWORK

The proposed framework consists of three main steps; 1) initial Design Of Computer Experiments (DOCE), 2) kriging models construction and validation, 3) surrogate-based optimization.

3.2.1 Initial DOCE

In this step, the original complex FPM of the process is explored in order to identify the output variables of interest $y \in R^M$ (variables establishing the objective and constraints), the input variables $x \in R^k$ (the degrees of freedom of the optimization) and their bounds or domain (surrogate models domain). Then, over the specified domain, a certain set of input values combinations (sample points) is selected, which is called "sampling plan" $[X]_{n\times k}$, where n is the number of sample points, and k is the number of input variables. The design of a sampling plan includes two issues: first, specifying a reasonable number of sample points n, and, second, designing the locations or the distribution of these data points over the surrogate model input domain. Many DOCE techniques have been developed such as Latin hypercube sampling (LHS) (Garud, et al., 2017), Space-filling Latin Hypercube Sampling (SLHS) (Forrester, et al., 2008), low discrepancy sequences as the Hammersley technique (Ibrahim, et al., 2019) and sequential or adaptive sampling (Kajero, et al., 2017). Most of these DOCE techniques show both desired and limiting characteristics in terms of the uniformity of the generated sampling plan and the demanded computational cost (Ibrahim, et al., 2019). In this Chapter, the SLHS design is used, as it achieves high uniformity (Garud, et al., 2017). More details about DOCE techniques can be found in Section 2.1.

After designing an efficient sampling plan $[X]_{n\times k}$, a computer experiment or a simulation run is carried out using the complex FPM at each point of the sampling plan, so as to obtain the response or output variables values $[Y]_{n\times M}$. Where M is the number of output variables that includes the objective function and the $M-1$ constraints. Consequently, M is also the number of kriging models to be fitted (a surrogate model for the objective, and a surrogate model for each of the $M-1$ constraints). It is worth mentioning that the objective and the constraints do not need to be direct measures of the process variables, but they can be, also, combinations of the process variables, which finally lead to process performance indicators.

3.2.2 Kriging construction and validation

The generated input ($[X]_{n\times k}$) and output ($[Y]_{n\times M}$) data are used to train a number of M kriging models (one surrogate model for each one of the objective and constraints to be considered).

For one output variable, y, the Ordinary Kriging (OK) (Forrester & Keane, 2009) assumes the predictor $\hat{y}(x) = \mu_{ok} + \mathbb{Z}(x)$, where the constant term μ_{ok} represents the main trend of the system to be approximated, and $\mathbb{Z}(x)$ is a deviation/residual from that trend, which accounts for detailed complex behavior of the system that could not be captured via the main trend μ_{ok} (Jones, et al., 1998). The residual $\mathbb{Z}(x)$ is modeled as a stochastic Gaussian process with expected value $E(\mathbb{Z}(x)) = 0$, and a covariance between two residuals $cov(\mathbb{Z}(x_i), \mathbb{Z}(x_j))$ that only depends on their corresponding input values x_i, x_j. Thus, it can be calculated as: $cov(\mathbb{Z}(x), \mathbb{Z}(x_j)) = \sigma_{ok}^2 R(x_i, x_j)$, being σ_{ok}^2 the process variance, and $R(x_i, x_j)$ a correlation function, $R(x_i, x_j) = exp\left(-\sum_{\ell=1}^{k} \xi_\ell \left|x_{i,\ell-} x_{j,\ell-}\right|^{p_\ell}\right) + \delta_{ij}\,\lambda$, where, $\xi_\ell, l = 1, \dots, k$ are the model hyper-parameters, δ_{ij} is the Kronecker delta, p_ℓ are smoothing parameters and λ is a regularization constant that enables the kriging predictor to regress noisy data (Forrester & Keane, 2009).

In order to estimate the values of the parameters $[\mu_{ok}, \sigma_{ok}^2, \xi_\ell, p_{\mathcal{L}}, \lambda]$, the likelihood function of the observed data $[Y]_{n\times 1}$ is maximized. The kriging predictor (Eq.(3.1)) and its estimated error (Eq.(3.2)) are obtained by deriving the augmented likelihood function of both the original training data set and a new interpolating point (x^*, y^*). In both equations, $[r]_{n\times 1}$ is the vector of correlations between the new point to be predicted x^* and the original training data points and calculated as $R\,(x_i, x^*)$, and $[\mathbf{1}]_{n\times 1}$ is the identity vector (Jones, et al., 1998; Caballero & Grossmann, 2008).

$$\hat{y}(x^*) = \mu_{ok} + r^T R^{-1}(Y - \mathbf{1}\mu_{ok}) \tag{3.1}$$

$$\hat{s}^2(x^*) = \sigma_{ok}^2(1 + \lambda - r^T R^{-1} r + (1 - \mathbf{1}^T R^{-1} r)^{-1}/(\mathbf{1}^T R^{-1}\mathbf{1})) \tag{3.2}$$

In practice, this maximization of the concentrated log-likelihood function is computationally challenging because of the high effort associated to the repetitive calculation of the correlation matrix inverse $[R]_{n\times n}^{-1}$ during the optimization iterations, which quickly grows with the size of the training data set and/or the model input dimensionality see Section 2.2.1. Besides, the nature of the concentrated log-likelihood function itself is quite complicated because it is flat near the optimum (Fang, et al., 2005). More details about these computational

challenges and the numerical methods and optimization techniques to reduce these obstacles can be found in (Fang, et al., 2005; Forrester, et al., 2008).

After fitting the M kriging metamodels, they are validated to verify that they show a sufficient level of accuracy. Cross-validation techniques allow performing this task without any additional data generation rather than the original training set (Kohavi, 1995; Meckesheimer, et al., 2002). Several methods have been proposed, as the "K-fold cross-validation", "leave-p-out cross-validation", and "leave-One-Out Cross-validation" (LOOCV). A detailed justification of the characteristics of these and other cross-validation techniques can be found in (Kohavi, 1995; Meckesheimer, et al., 2002). In this Chapter, the LOOCV method is used based on computing the Root Mean Square Error (RMSE) by Eq.(3.3), where y_i is the real value of the left out point and $\hat{y}_i$ is its corresponding estimated value.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{3.3}$$

3.2.3 Optimization using surrogate models

Once the kriging surrogate models have been fitted and validated, they take the place of the original complex process model. Hence, Eq.(3.4) and Eq.(3.5) represent the nonlinear constrained optimization problem, in which one kriging model represents the objective, and $M-1$ kriging models represent the $M-1$ constraints. In this problem, the objective $Y_{obj}(x)$ and the constraints $Y_{const(m)}(x),\ m = 1,2,\dots,M-1$ are considered as normally distributed random variables with means equal to the kriging models predictions, $\hat{y}_{obj}(x)$, $\hat{y}_{const(m)}(x)$, and variances equal to the kriging models variances, $\hat{s}^2_{obj}(x)$, $\hat{s}^2_{const(m)}(x)$, where T_m is the constraint limit.

$$Min\ Y_{obj}(x)\,, \qquad Y_{obj}(x) \approx \mathcal{N}(\hat{y}_{obj}(x), \hat{s}^2_{obj}(x)) \tag{3.4}$$

$$\begin{gathered} S.T.: Y_{const(m)}(x) \leq T_m, \\ Y_{const(m)}(x) \approx \mathcal{N}\left(\hat{y}_{const(m)}(x), \hat{s}^2_{const(m)}(x)\right), \\ m = 1,..,M-1 \end{gathered} \tag{3.5}$$

In the literature, the Expected Improvement (EI) criterion (Jones, 2001; Zuhal, et al., 2019) is used for optimizing an unconstrained objective function represented by a kriging surrogate model (i.e., Eq. (3.4)) via sequential sampling. Assuming that the objective function is to be minimized and the current best value of this objective is f_{min}, hence, if a new point x^*

is to be explored, the current best solution is expected to get improvement by an amount $I\ (x^*) = max[0, f_{min} - Y_{obj}(x^*)]$. Hence, the likelihood of achieving this improvement is given by a normal density function. By integrating over this density function, the EI is obtained in Eq.(3.6), where ϕ is the normal cumulative distribution function and φ is the density function (Jones, et al., 1998; Regis, 2016). The approach works iteratively through sequential sampling. In each iteration, the EI criteria, Eq.(3.6), is maximized to find a potentially improved solution x^*, the original complex FPM is evaluated at this solution to obtain the real response y^*, the new input-output data point $[x^*, y^*]$ is added to the initial set of training data and, then, the surrogate model is refitted. The method has been widely tested and it has been proven that it usually converges to the global optimum (Zhang, et al., 2018; Forrester, et al., 2008).

$$E[I(x^*)] = \hat{s}^2_{obj}(x^*)\ [u\ \phi(u) + \varphi(u)], \quad u = \frac{f_{min} - \hat{y}_{obj}(x^*)}{\hat{s}^2_{obj}(x^*)} \tag{3.6}$$

However, the EI can manage only the optimization of the objective and its uncertainty (i.e., Eq.(3.4)). The existence of constraints and their uncertainties, in Eq.(3.5), requires the use of an additional technique to manage the feasibility of the search and the uncertainty about this feasibility, as well. A straightforward approach is to use the EI method coupled with a penalty function for the violation of the constraints. But this approach would neglect the uncertainty about the constraints (i.e., uncertainty about the feasibility), and could easily lead to a deceptive solution (Parr, et al., 2010; Parr, et al., 2012; Qian, et al., 2019). The kriging variance enables the use of an additional technique to account for the constraints uncertainty, which is the Probability of Improvement (PI) (Schonlau, et al., 1998; Jones, et al., 1998; Durantin, et al., 2016). This technique considers the expected value of a kriging surrogate model (that represents a constraint) at a certain untrained point as a random variable $Y_{const}(x)$, which is normally distributed with a mean equal to the kriging prediction at this point, $\hat{y}_{const(m)}(x)$, and a variance equal to the kriging variance, $\hat{s}^2_{const}(x)$. Assuming that the maximum acceptable value of the constraint is T (the constraints limit) and its current value is f_{const}, when exploring a new point x^*, the probability of improving the current value f_{const} beyond T (the probability of feasibility) is modeled as the probability that $Y_{const}(x^*) \leq T$ (Parr, et al., 2012; Durantin, et al., 2016). Assuming the random variable is normally distributed, this PI is given by Eq.(3.7). The PI is calculated for each one of the constraints surrogate models (Parr, et al., 2012), see Figure 3.1.

$$PI(x^*) = p(Y_{const}(x^*) \leq T\ ) = \phi\left[\frac{T - \hat{y}_{const}(x^*)}{\hat{s}^2_{const}(x^*)}\right] \tag{3.7}$$

$$CEI(x^*) = E[I(x^*) \cap PI(x^*)] = E[I(x^*)]\ p(Y_{const}(x^*) \leq T\ ) \tag{3.8}$$

$$CEI(x^*) = \log(E[I(x^*)])\ + \sum_{m=1}^{M-1} \log(PI_m(x^*)) \tag{3.9}$$

Combining the EI criterion and the PI criterion of each constraint, we obtain the Constrained Expected Improvement (CEI) criterion or method (Eq.(3.9)) (Parr, et al., 2012), which minimizes an objective function subjected to constraints, all of them represented by kriging surrogate models. The considered SBO procedure works iteratively: in each iteration, it finds the point x^* that maximizes the CEI criterion, evaluates the complex FPM of the process at x^* to obtain the corresponding output y^*, adds the point $[x^*, y^*]$ to the original training dataset and, then, refits the kriging surrogate models. The point that maximizes the CEI criterion is the point in the surrogate models domain that has minimum predicted value of the objective surrogate model, maximum prediction variances and highest probability of satisfying the constraints (probability of feasibility). So, the CEI method does not only conduct the search to well suited solutions to the proposed optimization problem, but also improves the surrogate models accuracy during the optimization search to reduce the uncertainties (Rehman & Langelaar, 2017). The most straightforward stopping criterion for CEI optimization method is the number of iterations specified by the modeler. The maximization of the CEI criterion is accomplished by a genetic algorithm, and the boundaries of the optimization problem decision variables are represented by the limits of the surrogate models input domain.

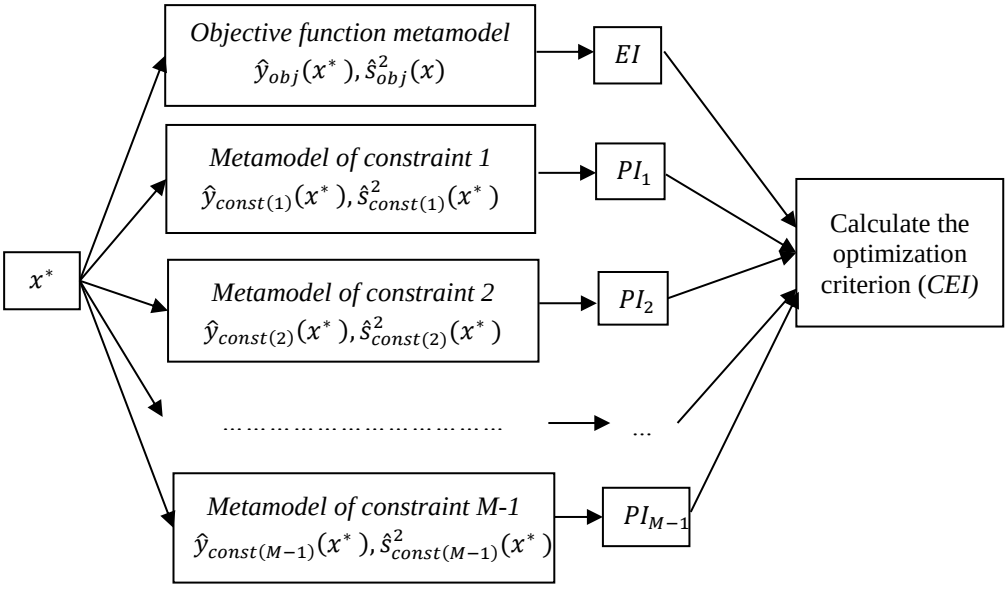


Figure 3.1. Calculation procedure of the CEI criterion for an interpolation point x^*.

3.3 METHODOLOGY STEPS

In order to reach to a robust and efficient optimization methodology, the previously described techniques and tools are coordinated through the following steps:

1. Explore the complex FPM to identify the M variables of interest (objective function and constraints). Then, identify the set of k independent variables (optimization/control variables) and their bounds (surrogate models domain),
2. Over the surrogate models domain, design a sampling plan with a certain number of sample points n. ($[X]_{n\times k}$),
3. Evaluate the FPM at these sampling points $[X]_{n\times k}$, and get the corresponding matrix of observations $[Y]_{n\times M}$.
4. Fit M kriging models by maximizing the likelihood of the observed data $[X]_{n\times k}$, $[Y]_{n\times 1}$.
5. Validate the kriging models (only in the first iteration).
6. Maximize the CEI criterion and get the optimal solution point x^*. Maximization of the CEI criterion is carried out using a genetic algorithm.
7. Evaluate the original FPM of the process at x^* and get y^*.
8. Add the new input-output point $[x^*, y^*]$ to the original matrix of observations $[X]$, $[Y\ ]$.
9. Stop if the stopping criterion is satisfied, otherwise return to step 4 and continue iterations.

As shown in the methodology steps and in Figure 3.2, the genetic algorithm plays an important role in the proposed SBO procedure: it is used for maximizing the CEI instead of classical derivative-based optimization techniques, which, if used, would face some obstacles resulting from the complex nature of the CEI criterion (i.e., entrapment in local optima) (Forrester, et al., 2008).

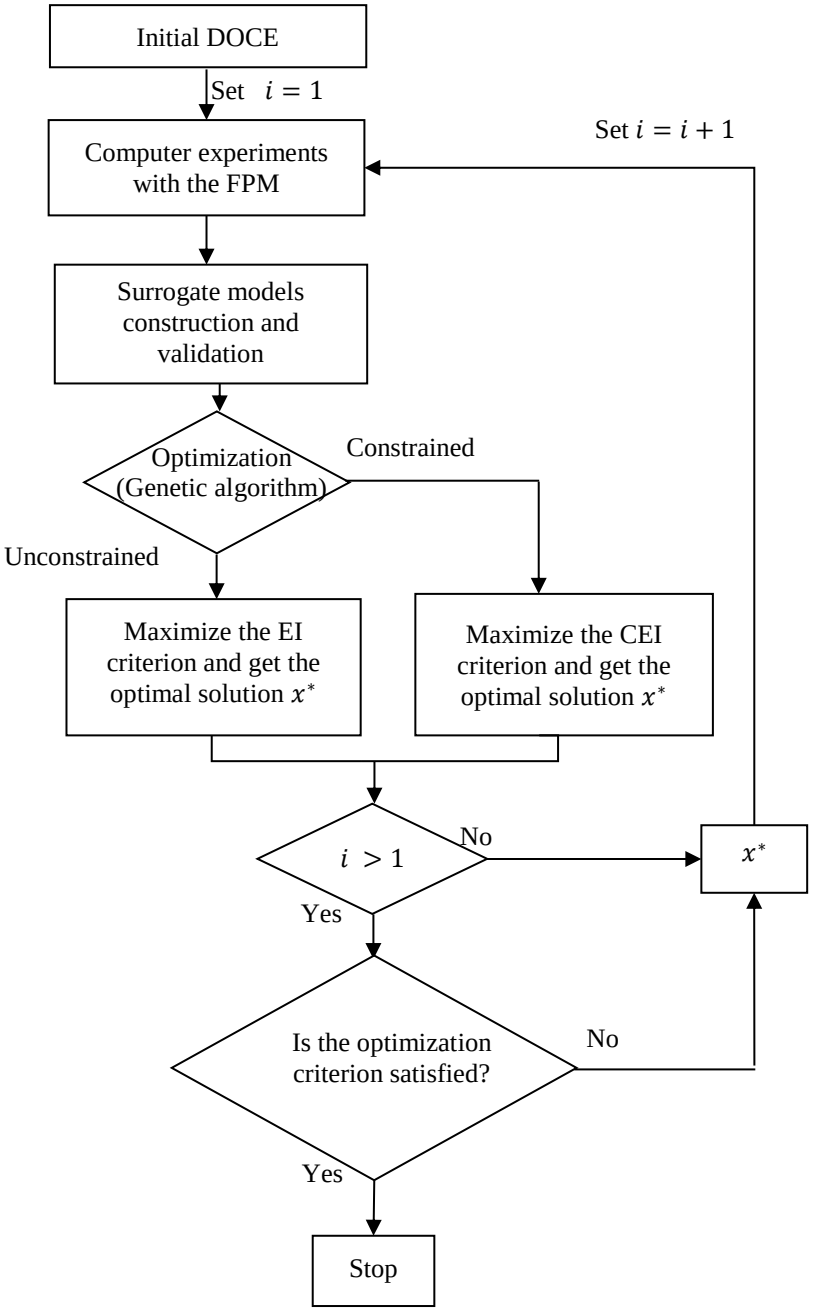


Figure 3.2. Proposed framework.

3.4 APPLICATIONS

3.4.1 Mathematical examples

Example (1): Minimize the Peaks function (Eq.(3.10)), subjected to the constraint in Eq.(3.11).

$$Min\ f_{Peaks} = 3(1-x_1)^2\ exp(-x_1^2-(x_2+1)^2)) - 10(\tfrac{x_1}{5} - x_1^3 - x_2^5)\ exp(x_1^2 - x_2^2) - \tfrac{1}{3} exp\ (((x_2+1)^2 - x_2^2)), \tag{3.10}$$

$$\text{S.T.:} \quad x_1^2 - 4x_2^2\ < 1.7\ , \quad -2 \le x_1, x_2 \le -2 \tag{3.11}$$

Example (2): Minimize the Branin function (Eq.(3.12)), subjected to the constraint in Eq.(3.13).

$$Min\ f_{Branin} = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10 \tag{3.12}$$

$$\text{S.T.:} \quad f_{Peaks}(x_1, x_2) < -2, \quad -5 \le x_1 \le 10\ ,\ 0 \le x_2 \le 15 \tag{3.13}$$

Example (3): Minimize the Six-hump Camel-back function (Eq.(3.14)), subjected to a constraint in Eq.(3.15).

$$f_{Camel} = \left(4 - 2.1\ x_1^2 + \tfrac{x_1^4}{3}\right) x_1^2 + x_1x_2 + (-4 + 4\ x_2^2)\ x_2^2, \quad -2 \le x_1 \le 2, -1 \le x_2 \le 1 \tag{3.14}$$

$$\text{S.T.:} \quad f_{Gomes}(x_1, x_2) > 3$$

$$f_{Gomes} = \left(4 - 2.1\ x_1^2 + \tfrac{x_1^4}{3}\right) x_1^2 + x_1x_2 + (-4 + 4\ x_2^2)\ x_2^2 + 3\sin(6(1-x_2)), \quad 0 \le x_1, x_2 \le -1 \tag{3.15}$$

Peaks, Branin, Six-hump Camel Back, and Gomez functions are well-known mathematical examples for global nonlinear optimization, because of their multimodality and high nonlinearity (Parr, et al., 2010; Durantin, et al., 2016; Qian, et al., 2019; Forrester & Keane, 2009). In this Chapter, scaled versions of those functions between [0 ,1] are used. In each example, a SLHS design technique is used to generate an initial sampling plan with 19 sample points to fit the kriging models.

The proposed methodology has been applied to each example with two different values of the stopping criterion (no. of iterations) to assess the algorithm abilities. The methodology is also compared to a classical Sequential Quadratic Programming (SQP) optimizer. In each of the three examples, the objective is subjected to only one constraint, so the methodology fits two kriging surrogate models (one for the objective and the other for the constraint).

The "scaled" results of the examples are summarized in Table 3.1, Table 3.2 and

Table *3.3* and visualized in Figure 3.3. The Tables show how the methodology provides accurate results with a smaller number of function evaluations than the SQP traditional method. Although, since the function evaluation was almost costless, in these cases the higher number of evaluations required implied less effort than the fitting process overhead, so the proposed procedure required higher computational effort than the use of traditional methods over the mathematical "real" model.

Moreover, the results also indicate the capability of the methodology to search over the whole domain of the problem, even moving among separated feasible regions (Figure 3.3-(b,c)), which facilitates not only the identification of the global optimum of the problem, but also to get information about alternative sub-optimal solutions. This fact is of essential importance in real engineering problems, where it is not always easy to fit all the information about the problem in the corresponding mathematical terms (objective function and/or constraints) and local optima may represent alternative solutions. So, human practical know-how may be supported through this additional information to make a final decision. Additionally, this characteristic eliminates the need to repeat the optimization departing from different initial points, which is a drawback of other local optimization methods.

Table 3.1. Results obtained for the constrained optimization of example (1).

	FPM + SQP	***Kriging+ CEI***	
***Solution* [*objective* (x_1, x_2)]**	-2.977 (0.183, 0.539)*	-2.886(0.200, 0.564)	-2.95(0.192, 0.553)
No. of kriging models	0	2	2
***No. of func. eval.* (*kriging*)**	0	19	19
***No. of func. eval.* (*optimization*)**	75+	4	8
Initial DOCE (sec)	0	10.632	10.632
Computer experiment time (sec)	0	0.008	0.008
Optimization time (sec)	0.131+	30.762	68.122
Computational reliability	50 %	100 %	100 %

+ The function evaluations and the optimization time in the SQP case are average values, as the SQP of the real model was carried out 50 times with different randomly selected initial solutions (applicable to the three examples).

**** Only 50 % of the optimization trails using SQP found the global solution. 42% of the optimization trails converged to trivial solution 0.1297 (0.607, 0.3461), and 8% to -0.0649 (0.57, 0.58).***

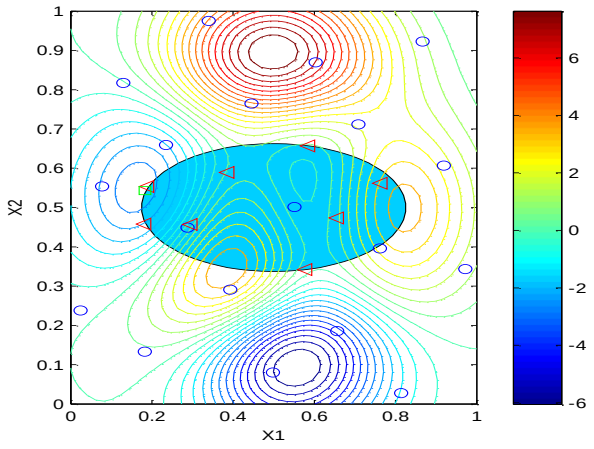
Table 3.2. Results obtained for the constrained optimization of example (2).

	FPM + SQP	***Kriging+ CEI***	
***Solution* [*objective* (x_1, x_2)]**	0.397 (0.542, 0.152)*	0.434(0.538, 0.163)	0.398 (0.543, 0.152)
No. of kriging models	0	2	2
No. of func. eval. (kriging)	0	19	19
No. of func. eval. (optimization)	80^{+}	4	8
Initial DOCE (sec)	0	10.490	10.490
Computer experiment time (sec)	0	0.0145	0.0145
Optimization time (sec)	0.214^{+}	25.819	50.531
Computational reliability	30 %	100 %	100 %
**** Only 30 % of the optimization trails with SQP found the global solution. 16% of the optimization trails found a local solution 2.47 (0.156, 0.677); 8% of the optimization trails found a local solution 18.60 (0.782, 0.082); 46% of the optimization trails converged to infeasible solutions***			

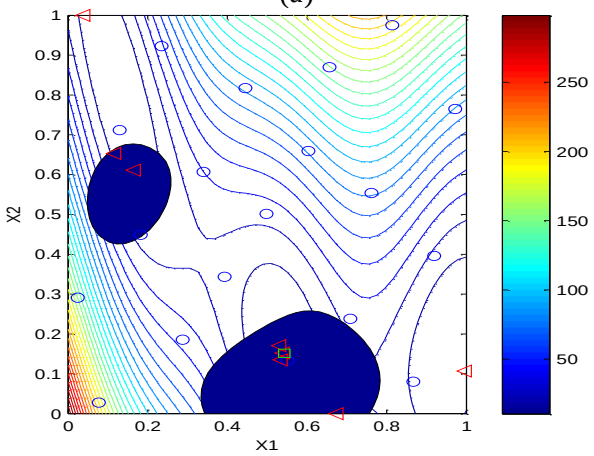
Table 3.3. Results obtained for the constrained optimization of example (3).

	FPM + SQP	***Kriging+ CEI***	
***Solution* [*objective* (x_1, x_2)]**	-0.975(0.447, 0.864) *	-0.667(0.397,0.877)	-0.913 (0.439,0.831)
No. of kriging models	0	2	2
No. of func. eval. (kriging)	0	19	19
No. of func. eval. (optimization)	35^{+}	4	8
Initial DOCE (sec)	0	10.47	10.47
Computer experiment time (sec)	0	0.017	0.017
Optimization time (sec)	0.577^{+}	34.42	63.91
Computational reliability	20 %	100 %	100 %
**** Only 20% of the optimization trails with SQP found the global solution. 40% of the optimization trails converged to infeasible solutions, and 40% local and trivial solutions.***			

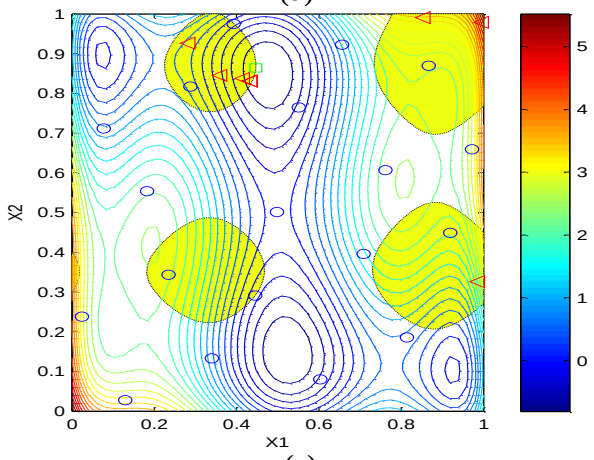
Finally, it is worth noting that the algorithm not only optimizes during iterations, but also improves the accuracy of the metamodels. So, it may explore infeasible regions - see Figure 3.3-(b) - if these regions show high prediction uncertainty; then, it smartly returns to search the feasible regions. Since, an area showing high prediction uncertainties will maximize the merit value (CEI in Eq.(3.9)) in spite of its eventual infeasibility, and will force the optimizer to explore it and add it to the set of sample points to reduce the uncertainties of surrogate models; in next iterations, the effect of the uncertainties on the merit value will be reduced, and the effect of the areas or points that have high probabilities of feasibility will dominate the merit and will force the optimizer to return to the feasible area. In this sense, the methodology is insensitive to the initial solution, simply because it does not need an initial solution to start the optimization. On the contrary, when optimizing examples 1, 2 and 3 with classical SQP optimizers, more than 50% of the optimization trials fail to find even feasible solutions, and a lot of effort was dedicated to find a feasible initial solution. Additionally, the methodology offers adjustable stopping criterion which allows the modeler to adjust the tradeoff between the required accuracy of the optimization results and the optimization time.



(*a*)



(*b*)



(*c*)

Figure 3.3. Constrained optimizations using the proposed SBO methodology; the colored areas are the feasible regions, ○ – original samples set, Δ – optimization samples (iterations), □- exact optimal solution.

3.4.2 Gas turbine case study

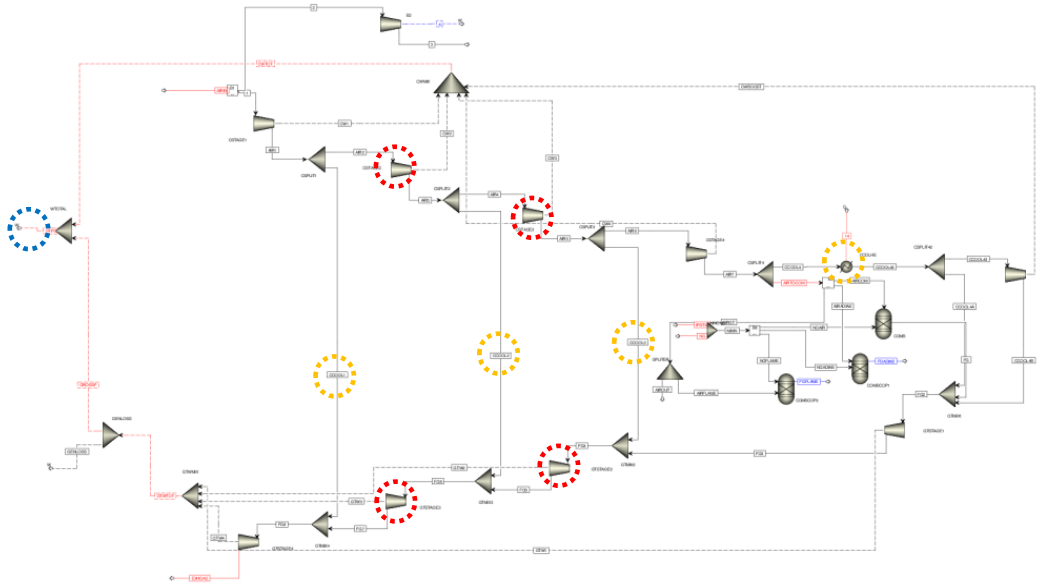


Figure 3.4. AspenPlus model of the Gas turbine case study.

The methodology has been also applied to the optimization of the operating parameters of a Gas Turbine (GT) (Bojarski, et al., 2010). The GT cycle is composed of a system of compressor-combustion chamber-turbine that uses natural gas. A saturation column, before the clean gas combustion, saturates this stream with vapor and nitrogen. The GT air cooling has been modeled by taking into consideration four stages GT, thus the air compressor has been modeled as a four steps process, as well. The combustion chamber is considered to operate at 15 bars, while air is fed to the compressor at atmospheric pressure. The combustion chamber is modeled as a Gibbs reactor (Bojarski, et al., 2010). Each corresponding stage could have the same pressure loss or gain ratio, but in this case the intermediate pressures have been left unspecified and are subject of optimization. All the model units of the turbines and compressors are taken from the AspenPlus model library and are considered to be isentropic. AspenPlus is an engineering software package, used for modeling and simulation of complex chemical/industrial processes, and takes the form of black box models.

The objective function pursued is the maximization of net power (NP) obtained (highlighted by blue circle in Figure 3.4), which is equal to the total work produced by the turbines minus the work consumed by the intermediate compressors. The optimization variables are:

- the cooling air split fractions $SF1, SF2, SF3, SF4$ (highlighted by yellow circles in Figure 3.4) that can vary within the range $[0.0 - 0.1]$, and

- the intermediate stage pressures of the compressors $PC2, PC3$ and of the turbines $PT2, PT3$ (highlighted by red circles in Figure 3.4), which can vary within the ranges $[2.5 - 7.5]$, $[8.0 - 17]$, $[2.0 - 4.0]$ and $[0.5 - 2.0]$ *bar*, respectively**.**

Table 3.4. Optimization results of the Gas Turbine case study.

	Case 1 (Aspen model +SQP)	**Case 2 (Kriging+ CEI)**
$\boldsymbol{SF1}$	0.080	0.00
$\boldsymbol{SF2}$	0.100	0.00
$\boldsymbol{SF3}$	0.100	0.00
$\boldsymbol{SF4}$	0.001	0.11
$\boldsymbol{PC2}$	4.0	8.12
$\boldsymbol{PC3}$	10	14.9
$\boldsymbol{PT2}$	3.2	2.0
$\boldsymbol{PT4}$	1.5	1.2
$\boldsymbol{NP}$	239610	239571
No. of kriging models	0	1
No. Func. eval. (kriging)	0	60
No. Func. eval. (optimization)	293	7
Initial DOCE time (sec)	0	71
Experiment time (sec)	0	46
Optimization time (sec)	234.4	58
Total no. of function eval.	**293**	**67**
Total time (sec)	**334.4**	**175**
Computational reliability	**70%**	**100%**

The case study has been solved with two different techniques summarized in Table 3.4:

- in case (1), the optimization has been achieved using the original complex FPM of the process (Aspen Plus) and the "*fmincon*" solver, based on a SQP algorithm, integrated in the Matlab optimization toolbox, and

- in case (2), the proposed SBO framework has been applied. Since, the addressed optimization problem is unconstrained, only one surrogate model is fitted, which approximates the objective (NP) as a function of the decision variables $[SF1, SF2, SF3, SF4, C2, PC3, PT2, PT3]$. The SLHS technique is used to generate a sampling plan of 60 points for the initial fitting of the

surrogate model. In this case, the proposed methodology is applied considering its unconstrained adjustment (see Figure 3.2), based on the maximization of the EI criterion (see, Eq.(3.6)).

The results show that in the case (1), a significant number of function evaluations (293 function evaluations) is required, whereas, in case (2), the number of function evaluations is significantly decreased (67 function evaluations) and the computational time as well (175 sec), achieving a solution with an objective function value that is very close to the optimal solution obtained in case (1).

3.4.3 Utility plant case study

This case study involves a utility plant, in Figure 3.5, which supplies the required energy to an industrial process, as electrical and thermal energy demands. The system is composed of a boiler that receives water and supplies high pressure steam, which is distributed to three steam turbines and to the low-pressure steam header that collects the outlet steam from the three steam turbines. The outlet steam is cooled and the water is taken to a deaerator to remove the dissolved gases from it. After that, demineralized water is added to compensate for plant losses and the water is pumped back to the boiler inlet. The process is modeled using the Aspen Hysys modeling environment. Aspen Hysys is an engineering software package, used for modeling and simulation of complex chemical/industrial processes, and takes the form of black box models.

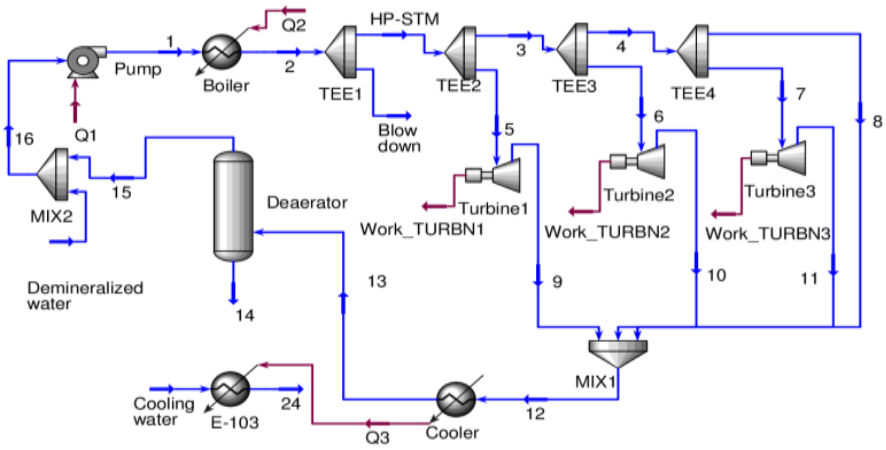


Figure 3.5. Aspen model of the utility plant case study.

The objective is the minimization of the operational cost of the utility plant, which is the summation of the operational cost of these described units (boiler, turbines, deaerator and pump) plus the cost of the required resources (cooling water, demineralized water and energy). These costs were calculated using the correlations presented in (Bruno, et al., 1998). The

operational cost [OPR_{cost}] is modeled as a function of five optimization variables (i.e., input or control variables) which include: the boiler outlet steam temperature and flow rate [STM_{temp}, STM_{flrt}] that can be manipulated within the ranges [160 ,170] (C^o) and [15-17]×3600 ($Kgmole/h$), respectively, and the steam split fractions to the three turbines [$SF_{TUR1}, SF_{TUR2}, SF_{TUR3}$] that can be tuned within the ranges [0.4, 0.6], [0.5, 0.7] and [0.8, 1.0], respectively . There are power demand constraints at the three turbines, which is required to maintain a minimum efficiency [$Work1 \geq 35000\ kW, Work2 \geq 25000\ kW, and\ Work3 \geq 15000\ kW$]. The case study has been solved with two different techniques summarized in Table 3.5:

- in case (1), the optimization has been achieved using the original complex FPM of the process (Aspen Plus) and the "*fmincon*" solver, based on SQP algorithm, integrated in the Matlab optimization toolbox, and

- in case (2), the proposed SBO framework with three different stopping criteria has been used. The process outputs in interest are four, which includes the objective and the three constraints [OPR_{cost}, $Work1, Work2, Work3$]. So, in case (2), four kriging surrogate models were fitted (one for each of the outputs) using SLHS technique to generate a sampling plan of 45 points.

Table 3.5. Optimization results of the utility plant case study.

	Case 1 (Aspen model +SQP)	**Case 2 (Kriging+ CEI)**		
		Max. no. of iter=3	***Max. no. of iter =5***	***Max. no. of iter =9***
STM_{flrt}	16.57	16.94	16.80	16.68
STM_{temp}	160.3	165.9	162.5	163.7
SF_{TUR1}	0.464	0.460	0.465	0.462
SF_{TUR2}	0.623	0.608	0.620	0.615
SF_{TUR3}	1.00	0.940	0.980	0.990
OPR_{cost} **(\$/year) × 10^6**	2.66	2.72	2.69	2.68
No. of kriging models	0	4	4	4
No. Func. eval. (kriging)	0	45	45	45
No. Func. eval. (optimization)	270	3	5	9
Initial DOCE time (sec)	0	49	49	49
Experiment time (sec)	0	108	108	108
Optimization time (sec)	993	64	106	226
Total no. of function eval.	**270**	**48**	**50**	**54**
Total time (sec)	**993**	**221**	**263**	**383**
Computational reliability	**60%**		**100%**	

When compared with case (1), the proposed SBO framework requires a significantly lower number of function evaluations, and the overall computational effort is also significantly reduced, leading to very similar operating set-points. The constraint violations of the SQP optimizer and the proposed optimization methodology were zeros for this case study, and also for the previous applications. The reduction of the computational effort is much greater in successive uses of the algorithm, which will make use of the already fitted surrogate models and so will avoid the computational load associated to the initial surrogate models generation. But the main advantage of the proposed procedure, especially when dealing with complex highly nonlinear systems, is its computational reliability, which is basic in the day-to-day optimization of the operating conditions in situations which require fast decision-making: further the computational load (case(1)) associated to the optimization itself, the evaluation of FPM during an optimization procedure may require a huge quantity of time and human effort to redress the computational system from eventual failures, inconsistencies and convergence problems caused by the evaluation of the model for incompatible input combinations the optimizer may try. And additionally, if specific simulation tools are used (e.g., Aspen), it is not easy to make them compatible with standard optimization software tools (e.g., Matlab). Finally, in the previous applications, the SQP optimizer failed many times to find the optimal solution, and it sometimes failed even to find a feasible solution. The proposed kriging-based methodology integrates the model with the optimization algorithm and uses the FPM just in a relatively few evaluations, reducing dramatically the problems associated to these computational issues.

3.5 CONCLUSIONS

In this Chapter, a SBO methodology for steady-state operation optimization of complex nonlinear chemical processes, which are modelled by black box functions, is presented. The method is based on replacing the entire complex FPM (e.g., modular process simulator) with a set of global surrogate models (based on the kriging technique) representing the objective function and the constraints, which are fitted and validated using input-output data generated by the simulation of the FPM at specific value combinations of the input variables selected by DOCE technique. The search space of the decision variables (i.e., the surrogate model input domain) is explored by means of an adaptive sampling procedure, based on the EI (for unconstrained problems) or the CEI (for constrained problems) criteria. During each iteration of this sequential search procedure, the surrogate models are updated with the obtained optimal solution point for refining the search around the candidate solution.

The effectiveness of the methodology, with respect to the targeted objectives, has been proven by its application to benchmark mathematical examples for nonlinear constrained

optimization and two case studies including the operation optimization of a multi-stage gas turbine and a utility plant, both modeled by ASPEN simulation environment. The methodology performance has been compared to that of classical optimization procedures. The results clearly confirm the methodology capabilities in terms of:

i) high accuracy of the provided optimal solutions (a normalized root mean square error less than 1%, in most cases),

ii) overcoming many limitations of the traditional optimizers in complex process operation optimization, by significantly increasing the reliability of the numerical process (30 %, in the worst cases) associated to the frequent failure of obtaining the correct optimal solution due to reasons such as failure to start from a good feasible initial solution, trapping in a local minima or failure of the calculations due to convergence problem of such complex FPM at specific combinations of the input variables values. This computational/numerical reliability is essential when online decisions are required.

iii) reduction in the number of required function evaluations (77% reduction, in worst cases) and optimization time (48% reduction, in worst cases).

The proven capabilities of the method satisfy the requirements of the hour-to-hour or day-to-day operation optimization of complex processes by providing accurate optimal solutions in a much quicker way and guaranteeing reliable computation.

Chapter 4: Machine Learning-based Multi-Parametric Solution of Chemical Processes Operation Optimization under Uncertainty

Chemical process operation optimization aims at obtaining the optimal operating set-points of the process by real-time solution of an optimization problem that embeds a steady-state model of the process. This task is challenged by unavoidable uncertainties and fluctuations. MultiParametric Programming (MPP) is an efficient approach for solving such kind of problems, where the optimal set-points must be updated, in real-time, in response to sudden changes in the Uncertain Parameters (UPs). MPP provides simple algebraic functions describing the optimal solution as a function of the UPs, which allows alleviating large computational cost required for solving the optimization problem each time the UPs values vary. However, MPP applicability requires a well-constructed mathematical model of the process, which is not suited for process operation optimization, where complex, highly nonlinear and/or black-box models are usually used. To overcome this issue, this chapter presents two novel machine learning-based methodologies for multiparametric solution of optimization problems. The first methodology, which targets general continuous optimization problems, is based on the offline development of machine learning models for regression (surrogate models) that approximate the multiparametric behavior of the optimal solution over the entire space of the UPs. The second method, which targets Mixed-Integer optimization problems, harnesses machine learning models for regression (surrogate models) and clustering techniques in order to approximate the relations between the optimal values of the continuous variables and the UPs, while machine learning models for classification are employed to identify the optimal values of the integer variables also as a function of the UPs. In both methodologies, the data-driven models are developed using data generated by running the optimization using the original complex process model under different UPs values. The trained models are, then used online to, quickly and accurately, predict the optimal solutions in response to UPs variation. The methodologies are tested on benchmark MPP mathematical examples and applied to three case studies of process operation optimization. The results

demonstrate the methodology effectiveness in terms of high prediction accuracy, robustness to deal with problems of different natures and significant reduction in the complexity of the solution procedure compared to the traditional MPP approach.

4.1 INTRODUCTION

Plant-wide optimization, or process operation optimization, is a principle layer in the decision-making hierarchy of chemical plants management. It receives, as inputs, the outcomes and decisions coming from the above layers (i.e., planning and scheduling), such as production rate targets over long time periods (weeks/days), assignment of resources to activities, sequencing of the activities. (Hauptman & Jovan, 2004; Roffel & Betlem, 2004).

The goal of process operations optimization is to obtain the optimal values of the process variables (temperatures, pressures, concentrations, flow rates, etc.) at which the plant and its units must operate to maximize certain performance criteria (e.g., efficiency, profit, operational cost), while satisfying all the constraints (equipment capacities, environmental restrictions, etc.) and requirements (product quality, production yields, safety, etc.) (Vaccari & Pannocchia, 2017; Biegler, 2010). This is achieved by solving, in real time, an optimization problem, which embeds a detailed and rigorous steady-state model of the process (Shao, et al., 2019). More, detailed description of the process operation optimization can be found in Chapters 1 and 3.

Recently, there is growing interest to use complex and high-fidelity First Principle Models (FPMs) of the process in the operation optimization task. Although these models are able to capture more detailed features and sophisticated characteristics of the process and, consequently, provide more accurate estimation of its behavior, they show many practical drawbacks and challenging characteristics, such as the high nonlinearity, complexity, intricate architecture and large computational cost of simulation (Henao & Maravelias, 2011; Norbert, et al., 2017; Quirante, et al., 2018). These challenging characteristics inherent to the FPMs of chemical processes represent an obstacle to their use for the operation optimization, especially for large scale and/or fast dynamic processes (Salback, 2004; Kajero, et al., 2017). Chapter 3, also, presents detailed information about these drawbacks and challenging characteristics.

To overcome the above drawbacks and cope with the above challenges, Surrogate Based Optimization (SBO) approaches (as the one presented in Chapter 3) have been proposed and have received significant attention in the chemical process industry area (Quirante, et al., 2018). The basic idea of SBO is to use the original complex FPM for generating input-output data points by “Computer Experiments”, and use them to develop accurate, but simple and fast-running, data-driven models (“metamodels” or “surrogate models”). Then, these data-

driven models are used in replacement of the complex FPM in the addressed optimization problem (Ochoa-Estopier & Jobson, 2015). More details about SBO methods developed in the chemical process engineering area is presented in Chapter 3, moreover, a comprehensive review about the same topic can be found in (Kajero, et al., 2017).

One challenge for the applications of SBO approaches to process operation optimization comes from the fact that surrogate models are trained on data generated by FPMs with values of parameters and conditions predefined so as to match the real process behavior. So, any sudden and uncertain change in these parameters and conditions in the real process makes the surrogate models and, consequently, the obtained optimal solution based on their analysis, no longer valid/realistic.

The presence of uncertainty in the process is unavoidable at various levels (Jiao, et al., 2012), including inherent physical properties (e.g., kinetic rates, heat transfer constants) (Flemming, et al., 2007; Norbert, et al., 2017; Diangelakis, et al., 2017) and process fluctuations (e.g., feed streams properties like temperatures, pressures and concentrations, recipe variations, processing time, equipment efficiencies) (Mesfin & Shuhaimi, 2010; Papathanasiou, et al., 2019), as well as external uncertainty (such as resources, prices, demands) (Li, 2010). Many methods have been developed for handling uncertainty in optimization problems, most of them can be categorized into two main approaches: proactive and reactive(Medina-González, et al., 2020). The proactive approach aims at providing conservative optimal decisions, which minimize the consequences of the uncertainty on the performance measures of the system (i.e., objective(s)) (Jiao, et al., 2012). Stochastic programming and robust optimization are among the most popular methods used of the proactive approach (Grossmann, et al., 2016). In stochastic programming methods, the UPs are treated as stochastic variables with "a-priori" known probability distribution functions, whose parameters are estimated from historical data. Then, the goal becomes to identify the optimal decision variables that maximize/minimize the expected value of the objective function(s) and achieve feasibility over the distribution of the UPs (Li, 2010). Robust optimization methods deal with unknown but bounded UPs and aim at finding robust optima that ensures the feasibility of the solution and the immunity of the performance measure over the entire range of realizations of the UPs (Norbert, et al., 2017). Nevertheless, two limitations associated to the use of these methods are: *i*) the large computational cost required, since obtaining the optimal solution using these methods implies the analysis of a large number of uncertain scenarios, which grows with the number of UPs, *ii*) the need of complete knowledge of the characteristics of the UPs to identify their types and probability distributions, which is unrealistic especially in dynamic environments and *iii*) the problem that the provided solution

becomes suboptimal for most of the realizations of the uncertainties during the operation/production (Li, 2010; Pistikopoulos, 2008).

The reactive approach, instead, is considered when it is necessary to promptly provide online update of the optimal values of the decision variables in response to real-time changes of the UPs values, which can be measured once unveiled. Since the reactive approach is able to provide the optimal solution for each realization of the UPs, they are preferred for the application in dynamic production environments (Pistikopoulos, et al., 2007). Among the reactive methods, MPP offers outstanding capabilities (Pistikopoulos, 2008): i) its solution provides simple mathematical functions mapping the optimal decisions variables and objective(s) over the entire space of UPs, ii) once uncertainty is unveiled, optimal decisions can be readily and immediately calculated by these simple functions, avoiding the large computational cost associated to repetitive optimization procedure, iii) MPP is not only able to handle uncertainty related to the process conditions, but also to the optimization problem parameters (e.g., relative weights or importances of different objectives). A review on the different MPP algorithms developed for problems of different natures (linear, quadratic, nonlinear, mixed, integer, convex, local, global, etc.) can be found in (Pistikopoulos, 2008; Pistikopoulos, et al., 2007).

Despite the attractive characteristics of MPP, its successful application is conditioned by two main requirements: the first is the deep and complex mathematical programming knowledge required for the development of such formulations and the second is the availability of a well-constructed mathematical model of the process (Bemporad, et al., 2002; Kouramas, et al., 2011; Rivotti, et al., 2012). This hinders the smooth applications of MPP to process operation optimization in practice, where complex, highly nonlinear and black-box models need to be considered (e.g., modular process simulators).

To tackle these limitations, the use of data-driven or Machine Learning (ML) techniques for the solution of MPP problems has recently emerged as a feasible alternative. This research direction has been considered by Katza, et al. (2020) and Katza, et al. (2020b), who have proposed the use of deep learning techniques for solving explicit MPC problems. Medina-González, et al. (2020) have used kriging models for approximating the multiparametric solution of multiobjective optimization of a bio-based energy chain subjected to uncertainties including electricity demand, environmental conditions and social dynamics. Lupera, et al. (2016) have proposed a similar approach for supporting reactive scheduling in a multiproduct batch chemical plant: a kriging metamodel is used to approximate the optimal management decisions as a function of UPs including equipment starting times and task-unit assignment. Lupera, et al., (2018) have addressed the solution of mixed-integer optimization problems by using a combination of regression (kriging) and classification (ANN) techniques to

approximate the optimal continuous and integer variables, respectively, as a function of the UPs. They have applied the methodology to a simple example of a supply chain problem. Lupera, et al. (2018) have proceeded to apply the latter method to solve mixed-integer reactive scheduling problem.

In the previous works, each of the developed ML-based method for the solution of MPP problems has been tailored and evaluated with respect to a particular application of interest, involving an optimization problem of specific nature, which, in most of the cases, is linear. Also, they have not addressed the optimization of the process and/or unit operations, where the benefits of such methodology, if successfully applied, would be significant. Because, the update of the optimal setpoints in response to UPs variations is typically required within very tight time slots (minutes or seconds), whereas, the models of the process are often complex, nonlinear and/or in the form of computationally expensive black-boxes.

In this chapter two novel data-driven methodologies are developed for the solution of general MPP problems. The first method addresses continuous optimization problems and aims at developing global MultiParametric Metamodels (MPMs), which are trained using input-output data (UPs-optimal variables and objective) to approximate the multiparametric behavior of the optimal solutions over the entire space of the UPs. The second methodology targets general Mixed-Integer optimization problems. The method models the multiparametric behavior of a continuous variable by using clustering techniques in order to isolate those potential local regions of the UPs space over which the optimal solution behaves significantly different. Then a local MPM is trained to approximate the optimal solution behavior of this continuous decision variable over each of the identified local regions. For integer decision variables, the methodology harnesses Classification Techniques (CT) to predict the optimal values of the integer variables also as a function of the UPs.

In both methodologies, the input-output data (i.e., UPs values-optimal variables and objective values) are generated through the repeated optimization of the original process model using state-of-art optimizers and considering different values of the UPs that are selected by DOCE techniques. The performance of the first methodology is assessed by its application to five MPP benchmark examples of different nature (linear, bilinear, quadratic and nonlinear optimization problems) and to two case studies, including the operation optimization of a utility plant modeled by a black-box modular process simulator and of a batch reactor. The second methodology is also validated by its application to benchmarks case study.

The main novelties of the work are:

i) the generality and applicability of the proposed methodologies which, unlike the reviewed data-driven method for the solution of MPP problems, is aimed at

solving different types of optimization problems (i.e., linear, bilinear, quadratic, nonlinear, black-box) in a systematic way,

ii) the capacity of the proposed methodologies to address process and unit operation optimization problems, where highly nonlinear and/or black-box models are typically used,

iii) a novel utilization of the kriging metamodel to approximate the "optimal" behavior of a system, unlike most of the chemical engineering literature, in which the kriging is used, as any machine learning technique, to approximate the response of a system.

The rest of the chapter is organized as follows. Sections 4.2 and its subsections correspond to the first methodology, and they include the problem statement, the details of the proposed method, its application and the obtained results. Section 4.3 and its subsections show the same elements for the second methodology. Section 4.4 concludes the chapter work and highlights possible future directions of research.

4.2 MPMS FOR CONTINUOUS OPTIMIZATION

4.2.1 Problem statement

The proposed methodology is aimed at overcoming the difficulty of solving the process operation optimization problem using classical MPP formulations. Generally speaking, the problem (Eq.(4.1)) is to find the optimal values of the decisions variables $x \in R^K$ that maximize the objective function $Z(x,\theta)$ representing a performance index of the process, for which the available FPM f is complex, nonlinear and/or black-box. The problem is subjected to set of constraints $g_l(x,\theta), l = 1,..,L,$ and is influenced by a set of bounded UPs $\theta \in R^k$ (Pistikopoulos, 1995; Pistikopoulos, et al., 2007; Caballero & Grossmann, 2008).

$$\left.\begin{array}{c} \min\limits_{x} Z(x,\theta) \\ S.T. \quad f(x,\theta)\,, \\ g_l(x,\theta) \le 0, \quad l = 1,2,\ldots.L, \\ x \in R^K, \quad \theta \in R^k, \\ lb_x \le x \le ub_x, \quad lb_\theta \le \theta \le ub_\theta \end{array}\right\} \tag{4.1}$$

The MPP solution (Eq.(4.2)) is in the form of a number of P simple mathematical relations describing each of the optimal decision variables x^* and the objective Z^* as functions of the UPs θ.

$$\left.\begin{array}{c} Z_p^* = \mathcal{F}_{0,p}(\theta) \\ x_{ip}^* = \mathcal{F}_{i,p}(\theta) \\ lb_{\theta_p} \leq \theta \leq lb_{\theta_p}, \quad i = 1,2,\dots,K, \quad p = 1,2,\dots P \end{array}\right\} \tag{4.2}$$

The $p-th$ relation, $p = 1,2,\dots,P$, is only valid for a certain partition of the UPs space, which is called "critical region", where the P critical regions are adjacent, non-intersecting subspaces and, hence, their union equals to the entire UPs space. So, as the UPs are unveiled, the optimal solution is simply and immediately calculated by evaluating these simple functions (Dua & Pistikopoulos, 1999).

In this work, we consider situations, such as process and unit operation optimization, in which classical MPP approaches cannot be applied due to the high nonlinearity and complexity of the process model, and consequently, the exact solutions in Eq.(4.2) cannot be attained. The proposed methodology, alternatively, develops ML models that act as accurate data-driven multiparametric relations, which are referred to as "MultiParametric Metamodels (MPM)", and expressed as follows:

$$\left.\begin{array}{c} \hat{Z}^* = f_0(\theta) \\ \hat{x}_i^* = f_i(\theta) \\ lb_{\theta_p} \leq \theta \leq lb_{\theta_p}, \quad i = 1,2,\dots,K \end{array}\right\} \tag{4.3}$$

where, $f_0, f_i, i = 1,2,..K$ are supervised ML regression models efficiently built to approximate the optimal objective function, $\hat{Z}^*$, and the decision variables, $\hat{x}_i^*$,as functions of the UPs, θ.

4.2.2 Methodology

The steps of the proposed methodology are schematically illustrated in Figure 4.1, and are detailed in the subsequent sections.

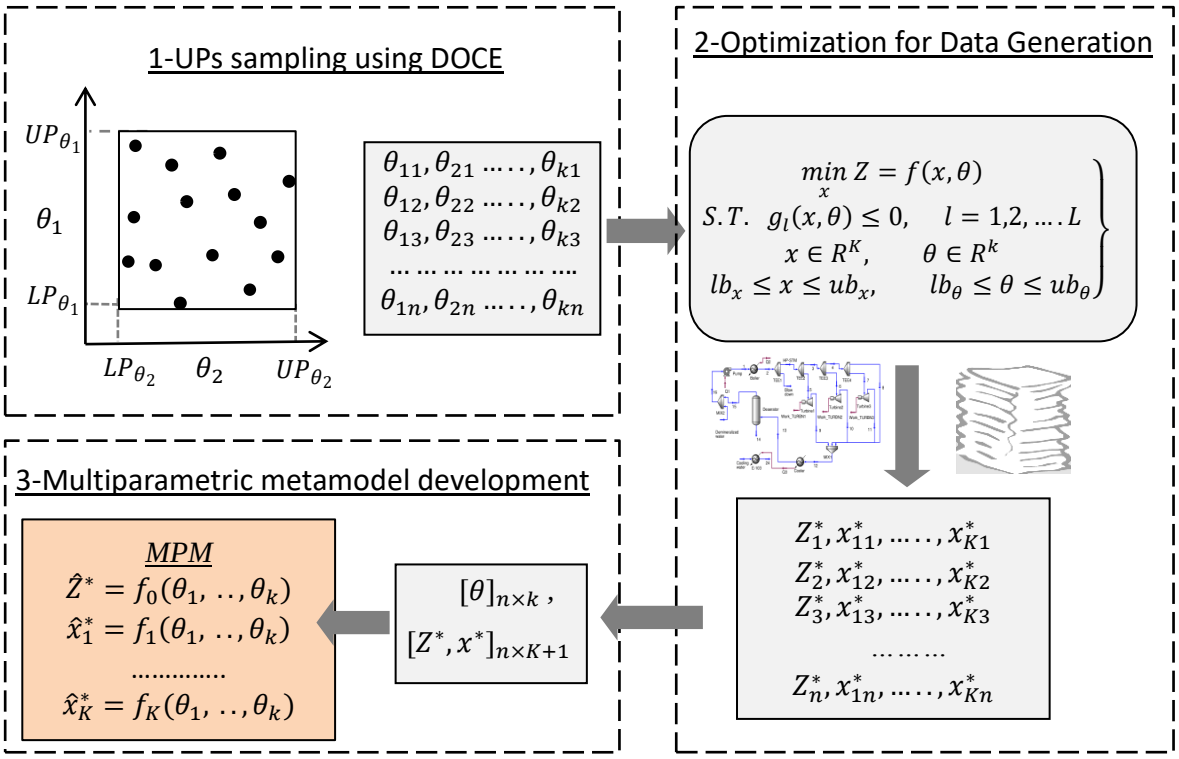


Figure 4.1. Schematic representation of the proposed methodology

4.2.2.1 UPs sampling using DOCE

In this step, the optimization problem and the involved FPM are explored in order to identify the influencing UPs, $\theta \in R^k$, and estimate their bounds $lb_\theta \leq \theta \leq ub_\theta$ (MPM input domain). Then, the goal becomes the selection of a proper set of combinations of the UPs values, $[\theta]_{n \times k}$, (i.e. the MPMs inputs) that uniformly cover the entire UPs space in order to collect information about the optimal solution behavior over all the sub-regions of the global domain.

Many DOCE techniques are available in the literature such as Latin hypercube sampling (Garud, et al., 2017; Forrester, et al., 2008), low discrepancy sequences as Hammersley technique (Ibrahim, et al., 2019), space-filling designs and sequential or adaptive sampling (Kajero, et al., 2017). Most of these DOCE techniques show both desired characteristics and limitations in terms of the uniformity of the generated sampling plan and of the required computational cost (Garud, et al., 2017). More details about different DOCE techniques can be found in (Ibrahim, et al., 2019). In this work, a hybrid technique of Hammersley sequence and full factorial design is used, as it achieves high uniformity with low computational cost

(Ibrahim, et al., 2019). The idea behind this hybrid technique is to employ the factorial design to compensate the limited ability of the Hammersley sequence to select sample points near the bounds and vertices of the input space, while at the same time, exploiting the high uniformity of the samples set of the Hammersley sequence over the bulk of the input space.

The number of required sample points, n, is proportional to the number of UPs, k, influencing the optimization problem and depends, also, on the complexity of the multiparametric behavior of the optimal solutions. On the other hand, as n increases, more computational effort is required for performing optimization runs and for the MPM training. So, the modeler should carefully balance this trade-off. More details about DOCE techniques can be found in Section 2.1.

4.2.2.2 Optimization for data generation

Once a good sampling plan, $[\theta]_{n\times k}$, is obtained, the optimization problem is solved n times, each time considering one of the UPs combinations, so as to obtain the matrix of the optimal values of the objective and decision variables $[Z^*, x^*]_{n\times K+1}$. In general, state-of-art optimization algorithms can be used to solve the optimization problem depending on its characteristics (Biegler, 2010). Particularly, this work addresses continuous optimization problems, including linear, quadratic and nonlinear types, and we employ the Matlab optimization toolbox for their solutions. For linear problems, the solver "*linprog*" is used based on a dual simplex algorithm; for quadratic problems, the solver "*quadprog*" is used based on an interior-point-convex algorithm; for nonlinear problems, the optimizer "*fmincon*" is used based on a sequential quadratic programming algorithm. Default values for the optimization algorithms parameters (such as maximum number of function evaluations, tolerance on the constraint violation, termination tolerance on the first-order optimality, termination tolerance on decision variables) are used.

4.2.2.3 MultiParametric metamodels development

The generated input ($[\theta]_{n\times k}$) and output ($[Z^*, x^*]_{n\times K+1}$) data are used to train a number of $K+1$ kriging-based MPM, $f_0, f_1, \dots, f_K$ (see Figure 4.1), each of them approximates the optimal behavior of each of the objective and decision variables as a function of all UPs.

For one optimal decision variable x^*, the kriging technique assumes the predictor $\hat{x}^*(\theta) = \mu_{ok} + \mathbb{Z}(\theta)$, where the constant term μ_{ok} represents the main trend of the system to be approximated, and $\mathbb{Z}(\theta)$ is a deviation/residual from that trend, which accounts for detailed complex behavior of the system that could not be captured via the main trend μ_{ok} (Jones, et al., 1998). The residual $\mathbb{Z}(\theta)$ is modeled as a stochastic Gaussian process with

expected value $E\big(\mathbb{Z}(\theta)\big) = 0$ and covariance between two residuals $cov(\mathbb{Z}(\theta_i), \mathbb{Z}(\theta_j))$ that only depends on their corresponding input values θ_i, θ_j. Thus, it can be calculated as: $cov(\mathbb{Z}(\theta_i), \mathbb{Z}(\theta_j)) = \sigma_{ok}^2 R(\theta_i, \theta_j)$, being σ_{ok}^2 the process variance and $R(\theta_i, \theta_j)$ a correlation function, $R(\theta_i, \theta_j) = exp\Big(-\sum_{\ell=1}^{k} \xi_\ell \big|\theta_{i,\ell-} \theta_{j,\ell-}\big|^{p_\ell}\Big) + \delta_{ij}\, \lambda$, where ξ_ℓ are the model hyper-parameters, δ_{ij} is the Kronecker delta, p_ℓ are smoothing parameters and λ is a regularization constant that enables the kriging predictor to regress noisy data (Forrester & Keane, 2009).

In order to estimate the values of the parameters $[\mu_{ok}, \sigma_{ok}^2, \xi_\ell, p_{\mathcal{L}}, \lambda]$, the likelihood function of the observed data $[x^*]_{n\times 1}$ is maximized. The kriging predictor (Eq.(4.4)) and its estimated error (Eq.(4.5)) are obtained by deriving the augmented likelihood function of both the original training data set and a new interpolating point $(\theta_{new}, x^*_{new})$. In both equations, $[r]_{n\times 1}$ is the vector of correlations between the new point to be predicted θ_{new} and the original training data points, and calculated as $R\ (\theta_i, \theta_{new})$, and $[\mathbf{1}]_{n\times 1}$ is the identity vector (Jones, et al., 1998; Caballero & Grossmann, 2008).

$$\hat{x}^*(\,\theta_{new}) = \mu_{ok} + r^T R^{-1}(x^* - \mathbf{1}\mu_{ok}) \tag{4.4}$$

$$\hat{s}^2(\theta_{new}) = \sigma_{ok}^2(1 + \lambda - r^T R^{-1} r + (1 - \mathbf{1}^T R^{-1} r)^{-1}/(\mathbf{1}^T R^{-1}\mathbf{1})) \tag{4.5}$$

In practice, the maximization of the concentrated log-likelihood function is computationally challenging because of the high effort associated to the repetitive calculation of the correlation matrix inverse $[R]^{-1}_{n\times n}$ during the optimization iterations, which quickly grows with the size of the training data set and/or the model input dimensionality. Besides, the nature of the concentrated log-likelihood function itself is quite complicated because it is flat near the optimum. More details about these computational challenges, and the numerical methods and optimization techniques to overcome or reduce these obstacles can be found in (Fang, et al., 2005; Forrester, et al., 2008).

After the training of the MPMs, they are validated using a new and different validation dataset, $[\theta^v]_{n^v\times k}$ $[Z^{*,v}, x^{*,v}]_{n^v\times K}$, where n^v is the number of samples. The MPMs are used to estimate the values of the optimal decision variables and objective, $[\hat{Z}^{*,v}, \hat{x}^{*,v}]_{n^v\times K}$, which are compared to their exact counterparts $[Z^{*,v}, x^{*,v}]_{n^v\times K}$ in order to calculate an accuracy measure, such as the Normalized Root Mean Square Error (NRMSE) (Eq.(4.6)), for each of the $K+1$ MPMs.

$$\begin{aligned} NRMSE &= 100 * \frac{RMSE}{(x^{*,v,max} - x^{*,v,min})}, \\ RMSE &= \sqrt{\frac{1}{n^v}\sum_{i=1}^{n^v}(x^{*,v,i} - \hat{x}^{*,v,i})^2} \end{aligned} \tag{4.6}$$

4.2.3 Applications

In this section, five benchmark examples with different characteristics selected from the MPP literature are used to assess the proposed methodology in terms of its accuracy of estimating the optimal solutions and its applicability to different types of continuous optimization problems, including linear, bilinear, quadratic and nonlinear. The methodology is, then, applied to two case studies regarding the operation optimization of a utility plant and a batch reactor.

4.2.3.1 Linear optimization: refinery blending problem

A refinery blending and production process (Pistikopoulos, et al., 2002; Pistikopoulos, et al., 2007) receives raw materials including two types of crude oils with flowrates x_1 and x_2 (bbl/day), which are processed in order to produce four types of products, namely, Gasoline, Kerosene, Fuel-oil and Residuals. It is required to select the optimal flowrates x_1 and x_2 that maximize the profit Z $(\$/day)$. The optimization problem (Eq.(4.7)) is subjected to three constraints associated to the maximum allowable production rates of the Gasoline, Kerosene and Fuel-oil from each crude oil type, and is affected by two UPs, θ_1 and θ_2, which are the maximum allowable production rates ((bbl/day)) of the Gasoline and Kerosene, respectively:

$$\left.\begin{array}{c} \underset{x}{Min}\ Z = \ 8.1\,x_1 + \ 10.8\,x_2 \\ S.T: \ \ 0.80\,x_1 + \ 0.44x_2 \le 24000 + \theta_1 \\ 0.05\,x_1 + \ 0.10x_2 \ \le 2000 \ + \theta_2 \\ 0.10\,x_1 + \ 0.36x_2 \ \le 6000 \\ x_1 \ge 0, \quad x_2 \ge 0 \\ 0 \ge \ \theta_1 \ge 6000, \ \ 0 \ge \ \theta_2 \ge 500 \end{array}\right\} \tag{4.7}$$

The methodology is applied following the steps described in Section 4.2.2. Only in this problem, different training sets of different sizes are used to build the MPMs in order to show the effect of the training set size on the MPMs accuracy, training time and prediction time. First, over the space of the UPs [0: 60000 bbl/day, 0: 500 bbl/day], five different sampling plans with different sizes, $([\theta_1,\theta_2]_{15\times2}\,,[\theta_1,\theta_2]_{30\times2}\,,[\theta_1,\theta_2]_{45\times2},[\theta_1,\theta_2]_{60\times2}\,,[\theta_1,\theta_2]_{75\times2})$, are designed by means of the hybrid technique of Hammersley sequence and two-levels fractional factorial design (Figure 4.2). For each of the five sampling plans, the LP problem is solved several times to obtain the corresponding matrix of the optimal objective and variables values, $[Z^*,x_1^*,x_2^*]_{15\times3},\dots,\dots,\dots.[Z^*,x_1^*,x_2^*]_{75\times3}$. The optimization problem is solved using the "*linprog*" optimizer of the Matlab optimization toolbox based on a dual simplex algorithm. Then, using each of the five input-output training datasets, e.g. $[\theta_1,\theta_2]_{15\times2}-[Z^*,x_1^*,x_2^*]_{15\times3}$,

three metamodels are fitted, $\hat{Z}^* = f_0(\theta_1, \theta_2),\ \hat{x}_1^* = f_1(\theta_1, \theta_2),\ \hat{x}_2^* = f_2(\theta_1, \theta_2)$, one for each of the optimal objective and decision variables.

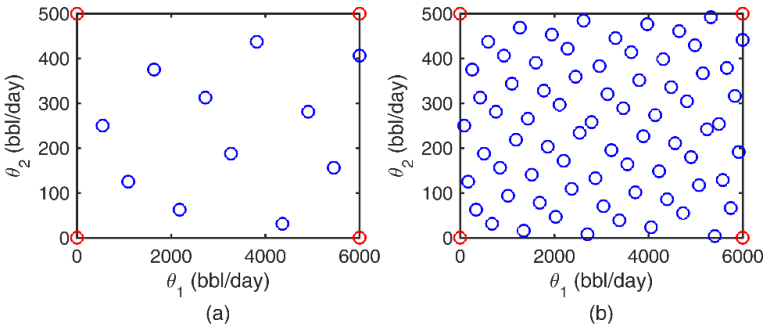


Figure 4.2. Two of the five sampling plans of the UPs ((a) $n = 15$ and (b) $n = 75$): Blue circles indicate UPs combinations generated by the Hammersley techniques, whereas red circles refer to those generated by the two-levels full factorial design..

One validation data set $[\theta_1^v, \theta_2^v]_{400\times2} - [Z^{*,v}, x_1^{*,v}, x_2^{*,v}]_{400\times3}$ is generated and used to assess the performances of all the MPMs, f_0, f_1, f_2, trained by the five different datasets. It is worth highlighting that the validation set is in the form of a uniform grid of 20×20 over the UPs space, so as to achieve a credible assessment of the MPMs predictions in all the local regions of the UPS. The NRMSE of the MPMs prediction is calculated by comparing their estimated outputs $[\hat{Z}^{*,v}, \hat{x}_1^{*,v}, \hat{x}_2^{*,v}]_{400\times3}$ with the exact ones $[Z^{*,v}, x_1^{*,v}, x_2^{*,v}]_{400\times3}$ provided by the rigorous optimization itself.

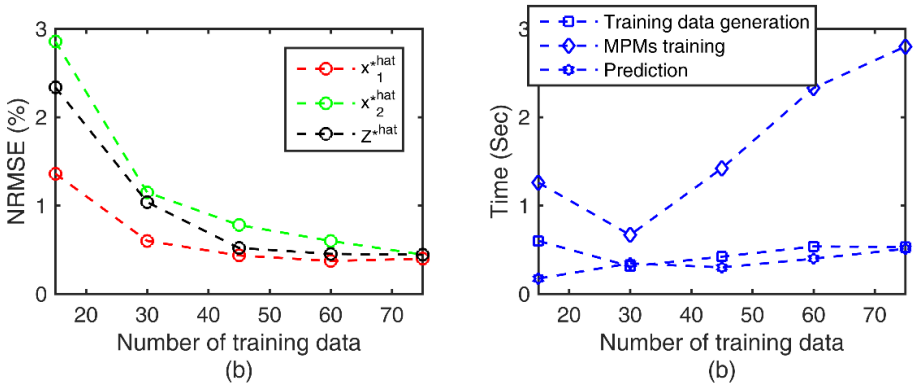
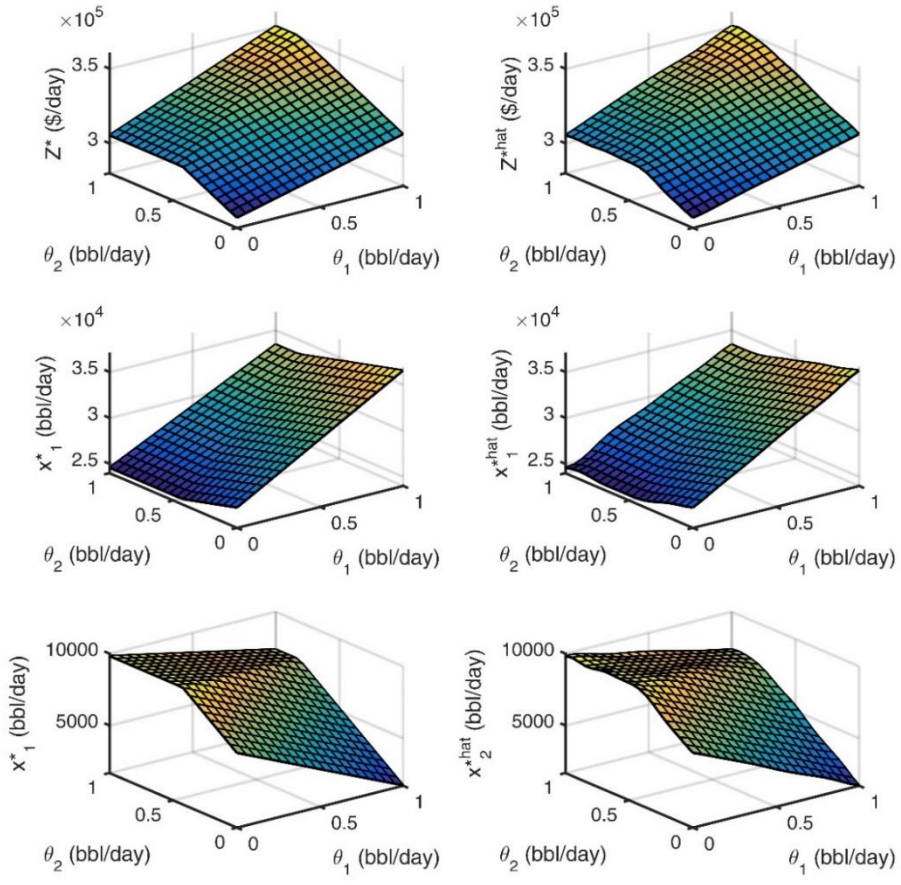


Figure 4.3. (a) NRMSE and (b) computational time required for training data generation, MPMs training and prediction times as a function of the training dataset size.

Figure 4.3 shows how the NRMSE of the MPMs decreases and how the training and prediction (of the validation set) times increase with increasing training dataset size. The Figure also indicates that even with few (i.e., 15) training patterns, the MPMs are able to achieve satisfactory accuracy (less than 3% of NRMSE). Table 4.1 illustrates the NRMSE, and the required training and validation times of the MPMs when the largest training dataset (75 patterns) is considered.

Figure 4.3 and Table 4.1 shows i) very high accuracy of the MPMs trained by the dataset containing 75 instances (NRMSE less than 0.5% of all the MPMs), ii) affordable "offline" computational time required for training 2.79 sec (0.86+0.80+1.13) and iii) very low computational time demanded in predicting the optimal solution of the validation set, 0.00127 sec (0.51/400), that saves 75.7% ((2.1-0.51)/2.1) of the computational time of the real optimization. More importantly, one multiparametric relation (i.e., a MPM) is able to describe the optimal solution behavior over the entire space of the UPs. In the literature (Pistikopoulos, et al., 2007), the classical MPP approach provides a solution for the same problem that divides the UPs space into two critical regions, consequently, two sets of mathematical parametric functions are obtained each of them is valid only for one of the two partitions of the UPs space.

As an additional assessment of the performance of the developed MPMs (and also of the correctness of the solution of the optimization problems used to generate the training and validation data), the deterministic multiparametric solution provided in (Pistikopoulos, et al., 2007) is used to calculate the optimal objective and decision variables values of the validation set (Figure 4.4).



(a) (b)

Figure 4.4. Refinery blending problem: comparison between (a) the results obtained by the classical MPP solution provided in (Pistikopoulos, et al., 2007) and (b) the results provided by the proposed methodology.

4.2.3.2 Bilinear optimization

In this mathematical example (Eq.(4.8)) (Ichihara & Anai, 2012), a bilinear objective function is to be minimized, subjected to two linear constraints which are affected by one uncertain parameter:

$$\left.\begin{array}{c} \underset{x}{Min}\, Z = x_1 x_2 \\ S.T.: 2\,x_1 + x_2 \geq \theta \\ x_1 + 3\,x_2 \geq 0.5\,\theta \\ -1 \leq x_1, x_2 \leq 1, \quad 0 \leq \theta \leq 1 \end{array}\right\} \tag{4.8}$$

The proposed method is applied starting by designing a sampling plan $[\theta]_{60\times1}$ over the space [0: 1] of the UP (i.e., the MPMs input). The optimization problem is solved 60 times considering the values of the UP in the sampling plan, to obtain the corresponding optimal decision variables and objective values (i.e., the MPMs outputs) $[Z^*, x_1^*, x_2^*]_{60\times3}$. The "fmincon" optimizer of the Matlab optimization toolbox is used, based on a sequential quadratic programming algorithm. Using these input-output training data, three MPMs ($\hat{Z}^* = f_0(\theta),\ \hat{x}_1^* = f_1(\theta),\ \hat{x}_2^* = f_2(\theta)$) are fitted, one for each of the optimal objective and decision variables. A different validation dataset including a uniform grid of 150 sample is generated ($[\theta^v]_{150\times1} - [Z^{*,v}, x_1^{*,v}, x_2^{*,v}]_{150\times3}$), the MPMs are used to estimate the optimal solutions $[\hat{x}_1^{*,v}, \hat{x}_2^{*,v}, \hat{Z}^{*,v}]_{150*3}$ and the NRMSE of the prediction is calculated for each MPM.

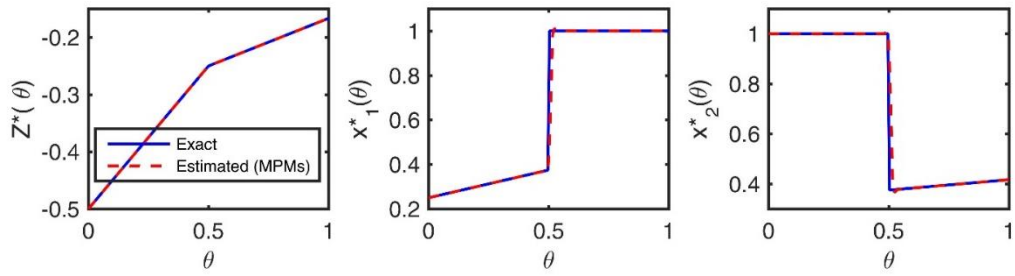


Figure 4.5. Bilinear optimization problem: comparison between the solutions obtained by the classical MPP provided in (Ichihara & Anai, 2012) (blue solid lines) and the solutions provided by the proposed methodology (red dashed lines).

Table 4.1 reports the results obtained by the proposed methodology, that indicate the very high accuracy of the three MPMs (NRMSE of 0.01%, 0.8% and 0.4%) and also a significant reduction in the computational time required for calculating the optimal solutions with respect to the real optimization, that reached to 98.9% ((16.2-0.167)/16.2). Also, the deterministic MPP solution provided in (Ichihara & Anai, 2012) is used to calculate the optimal objective and variables of the validation set (Figure 4.5).

4.2.3.3 Quadratic optimization

The second application (Dua, et al., 2002) involves the minimization of a quadratic objective function subjected to a set of six constraints, which are affected by two UPs in their right-hand side (Eq.(4.9)).

$$\left.\begin{array}{c} \underset{x}{Min}\, Z = c^T[x_1, x_2]^T + 0.5[x_1, x_2]Q[x_1, x_2]^T \\ S.T.: A[x_1, x_2]^T \le b + F[\theta_1, \theta_2]^T \\ -1 \le x_1, x_2 \le 1, \quad 0 \le \theta_1, \theta_2 \le 1 \end{array}\right\} \tag{4.9}$$

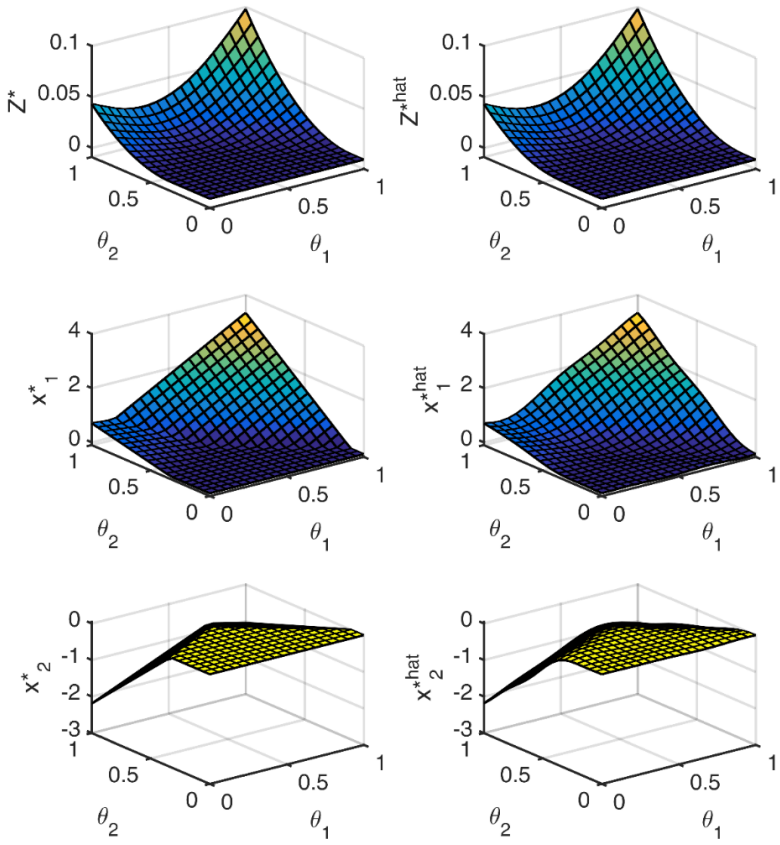
$$c = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, Q \begin{bmatrix} 0.0196 & 0.0063 \\ 0.0063 & 0.0199 \end{bmatrix},$$

$$b = \begin{bmatrix} 0.417425 \\ 3.582575 \\ 0.413225 \\ 0.467075 \\ 1.090200 \\ 2.909800 \end{bmatrix}, A = \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ -0.0609 & 0 \\ -0.0064 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix}, \quad F = \begin{bmatrix} 3.16515 & 3.7546 \\ -3.16515 & -3.7546 \\ 0.17355 & -0.2717 \\ 0.06585 & 0.4714 \\ 1.81960 & -3.2841 \\ -1.81960 & 3.2841 \end{bmatrix}$$

The same procedure is considered. A sampling plan $[\theta_1, \theta_2]_{80\times2}$ is designed over the UPs space. The problem is solved 80 times to obtain the matrix $[Z^*, x_1^*, x_2^*]_{80\times3}$, using the "quadprog" optimizer of the Matlab optimization toolbox based on an interior-point-convex algorithm. Using these input-output data, three MPMs are fitted: $\hat{Z}^* = f_0(\theta_1, \theta_2)$, $\hat{x}_1^* = f_1(\theta_1, \theta_2)$, $\hat{x}_2^* = f_2(\theta_1, \theta_2)$. The validation is accomplished using a different dataset of size 400 ($[\theta_1^v, \theta_2^v]_{400\times2} - [Z^{*,v}, x_1^{*,v}, x_2^{*,v}]_{400\times3}$), where the MPMs are used to estimate the outputs $[\hat{x}_1^{*,v}, \hat{x}_2^{*,v}, \hat{Z}^{*,v}]_{400*3}$ and the NRMSE of the MPMs predictions is calculated (Table 4.1). In this case, the accuracy of the MPMs (NRMSE of 0.0001%, 0.028 % and 0.018%) is significantly higher than that of the MPMs in the precious two examples, which can be explained by the smooth and continuous multiparametric behavior of the optimal solutions over the entire UPs space (Figure 4.6), which is relatively easy to capture by data-driven models. In contrast, the multiparametric behavior of the optimal solution in the previous two cases shows discrete features, which represent a challenge for the data-driven models.

The MPMs were also able to save a considerable percentage (67%) of the time required to calculate the optimal solutions through real optimization, but the saving percentage is not as high as the previous two cases. This is, again, due to the relative simplicity of the optimization problem solution (i.e., a quadratic objective function subjected to linear constraints).

The deterministic multiparametric solution obtained by (Dua, et al., 2002) for the same problem, divides the UPs space into four critical regions, over each of them a different set of mathematical parametric functions are used to calculate the optimal solutions.



(*a*) (*b*)

Figure 4.6. Quadratic optimization problem: comparison between (a) the results obtained by the classical MPP solution provided in (Dua, et al., 2002) and (b) the results provided by the proposed methodology.

4.2.3.4 Quadratic optimization: milk surplus problem

This application considers a Dutch agriculture cooperative company that produces four products including milk for direct consumption, butter, fat and cheese with prices x_1, x_2, x_3 and x_4, respectively (Pistikopoulos, et al., 2007). The consumer demand from each product is modelled as an inverse function of the product price. The cooperative company must decide the optimal prices (that indirectly set the optimal quantities of products) that maximize the profit, disregarding the production costs. The optimization problem is subjected to capacity constraints and an escalation of the price constraint, and is influenced by four UPs, $\theta_1, \theta_2, \theta_3$ and θ_4, related to the consumer demand, and another UP θ_5, associated to the escalation of the prices. Notice that, in this application, the UPs are affecting both of the constraints and the objective function:

$$\min_x Z = -1.2338\,x_1^2 - 0.0203\,x_2^2 - 0.0136\,x_3^2 - 0.0027\,x_4^2 + 0.0031\,x_3\,x_4 + 2139\,x_1 + 135\,x_2 + 103\,x_3 + 19\,x_4 + x_1\theta_1 + x_2\theta_2 + x_3\theta_3 + x_4\theta_4$$

S.T.:

$$-0.0321\,x_1 - 0.0162\,x_2 - 0.0038\,x_3 - 0.0002\,x_4 \leq -80.5 - 0.026\,\theta_1 - 0.800\,\theta_2 - 0.306\,\theta_3 - 0.245\,\theta_4,$$

$$-0.1061\,x_1 - 0.0004\,x_2 - 0.0034\,x_3 - 0.0006\,x_4 \leq 26.6 - 0.086\,\theta_1 - 0.020\,\theta_2 - 0.297\theta_3 - 0.371\,\theta_4,$$

$$1.2334\,x_1 \leq 2139 + \theta_1,$$
$$0.0203\,x_2 \leq 135 + \theta_2,$$
$$0.0136x_3 - 0.0015\,x_4 \leq 103 + \theta_3,$$
$$-0.0016x_3 + 0.0027\,x_4 \leq 19 + \theta_4,$$
$$0.0163x_1 + 0.0003x_2 + 0.0006x_3 + 0.0002\,x_4 \leq 10 + \theta_5,$$
$$-150 \leq \theta_1 \leq 150,$$
$$-5 \leq \theta_2 \leq 5,$$
$$-6 \leq \theta_3 \leq 6,$$
$$-2 \leq \theta_4 \leq 2,$$
$$-1 \leq \theta_5 \leq 1$$

The methodology is, again, applied following the same procedure. A sampling plan, $[\theta_1, \theta_2, \theta_3, \theta_4, \theta_5]_{80\times5}$, is created including different combinations of the UPs values, which are selected by the hybrid DOCE method within the known bounds [-150:150, -5:5, -6:6, -2:2, -1:1]. The quadratic optimization problem is solved 80 times using the "*quadprog*" optimizer of the Matlab optimization toolbox, based on the interior-point-convex algorithm, to yield the matrix $[Z^*, x_1^*, x_2^*, x_3^*, x_4^*]_{80\times5}$. Five MPMs, $\hat{Z}^* = f_0(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)$, $\hat{x}_i^* = f_i(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)$, $i = 1, \dots 4$, are fitted to approximate the optimal profit and prices as a function of the demand and the price escalation uncertainties.

The MPMs performances are assessed relying on a new dataset, $([\theta_1^v, \theta_2^v, \theta_3^v, \theta_4^v, \theta_5^v]_{400\times5} - [Z^{*,v}, x_1^{*,v}, x_2^{*,v}, x_3^{*,v}, x_4^{*,v}]_{400\times5})$, and their accuracies, in terms of NRMSE, are reported in Table 4.1. As in the third case (Section 4.2.3.3), a significantly high prediction accuracy of the MPMs is obtained (a maximum NRMSE of 0.0035 %), which can be justified by similar reasons. This also is supported by the fact that despite the high dimensionality of the optimization problem (four decision variables) and the high number of the UPs (five) with respect to the other examples, the same order of magnitude of training data (80 points) was enough to achieve such high accuracy.

Classical MPP approaches (Pistikopoulos, et al., 2007) provides a deterministic solution to this problem that is characterized by two critical regions over the UPs space. This deterministic solution is compared, using the validation set, to the approximate one provided by our methodology in Figure 4.7.

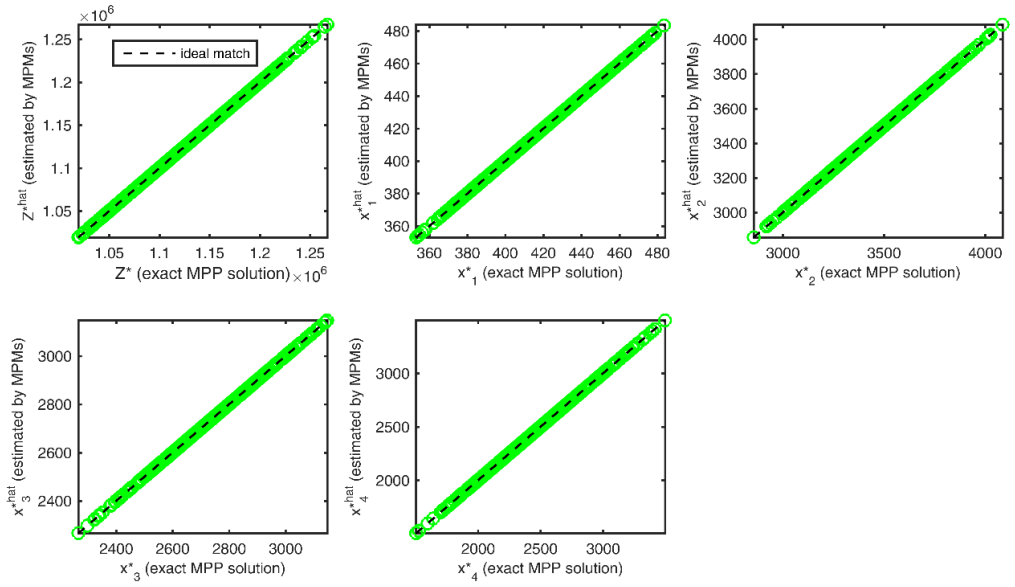


Figure 4.7. Milk surplus problem: comparison between the solutions obtained by the classical MPP provided in (Pistikopoulos, et al., 2007) and the solutions provided by the proposed methodology.

4.2.3.5 Nonlinear optimization

The last multiparametric optimization benchmark example (Eq.(4.10)) (Domínguez, et al., 2010) includes a nonlinear objective subjected to two constraints, each one involving an uncertain parameter in the right-hand side:

$$\left.\begin{aligned} \underset{x}{Min}\ Z = x_1^3 + 2\,x_1^2 - 5\,x_1 + x_2^2 - 3\,x_2 - 6 \\ S.T:\ \ 2.0\,x_1 + x_2 \le 2.5 + \theta_1 \\ 0.5\,x_1 + x_2 \le 1.5 + \theta_2 \\ 0.5\,x_1 + x_2 \le 1.5 + \theta_2 \\ 0 \ge \theta_1 \ge 1,\ \ 0 \ge \theta_2 \ge 1 \end{aligned}\right\} \qquad (4.10)$$

Three MPMs, $\hat{Z}^* = f_0(\theta_1, \theta_2)$, $\hat{x}_1^* = f_1(\theta_1, \theta_2)$, $\hat{x}_2^* = f_2(\theta_1, \theta_2)$, are fitted using the input-output training data, $[\theta_1, \theta_2]_{140\times2} - [Z^*, x_1^*, x_2^*]_{140\times3}$, generated as explained earlier for the other examples. The performance of the MPMs is evaluated using a new validation set, $[\theta_1^v, \theta_2^v]_{400\times2} - [Z^{*,v}, x_1^{*,v}, x_2^{*,v}]_{400\times3}$, and the result is shown in Table 4.1. Figure 4.8 compares the approximated multiparametric solution obtained by the MPMs to the deterministic one obtained by (Pistikopoulos, et al., 2007; Domínguez, et al., 2010). In (Domínguez, et al., 2010) different classical MPP algorithms have been used to solve the problem, and the best one was the quadratic approximation algorithm that partitioned the UPs into four critical regions.

In this application, a relatively high number of training data was required to fit the MPMs, and although the resulted accuracy (NRMSE of 0.62%, 1.5% and 1.9%) is very good,

but it is not as high as in the previous examples, where the NRMSE is in the worst cases less than 1%. Again, this is because of the challenging discrete or piecewise characteristics of the multiparametric solution, which can be clearly noticed in Figure 4.8-(a).

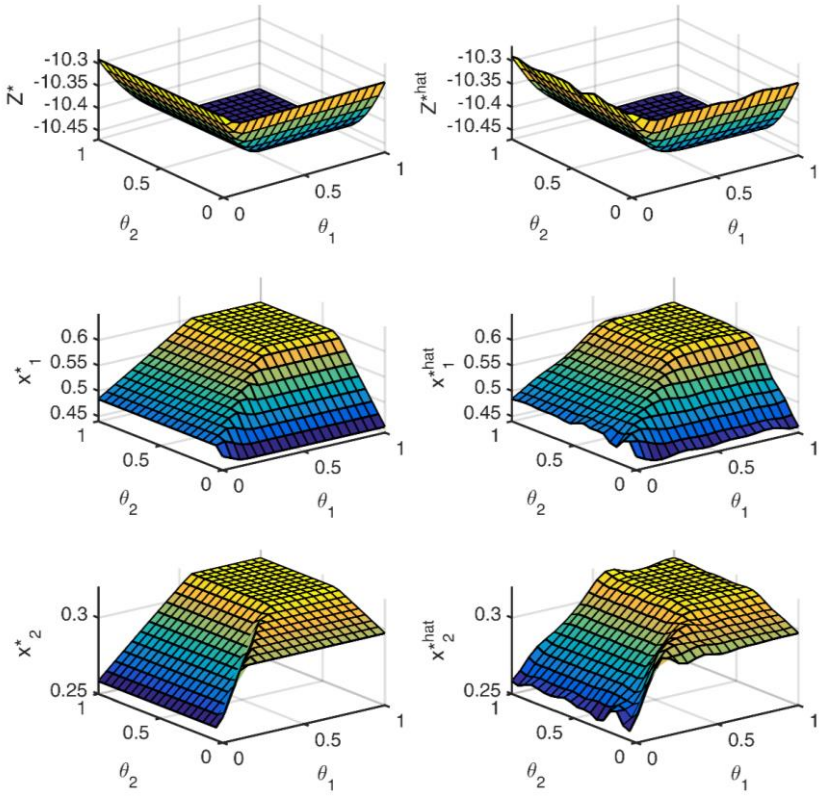


Figure 4.8. Nonlinear optimization problem: comparison between (a) the results obtained by the classical MPP solution provided in (Domínguez, et al., 2010) and (b) the results provided by the proposed methodology.

4.2.3.6 Case Study 1: operational cost optimization of utility system

A utility system (Figure 4.9) supplies mechanical energy to an industrial process is considered. The system is composed of a boiler (E-1) that receives water and supplies high pressure steam to a steam turbine (T1), whose outlet steam is condensated to water that is fed-back to the boiler inlet by the pump (P-1).

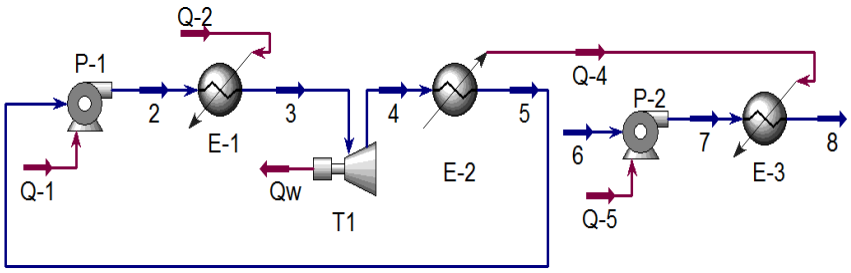
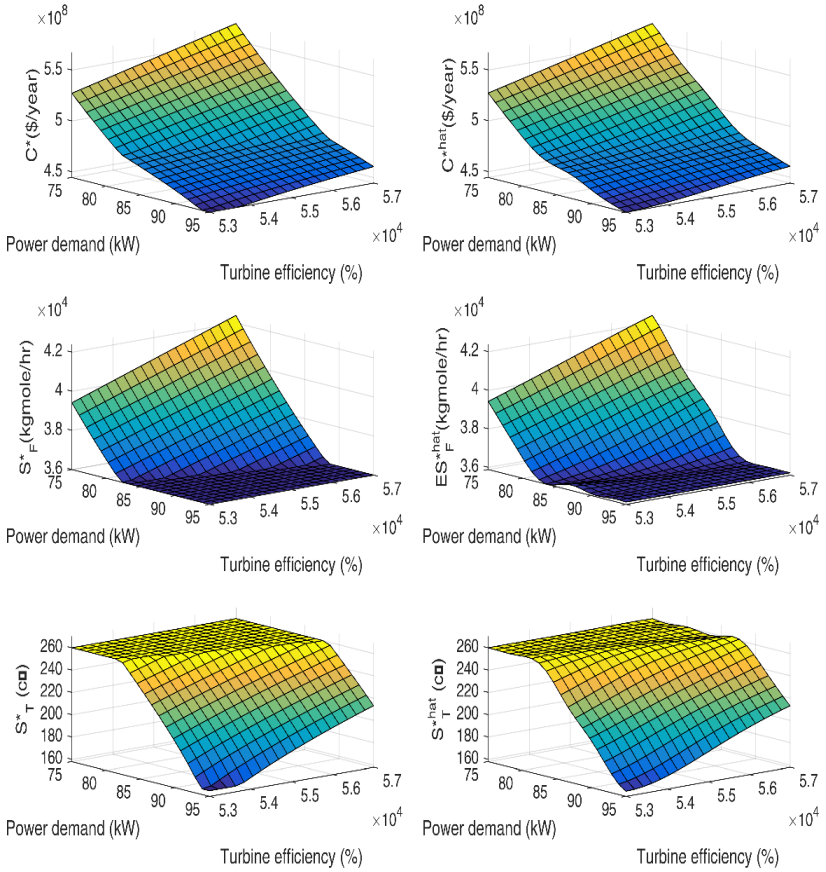


Figure 4.9. Utility plan model.

The objective is to minimize the system operational cost, C, that includes the costs of energies, Q2, Q1 and Q5, consumed by the boiler, T1, and pumps, E-1 and E-2, respectively, and the cooling water cost. The operational cost is modeled as a function of the boiler outlet steam flowrate (SF) and temperature (ST). However, two UPs, θ_1 and θ_2, affect the system, which are: the power demand that must be satisfied by the turbine and varies between [53000, 57000] kW, and the turbine efficiency that varies between [75, 95] %. Such case study represents a difficulty for the classical MPP approaches, as it is a simulation-based optimization case, in which the simulation model (Figure 4.9) is a black-box one (ASPEN HYSYS modeling and simulation environment) that includes complex thermodynamics relations. The problem is formulated as:

$$\left.\begin{array}{c} \underset{S_F,S_t}{Min}\ C = f(S_F, S_t, \theta_1, \theta_2) \\ S.T: \\ the\ process\ model \\ Qw\ (S_F, S_t, \theta_2) \le \theta_1 \\ 36000 < S_F < 79200\ kgMole/hr\ , \qquad 162 < S_t < 360\ c\circ \end{array}\right\} \tag{4.11}$$

A sampling plan $[\theta_1, \theta_2]_{70\times2}$, is designed over the domain [53000:57000, 75:95]. The black-box simulation-based optimization problem (Eq.(4.11)) is solved 70 times (the "*fmincon*" Matlab optimizer is used) to obtain the optimal values of the objective and decision variables $[C^*, S_F^*, S_T^*]_{70\times3}$. Using this dataset, three MPMs are trained, which approximate the optimal behavior of the operational cost, steam flowrate and temperature as a function of the power demand and turbine efficiency: $\hat{C}^* = f_0(\theta_1, \theta_2)$, $\hat{S}_F^* = f_1(\theta_1, \theta_2)$, $\hat{S}_T^* = f_2(\theta_1, \theta_2)$. The validation is accomplished using another dataset of size 400 samples, $[\theta_1^v, \theta_2^v]_{400\times2} - [C^{*,v}, S_F^{*,v}, S_T^{*,v}]_{400\times3}$, where the three MPMs are employed to predict the outputs $[\hat{C}^{*,v}, \hat{S}_F^{*,v}, \hat{S}_T^{*,v}]_{400\times3}$ and the NRMSE is calculated (Table 4.1).



(*a*) (*b*)

Figure 4.10. (a) Exact versus (b) approximate multiparametric behavior of the utility system.

Table 4.1shows the high potential of the method: the optimal decisions are accurately predicted via simple interpolations using the MPMs, in a very slight time (0.2 sec), saving a large amount of time consumed by the real simulation-based optimization (300.8 sec). Thus, the method represents a powerful tool to promptly manage the UPs variations during the process online operations.

4.2.3.7 Case Study2: operational optimization of a batch reactor

The second case study (Hale & Qin, 2004) considers a hypothetical scenario in which an engineer is charged with starting up a new chemical process based on the reactions:

$$A \rightarrow B, \qquad r_1 = k_1 C_A, \qquad k_1 = 0.05\ s^{-1}$$

$$B \rightarrow C, \qquad r_2 = k_2 C_B, \qquad k_2 = \theta_1 k_1\ s^{-1}$$

$$A \rightarrow C, \qquad r_3 = k_3 C_A, \qquad k_3 = \theta_2 k_1\ s^{-1}$$

where A is the reactant, B is the desired product, C is a secondary undesired product, $r_i, i = 1,2,3$ are the respective reaction rates, $k_i, i = 1,2,3$ are the reaction rate constants, C_A is the reactant concentration and C_B is the desired product concentration. It has

been assumed that the reaction constants, k_2, k_3 of the two side-reactions, are not precisely estimated with UPs θ_1 and θ_2. The process is to be run in a batch reactor with maximum capacity $V_{max} = 1000L$, on which an automatic feed and emptying system is installed, with flowrate $0 \leq F \leq F_{max} = 1\ L/s$. The process can be operated in two scenarios:

- A complete batch mode, where the process starts with the reactor full of reactants, i.e., the initial volume equals to the reactor maximum capacity: $V_0 = V_{max} = 1000L$.
- A hybrid mode of fed-batch then batch, according to which the process starts with initial volume, V_0, of the reactant less than the maximum capacity, i.e., $V_0 < V_{max}$. Then, the rest of the reactant volume ($V_{max} - V_0$) is continuously fed into the batch with constant flowrate, F, until the time instance τ at which the reactor is full ($V(\tau) = V_{max}$), the flow is shut-off and, then, the process continues in a batch mode.

Given that the duration of the batch is T, and the time instance $\tau = (V_{max} - V_0)/F$ (i.e., for a full batch mode scenario $\tau = 0$), the process can be modeled as:

for the time period $0 \leq t < \tau$

$$\frac{dC'_A}{dt} = C_{A0}\,F - (1+\alpha_2)\ \ k_1 C'_A\,, \qquad C'_A(0) = C_{A0}V_0$$
$$\frac{dC'_B}{dt} = k_1 C'_A - \alpha_1 k_1 C'_B, \qquad C'_B(0) = 0$$
$$\frac{d\,V'}{dt} = F \qquad V'(0) = V_0$$

for the time period $\tau \leq t \leq T$

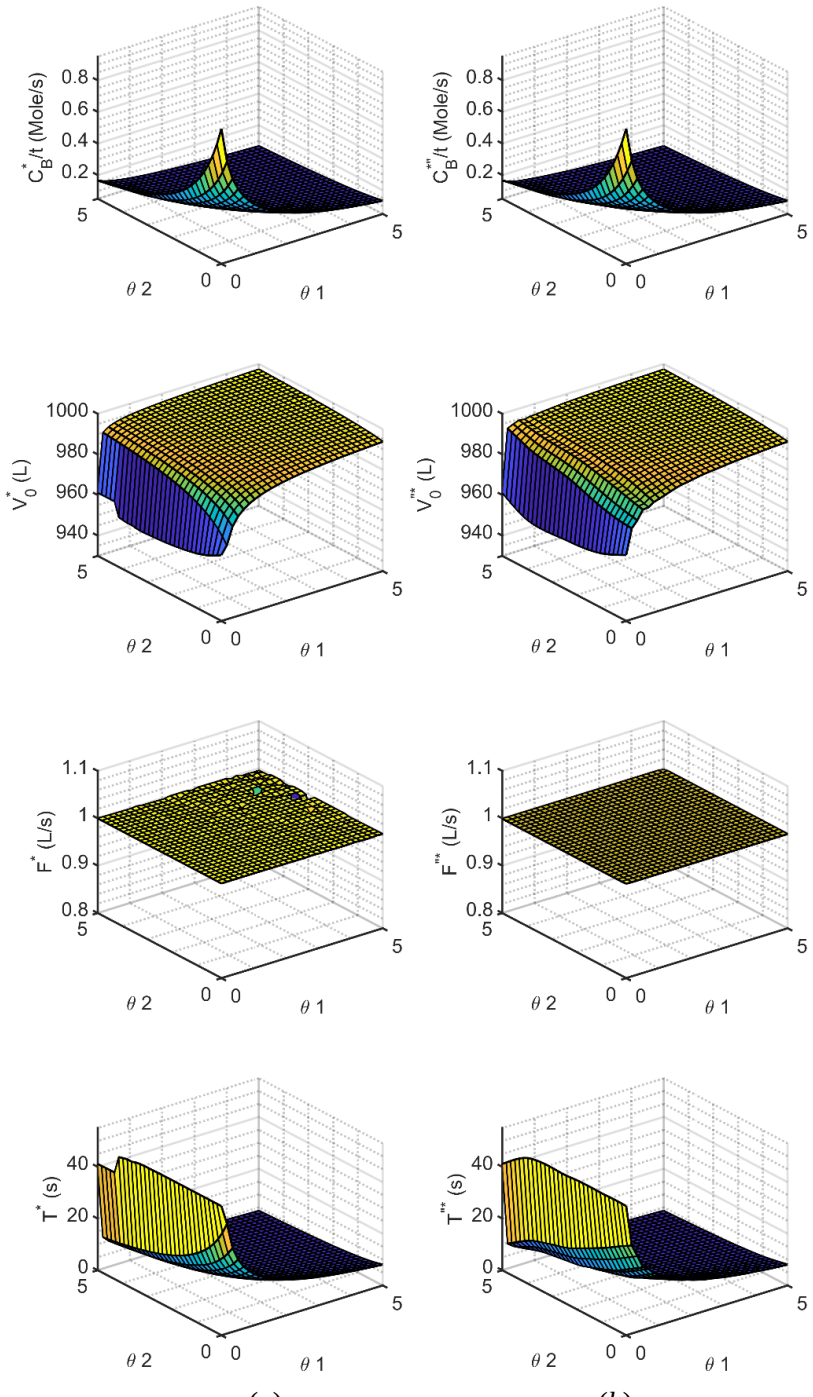
$$\frac{dC_A}{dt} = -(1+\alpha_2)\ \ k_1 C_A\,, \qquad C_A(\tau) = C'_A(\tau)$$
$$\frac{dC_B}{dt} = k_1 C_A - \alpha_1 k_1 C_B, \qquad C_B(\tau) = C'_B(\tau)$$
$$\frac{dV}{dt} = 0 \qquad V(\tau) = V'(\tau)$$

where the dash superscript is used to distinguish the process variables during the fed-batch period, and $C_{A0} = 2\ M$ is the concentration of feed. The process total time, T_P, is the summation of the time required to fill the reactor with the initial volume of reactant, V_0, plus the batch duration, T, plus the time to empty the batch, i.e., $T_P = \frac{V_0}{F_{max}} + T + \frac{V_{max}}{F_{max}}$. The objective of the batch operation is to select the optimal values of the decision variables V_0, F, T that maximize the amount of the desired product B produced per unit time, considering bounded uncertain parameters θ_1 and θ_2.

$$\left.\begin{array}{c} \underset{V_0,T,\mathrm{F}}{\mathrm{Max}}\ C_B\,(V_0\,,\mathrm{F},T)/T_P \\ S.T.: \\ the\ process\ model \\ 0 \le F \le F_{max} \\ 0 \le V_0 \le max \\ 0 \le \tau \le T \\ 0 \le \theta_1\,,\theta_2 \le 5 \end{array}\right\} \qquad (4.12)$$

Before the application of the proposed method, the correctness of the optimization procedure is checked by solving the problem considering the nominal values of the UPs $\theta_1 = 0.05$ and $\theta_2 = 0,$ as in (Hale & Qin, 2004), and exactly the same solution of the nominal problem (${V_0}^* = 978.7$ L , $F^* = 1\ L/s$, $T^* = 27.7\ s$) is obtained.

The proposed method is straightforwardly applied with the same steps previously illustrated. A sampling of 150 points is designed, the optimization problem (Eq.(4.12)) is solved (using the Matlab "*fmincon*" algorithm) to obtain the input-output training data $[\theta_1, \theta_2]_{150\times2} - [C_B^*, {V_0}^*, F^*, T^*]_{150\times4}$ and, finally, four MPMs, $C_B^* = f_0(\theta_1, \theta_2)$, ${V_0}^* = f_1(\theta_1, \theta_2)$, $F^* = f_2(\theta_1, \theta_2), T^* = f_3(\theta_1, \theta_2)$, are trained. The MPMs validation is accomplished in the same way as previously mentioned, considering a new input-output dataset $[\theta_1^v, \theta_2^v]_{300\times2} - [C_B^{*,v}, {V_0}^{*,v}, F^{*,v}, T^{*,v}]_{300\times4}$ different from the training one.



(*a*) (*b*)

Figure 4.11. (a) Exact versus (b) approximate multiparametric behavior of the batch reactor.

The performance, shown in Figure 4.11 and reported in Table 4.1, further emphasizes the methodology capabilities in terms of high prediction accuracy and significant reduction in the optimization time (99.92% = (4887-3.88)/4887). Notice that the optimal flowrate, $\hat{F}^*$, is insensitive to the UPs variation (see Figure 4.11) and always takes the maximum allowable value, which make sense because using the maximum flowrate minimizes the time of filling and emptying the reactor and, consequently, maximizes the objective function (maximum

desired product B produced per unit time). Also, since the variability range of $\hat{F}^*$ is almost zero, the calculation of its NRMSE is meaningless (see Eq.(4.6)), as it will lead to an extremely high "numerical" value that is not expressing the actual performance of the MPM. Therefore, to evaluate the performance of the MPM of $\hat{F}^*$, we consider the RMSE, which equals to 0.0039.

Table 4.1. NRMSE (%) of the MPMs and the computational time of their training and validation.

Problem	MPMs	Training				Validation			NRMSE
		No. of training samples	Time(sec)*			No. of validation samples	Time (sec)*		
			Ups Sampling	Optimization for data generation	Fitting		Optimization for data generation	Prediction	
Refinery blending problem	$\hat{Z}^*$	75	0.002	0.53	0.86	400	2.1	0.51	0.44
	$\hat{x}_1^*$				0.80				0.39
	$\hat{x}_2^*$				1.13				0.44
Bilinear objective with linear constraints	$\hat{Z}^*$	60	0.012	6.9	0.53	150	16.2	0.17	0.01
	$\hat{x}_1^*$				1.08				0.80
	$\hat{x}_2^*$				1.56				0.40
Quadratic optimization	$\hat{Z}^*$	80	0.015	0.815	5.76	400	4.51	1.49	0.00011
	$\hat{x}_1^*$				2.52				0.02813
	$\hat{x}_2^*$				4.46				0.01853
Quadratic optimization: milk surplus problem	$\hat{Z}^*$	80	0.023	1.07	1.47	400	8.17	1.59	0.00350
	$\hat{x}_1^*$				1.52				0.00020
	$\hat{x}_2^*$				1.42				0.00016
	$\hat{x}_1^*$				1.39				0.00021
	$\hat{x}_2^*$				155				0.00010
Nonlinear optimization	$\hat{Z}^*$	140	0.039	4.67	4.60	400	13.85	1.60	0.62
	$\hat{x}_1^*$				4.00				1.5
	$\hat{x}_2^*$				6.00				1.9
Operational cost minimization of utility plant	$\hat{C}^*$	70	0.015	48.4	1.20	400	300.8	0.20	0.31
	$\hat{S}_F^*$				1.40				0.61
	$\hat{S}_T^*$				1.07				0.77
Operational cost minimization of batch reactor	$\hat{C}_B^*$	150	0.038	861	14.0	300	4887	3.88	0.08
	$\hat{V}_0^*$				17.5				1.50
	$\hat{F}^*$				0.66				*NA*
	$\hat{T}^*$				11.1				1.45

Finally, the Table 4.1, as a whole, also shows that the method advantage increases as the optimization problem complexity increases: in examples 2 and 3, a quadratic optimization problem is quite easy to solve (single global optima), and the percentage of the time saved using the method is 67% (100 ×(4.5-1.49)/4.5) and 80% respectively. However, as the problem complexity increases in example 1 (bilinear objective function including saddle behavior) and in example 5, the percentage of the time saved increases to 98.7% and 88.5% respectively. Finally, when the optimization problem involves a complex, nonlinear and/or black-box model (case studies) the amount of the saved time reaches 99.9%.

4.3 MPMS FOR MIXED-INTEGER OPTIMIZATION

4.3.1 Problem statements

A general Mixed Integer optimization problem under uncertainty can be generally expressed as in Eq.(4.13), where Z is the objective function, $g_l(x), l = 1,2,\dots,L$ represents a set of constraints, $\theta \in R^k$ is a set of the bounded UPs affecting the problem, $y \in \{0,1\}^{K_{int}}$ is a vector of binary variables and $x \in R^{K_{cnt}}$ is a vector of continuous variables (Dua & Pistikopoulos, 1999; Pistikopoulos, 1995).

$$\left.\begin{array}{c} \min\limits_{x,y} Z = f(x,y,\theta) \\ S.T.\ \ g_l(x,y,\theta) \le 0, \quad l = 1,2,\dots.L \\ lb_x \le x \le ub_x, \qquad lb_\theta \le \theta \le ub_\theta \\ x \in R^{K_{cnt}}, \qquad y \in \{0,1\}^{K_{int}}, \qquad \theta \in R^k \end{array}\right\} \tag{4.13}$$

The application of MPP approaches provides the optimal solution (Z^*, x^*, y^*) as P mathematical functions of the UPs (Eq.(4.14)), each one is valid for a certain partition of the UPs space which is called "*critical region*". So, as the UPs vary, the optimal solution is calculated by evaluating these simple functions, and avoiding repeated optimization tasks (Pistikopoulos, et al., 2002).

$$\begin{array}{c} Z_p^* = f_p\ (\theta), \qquad x_{i,p}^* = f_{i,p}(\theta), \qquad y_{j,p}^* = f_{j,p}(\theta) \\ i = 1,2,\dots K_{cnt}, \qquad j = 1,2,..K_{int} \quad , \quad p = 1,2,..P \ , \ lb_{\theta_p} \le \theta \le lb_{\theta_p} \end{array} \tag{4.14}$$

The solution of this problem is beyond the capabilities of the first methodology, because of two reasons:

1) Molding the multiparametric behavior of the optimal integer variables $y \in \{0,1\}^{K_{int}}$ implies the use of a totally different tool (*classification* technique) rather than metamodels, which should be able to approximate binary or categorical outputs.

2) The existence of the integer variables $y \in \{0,1\}^{K_{int}}$ adds extra discrete features to the multiparametric behavior of the optimal continuous variables $x \in R^{K_{cnt}}$, so auxiliary techniques (*clustering*) should be used in order to assist the metamodels to capture this high discrete behavior.

Therefore, this part is focused on extending the capabilities of the first method in order to be able to address Mixed-Integer optimization models. Before describing the methodology details, it is worth highlighting the basics of the classification and clustering techniques.

- *Classification* techniques allocate an observation into a specific class among a set of predefined classes. In this framework, they are used to construct pattern recognition models that characterize the optimal values of the binary/integer variables as a function of the UPs. The classifier should be trained using an input-output dataset $[\theta, y]_n$, $\theta \in R^k$, $y \in R^{int}$, where y is a categorized variable consisting of specific classes (e.g. 0 or 1). Artificial Neural Network (ANN) classifiers are used in this work (Matlab ANN toolbox) due to their high generalization properties and efficiency.
- *Clustering* methods allow identifying groups (clusters) of data/results, associating into each single cluster similar samples according to some performance measure. The *K*-means clustering method (Matlab "*kmeans*" function) is used due to its simplicity (Kaufman & Rousseeuw, 1990).

4.3.2 Methodology

4.3.2.1 UPs sampling using DOCE

In order to obtain accurate predictions, the training data should include -as much as possible- information about the optimal solution behavior in every sub-region of the total input (UPs) space. So, DOCE methods are used to select an input sampling plan $[\theta]_n, \theta \in R^k$ including sufficient combinations of the UPs values that uniformly span the whole input domain $lb_\theta \leq \theta \leq ub_\theta$. A hybrid technique of Hammersley sequencing and full factorial design is proposed, due to its uniformity and simplicity (Ibrahim, et al., 2019).

4.3.2.2 Data generation and feasibility modelling

After designing a sampling plan $[\theta]_{n\times k}$, the optimization problem in Eq.(4.13) is solved *n* times, each one using a different combination of the UPs values in order to obtain the corresponding optimal output $[Z^*, x^*, y^*]_n$, and additional Feasibility Information (FI): $n = n_f + n_{inf}$, where n_f is the number of feasible samples and n_{inf} is the number of optimization runs leading to an unfeasible situation. The optimization model was written in GAMS 23.8.2 and solved using the solver DICOPT, which combines solvers of the sub-problems nonlinear programming using CONOPT and Mixed Integer programming using CPLEX. Then, an ANN classifier $\widehat{FI} = g^{FI}(\theta)$ is constructed using the UPs values and the feasibility labels $[\theta, FI]_n$ in order to examine the optimization problem feasibility for new UPs values.

4.3.2.3 Integer variables modelling

For each integer variable y_i^* , a classifier $\hat{y}_i^* = g^{y_i}(\theta)$ is constructed and trained using the UPs feasible inputs, and the corresponding output data of the optimal solutions, i.e. $[\theta,\ y_i^*]_{n_f}$. Thus, the resulting set of K_{int} classifiers can be used to identify the optimal values of the integer variables for any further change in the UPs.

4.3.2.4 Continuous variables modelling

The optimal values of the continuous variables x^* are explored, processing them by the clustering algorithm, in order to identity the $k_{cnt'}$ continuous variables whose multiparametric behavior shows a continuous response over the UPs space ($x_i^*, i = 1,2..k_{cnt'}$). Besides, additional $k_{cnt''}$ continuous variables that show discrete/piecewise multiparametric behavior over the UPs domain are also identified, being $k_{cnt} = k_{cnt'} + k_{cnt''}$.

The optimal values of each variable $[x_i^*]_{n_f}$ are processed by the K-means algorithm, specifying different values for the number of clusters K, as there is no prior knowledge about the number of distinct behaviors really present in the system. Each time (i.e.: for each feasible situation) the clustering quality is assessed via the average silhouette value measure (Kaufman & Rousseeuw, 1990): high values indicate high probabilities of distinct clusters existence, and also righteousness of the assumed number of clusters K, while lower values indicate no sharp piecewise behavior.

For each one of the continuous variables with a continuous multiparametric behavior, a metamodel $\hat{x}_i^* = f_i^{cnt}(\theta)$ is trained, so as to obtain the set of $k_{cnt'}$ metamodels.

Regarding each of the continuous variables with discrete multiparametric behavior, once the best clustering is identified (K), a classifier $\hat{j}_{xi} = g^{x_i}(\theta)$ is constructed using the input information of the UPs $[\theta]_f$ and the labels obtained from the clustering step as the output. In parallel, K metamodels $\hat{x}_{ij}^* = f_{ij}^{cnt}(\theta), j = 1,2,..,$K are fitted, each one trained using the data of each cluster $[\theta_j, x_{ij}^*]_{n_{f,j}}$.

Hence, the classifier-metamodels system is employed as follows: for a new value of a certain UPs combination, the classifier is used to decide to which one of the K clusters or behaviors (i.e. metamodels) the UPs values belong to. And then, the specific metamodel associated to the identified cluster – i.e. the one that describes a distinct multiparametric behavior over a certain (local) area of the total UPs space- is used to estimate the optimal values of the continuous variable. The same procedure is also applied to explore the multiparametric behavior of the objective Z^*.

4.3.3 Application

A case study from the work by (Dua & Pistikopoulos, 1999) has been used to illustrate the application of the proposed method. In this case (Figure 4.12), a product *C* is manufactured from a chemical *B*, where the latter can either be purchased from the market or manufactured from a chemical *A* by two different alternatives. The supply of *A* and the demand of *C* are represented by UPs $[\theta_1, \theta_2]$ affecting the problem as illustrated in Figure 4.12.

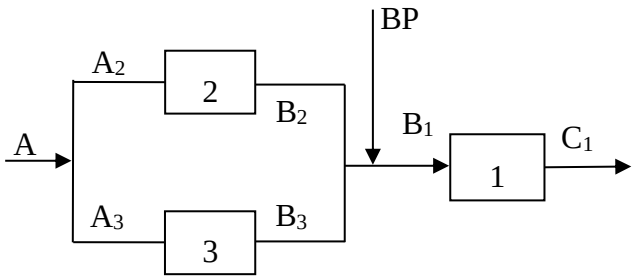


Figure 4.12. Case-study illustration.

The optimization problem is formulated as follows:

$$Z(\theta) = \min_{x,y}(3.5y_1 + y_2 + 1.5y_3 + B_2 + 1.2B_3 + 1.8(A_2 + A_3) + 7BP - 11C_1)$$

$$S.T.: -0.9B_1 + C_1^2/15 \leq 0,$$
$$-A_2 + exp(B_2) \leq 1,$$
$$-A_3 + exp(B_3/1.2) \leq 1,$$
$$-A + A_2 + A_3 = 0,$$
$$-B_2 - B_3 - BP + B_1 = 0,$$
$$BP \leq 1.95,$$
$$y_1 + y_2 + y_3 \leq 2,$$
$$C_1 \leq 20y_1,\ B_2 \leq 20y_2,\ B_3 \leq 20y_3, A_2 \leq 20y_2, A_3 \leq 20y_3$$
$$0.50 \leq (\theta_1 = A) \leq 0.75,\quad 5.50 \leq (\theta_2 = C_1) \leq 6.00$$

Over the inputs (UPs) domain [0.5: 0.75; 5.5: 6], a sampling plan $[A, C]_{250}$ is designed, and the optimization problem is solved 250 times, to obtain the optimal variables and objective $[A^*, B1^*, BP^*, C1^*, A2^*, A3^*, B2^*, B3^*, Z^*, y1^*, y2^*, y3^*]_{250}$ (outputs). From the 250 optimization runs, 172 input-output samples are found feasible (Figure 4.13-(b)).

Using the training data ($[A, C]_{250}$–[FI]$_{250}$), an ANN classifier $\widehat{FI} = g^{FI}(A, C)$ is trained which is used to check the problem feasibility for new values of *A* and *C*. And using the feasible samples ($[A, C]_{172} - [y1^*, y2^*, y3^*]_{172}$) three ANN classifiers are trained ($\hat{y}1^* = g^{y1}(A, C)$, $\hat{y}2^* = g^{y2}(A, C)$, $\hat{y}3^* = g^{y3}(A, C)$), to define the optimal value of each integer variable as a function of the UPs (*A* and *C*).

On the other side, the K-means algorithm is used to explore each of the continuous variables and the objective, in order to look for any eventual cluster -i.e. distinct local multiparametric

behaviors. For each variable, different values of Ƙ are tested, and the obtained clustering qualities are assessed (Figure 4.13-(a)): variables $[A^*, B1^*, BP^*, C1^*, Z^*]$ show relatively lower silhouette values, and the change of K does not significantly affect their silhouette values. So, these variables (including the objective function values) do not show severe discrete multiparametric behavior over the UPs feasible domain. Thus, five OK models are fitted, one for each of these variables: $[\hat{A}^* = f_1^{cnt'}(A, C), \ldots\ldots \hat{Z}^* = f_5^{cnt'}(A, C)]$.

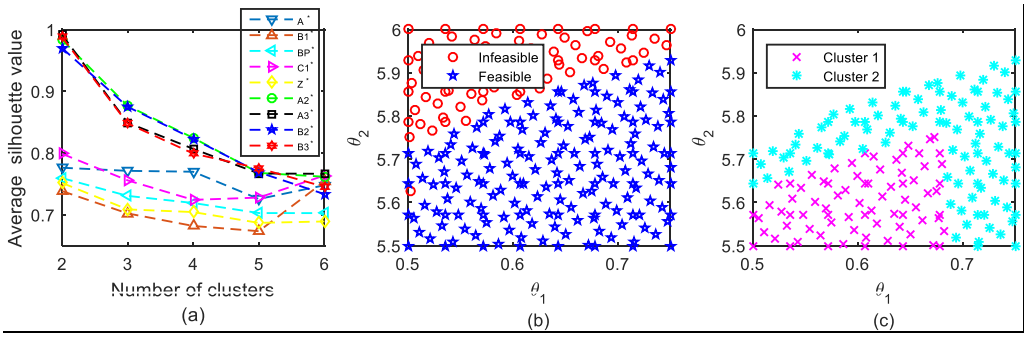


*Figure 4.13. (a) Clustering analysis, (b) Feasibility information and (c) B3*clustering.*

In contrast, Figure 4.13 -(a) also shows how the variables $[A2^*, A3^*, B2^*, B3^*]$ exhibit higher silhouette values, which significantly depend on the value of Ƙ dramatically. Thus two (Ƙ=2) distinct clusters over the UPs feasible space are characterized. Figure 4.13-(c) shows the clustering results of $B3^*$. The multiparametric behaviors of the other variables $[A2^*, A3^*, B2^*]$ show similar cluster patterns, so finally, a classifier is trained for each of the four variables using the UPs and their corresponding labels obtained from the clustering step ($\hat{\jmath}_{A2} = g^{A2}(A, C)$, ..., ($[A, C]_{172}$–[clustering labels]$_{172}$).

Besides, two additional metamodels are trained for each one of these variables, each using the data associated to the specific cluster, in order to approximate the multiparametric behavior at each UPs local area: $[\widehat{A2}_1^* = f_{1,1}^{cnt''}(A, C), \quad \widehat{A2}_2^* = f_{1,2}^{cnt''}(A, C), \ldots]$.

In order to validate the proposed system, a different dataset with 400 samples is generated, and the trained system of classifiers and metamodels is used to predict the optimal solutions of the validation set. Figure 4.14-(a) shows the estimated feasibility labels (crosses and triangles) using the feasibility classifier g^{FI}compared to the exact ones (squares and circles). Additionally, Figure 4.14-(b) presents the estimated optimal solutions of $\widehat{y2}^*$ (crosses and triangles) using the classifier g^{y2}, compared to their exact values (squares and circles). Figure 4.14-(c, d) displays the exact multiparametric behavior of $B3^*$ and the estimated one using the metamodels $\widehat{B3}_1^* = f_{4,1}^{cnt''}(A, C)$, $\widehat{B3}_2^* = f_{4,2}^{cnt''}(A, C)$ assisted by the classifier $\hat{\jmath}_{B3} = g^{B3}(A, C)$.

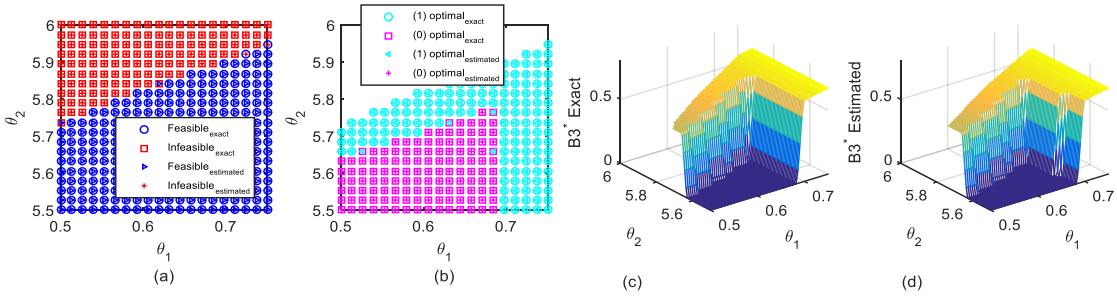


Figure 4.14. Validation: (a) Feasibility classifier, (b) y2 classifier and (c) $\widehat{B3}^*$ multiparametric behavior.

Table 4.2 reports the accuracy values of all the classifiers (feasibility and integer variables) in terms of their $f1$-score, and the accuracy of the predicted multiparametric behavior of the continuous variables (metamodels) in terms of the Normalized Root Mean Square Error (NRMSE). The metamodels of the variables that show continuous multiparametric behavior ($[\hat{A}^*, \widehat{B1}^*, \widehat{BP}^*, \widehat{C1}^*, \hat{Z}^*]$) are very accurate. However, some of the metamodels of the continuous variables behaving in a piecewise manner ($\widehat{A2}_2^*$ $\widehat{B2}_2^*$, $\widehat{A3}_2^*$, $\widehat{B3}_2^*$) show poor accuracies, although this is not evident in the visual comparison to the exact multiparametric behavior (Figure 4.14-(c,d)).

Table 4.2. f1 score (%) of the classifiers and NRMSE (%) of the metamodels.

Classifiers (ANN)	$\widehat{Fl}$	$\widehat{y1}^*$	$\widehat{y2}^*$	$\widehat{y3}^*$									
f1-score (%)	99.0	100.0	98.5	97.0									
Metamodels (OK)	$\hat{A}^*$	$\widehat{B1}^*$	$\widehat{BP}^*$	$\widehat{C1}^*$	$\hat{Z}^*$	$\widehat{A2}_1^*$	$\widehat{A2}_2^*$	$\widehat{B2}_1^*$	$\widehat{B2}_2^*$	$\widehat{A3}_1^*$	$\widehat{A3}_2^*$	$\widehat{B3}_1^*$	$\widehat{B3}_2^*$
NRMSE (%)	1.2	1.1	4.8	0.1	0.5	0.4	21.5	0.0	17.0	0.4	20.7	0.0	20.1

Actually, this shortage is not directly related to the metamodels themselves, but to the critical regions classifiers $g^{A2}, g^{A3}, g^{B2}, g^{B3}$, which are not accurate enough at the limits between the identified clusters (UPs local regions). Thus, a classifier (e.g. g^{B3}) may incorrectly select a metamodel (e.g. $f_{4,2}^{cnt''}$ instead of $f_{4,1}^{cnt''}$), leading to a totally different estimated multiparametric behavior and, consequently, to a significant metamodel prediction error. Even when the misclassified points are few (7/271), the effect in the NRMSE is significant.

Figure 4.15-(a) shows the estimated multiparametric response of $B3^*$, composed by the local behaviors $\widehat{B3}_1^*$ (red circles) and $\widehat{B3}_2^*$ (blue circles), and the effect of misclassification. In the same way, Figure 4.15-(b,c) shows the absolute error of the estimated values, including the very few points showing significant errors, again due to this mentioned misclassification.

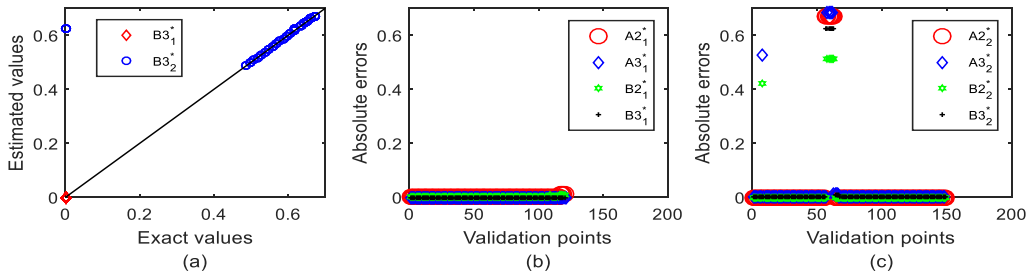


Figure 4.15. Validation: (a) $\widehat{B3}^*$ versus $B3^*$, (b) and (c) absolute errors of $[\widehat{A2}^*, \widehat{A3}^*, \widehat{B2}^*, \widehat{B3}^*]$.

Finally, Table 4.3 indicates the computational effort required in this case for data generation, training of the involved machine learning techniques (classification, clustering and metamodels), and the validation/prediction time. The table shows that the computational effort required for predicting the optimal solution corresponding to certain UPs using the proposed method is about 1/500 000 of the effort required to perform an optimization run.

Table 4.3. Computational time (i5-6200U CPU@2.3GHz).

Problem	***Training***						***Validation***		
	No. of training samples	*Time(sec)**					*No. of validation samples*	*Time (sec)**	
		UPs sampling	*Optimization for data generation*	*Fitting*				*Optimization for data generation*	*Prediction*
				classifiers	*clustering*	*metamodels*			
Case study	250	0.04	1250	1.52	10.0	107.5	400	4000	0.006

4.4 CONCLUSIONS

This chapter proposes two efficient machine learning-based methods for multiparametric solution of general optimization problems subjected to uncertainties, with special emphasis on chemical processes operation optimization problems. The first method addresses continuous optimization problems, while the second targets Mixed-Integer problems. The methods combine different tools and techniques as DOCE, state-of-art optimization algorithms, machine learning models for regression (i.e., surrogate models), clustering techniques and machine learning models for classification.

The methods have been tested on several benchmark MPP examples including linear, bilinear, quadratic and nonlinear problems and applied to three case studies of process operation optimization. The results show that the methods are able to approximate the multiparametric solutions using a relatively small number of training data, with very good accuracy. More importantly, significant differences with the results of the standard MPP appear; 1) in all the tested cases, a single (or maximum two) MPMs was enough to correctly reproduce the multiparametric behavior of the optimal solution over the whole UPs domain, instead of several mathematical function (provided by classical MPP approaches) each is applicable to a certain partition of the UPs space, 2) the methods are able to solve problems of

different types (linear, bilinear, quadratic, nonlinear) in a systematic and robust way, instead of many classical MPP algorithms each is able to solve one specific type and 3) the methods are capable of solving process operation optimization problems where complex, black-box and/or highly nonlinear models are used, providing a huge reduction in the online optimization time. With respect to the second method, modeling the multiparametric behavior of the continuous variables that show severe changes over the UPs domain is challenging: although clustering techniques help to define local metamodels, they suffer from poor approximation in the limits of the critical regions.

Future research will investigate the extension of the methodology capabilities for improving the modeling of the multiparametric behavior of continuous variables that show significant/discrete changes over the UPs space (e.g., the example in Section 4.2.3.5), the quantification of uncertainty in the MPMs predictions and also the suitability of the methods for other problem types.

Chapter 5: Data-Driven Explicit MPC of Chemical Processes

This Chapter proposes a Data-Based MultiParametric-Model Predictive Control (DBMP-MPC) methodology, which enables simple and efficient implementations of explicit MPC in situations when the deep mathematical knowledge required to develop traditional multiparametric MPC techniques is not available. Additionally, it represents a powerful alternative in cases when it is difficult to apply traditional multiparametric MPC due to the complexity and/or high nonlinearity of the process First Principle Model (FPM). The proposed method builds machine learning models (kriging, Support Vector Regression (SVR) or Artificial Neural Networks (ANNs)), which are trained offline using input-output information, to approximate the optimal values of the control variables that must be applied the next sampling period as a function of the state variables value at the current sampling period. Then, during the online application, the optimal control is calculated through simple interpolations using these machine learning models, avoiding the need for solving the open-loop optimal control problem. The method is tested on benchmark problems adopted from multiparametric MPC literature. The results show high accuracy and robustness using a simple method, bypassing complex mathematical formulations.

5.1 INTRODUCTION

MultiParametric Programming (MPP) is an efficient tool widely used to proactively manage the uncertainty in some of the process parameters, avoiding the need to re-run the optimization model when the uncertainty is unveiled (Pistikopoulos, et al., 2002). A remarkable millstone in the MPP development is its incorporation to MPC (Kouramas, et al., 2011): a nominal process model (Eq.(5.2)) is used to control the process in a receding horizon fashion over a finite time horizon (sampling period). The optimal manipulated inputs $u(t)$ that optimize the process (desired performance or economic criteria, Eq.(5.1)) are predicted through solving a dynamic optimization problem (open loop optimal control), using the values of the measured states of the previous sampling period as the initial values for the system. The calculated optimal inputs are implemented for the next sampling period, and at its end, the state variables $x(t)$ are measured and their values are used to set up the next optimal control problem over the next sampling period, and so on (Tenny & Rawlings, 2004).

$$\min_{u} J(u, x_t) = x'_{t+N}\, P\, x_{t+N} + \sum_{k=0}^{N-1} [x'_{t+k}\, Q\, x_{t+k} + u'_{t+k}\, R\, u_{t+k}] \quad (5.1)$$

$$S.T.: \; x_{(t+1)} = A\, x_{(t)} + B\, u_{(t)}\,, x_j \in R^k, u_i \in R^K \quad (5.2)$$

$$g_l(x_{(t)}, u_{(t)}) \leq 0\,, \; l = 1,2,\dots L \quad (5.3)$$

But MPC technology faces a common obstacle associated to the time (computational effort) required to repeatedly solve the online open loop control problem at each sampling period, which may become infeasible for fast dynamic systems and/or when the optimization problem is complex (highly nonlinear / high size). The usage of the multiparametric MPC framework would provide a smooth solution, since it shall identify - offline - the optimal control strategy as *P* explicit simple mathematical functions of the state variables (Eq.(5.4)), each one is valid for a certain partition of the state variables space ("*critical region*") (Katz, et al., 2020). During the online MPC application, the optimal control values are calculated via simple evaluation of these functions, avoiding the need for online optimization.

$$J_p^* = f_{0p}(x_j)\,, U_{ip}^* = f_{ip}(x_j)\,, \qquad lb_{x_j p} \leq x_j \leq lb_{x_j p}, p = 1,2,\dots P \quad (5.4)$$

However, further to the complex mathematical knowledge required to develop the MPP analysis; the availability of a clear discrete-time linear state-space model of the process is usually a necessity for the practical application of the multiparametric MPC (Pistikopoulos, et al., 2002; Pistikopoulos, et al., 2007) . This, again, may hinder the multiparametric MPC usage in cases where the available process model is highly nonlinear, high dimensional, with a complicated structure (sequential simulation models), and/or non-transparent (black box) (Rivotti, et al., 2012; Medina-González, 2019). Recently, model approximation and order reduction techniques have been proposed (Rivotti, et al., 2012); however, this may over simplify the processes behavior and, consequently, degrade the controller performance; additionally, the effort dedicated to this model simplification step should be also considered. Other works (Medina-González, 2019) have proposed data-based multiparametric analysis techniques that can be only used in such situations for design and steady state optimization problems.

This Chapter proposes DBMP-MPC methodology aimed to achieve two goals: the first is to enable rapid and ease implantations of multiparametric MPC in situations when the deep mathematical knowledge required to develop it based on traditional approaches is not obtainable, even when a clear discrete-time linear state-space model is available. The second is the assistance in situations where it is difficult to apply traditional multiparametric MPC techniques, due to the complexity and/or nonlinearity of the available process dynamic FPM.

5.2 METHODOLOGY

The proposed methodology is based on the use of machine learning techniques which are trained offline using input-output data (initial state variables vs. optimal control), to obtain metamodels that approximate the optimal control of the future sampling period as a function of the initial state variables ("*MultiParametric Metamodels*" (MPMs)). Then, during the online application, the optimal control is calculated through simple interpolations using these MPMs, avoiding the need for the dynamic optimization. The most significant elements of this general procedure and some application details corresponding to its implementation over the cases presented in Section 5.3, include the input-output data generation and the employed modeling techniques:

The input-output data for the MPMs training are generated offline, by solving the open loop optimal control problem several times, each using different combination of the initial state variables values, to find the corresponding optimal control results. To obtain accurate metamodel predictions, these data should include -as much as possible- information about the output (the optimal control) behavior in every sub-region of the whole input (initial state variables) space. Consequently, a sampling plan $[x]_n$ $(x \in R^K)$, should be designed to uniformly cover/span the metamodel whole input domain (Ibrahim, et al., 2019). In this work, a hybrid technique combining Hammersley sequence and fractional factorial design is used, because it achieves high uniformity with low computational cost. More details about DOCE techniques can be found in Section 2.1.

After designing the sampling plan, the open control problem is solved (n times), to obtain the outputs $[J^*, u^*]_n$, $(u \in R^K)$.

5.2.1 Ordinary kriging

Given a set of training data $[x_i, y_i]$, $i = 1,2,..n, x \in R^k, y \in R$, OK assumes a predictor $\hat{y}(x) = \mu_{ok} + Z(x)$, where $Z(x)$ is a deviation from the constant mean value μ_{ok}, and it is expressed as a stochastic Gaussian process with expected zero value, $E(Z(x)) = 0$, and a covariance $cov(Z(x_i), Z(x_j)) = \sigma_{ok}^2 R(x_i, x_j)$. Being σ_{ok}^2 the process variance and $R(x_i, x_j)$ a correlation function. The OK final predictor and estimated variance are given by Eq.(5.5) and Eq.(5.6), respectively. Where, $[r]_{n\times 1}$ is the correlations vector between the point to be predicted x^* and the training data. More details about the OK model can be found in Section 2.2.1.

$$\hat{y}(x^*) = \mu_{ok} + r^T R^{-1}(Y - 1\,\mu_{ok}) \tag{5.5}$$

$$\hat{s}^2(x^*) = \sigma_{ok}^2(1 + \lambda - r^T R^{-1} r + (1 - r^T R^{-1} r)^2/(1^T R^{-1} 1)) \tag{5.6}$$

5.2.2 Support vector regressions

Given a set of n input-output training data $[x_i, y_i]$, $i = 1,2,..n$, $x \in R^k, y \in R$, SVR (Vapnik, 1995) maps the input data original space into a high-dimensional feature space, often through a basis or kernel function $\Phi(x_i, x_j)$ that may be represented by different styles as linear, polynomial, Gaussian, etc. Then, the modeling problem becomes the determinations of the optimal (flattest) surface $\hat{y}(x) = b + \sum_{i=1}^{n} w_i \Phi(x_i, x_j)$ in this feature space that fits the data, through the minimization of the weights vector norm $|w|^2, w \in R^n$, where $b = \mu_{svr}$ is a base or bias (Forrester & Keane, 2009). The final predictor of the SVR is given by Eq.(5.7), where α_j^+, α_j^- are Lagrange multipliers resulting from the solution of the aforementioned minimization problem. The detailed mathematical description and derivations can be found in Section 2.2.3. This work uses the SVR algorithm based on the function "*fitrsvm*" included in the Matlab statistics and machine learning toolbox.

$$\hat{y}(x^*) = b + \sum_{i=1}^{n} (\alpha_j^+ - \alpha_j^-)\, \Phi(x^*, x_i) \tag{5.7}$$

5.2.3 Artificial neural networks

The ANNs are very well-known efficient machine learning models, which are widely used for data-driven modelling of nonlinear systems. In this work, the Matlab ANN toolbox and the function "*feedforwardnet*" have been used to create multilayer feedforward ANNs. In each of the following application cases, the number of layers, number of neurons and the training algorithm were selected based on a trial and error procedure in order to balance the ANN structure simplicity and its prediction accuracy. More details about ANNs can be found in Section 2.2.2.

5.3 APPLICATIONS

The methodology is illustrated with a benchmark problem widely used in the multiparametric MPC literature, and also through its application to a simulation case study. It is worth to mention that, in all the examples, the Matlab "*fmincon*" function is used as the optimization algorithm, with the computer capacities illustrated at the bottom of Table 5.1.

5.3.1 MPC of a discrete time state-space Model

The first application is an unconstrained MPC problem (Pistikopoulos, et al., 2002; Kouramas, et al., 2011) with the same formulations as Eq.(5.1) and Eq.(5.2), considering the state space model in Eq.(5.8).

$$x(t+1) = \begin{bmatrix} 0.732 & -0.0861 \\ 0.1722 & 0.9909 \end{bmatrix} x(t) + \begin{bmatrix} 0.0609 \\ 0.0064 \end{bmatrix} u(t), \qquad -2 \leq u(t) \leq 2 \quad (5.8)$$

Over an expected domain [-2: 2, -2:2] of the state variables (inputs) x_1, x_2, a sampling plan of 150 values is designed ($[x_1, x_2]_{150}$), and the open loop optimal control problem is solved 150 times (requiring a CPU time of 2.7 *s*) over one sampling period (for simplicity), in order to obtain the optimal objective and control values (output variables) $[j^*, u^*]_{150}$. After that, two MPMs are fitted: $\hat{j}^* = f_0(x_1, x_2), \hat{u}^* = f_1(x_1, x_2)$, each one to predict each output variable as a function of the initial state variables. The MPMs validation is carried out using a different input-output validation data set ($[x_{1v}, x_{2v}]_{400}$, ($[J_v^*, u_v^*]_{400}$), generated in the same way (requiring a CPU time of 5.3 *s*). The two MPMs are then harnessed to predict the outputs$[\hat{j}_v^*, \hat{u}_v^*]_{400}$, which are then compared with the corresponding real ones $[J_v^*, u_v^*]_{400}$ obtained from the real open loop optimal control problem (the Root Mean Square Error (RMSE) is calculated as the accuracy measure).

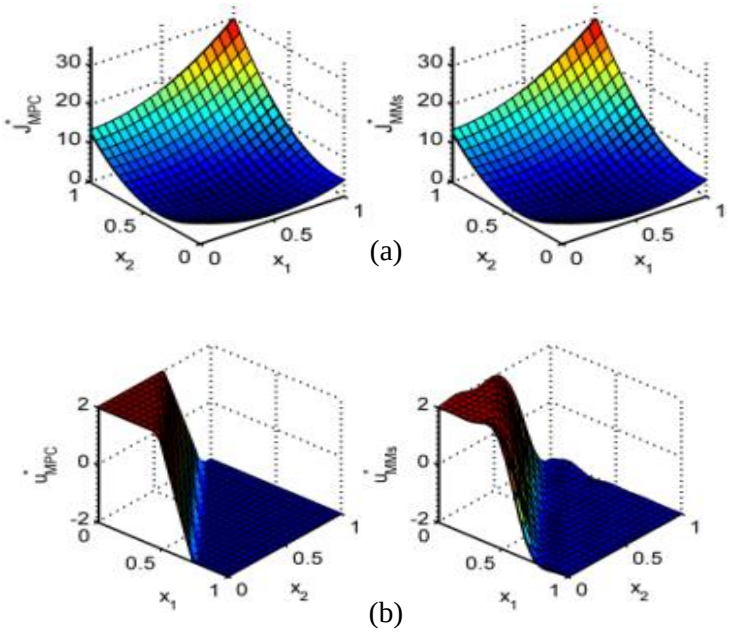


Figure 5.1. Open loop behavior of the optimal objective (a) and control (b) obtained from the OK-based MPMs (right) and the MPC (left).

Figure 5.1 shows the accuracy of the MPMs of the objective (Figure 5.1-(a) right) and the optimal control (Figure 5.1-(b) right) compared to the exact ones obtained by the MPC scheme (Figure 5.1-(a),(b) left). Table 5.1 shows the MPMs fitting and the prediction/validation CPU times, and additionally the validation RMSE. It is clear that the three modeling techniques achieve high accuracy.

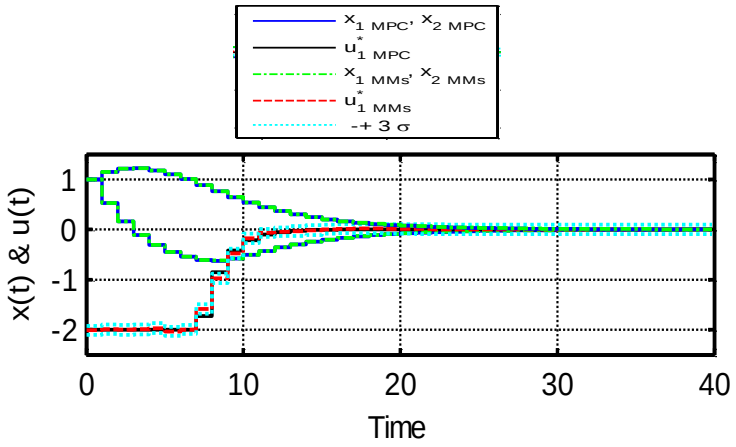


Figure 5.2. Closed loop behavior of the OK-based MPMs compared to MPC.

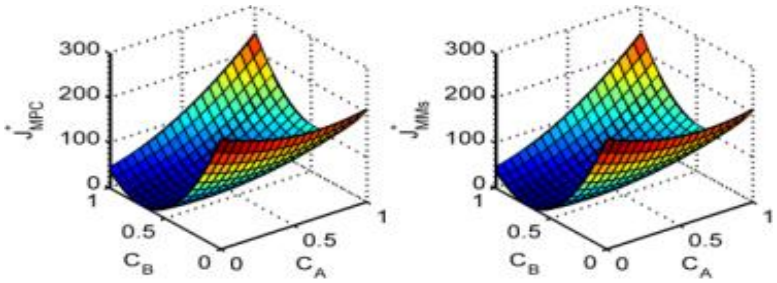
After the validation of the MPMs, they are ready for the online application, so they are employed to predict the closed loop control action of the system via recursive interpolation; at each sampling period, the initial state variables values are used as the inputs for the MPMs interpolation. Figure 5.2 shows that the closed loop control of the MPMs (red dotted line) is too close to the real closed loop control (black solid line) obtained from solving the MPC optimization problem, resulting in approximately the same system state behavior (see also Table 5.2). But more importantly, the MPMs determine the online closed loop control behavior in a very small time, compared with the one obtained by the MPC problem online solution (Table 5.2), besides that the OK estimated error can provide an uncertainty measure (cyan dotted line) about the prediction, which is very useful to evaluate the confidence about the control action.

5.3.2 MPC of a stirred tank reactor

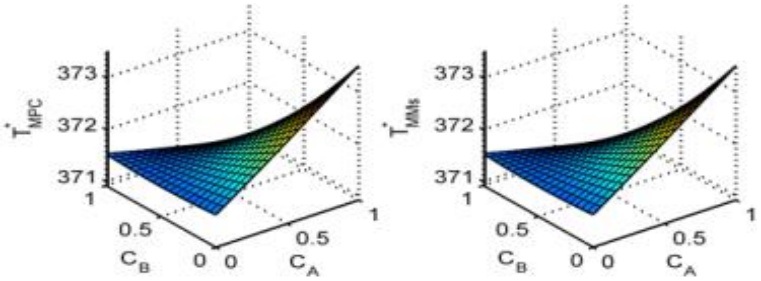
The second application (Tenny & Rawlings, 2004) corresponds to a continuously stirred tank reactor in which the irreversible reactions $A \rightarrow B \rightarrow C$ are taking place, where, C_A, C_B and T represent the concentrations of A, B (states) and the reactor temperature (manipulated variable) in the process model (Eq.(5.9)). The feed to the reactor is pure A, and the maximum conversion to B is desired. A mismatch between the model and the plant exists, as their activation energy values are slightly different, which makes their maximum yield of B different too (0.670 mol/L for the model, and 0.654 mol/L for the plant). The plant is to be operated at its point of maximum yield. Therefore, the output set point in the target calculation is defined as the maximum yield of the model. Traditional multiparametric MPC tools cannot be applied directly to this type of problem, since the process model is nonlinear differential and requires initial steps of model order reduction and approximation to obtain a discrete time model.

$$\dot{C}_A = \frac{F}{V}\left(c_{Af} - c_A\right) - k_1 c_A\, e^{-\frac{E_1}{RT}},$$

$$\dot{C}_B = k_1 c_A\, e^{-\frac{E_1}{RT}} - k_2 c_B\, e^{-\frac{E_2}{RT}} - \frac{F}{V} c_B \tag{5.9}$$



(a)



(b)

Figure 5.3 Validation of the open loop behavior of the OK MPMs compared to real MPC.

The methodology is applied in the same procedure: over an expected domain [0: 1, 0: 1] of the state variables $[C_A, C_B]$, a sampling plan is designed $[C_A, C_B]_{65}$, and the open loop optimal control problem is solved (65 times, requiring a CPU time of 2.8 s) to obtain the optimal objective and control values $[j^*, T^*]_{65}$. After that, two MPMs are fitted: $\hat{j}^* = f_0(C_A, C_B), \hat{T}^* = f_1(C_A, C_B)$. A different input-output data set ($[C_{Av}, C_{Bv}]_{400}$, $[j_v^*, T_v^*]_{400}$) is generated (11 s of CPU time was required) and used to validate the MPMs: the two MPMs are harnessed to predict the outputs $[\hat{T}_v^*, \hat{j}_v^*]_{400}$ using the corresponding inputs $[C_{Av}, C_{Bv}]_{400}$, the predicted outputs are then compared with the corresponding real outputs $[j_v^*, T_v^*]_{400}$ obtained from the real open loop control problem solution.

Figure 5.3 shows the high accuracy of the predicted optimal objective (a) and the predicted optimal control strategy (b) using the MPMs, compared to the optimized ones (see also Table 5.1). Then these MPMs can be used online to estimate the closed loop optimal control with very high accuracy. Figure 5.4 shows how the closed loop control proposed by the MPMs (red dotted line), and the resulting system states, are very close to the optimal closed loop control strategy (black solid line) obtained by solving the MPC problem.

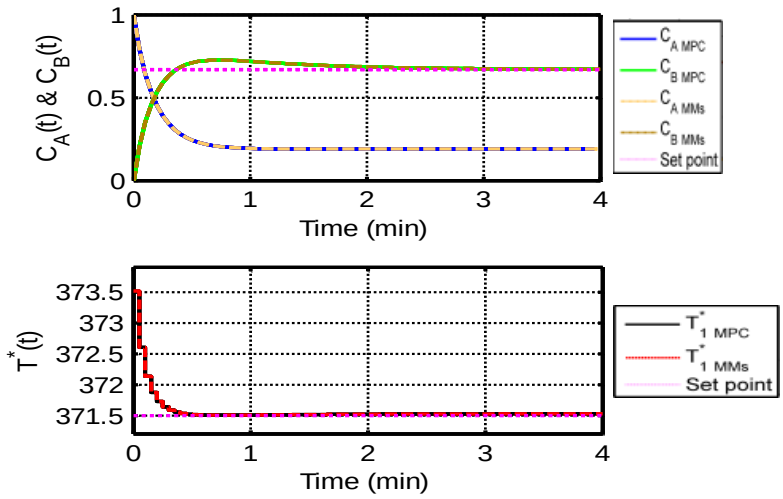


Figure 5.4. Online closed loop behavior of the OK MPMs compared to MPC

Table 5.2 shows that the proposed method achieves a huge saving in the required computational effort ranging between 78% to 99%. Additionally, the results in Table 5.2 illustrate that the methodology advantages increase as the complexity/nonlinearity of the process model increases.

Table 5.1. Offline results: training and validation CPU times, and validation RMSE.

Example	*MPM*	*CPU Time (sec)**						*Validation RMSE*		
		Fitting			*Prediction*					
		OK	*ANN*	*SVR*	*OK*	*ANN*	*SVR*	*OK*	*ANN*	*SVR*
1	***J***	5.30	2.50	0.06	0.30	5.00	0.28	0.002	0.009	0.090
	u_1							0.060	0.050	0.088
2	***J***	0.80	2.30	0.14	0.14	4.90	0.28	0.009	0.031	3.760
	u_1							0.001	0.031	0.113

(*)Intel core (TM) i7-4790 CPU@ 3.6 GHz, 16 GB RAM.

Table 5.2. Online results: RMSE of the MPMs, and CPU times of MPC and DBMP-MPC.

Example	*MPMs*	*RMSE*			*Time per sampling period*		*Time saved (%)*
		u	x_1	x_2	**MPMs**	**MPC**	
1	***OK***	0.03200	0.001800	0.003000	0.00029	*0.0170*	98.30
	ANN	0.03600	0.002600	0.007000	0.00361		78.75
	SVR	0.05200	0.003600	0.002500	0.00028		98.38
2	***OK***	0.00020	0.000001	0.000001	0.00125	0.0300	99.58
	ANN	0.02340	0.000230	0.000090	0.00758		97.47
	SVR	0.05760	0.057630	0.057630	0.00141		99.53

On another side, the OK was able to achieve the highest accuracy in both applications (see Table 5.1). The SVR shows the least accuracy, but it requires the least computational effort for fitting, as the nature of the optimization problem solved for tuning its parameters is quite easy: unconstrained quadratic programming problem (one global optima). On the contrary, the optimization problem solved to adjust the OK (maximize the likelihood) or the

ANN (minimize the error function) parameters is relatively complex: nonlinear optimization problem with multiple local optima. Regarding the interpolation/prediction times (see Table 5.1), the ANN requires the higher computational effort as the prediction is accomplished via series of multiplications of several weight matrices (depending on the number of layers and neurons), additional to the transfer function calculation at each node/neuron output. However, the interpolation via OK and SVR is relatively simple and similar as well: calculations of a simple weight vector, between the new point to be interpolated and the training data, see Eq.(5.5) and Eq.(5.7).

5.4 CONCLUSIONS

A DBMP-MPC is presented, which includes different techniques as sampling design for computer experiments, state of art optimization techniques and machine learning techniques. Its application results to benchmark problems show that the method has achieved the two initially stated goals. The method can approximate optimal control laws using relatively small number of training data, showing very high accuracy and robustness overpassing complex mathematical formulations of the traditional multiparametric MPC. More importantly, a significant difference with the results of the standard multiparametric MPC technique appears; in all the tested cases a single relation was enough to correctly reproduce the optimal control law over the whole initial state variables domain. So, it is not required to use several mathematical relations, each for a certain partition of the initial state variables space. Among the different tested techniques, kriging shows higher accuracy and higher flexibility and robustness to tune its parameters. Additionally, it provides error estimations which can be used as an uncertainty measure of the proposed control actions. The work is now progressing to develop advanced sampling techniques -during the offline MPMs construction-as the sequential sampling-, in order to be able to address more complex case studies.

Chapter 6: Dynamic Surrogate Modelling for Multistep-Ahead Prediction of Multivariate Nonlinear Chemical Processes

This work proposes a methodology for multivariate dynamic modeling and multistep-ahead prediction of nonlinear systems using surrogate models for the application to nonlinear chemical processes. The methodology provides a systematic and robust procedure for the development of data-driven dynamic models capable of predicting the process outputs over long time horizons. It is based on using surrogate models to construct several Nonlinear AutoRegressive eXogenous models (NARX), each one approximating the future behavior of one process output as a function of the current and previous process inputs and outputs. The developed dynamic models are employed in a recursive schema to predict the process future outputs over several time steps (multistep-ahead prediction). The methodology is able to manage two different scenarios: 1) one in which a set of input-output signals collected from the process is only available for training, and 2) another in which a mathematical model of the process is available and can be used to generate specific datasets for training. With respect to the latter, the proposed methodology includes a specific procedure for the selection of training data in dynamic modeling based on Design Of Computer Experiment (DOCE) techniques. The proposed methodology is applied to case studies from the process industry presented in the literature. The results show very high prediction accuracies over long time horizons. Also, thanks to the flexibility, robustness and computational efficiency of surrogate modeling, the methodology allows dealing with a wide range of situations, which would be difficult to address using first principle models.

6.1 INTRODUCTION

In the process engineering area, a reliable dynamic model of the process is necessary for its optimal operation, control and management. In particular, a dynamic model able to accurately predict the future values of the process outputs in reasonable computational times

is the base of most online applications, e.g. Real Time Optimization (RTO), Model Predictive Control (MPC), Dynamic Data Reconciliations, Fault Detection and Diagnosis.

Although analytical models (hereafter also called "First Principle Models" – FPMs) are available to describe the dynamics of many chemical processes, practical limitations often hinder their usage, especially in applications, such as RTO and MPC, which require the online repetitive solution of an optimization problem which, in itself, requires the evaluation of the model several times (Nagy, 2007; Caballero & Grossmann, 2008). This may result in an unaffordable computational effort, especially for large-scale or fast dynamic systems (Ažman & Kocijan, 2011), due to the complexity of the solution procedure –e.g. iterative schemes and/or integration techniques- used to solve such mathematical models (Davis & Ierapetritou, 2007; Davis & Ierapetritou, 2008).

Furthermore, the available FPMs are often developed under the assumption of favorable (ideal) working conditions, which are typically not encountered at the industrial scale, that is characterized by uncontrolled disturbances, different operating conditions, continuously varying parameters (e.g. heat transfer coefficients) and, possibly, different units/reactors geometries, etc. (Qin, 2012; Kajero, et al., 2017). Also, since process FPMs typically do not take into account the physical characteristics of mechanical and electrical components, connections and piping, which remarkably influence the real process, the accuracy of the FPMs predictions are reduced (Ali, et al., 2015). In other cases, the development of a detailed analytical FPM is conceptually difficult or even unaffordable, due to the limited knowledge about the nonlinear behaviors and complex phenomena characterizing the process, such as reaction kinetics, thermo-dynamic relationships, heat and mass transfer, etc. (Bradford, et al., 2018; Ali, et al., 2015). In these situations, on another hand, real data collected from the process are available, but there is no support of a well-founded conceptual/mathematical model for describing the process based on first principles (Nelles, 2001; Boukouvala, et al., 2011; Baraldi, et al., 2013).

In all these cases, system identification or data-driven dynamic modeling methods can be used to construct empirical dynamic models for predicting the future values of the process outputs (Nelles, 2001). Many methods have been developed for linear dynamic system modelling, but their application to nonlinear processes provides unsatisfactory results (Nagy, 2007). This is due to the fact that linear approximations severely simplify the nonlinear behavior of the process, resulting in poor prediction accuracy (Nagy, 2007; Amozeghar & Khorasani, 2016). Advanced data-driven nonlinear modelling techniques, such as Artificial Neural Networks (ANNs) (e.g. radial basis-ANNs, recurrent-ANNs etc.) (Tsai & Chang, 1995; Adebiyi & Corripio, 2003), Fuzzy models (Nelles, 2001), Neuro-fuzzy models (Banu & Umab, 2011) and recently Gaussian Process (GP) models (Zhou, et al., 2015; Mattosa, et al.,

2017), have been widely proposed to capture nonlinear dynamic relations between the nonlinear process inputs and outputs. These techniques, which are also referred to as metamodels or surrogate models, establish nonlinear relationships between inputs and outputs variables, using input-output training data, which can be either generated from complex FPM simulations or measured from the real process (Kajero, et al., 2017).

6.1.1 Review on data-driven dynamic modelling in chemical processes

ANNs have become a popular choice for nonlinear dynamic modeling and identification (Adebiyi & Corripio, 2003; Himmelblau, 2000; Poznyak, et al., 2019), due to their universal approximation abilities (Dua, 2010; Amozeghar & Khorasani, 2016). Although they exhibit very powerful capabilities, their usage has two main practical drawbacks: *i*) large effort is required to select a good network structure (numbers of layers and the included neurons) and configurations (type of activation function, training algorithm, cost/error function, etc.) (Kajero, et al., 2017), and *ii*) the curse of dimensionality, i.e. the increase of the number of inputs causes the growth of the number of the ANN neurons, and consequently, of the number of parameters (weights and biases) to be set: then, the quantity of data needed for training the ANN grows exponentially with the number of inputs (Ažman & Kocijan, 2011).

Although different algorithms have been developed to automatically select ANNs structures and configurations (Dua, 2010), their application requires additional computational effort, since they solve a complex optimization problem, in which the network configuration and its parameters are treated as decision variables to be tuned to minimize an objective associated to the output prediction error (Ludermir, et al., 2006; Benardos & Vosniako, 2007; Leperi, et al., 2019). As a result, their application to cases involving high dimensional systems, large-scale databases and/or online fitting and updating has been quite limited.

In spite of these difficulties, a significant number of successful applications of ANNs for dynamic modelling are reported over a wide spectrum of fields (Nelles, 2001; Masters, 1993; Himmelblau, 2000; Rigamonti, et al., 2018). Especially in the process engineering area, ANNs have been extensively used as Nonlinear AutoRegressive eXogenous (NARX) models for dynamic system identification of both univariate (single output) (Godarzi, et al., 2014; Nagy, 2007; Panapakidis & Dagoumas, 2016; Sadeghassadi, et al., 2018; Xu, et al., 2014; Poznyak, et al., 2019) and multivariate (multi-output) problems (Adebiyi & Corripio, 2003; Caccavale, et al., 2010; Banu & Umab, 2011; Li & Li, 2015; Amozeghar & Khorasani, 2016; Lee, et al., 2018). In the literature, multivariate systems are usually approximated either using a multi-output ANN model or an ensemble of single-output ANNs models, where, in the latter case, a set of independent single-output ANN models, each approximating one output as a function of the inputs, is built.

On the other hand, Gaussian Process (GP) models have been proposed in the Bayesian inference area by O'Hagan et al. (1978; 1999) for the approximation of complex static computer codes, representing a generic class of non-parametric probabilistic models. GP models have shown promising accuracy and ability to reduce the previously mentioned problems of ANNs (Ažman & Kocijan, 2011; Deisenroth, et al., 2009). This is due to their nonparametric nature: they do not approximate the system by fitting the parameters of a selected structure or functional shape but, instead, they search for relationships among the measured data through a correlation function/model. Therefore, the number of the metamodel parameters to be identified is significantly low compared to other parametric models (e.g. ANNs models) and, consequently, the size of the required set of training data is significantly reduced, too (Azman & Kocijan, 2007). Besides, GP models offer high approximation accuracy, tuning flexibility and ability to estimate a measure of uncertainty about the prediction in the form of prediction error or variance (Boukouvala, et al., 2011; Rasmussen & Williams, 2006).

Thanks to the pioneer works of Murray-Smith, et al. (2003), Kocijan, et al. (2005) Girard, et al. (2002), and Rasmussen & Deisenroth (2008), among others, GP models have gained a wide popularity for dynamic modeling and identification of nonlinear systems, and shown performances comparable and competitive to other state-of-art techniques. The main limitation of the GP models is the large computational cost for optimizing/fitting their parameters, especially when considering a large amount of training data and/or addressing a high dimensional system (Ažman & Kocijan, 2011). With respect to the problem of performing multi-step ahead predictions, some works have been able to successfully propagate the GP estimated error when it is used in recursive prediction (Girard, et al., 2002). But, again, the computational cost associated to the uncertainty propagation is still significant.

The Ordinary Kriging (OK) techniques can be considered specific form of GP models (Boukouvala, et al., 2011) and share similar advantages, such as accurate approximation capabilities, required small number of training data, flexible tuning of the model parameters (Forrester, et al., 2008) and ability of estimating a prediction error. Also, alike to the GP model, OK suffers from the high computational training effort. Thanks to the works of Davis and Ierapetritou (2007) and Caballero and Grossmann (2008), the OK surrogate models has been introduced to the chemical process engineering area and, since this time, it is attracting increasing attention for surrogate-based optimization and analysis of complex nonlinear static processes (Kajero, et al., 2017; Wang & Ierapetritou, 2017; Beck, et al., 2015; Egea, et al., 2007).

Nowadays, the GP and OK models have been proposed for univariate dynamic modeling of nonlinear chemical processes (Ažman & Kocijan, 2011; Zhou, et al., 2015), where they are employed as NARX models to estimate the future value - over one step-ahead - of an output of interest, as a function of the process current inputs and output values. The developed model is, then, used to perform multistep-ahead prediction via recursive calculation, where the predicted output at the current time is fed-back to the model as a part of its input for the next prediction step.

To the best of the authors' knowledge, few works have extended the GP and OK capabilities to multivariate dynamic modeling of chemical processes: Hernandez and Grover (2010) developed a method for multivariate dynamic modeling based on a set of GP models, each one representing a discrete-time state space model predicting the time evolution of one process output; they also proposed a sequential sampling technique to select the training data to be used for training the GP-based dynamic models; the method was successfully applied to approximate a stochastic zero-input/multi-output dynamic model describing nanoparticle size evolution. In an area more related to process and system engineering, Boukouvala et al. (2011) proposed a similar approach based on a set of kriging metamodels, each one predicting the future values of one process output through recursive prediction over several time steps, and applied it to the simulation of a powder-roller-compaction pharmaceutical process. The approach has shown good accuracy in the identification of the dynamic behavior of the process outputs (ribbons density and roll gap) that are influenced by three control inputs (roll speed, roll pressure, feed speed); they proposed the use of a full factorial design for selecting the initial training dataset, and a sequential procedure to update the trained models during their online usage by adding to the initial training set the predicted instances for which the summation of the OKs estimated variances/errors was lower than a specific threshold.

However, these two works share some common limitation: 1) they have been validated considering processes characterized by very smooth/steady dynamics, without any influencing control/external inputs (Hernandez & Grover, 2010) or with very simple changes in them (Boukouvala, et al., 2011), 2) both works provided simple Markovian state-space models and they have not illustrated the ability of their methodologies to develop dynamic models with delayed/lagged inputs, 3) they presumed that a FPM is always available, which is combined with DOCE methods to produce optimized data for training, and 4) the robustness of their methodologies to handle different cases studies, and their flexibility to integrate different metamodel types are not explored. Finally, the methodology proposed by Boukouvala et al. (2011) has not been proven to provide one compact set of models able to simulate the future behavior of the system outputs corresponding to simultaneous changes in the process inputs,

since, in this method, a new set of dynamic models should be fitted several times, each time to approximate the system behavior corresponding to a simple step change in one of the control input variables, keeping the rest of the control input variables fixed.

The aforementioned limitations obstacle the use of these methodologies for the dynamic modelling of real processes or systems, where remarkable challenges are posed: *i*) in real processes, many external inputs exist, which control or disturb the process causing significant changes in its outputs behavior, *ii*) incorporating lags or delays in the model inputs is a basic requirement in data-driven dynamic modeling, in order to capture the possible delayed behavior of the process itself and/or to compensate for missing repressors of the model (Espinosa & Vandewalle, 1998a; Espinosa & Vandewalle, 1998b), *iii*) in many practical situations, data collected from the process can be the only source of information available (i.e. no FPM).

More recently, Bradford et al., 2018 presented a method for multivariate dynamic modeling that relies on a set of GP-based NARX models. The method was applied to model the multivariate behavior of a real Algal lutein production batch process that involves two control inputs and three process outputs. Although the method provided good prediction accuracy, the addressed case study is characterized by simple dynamics, since one control input is kept constant in all the different batches, while the second is allowed to vary from one batch to another, but its value within the same batch is kept constant. Hence, practically, the control inputs became constant parameters and, consequently, the set of dynamic models are validated by predicting the simple behavior of zero-input batches. Also, when the validated set of GP dynamic models is further used for dynamic optimization, the predicted optimal "offline" profiles of the process outputs are not compared to those of the real batch system.

This Chapter presents a generic multivariate dynamic modeling and multi-step ahead prediction methodology. The methodology is based on training a set of OK-based NARX models; each model predicts the upcoming value of one process output over a constant time step as a function of the preceding values of the process inputs and outputs, over a suitable time lag. The obtained models represent discrete state-space models (also called single-step or one-step ahead simulators) that mimic the incremental evolution of the process outputs. The trained dynamic models interact through a recursive scheme to predict the system outputs over several time steps (multistep-ahead prediction),

The main contributions of this work are:

1) the development of a novel, generic and robust methodology for multivariate dynamic modeling and multi-step ahead prediction of complex nonlinear chemical processes

using surrogate models. The properties of generality and robustness are fundamental in order to address the main limitations currently attributed to the existing approaches in terms of a) the ability to provide accurate data-driven dynamic models for general multi-input/multi-output processes that may involve complex dynamic behaviors (complex control input profiles, delayed behaviors, etc.), b) the ability to simulate the process future outputs over large time horizons, c) the capability to accommodate different types of data modeling techniques and d) the ability of handling different situations, either when a limited set of input-output data signals are available, or when the training data can be optimally generated using a FPM and design for computer experiment techniques.

2) the introduction of the use of OK models for the multivariate dynamic modeling in the chemical process field in a robust and flexible manner, and the comparison of its capabilities with most popular techniques (i.e. ANNs).

3) the development of a novel Design Of Computer Experiments procedure for dynamic modeling, considering the purpose of the simplification and complexity reduction of expensive dynamic FPMs.

The rest of the Chapter is structured as follows. Section 6.2 gives a general view over the considered DOCE and surrogate modeling techniques (i.e.: OK and ANN), including their mathematical/statistical basis and implementation details. Section 6.3 presents the proposed dynamic metamodeling method, and the new procedure proposed for the design of computer experiments in the case of dynamic modelling. Section 6.4 shows the method application to three different case studies (different natures, i.e. continuous and batch and, different areas, i.e. biochemical, industrial and petrochemical) and discusses the obtained results. Finally, Section 6.5 concludes the work, stresses its advantages and discusses its limitations, which would be further investigated in future works.

6.2 SURROGATE MODELS BUILDING TECHNIQUES

Surrogate models are data-driven techniques which are used to build empirical relations describing the mapping between input and response variable(s) (Forrester, et al., 2008; Wang, et al., 2019). Although this definition can involve a very wide range of data-based models, including the simplest types (e.g.: linear or polynomial regressions), the term is usually associated to nonlinear multivariate models, like ANNs, GPs, OK, Support Vector Regression (SVR), etc. (Fang, et al., 2005). Surrogate models can be trained using real data collected by sensors from the physical systems or using simulation data generated from a complex FPMs, for the purpose of its simplification. The following subsections review most common DOCE

techniques used for training data selection in cases where a FPM is available, highlight the basics of the two common nonlinear data-driven modeling techniques, namely OK and ANNs, which have been used in this work, and review basics of common DOCE methods.

6.2.1 Design of computer experiments

Design Of Computer Experiments (DOCE) techniques (Jurecka, 2007) aim at selecting the best combinations of the input variables values -within specific domain or bounds- that can be used for the simulation of the complex FPM providing the most representative information/knowledge about the output behavior (Garud, et al., 2017). The set of combinations of the input variables values is called "sampling plan", $[X]_n$, where n is the number of sample points or instances. The objective of these techniques is to collect as much information as possible about the output behavior over all the local sub-regions in the input space. Therefore, DOCE techniques consider samples selection criteria including, mainly, the space-fillingness and the stratification of the sampling plan (Forrester, et al., 2008; Garud, et al., 2017), and both lead to increase the uniformity of the sampling plan over all the local sub-regions of the input space to be covered. More details about DOCE techniques can be found in Section 2.1.

Many DOCE techniques have been developed in the literature, basically, for "static" surrogate modeling. This work considers the Hammersley design technique, due to its ability to provide sampling plans of good uniformity and stratification properties with very low computational cost (Garud, et al., 2017; Ibrahim, et al., 2019). In each case, the optimal selection of the number of sample points (n) required to capture the output behavior depends on the input dimensionality of the surrogate model (k), the size of the input space and, also, on the intricacy and nonlinearity of the considered output behavior. In general, as n increases, the effort (time/cost) required not only for executing the experiments, but also to design the sampling plan and for the surrogate model fitting increases. Then, the modeler should carefully balance the trade-offs between the required surrogate model accuracy, the computational cost and the eventual application benefits of the surrogate model.

6.2.2 Ordinary kriging

Given a set of n input-output training data $[x_i, w_i]$, $i = 1,2,..n$, $x \in R^k, w \in R$, the OK assumes the predictor $\widehat{w}(x) = \mu_{ok} + Z(x)$, where the constant term μ_{ok} represents the main trend of the system to be approximated, and $Z(x)$ is a deviation from that trend. The deviation $Z(x)$ is modeled as a stochastic Gaussian process with expected value $E(Z(x)) = 0$, and a covariance between two residuals $cov(Z(x_i), Z(x_j))$ that only depends on their corresponding inputs x_i, x_j. Thus it can be calculated as: $cov\left(Z(x_i), Z(x_j)\right) = \sigma_{ok}^2 R(x_i, x_j)$, being σ_{ok}^2 the

process variance and $R\left(x_i, x_j\right) = exp\left(-\sum_{l=1}^{k} \xi_l \left|x_{i,l-} x_{j,l-}\right|^{p_l}\right) + \delta_{ij}\,\lambda$ a correlation function, where, $\xi_l, l = 1, \ldots k$ are the model hyper-parameters, δ_{ij} is the Kronecker delta, p_l are smoothing parameters and λ is a regularization constant that enables the kriging predictor to regress noisy data (Azman & Kocijan, 2007). The kriging predictor and its estimated error are given by Eq.(6.1) and Eq.(6.2), respectively, where (x^*) is a new interpolating point (different from the training data). In Eq.(6.1), $[r]_{n\times 1}$ is the vector of correlations between the point to be predicted x^* and the original training data points and calculated as $R(x_i, x^*)$, $[R]_{n\times n}$ is the correlation matrix between the training inputs, $[W]_{n\times 1}$ is the vector of the training outputs and $[\mathbf{1}]_{n\times 1}$ is the identity vector.

$$\hat{w}(x^*) = \mu_{ok} + r^T R^{-1}(W - \mathbf{1}\mu_{ok}) \tag{6.1}$$

$$\hat{s}^2(x^*) = \sigma_{ok}^2(1 + \lambda - r^T R^{-1} r + (1 - \mathbf{1}^T R^{-1} r)^{-1}/(\mathbf{1}^T R^{-1}\mathbf{1})) \tag{6.2}$$

This work considers the OK implementation developed by Forrester, et al., (2008), because of its high efficiency and applicability. Besides, the "*fmincon*" algorithm included in the Matlab optimization toolbox is used for the maximization (nonlinear optimization) of the concentrated likelihood function, see Section 2.2.1. The work, also, considers another software implementation for the GP model construction: the GP-Regression (GPR) algorithm based on the function "*fitrgp*" included in the Matlab statistics and machine learning toolbox. Here, it is worthy to emphasize that the objective of this work is not to compare different specific implementations of the GP models but to explore the robustness and flexibility of the proposed methodology by handling different data-based modelling techniques and software.

6.2.3 Artificial Neural Networks

The ANNs are very well-known efficient machine learning models, which are widely used for data-driven modelling of nonlinear systems. In this work, the Matlab ANN toolbox and the function "*feedforwardnet*" have been used to create multilayer feedforward ANNs. In each of the following application cases, the number of layers, number of neurons and the training algorithm were selected based on a trial and error procedure in order to balance the ANN structure simplicity and its prediction accuracy. More details about ANNs can be found in Section 2.2.2.

6.3 DYNAMIC MODELLING BASED ON SURROGATE MODELS

This part presents *i*) an overview on the most common approaches considered in the literature (Conti, et al., 2009; Ažman & Kocijan, 2011) for the univariate dynamic modeling and multi-step ahead prediction using black box models (Section 6.3.1), *ii*) the proposed methodology for multivariate dynamic modeling and multi-step ahead prediction of chemical

processes based on surrogate models (Section 6.3.2), *iii*) the proposed DOCE procedure for training data generation in dynamic modeling in cases when the purpose is the simplification and complexity reduction of expensive dynamic FPMs (Section 6.3.3) and *iv*) the procedure for training data generation that mimics practical situations where a FPM of the process is not available and, only input-output signals, measured and collected from the process by the physical sensors network are available (Section 6.3.4).

6.3.1 Univariate dynamic modelling and multi-step ahead prediction

Let us consider a univariate dynamic system or process, characterized by D_u control inputs $\boldsymbol{U} \in R^{D_u}$ and one process output $\boldsymbol{Y} \in R$, where both can be real data collected from actual plant or simulated data generated by a FPM over discrete, successive and uniform time intervals or sampling periods $\Delta t = (t_i - t_{i-1})$: $[t_0, t_1, t_2, t_3, \dots t_i, \dots t_{f-1}, t_f]$, where t_0 and t_f are the first and the final time instances, respectively. Hence, the measured control input and process output signals become $\boldsymbol{U} = [U_0, U_1, U_2, \dots U_i, \dots U_{f-1}, U_f]$ and $\boldsymbol{Y} = [Y_1, Y_2, Y_3, \dots Y_i, \dots Y_{f-1}, Y_f]$.

Using this input-output training information, it is required to construct a data-driven or black-box model that is able to forecast the future values of the output over q time steps-ahead from the current generic time instance t, i.e., $\left[\hat{Y}_{t+1}, \hat{Y}_{t+2}, \dots \hat{Y}_{t+q}\right]$. For this purpose, three main dynamic modeling approaches have been usually considered (Conti, et al., 2009; Ažman & Kocijan, 2011):

i) The first approach is the "Multi-Output" (MO) emulator that considers a q-output data-driven model, where each output of this model corresponds to the process output value at the *j-th* time step, $j = 1,2,q$. In this case, the model input, x, must include the previously measured values of the process outputs and the corresponding control inputs over a specific time lag L, i.e. $\left[\hat{Y}_{t+1}, \hat{Y}_{t+2}, \dots \hat{Y}_{t+q}\right] = \text{Ŧ}(Y_t, Y_{t-1}, .. Y_{t-L}, U_t, U_{t-1}, .. U_{t-L})$, where Ŧ is the multi-output black-box model. In this case, a multi-output surrogate model must be used, e.g. multi-output ANN.

ii) A second alternative is the "Ensemble of Single-Output" models (ESO) approach in which q single-output black box models are considered: each model predicts the single output at each of the q required times, hence $\hat{Y}_{t+1} = \mathbb{f}_1(Y_t, Y_{t-1}, .. Y_{t-L}, U_t, U_{t-1}, .. U_{t-L}), \dots, \hat{Y}_{t+q} = \mathbb{f}_q(Y_t, Y_{t-1}, .. Y_{t-L}, U_t, U_{t-1}, .. U_{t-L})$, where $\mathbb{f}_1, \dots \mathbb{f}_q$ are single-output black-box models.

iii) The third approach is the recursive single-step emulator, which employs one black box model to approximate the evolution of the process output over a single time increment or step Δt, such that $\hat{Y}_{t+1} = f(Y_t, Y_{t-1}, .. Y_{t-L}, U_t, U_{t-1}, .. U_{t-L})$.

The single-step emulator approximates the future value of the process output as a function of the process previous control input and output values, considering a specific time lag L. However, it is used in a recursive way for forecasting the output value along q intervals of time. Hence, at every prediction step, the forecasted value of the process output is sent back to the model acting as a part of its input for the next time step prediction, jointly with the new values of the process control inputs.

The single-step emulator is also known as autoregressive model, and it has proved to be much more efficient than the two previous approaches, because of its capability to predict the output variable values over any number of time steps through a recursive procedure. This capability is not obtainable when using the other two approaches (MO and ESO), because they are designed and trained to predict the output value over a fixed or rigid number of time steps. Thus, if it is required to change the prediction horizon (i.e. number of prediction time steps), a completely new model (MO case) or set of models (ESO case) must be constructed. Additionally, the single step emulator approach is simpler/more practical in terms of the computational effort required for its implementation, since only one single-output model is constructed and used, instead of the construction/training of a MO model or ESO models. And, finally, it is worth noting that, when considering a multivariate (i.e. multi-output) process, the effort and time required for the construction of data-driven dynamic models based on the MO or ESO approaches will be dramatically magnified. For all the aforementioned reasons, the single-step emulator scheme is considered in this study.

6.3.2 Proposed multivariate dynamic modelling and multi-step ahead prediction methodology

Assuming a general multivariate dynamic process involving the inputs $\boldsymbol{U} \in R^{D_u}$ and outputs $\boldsymbol{Y} \in R^{D_y}$, and keeping the same assumption that all the process inputs and outputs are either measured (real process) or simulated (computer code) at constant, successive and equal time intervals $[t_0, t_1, t_2, t_3, \dots t_i, \dots t_{f-1}, t_f]$, the proposed method is based on the construction/training of a set of D_y NARX models (see Eqs.(6.3)) in order to capture the incremental evolution of the process outputs, $\hat{Y}_{t+1}$, over one step-ahead time interval. Thus, each model f_j, $j = 1,2, \dots D_y$ approximates the future value of the *j-th* process output at the next time step $t + 1$, i.e. $\hat{y}_{j,t+1}$, as a function of the previous process inputs and outputs, considering a specific time delay L. In this way, any possible correlation between the

upcoming value of a certain output $\hat{y}_{j,t+1}$ and any of the process previous input and output can be captured.

$$\left.\begin{aligned} \hat{y}_{1,t+1} &= f_1\left[\hat{Y}_t,..,\ \hat{Y}_{t-L},\ U_t,..,U_{t-L}\right], \\ &\vdots \\ \hat{y}_{j,t+1} &= f_j\left[\hat{Y}_t,..,\ \hat{Y}_{t-L},\ U_t,..,U_{t-L}\right], \\ &\vdots \\ \hat{y}_{D_y,t+1} &= f_{D_y}\left[\hat{Y}_t,..,\ \hat{Y}_{t-L},\ U_t,..,U_{t-L}\right] \end{aligned}\right\} \quad (6.3)$$

After the models group (in Eq.(6.3)) is trained, they are used to forecast the evolution of the process outputs over longer period of time associated to a "totally new" and known profile of the process control input $U^v = [U^v_{t_0^v}, U^v_{t_1^v} \ldots . U^v_{t_i^v} \ldots .. U^v_{t^v_{f-1}}, U^v_{t_f^v}]$ that affects the process over the "totally new" time sequence $[t_0^v, t_1^v, \ldots., t_i^v, \ldots .. t^v_{f-1}, t_f^v]$ (the superscript *v* refers to "validation"), i.e. performing multi-step ahead prediction. The latter goal is achieved through recursive prediction, assuming that the first L values of the outputs are known, ($Y^v_{t_0^v}, \ldots\ Y^v_{t_{nL}^v}$), $nL = L$. The recursive prediction starts using the known inputs $x_1^v = [\ Y^v_{t_0^v}, \ldots\ Y^v_{t_{nL}^v},\ U^v_{t_0^v}, \ldots\ U^v_{t_{nL}^v}\ ]$ to predict the process output values at the next time step, $\widehat{Y}^v_{t^v_{nL+1}}$. These predicted output values are used, jointly with the new control input values, as the new models input, $x_2^v = [Y^v_{t_1^v}, \ldots\ Y^v_{t_{nL}^v},\ \widehat{Y}^v_{t^v_{nL+1}},\ U^v_{t_1^v}, \ldots\ U^v_{t_{nL}^v},\ U^v_{t^v_{nL+1}}]$, for the next time step, so as to predict the output values, $\widehat{Y}^v_{t^v_{nL+2}}$. The recursive prediction continues until the last time step, at which the prediction input $x_n^v = [\ \widehat{Y}^v_{t^v_{f-1-L}}, \ldots\ \widehat{Y}^v_{t^v_{f-1}}, U^v_{t^v_{f-1-L}}, \ldots, U^v_{t^v_{f-1}}]$ are used to predict the output $\widehat{Y}^v_{t_f^v}$. Notice that $n_v = t_f^v - nL$ is the number of prediction steps or times recursively performed by the models in order to predict the future outputs behavior of the validation signals.

The dynamic models performance can be assessed considering an accuracy metric (e.g., Normalized Root Mean Square Error – NRMSE - Eq.(6.5)) that computes the difference between the exact and the predicted values of each of the D_y output signals, respectively, $y^v_{j,t_i^v} \in\ Y^v_{t_i^v}$ and $\hat{y}^v_{j,t_i^v} \in\ \widehat{Y}^v_{t_i^v}$, $i = nL+1, \ldots, f,\ j = 1,2, \ldots D_y$.

$$RMSE_j = \sqrt{\frac{1}{n_v}\sum_{i=nL+1}^{f}(\ y^v_{j,t_i^v} - \widehat{y}^v_{j,t_i^v})^2} \quad (6.4)$$

$$NRMSE_j = 100\frac{RMSE_j}{(max\ (y^v_{j,t_i^v}) - \min(\ y^v_{j,t_i^v}))} \quad (6.5)$$

It is worth to highlight that the mathematical structure/design of the proposed modeling approach (Eq.(6.3)) does not directly or explicitly assume any correlation between the outputs of the single-step emulator $[\hat{y}_{1,t+1}, \ldots \hat{y}_{j,t+1}, \hat{y}_{D_y,t+1}]$, since each dynamic model is constructed

and trained independently. However, the information about the eventual correlations among the dynamic model outputs is introduced by two mechanisms: i) the fact that each model output $\hat{y}_{j,t+1}$ is computed as a function of the whole set of former values of the process state (inputs and outputs), and ii) the recursive nature of the prediction scheme (Figure 6.1), which makes each dynamic model f_j to contribute with its prediction $\hat{y}_{1,t+1}$ to the overall prediction of the process output $\hat{Y}_{t+1} = [\hat{y}_{1,t+1}, \hat{y}_{2,t+1}, \dots, \hat{y}_{j,t+1}, \dots, \hat{y}_{D_y,t+1}]$ which, at the end, will constitute the prediction/model input at the next time step. In other words, the output of each dynamic model at the current time step depends on the delayed outputs predicted by other dynamic metamodels at previous time steps, interacting among them during the recursive calculations, so every sole model benefits from the knowledge supplied by the other models in former time steps.

On another hand, it is unlikely that each process output will be dependent on the complete set of the process input and output variables -including their lagged values-, see Eq.(6.3). But, since there is no prior knowledge about the process behavior, it is useful to allow for all the possible correlations between the process variables, and to let the training task extract the knowledge about the strength of the allowed correlations. However, this may be also a limitation when a large-scale process is considered, since this will increase the input dimensionality, complicate the model structure and, consequently, increase the number of model parameters. Therefore, this might pose many challenges to the training task: not only the computational effort will increase, but also a higher number of training data will be required in order to face the tuning of the additional model parameters. In this case, previously to the modeling task, an initial analysis can be carried out in order to reduce the models input dimensionality. Although this is not in the scope of this work, it is worth to mention that this can be achieved either based on the knowledge about the system variables and their relations, or using computational techniques as cross-correlation, sensitivity analysis, feature selection and extraction techniques, etc.

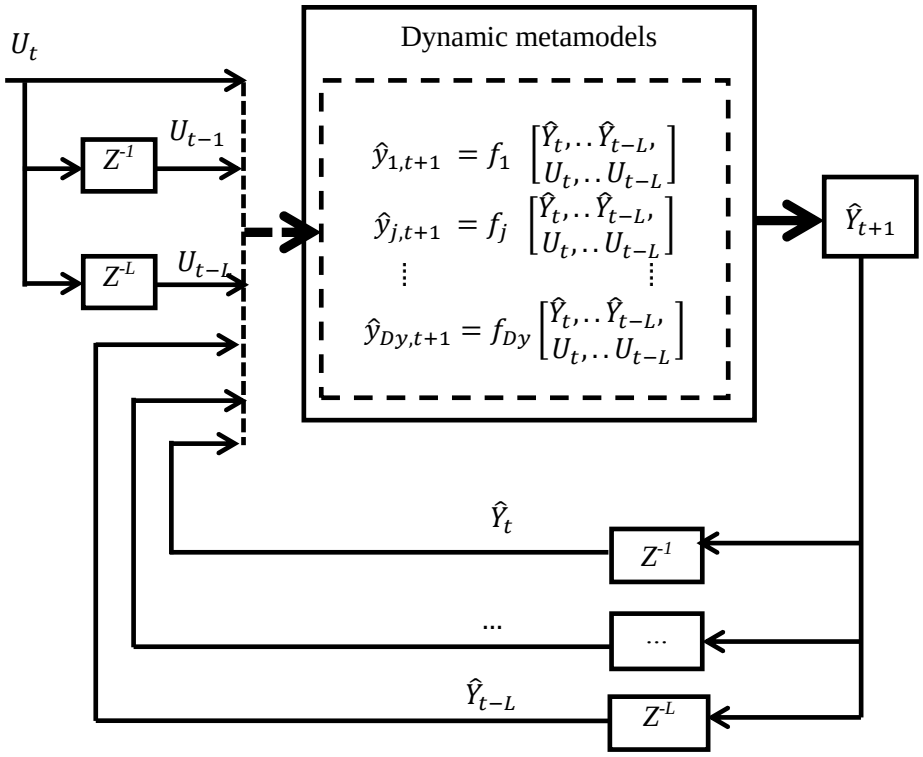


Figure 6.1. Scheme of the proposed multistep-ahead prediction using the fitted multivariate dynamic models.

Another factor to be selected at this stage is the model lag/order, L, which will affect the resulting model quality and complexity (and, obviously, will also determine the effort required for model training/tuning). Several methods can be found in the literature for making this selection. For linear dynamic models, the cross-correlation between the model output and input has been used (Nelles, 2001; Espinosa & Vandewalle, 1998b). This technique exploits the linear relationships assumed by choosing a linear model. Thus, the cross-correlation between the model output, Y_{t+1}, and the input including different delayed information, $Y_t, Y_{t-1}, \dots, Y_{t-L}, U_t, U_{t-1}, \dots, U_{t-L}$, would give an indication about the delay within which the model input mostly influences its output. Similarly, the correlation between the model inputs and the model prediction error, $e = Y_{t+1} - \hat{Y}_{t+1}$, based on a test set, can reveal the missing regressors, i.e. delayed inputs. Another technique that has been commonly used for the inference/selection of the time lag associated to a linear dynamic model is based on the use of Akanke's information criterion. More details can be found in (Espinosa & Vandewalle, 1998a).

For nonlinear dynamic models, a common technique for the estimation of a suitable lag is the calculation of the Lipschitz index from the training data only without any dependence or assumption about the model nature (Espinosa & Vandewalle, 1998b; Cho, et al., 2007; Suykens, et al., 1996). The method is based on the continuity property of the nonlinear functions that represent input-output models of continuous dynamic systems. The Lipschitz

index is computed considering different lags or delays starting from $L = 0$, and the best embedding dimension is obtained when the index stops decreasing.

Most of the techniques proposed for estimating the data-driven dynamic models order consider only univariate cases. When dealing with multivariate dynamic models, defining a specific different lag for each input with respect to each output is an optimal, but utopic, objective, and to the authors' knowledge, a way for achieving this is not yet available in the literature because it is practically/numerically complicated, mainly due to the eventual combined interactions. A practical and simple approach is to consider a model structure with the same lag for all the input variables (Nagy, 2007; Azman & Kocijan, 2007; Bradford, et al., 2018), see Eq.(6.3). Although this may seem restrictive, as each process variable, in fact, will present a different physical behavior, the idea is that the importance of the lagged inputs will be adjusted/balanced during the model training according to their significance with respect to the model output, through the manipulation of the values of the weights and biases in the ANN model, or of the parameters ξ_l in the OK model.

In this work, a simple and common try and cut procedure is considered for this selection. So, different sets of multivariate dynamic models are built with different lag values, and the lag that achieves the minimum prediction error of all the D_y models - over a new test set - is selected.

6.3.3 DOCE for dynamic modelling

As mentioned in Section 6.2.1, different techniques for the DOCE have been commonly used for determining the most convenient training set in the case of data-driven modeling with the purpose of approximating static complex computer models. But these techniques are rarely applied to situations where the purpose is the approximation of "dynamic" computer models.

As indicated before, the few methods already proposed for DOCE in dynamic modeling (Hernandez & Grover, 2010; Boukouvala, et al., 2011) show different limitations: 1) their capabilities to handle general dynamic processes that often include control inputs and lagged behavior are not illustrated, 2) their robustness to handle different case studies, and their flexibility to integrate different metamodel types are not explored, 3) these sampling procedures are based on the estimated prediction error of the GP/OK metamodels and, therefore, their application with important metamodelling techniques that do not possess this characteristic (e.g. ANN, SVR, etc.) is not feasible, and 4) the sequential nature of these sampling procedure would easily lead to a high computational burden, especially if it is applied to cases characterized by high dimensionality (e.g., several control inputs and process outputs with lagged behavior) and/or include high numbers of training data, see Section 6.2.1 .

In this section, a DOCE is proposed for data-driven multivariate dynamic modeling of complex processes, assuming the availability of a reliable and accurate FPM. The method is based on the use of Hammersley sampling design, which is the one selected in this work, as previously justified. However, any efficient alternative can be also used (e.g. optimized Latin hypercube designs, etc.). The proposed procedure is aimed at alleviating the limitations just mentioned at the beginning of this section.

As an important principle of the proposed sampling procedure, it must be taken into account the different nature of the dynamic model (or metamodel) inputs, when compared to steady-state model/metamodel inputs. Since the inputs of a steady-state model are assumed to be independent (e.g. temperature, pressure, volume etc.), the selection of their values combinations $[X]_n$ within their specific bounds is a straightforward task. However, in the case of a dynamic model (Eq.(6.3)), the model inputs $x = [Y_t, Y_{t-1}, .. Y_{t-L},\ U_t, U_{t-1}, .. U_{t-L}]$ can not be considered independent since, in general, x must include some model inputs $(Y_t, Y_{t-1}, .. Y_{t-L})$ which actually correspond to previous outputs (over a certain time lag). Thus, the DOCE technique can freely select any possible combination of values for the process control inputs and their delayed counterparts, $U_t, U_{t-1}, .. U_{t-L}$, since these values correspond to external actions applied to the system and, as a consequence, they can be considered neither correlated nor dependent over time (i.e. U_{t-1} does not depend on U_{t-2}). But, in contrast, it is not possible to freely select any arbitrary combination of values for the process outputs and their delayed counterparts, $Y_t, Y_{t-1}, .. Y_{t-L}$, because the process outputs may be correlated among others (i.e., y_j depends on $y_{j'}, j\ and\ j', = 1,2, .. D_y,\ j \neq j'$), they will probably depend on their delayed values (i.e. $y_{j,t-1}$ will probably depend on $y_{j,t-2}$) and, of course, they will depend on the process inputs and their lagged values also (i.e. $Y_{t-i} = f\left(Y_{t-(i+1)}, U_{t-(i+1)}, \ldots\right)$. Thus, Y_{t-L} are the only output values that can be freely selected, since they are the initial values in the generated profile.

So, the proposed procedure harnesses the Hammersley technique to design a sampling plan that includes n combinations of values of independent models inputs, $[Y_{t-L},\ U_t, U_{t-1}, .. U_{t-L}]$, over the expected operational domain of the process variables $[U_{t\,min} :\ U_{t\,max},\ Y_{t\,min} :\ Y_{t\,max}]$. Each combination (row of the sampling plan matrix) consists of the D_y initial process output values, Y_{t-L}, besides the $D_u \times (L+1)$ values of the process control inputs and their lagged counterparts, $[U_t, U_{t-1}, .. U_{t-L}]$. The rest of the dynamic metamodel inputs, $[Y_t, Y_{t-1}, .. Y_{t-L+1}]$, together with the dynamic metamodel output, Y_{t+1}, are obtained by the simulation of the process model considering the initial process outputs, Y_{t-L}, and the control input profile value, $[U_t, U_{t-1}, .. U_{t-L}]$, previously selected by the DOCE technique (Figure 6.2). Finally, the input–output training data, $[X]_n =$

$[Y_t, Y_{t-1}, .. Y_{t-L}, U_t, U_{t-1}, .. U_{t-L}]_n$, $[W]_n = [Y_{t+1}]_n$, are used to train the set of D_y dynamic metamodels (Eq.(6.3)).

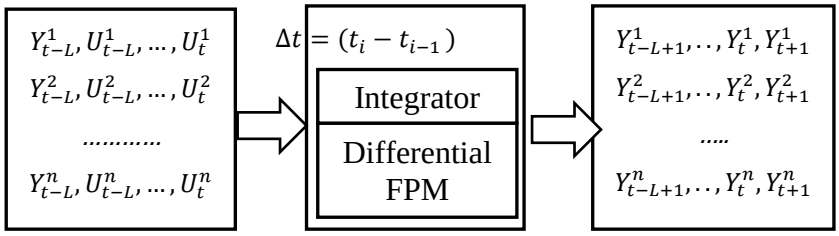


Figure 6.2. Scheme of the proposed dynamic DOCE.

The sampling procedure becomes simpler when no lag exists (i.e. Markovian process, *L=0,*). Hence, the DOCE technique is used to directly select/design a sampling plan $[X]_n = [Y_t, U_t]_n$. After that, *n* simulation runs are carried out using the process FPM in order to obtain the dynamic model output values $[W]_n = [Y_{t+1}]_n$.

6.3.4 Random input-output signals

In common practical situations, a FPM of the process may not be available and, consequently, the selection of the best training data through the application of the proposed DOCE procedure is not possible. Therefore, this work also considers cases where only input-output signals, measured and collected from the process by the physical sensors network are available. We mimic this situation through considering the process FPM as a real plant that generates these input-output data signals.

The first step in the generation of input-output signals is the synthesis of a piecewise-constant set of the process control inputs $U_t \in R^{D_u}$, which are composed by random step changes of the control input values along the time within the allowable control limits $U_{t\,min}$: $U_{t\,max}$. Each step change is expected to hold for some intervals, Δt, to catch the entire dynamic conduct of the process outputs corresponding to this step change. At the same time, the number of sampling periods over which the control input values hold should not be large, in order to avoid gathering redundant information about the steady-state mode of the process. The synthesized control input signals are, then, simulated by the process plant (i.e., model) in order to obtain the corresponding process output signals, to which Gaussian noise is added to emulate the sensors noise. The initial values of the process output signals are selected to be in the middle of their estimated variability domain, in order to maximize the likelihood that during their evolution they could span the sub-regions of the whole domain. These input-output signals, U_t, Y_t, are used to train the system of dynamic surrogate models considering a suitable lag, L.

Usually, the domain within which the process control inputs are allowed to be manipulated $U_{t\,min}$: $U_{t\,max}$ is known from the process operational specification. However, the variation domain of the process outputs [$Y_{t\,min}$: $Y_{t\,max}$] should be also checked in front of the recorded process historical data. In the case of the considered simulated case studies, the domain $Y_{t\,min}$: $Y_{t\,max}$ has been estimated through several trial and error simulations, using control profiles within the specified limits of the process control inputs. Also, it is worth to mention that the time step length Δt is conditioned by the subsequent application of the multivariate dynamic models. For example, if these dynamic models are to be employed for monitoring, fault detection and diagnosis, or model predictive control applications, Δt will be the sampling period over which the process must be supervised or controlled. In this work, we have considered the same Δt previously used in the literature for each one of the addressed case studies.

6.4 APPLICATIONS

In this section, three benchmark models from the chemical process engineering literature are used to evaluate the proposed modeling methodology, including the sampling procedure, and to compare different metamodels types. These benchmarks are representative examples of nonlinear dynamic systems from three different sub-domains, namely, biochemical, industrial and petrochemical engineering.

The first case involves the model of a continuous bioreactor system that has been considered in different dynamic modeling and control studies, e.g., for data-driven univariate dynamic modeling (Azman & Kocijan, 2007), Quasi-sliding mode control (Cho, et al., 2007), and for the design of nonlinear observers (Gauthier, et al., 1992). The second application considers the model of a three-tank system that has been commonly used as a benchmark in different monitoring, control and fault detection and diagnosis studies (Frank & Ding, 1997; Kouadri, et al., 2012; Sarailo, et al., 2015; Patton, et al., 1994). The third case study involves the model for a shale-oil pyrolysis batch system that has been frequently addressed as an example of batch processes dynamic optimization (Wen & Yen, 1977; Carrasco & Banga, 1997).

As previously mentioned, in all these case studies two application scenarios will be considered: the first one would mimic a realistic situation where only input-output signals are available for training the models (see Section 6.3.4) and, thus, the FPM is used as the process plant from which these signals are collected. The second scenario assumes that the FPM is available for the application of the proposed DOCE procedure in order to optimally select the training data (see Section 6.3.3). Finally, in both scenarios, the trained dynamic models are tested with a set of totally new data, independently generated in the form of input-output

signals. The dynamic models are harnessed for forecasting the process output values, given the values of the validation control inputs, by interacting in a coordinated way through the recursive time integration process proposed (Section 6.3.2). Finally, the NRMSE (Eq.(6.5)) is calculated between the predicted outputs and the corresponding known real values.

6.4.1 Bioreactor

A bioreactor consists in a system inside which microorganisms grow by feeding on the substrate in order to produce the desired product. The difficulties to model the biochemical dynamics associated to the involved processes, usually depending on many factors and conditions not easy to control, convert such systems in challenging situations where to test nonlinear dynamic modeling methods and their applications (Gauthier, et al., 1992; Cho, et al., 2007; Azman & Kocijan, 2007). A second-order discrete dynamic model of the bioreactor is considered to describe the evolution of the concentrations of the microorganisms, C_m, and the substrate, C_s, inside the reactor, which are affected by the reactor outlet flowrate, U, as detailed by Eqs.(6.6):

$$\left.\begin{aligned} C_{m(t+1)} &= C_{m(t)} + 0.5\,\frac{C_{m(t)}\,C_{s(t)}}{Cm_t + \ C_{s(t)}}C - 0.5\,U_{(t)}C_{m(t)} \\ C_{s(t+1)} &= \ C_{s(t)} - 0.5\,\frac{C_{m(t)}\,C_{s(t)}}{C_{m(t)} + \ C_{s(t)}}C - 0.5\,U_{(t)}C_{s(t)} + 0.05\,U_{(t)} \end{aligned}\right\} \quad (6.6)$$

The objective is building a group of data-driven models (Eqs.(6.7)), which are able to accurately approximate the bioreactor output evolution, $[C_{m(t+1)}, C_{s(t+1)}]$:

$$\left.\begin{aligned} \hat{C}_{m(t+1)} &= f_1\big(C_{m(t)}, \dots C_{m(t-L)},\ C_{s(t)} \dots,\ C_{s(t-L)}, U_{(t)}, \dots, U_{(t-L)}\big) \\ \hat{C}_{s(t+1)} &= f_2\big(C_{m(t)}, \dots C_{m(t-L)},\ C_{s(t)}, \dots,\ C_{s(t-L)}, U_{(t)}, \dots, U_{(t-L)}\big) \end{aligned}\right\} \quad (6.7)$$

As previously mentioned, the situation where only signals measured from the process plant are available $[U_{(t)}, C_{m(t)}, C_{s(t)}]$ is first considered for model training. Thus, a flowrate signal, $U_{(t)}$, is synthesized by arbitrarily changing its amplitude along the time, where every change lasts over 20 sampling intervals (Figure 6.3-left). The amplitude values of the step changes are randomly chosen within the known operating range [0:0.7] of the outlet flowrate, $U_{(t)}$.

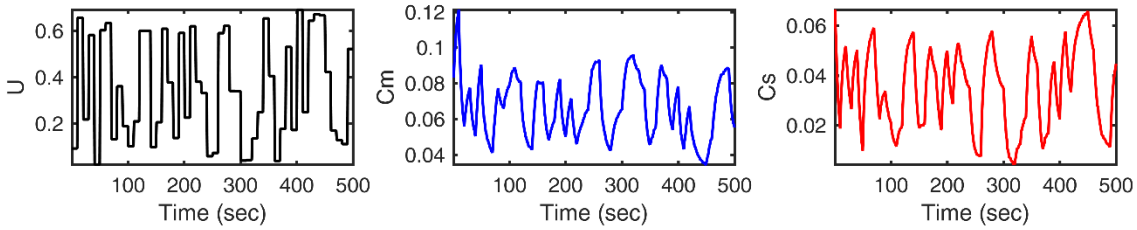


Figure 6.3. Training signal (bioreactor).

This outlet flowrate signal is introduced to the process FPM (Eq.(6.6)) in order to obtain the corresponding process output signals: concentrations of microorganisms, $C_{m(t)}$, and substrate, $C_{s(t)}$. To these calculated values, $C_{m(t)}, C_{s(t)}$, a Gaussian noise $\mathcal{N}(\mu = 0, \sigma = 0.0025\%)$ is added to emulate the kind of information which would be available in this case (Figure 6.3-(middle, right)), where σ is a percentage of the variability domain ([0: 0.15, 0:0.15]) of these variables, $C_{m(t)}, C_{s(t)}$. As previously mentioned in Section 6.3.4, the variation ranges of the output, $C_{m(t)}$ and $C_{s(t)}$ are estimated by carrying out different trial and error simulations using random values of the outlet flowrate, whose variation range is already specified, [0:0.7]. Besides, the initial values of the substrate and microorganisms concentrations, $[C_{m(0)}, C_{s(0)}]$, are selected to be in the middle of their variation ranges.

In parallel, a second situation where the training data is generated by means of the proposed sampling procedure for the dynamic modeling has also been considered. Hence, the Hammersley technique is used to sample over the expected variation domain of the dynamic models input, [0: 0.15, 0: 0.15, 0: 0.7], so as to generate a sampling plan which includes 300 sample points (input values combinations), as described in Section 6.3.3. It should be noted that a different sampling plan is designed for each one of the different lag values considered, since a different lag implies a different number of the dynamic model inputs (i.e. model delayed input).

The procedure application becomes straightforward when no lag is considered ($L = 0$): the DOCE is used to design a sampling plan over the dynamic models input variables $[C_{m(t)}, C_{s(t)}, U_{(t)}]$ and, then, the FPM is used to simulate the model output $[C_{m(t+1)}, C_{s(t+1)}]$; after that, the input-output training data matrices, $[C_{m(t)}, C_{s(t)}, U_{(t)}]_{300} - [C_{m(t+1)}, C_{s(t+1)}]_{300}$, are used to train the models. However, if a lag is considered, just for example, $L = 1$, the Hammersley technique used to design a sampling plan should only consider the independent inputs of the dynamic model, $[C_{m(t-1)}, C_{s(t-1)}, U_{(t-1)}, U_{(t)}]$, and the FPM is employed to simulate the dependent inputs, $[C_{m(t)}, C_{s(t)}]$, of the dynamic model and also the model output, $[C_{m(t+1)}, C_{s(t+1)}]$, as described in Section 6.3.3. Similarly, in this case, a Gaussian noise with the same mean and standard deviation is added to the all process output

data. Finally, the input-output training data matrices, $\left[C_{m(t)}, C_{m(t-1)}, C_{s(t)}, C_{s(t-1)}, U_{(t)}, U_{(t-1)}\right]_{300} - [C_{m(t)}, C_{s(t)}]_{300}$, are obtained.

Both training datasets (input-output signals or DOCE) have been used to train different groups of the multivariate models in Eqs.(6.7), considering the OK and ANNs techniques and various lags ($L = 0, 1, 2\ or\ 3$).

In case of the ANN, its structure has been selected by a search procedure, trying to balance the accuracy and simplicity of the resulting network. Specifically, for any of the models in Eqs.(6.7), four different ANNs-based dynamic models, corresponding to four different lags ($L = 0, 1, 2\ or\ 3$), have been fitted. Since in each case the number of the model inputs will be different, a single fixed ANN structure is not likely to be suitable for all these different dynamic models. In this case, a two layer ANN is used, where the number of neurons in each layer equals to double of the number of input variables of the dynamic model. Besides, a log-sigmoid transfer function is used for the hidden layer neurons, whereas a linear transfer function is used for the output layer. The network training is trained by means of Bayesian regularization backpropagation algorithm, which updates the weights and biases according to Levenberg-Marquardt optimization. This training algorithm usually provides the ANNs with good generalization properties. Again, it is worthy to stress that the selection of the ANN structure and configurations is a time and effort consuming task, even when addressing a low dimensional problem, as the case in hand. This challenge will be magnified as the problem dimensionality and/or the number of training data increases.

Regarding the OK-based models, the "*fmincon*" algorithm for nonlinear optimization of the Matlab optimization toolbox is used to tune the parameters [ξ_l, λ] (see Section 6.2.2). Unlike the ANN, all the OK parameters are automatically optimized. However, a main obstacle which complicates the fitting of the OK is the choice of proper initial values necessary for starting the optimization search: a derivative-based optimization algorithm is relatively fast but it can, readily, end up at a local optima, because of the intricacy of the likelihood function. In this work, few optimization runs (each departing from distinct initial values of the parameters) are considered, to ensure effective training of the OK. Although derivative-free optimizers (e.g., genetic algorithms, swarm intelligence-based algorithms) guarantee global search, their search mechanism may demand a huge computational burden considering the expensive evaluation of the likelihood function (see Section 6.2.2). For assessing the trained models performance, two validation signals have been randomly generated in the same previously mentioned manner (Section 6.3.4), where the amplitude value of the control scenarios (reactor outlet flowrate, U) has been randomly selected within the specified domain [0:0.7]. However, the time length over which each amplitude value holds has been selected

differently for each control scenario. The objective is to assess the accuracy and robustness of the multivariate metamodels under different operational conditions and dynamics (Figure 6.5-top solid black lines) and also to avoid any correspondence with the training conditions.

The dynamic metamodels are harnessed to emulate the evolution of the microorganisms concentration, $C_{m(t+1)}$, and substrate concentration, $C_{s(t+1)}$, along the entire time period (five hundred steps) of each of the two validation scenarios of the output flowrate, U, (Figure 6.5) through the recursive procedure illustrated in Section 6.3.2. Table 6.1 and Figure 6.4 illustrate the NRMSE of the multivariate dynamic models $\hat{C}_{m(t+1)}$ and $\hat{C}_{s(t+1)}$ when they are trained using the considered techniques (i.e.: ANN, OK), lags (L=0, 1, 2, 3) and procedures for training data selection (input-output signals, DOCE).

Table 6.1. NRMSE (%) of the multivariate dynamic metamodels (bioreactor).

Training data type	***Lag***	$\widehat{C}_{m(t+1)}$		$\widehat{C}_{s(t+1)}$		***Average*** $(\widehat{C}_{m(t+1)}$ & $\widehat{C}_{s(t+1)})$	
		OK	***ANN***	***OK***	***ANN***	***OK***	***ANN***
Signal	**0**	4.0	2.9	3.1	3.0	3.5	3.0
	1	3.4	4.6	3.0	2.7	3.2	3.6
	2	2.9	3.7	3.0	3.0	2.9	3.4
	3	2.9	4.4	3.0	2.9	3.0	3.6
						$\boldsymbol{\mu = 3.2}$, $\boldsymbol{\sigma = 0.2}$	$\boldsymbol{\mu = 3.4}$, $\boldsymbol{\sigma = 0.3}$
DOCE	**0**	2.3	1.2	0.4	0.3	1.4	0.8
	1	0.7	2.1	0.8	2.3	0.7	2.2
	2	1.6	1.2	0.4	0.3	1.0	0.8
	3	2.3	1.0	0.9	0.4	1.6	0.7
						$\boldsymbol{\mu = 1.2}$, $\boldsymbol{\sigma = 0.4}$	$\boldsymbol{\mu = 1.1}$, $\boldsymbol{\sigma = 0.7}$

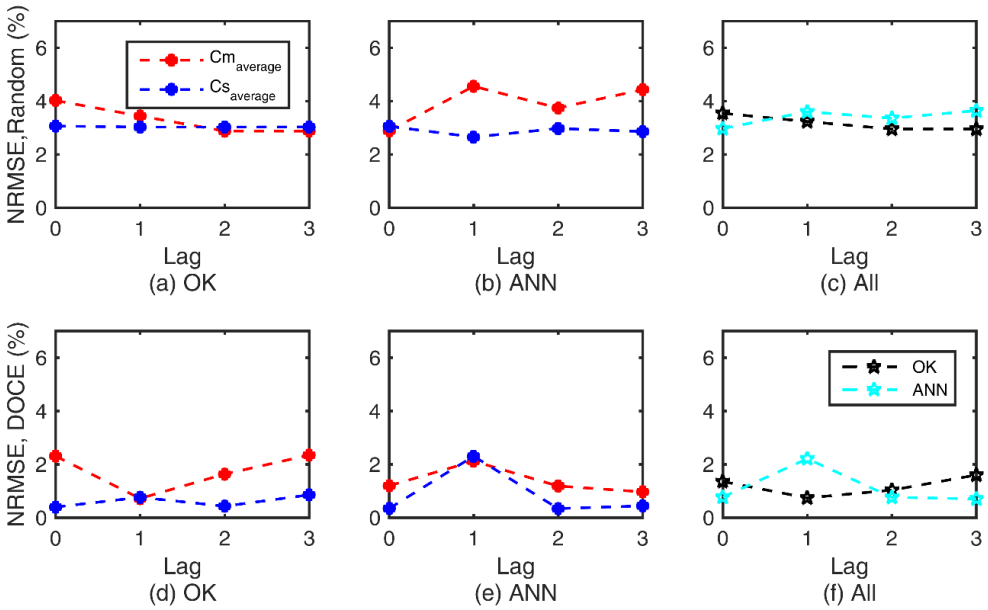


Figure 6.4. NRMSE of the multi-step-ahead predictions of the output variables (C_m, C_s) of the Bioreactor system versus the considered lag of the dynamic models: (a,b,c) training using signals data and (d,e,f) training using DOCE data.

Notice that, generally, all the models trained with the different training data types (signals, DOCE), techniques (ANN, OK) and lags ($L = 0,1,2,3$) achieved very good performances. In particular, the DOCE further enhances the performance of the multivariate dynamic models, even when only 300 data points have been used for training in these cases, in comparison to the 500 training points used in the cases using input-output signals, (see the overall mean, μ, and standard of deviation , σ , of the different sets of models built with different lags). Also, it is worth to highlight that, regarding the signals-based training procedure, the set of multivariate dynamic models based on ANNs with L=0, and OK with L=2 achieved the best performances, respectively NRMSE of 3.0 %, and 2.9 %. In relation to the DOCE training procedure, dynamic models based on ANNs with L= 3, and OK with L= 1 provided the best performance, respectively NRMSE of 0.7 %, and 0.7 %.

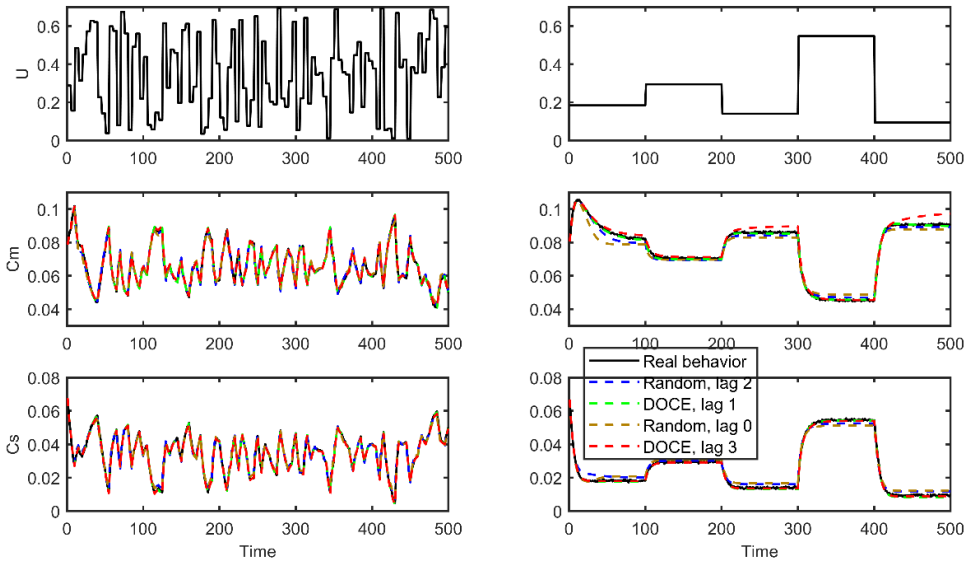


Figure 6.5. Multi-step ahead prediction of the Bioreactor output variables (C_m, C_s) in two validation scenarios (left and right), predicted by different sets of OK-based dynamic models, trained using different data selection procedures and considering different lags of the dynamic models: solid black line is the exact behavior of the process, blue and brown dashed lines are, respectively, the best and worst predictions of the metamodels set trained using input-output signals and the green and red dashed lines are, respectively, the best and worst predictions of the metamodels set trained using the data selected by the proposed DOCE.

Figure 6.5 shows the step-ahead predictions of the microorganisms, $C_{m(t+1)}$, and substrate, $C_{s(t+1)}$, concentrations, corresponding to two validation scenarios by means of the multivariate dynamic models set based on the OK technique. The Figure compares -in terms of the prediction accuracy, see Table 6.1- the best and the worst models in both training cases: using the input-output signal (blue and brown dashed lines for worst and best respectively) and the DOCE (red and green dashed lines for worst and best, respectively). Similar Figures for the dynamic models based on ANN techniques are illustrated in the Appendix. These Figures not only emphasize the very high prediction accuracy of the best multivariate metamodels, but also show that even in the worst modeling trials (e.g.: blue and red dotted lines) quite satisfactory levels of accuracy are achieved for both the OK and ANN cases. The step-ahead prediction of the multivariate dynamic models set based on the ANN technique are shown in Figure 6.6.

Azman et al. (2007) have used the same case study to illustrate their proposal of univariate dynamic modeling based on GP models, where a single-input-single-output system, $U - C_m$, was considered. They used an input-output training signal of 602 samples with added normal random noise to the C_m data ($\mu = 0, \sigma = 0.0025$), and a random validation scenario

that involves 60 time steps. In their work, a dynamic model with a lag $L = 2$ achieved the best prediction accuracy, with a RMSE of 3.44×10^{-3}. Using the methodology proposed in this work, extended prediction capabilities have been achieved with equal (600 samples for the input-output signals training set) or much less (300 samples for the DOCE training set) training data sizes, since all the system outputs (C_m and C_s) have been considered and equivalent RMSEs have been achieved (3% NRMSE that corresponds to a RMSE of 3.1×10^{-3}) over much larger prediction horizons (500 steps-ahead predictions).

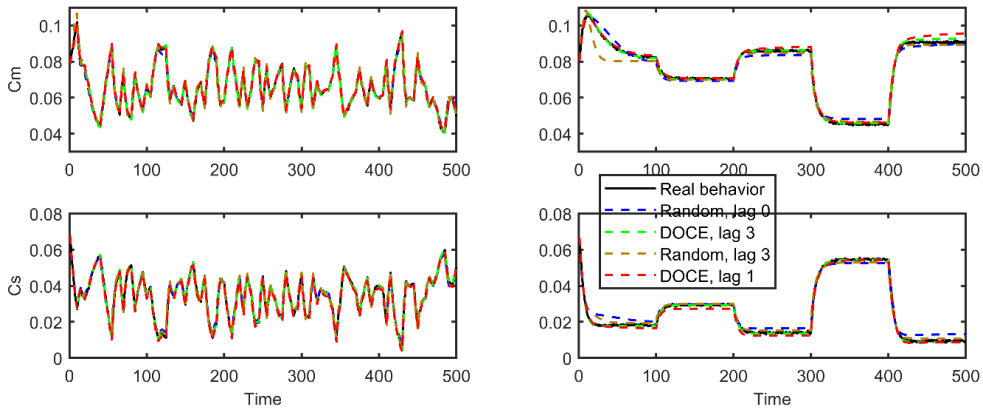


Figure 6.6. Multistep-ahead prediction of the bioreactor output variables (C_m, C_s) in two validation scenarios (left and right) predicted by different sets of ANN-based dynamic models, trained using different data selection procedures, and considering different lags of the dynamic models: the solid black line is the exact behavior of the process, blue and brown dashed lines are the best and worst predictions of the metamodel set trained using input−output signals, respectively, and the green and red dashed lines are the best and worst predictions of the metamodel set trained using the data selected by the proposed DOCE, respectively.

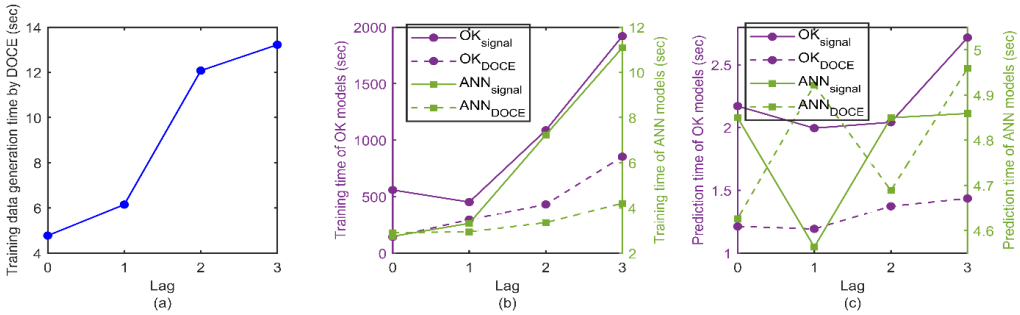


Figure 6.7. Computational times required for the: (a) generation of the training datasets using the proposed DOCE, (b) training of the multivariate dynamic models sets based on OK and ANN and for (c) the prediction of the testing scenarios of the bioreactor case study. (Intel core i5-6200U CPU@2.3GHz.)

Figure 6.7-(a) shows that the computational effort required for training data generation using the proposed DOCE procedure increases with the considered lags in the dynamic models: larger considered lags require more integration steps in the analytical model simulation runs (Section 6.3.3 and Figure 6.2). Notice that the time required for generating the other type of training data (signals) is not illustrated since it is independent of the model lag (an average of 5.6 sec for generating input-output signal as in Figure 6.3). Figure 6.7-(b) shows that, generally, 1) the increase in the dynamic models lag escalates the training time due to the increase in dynamic model input dimensions and, consequently, the growth of the model parameters to be identified, 2) the training time of the OK-based dynamic models (mauve color) are much larger compared with that of the ANN (green color), because of the very expensive evaluation of the objective function involved in its parameters tuning task (the concentrated likelihood function that implies the expensive calculations of the inverse of the correlation matrix $[R]_{n\times n}$, where n is the number of the training data). Nevertheless, given the fact that the training of the multivariate dynamic models is aimed to be an offline task, the high training computational efforts should be affordable. Figure 6.7-(c) shows the average prediction time of the entire 500 steps ahead of one testing profile (as in Figure 6.5) required by the multivariate dynamic models sets with different lags. Notice that the prediction time of the OK-based models are much lower than those of the ANN-based ones, due to the very simple predictor formula associated to OK (see Eq.(6.1)) compared with the relatively expensive calculations required by the ANN to perform the prediction, which include multiplication of matrices of inputs and weights at each layer besides processing their result by the transfer functions. In general, the prediction time is quite suitable for any online application, as one-step ahead prediction requires an order of magnitude of 10^{-3} sec in a simple Personal Computer.

It is worth noticing that, in this case study, as well as in the next ones, the analysis of the computational time are perturbed by different uncontrolled uncertainties and randomness, which lead to some outliers and noise in the trends of the curves in Figure 6.7. These uncertainties include the random initial values of the parameters of the metamodels (OK and ANN), the possible change of the behavior of the objective function involved in the parameter tuning task with the increase of the model lags (i.e. increase in the model input dimensions) and, also, the online availability of the processors and RAM of the computer while performing the calculations.

Finally, it should be emphasized that the performance of the proposed methodology in all cases will be affected by the general limitations and criticalities of any data-driven / machine-learning technique, including the one that refers to the size and the quality of the

training data: to ensure a satisfactory prediction accuracy level, sufficient number of training data should be available, including enough information about the different dynamic conditions/states/scenarios that the process will face. Also, the quality of the training data in terms of the measuring error/noise (unavoidable in real systems) is an important factor affecting the model performance, as the excess of noisy measurements could lead to poor model performance.

Figure 6.8 shows two experiments that address the effects of the training dataset size and noise over the model prediction accuracy in this case, based on the OK model, trained with data generated via the DOCE procedure and considering lag =1 (best overall prediction accuracy in this case). Figure 6.8-(a) shows how the size of the training dataset (fixing the noise standard deviation to 0.0025%) affects the average prediction accuracies of the model. Considering the overall accuracy (black stars) the initial positive effect of increasing the size of the training dataset achieves an optimum situation and, from this point, an increase of the training data does not necessarily enhance further the accuracy (as usually happens with these techniques). Figure 6.8-(b) shows how the noise/error in the data also affects the average prediction accuracy of the models (fixing the number of training data to 300, which was the best value for the nominal conditions, with a noise standard deviation of 0.0025%). The Figure also shows that the methodology behaves robust with respect to the change of the training dataset size and the noise.

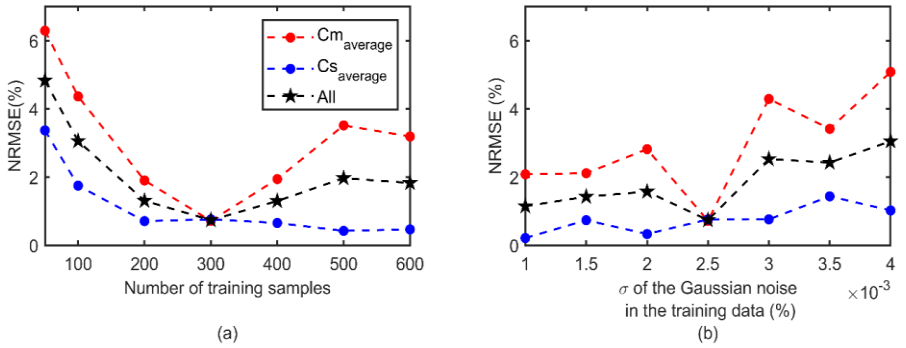


Figure 6.8. Effect of the training dataset size (a) and the amount of noise (b) on the performance of the multivariate dynamic models set based on the OK technique, trained by data selected via the DOCE procedure and considering lag=1.

6.4.2 Three-tanks system

The second application is based on the three-tank system illustrated in Figure 6.9. It is a well-known nonlinear process that has been commonly used as a benchmark in different monitoring, control and fault detection and diagnosis studies (Frank & Ding, 1997; Kouadri, et al., 2012; Sarailo, et al., 2015). Its popularity stems from the fact that it involves

characteristics of fluid distribution systems (tanks, pumps, and pipelines) often encountered in real plants (Patton, et al., 1994; Theilliol, et al., 2002), as cooling water circuits of distillation columns and feed water systems in power stations, etc.

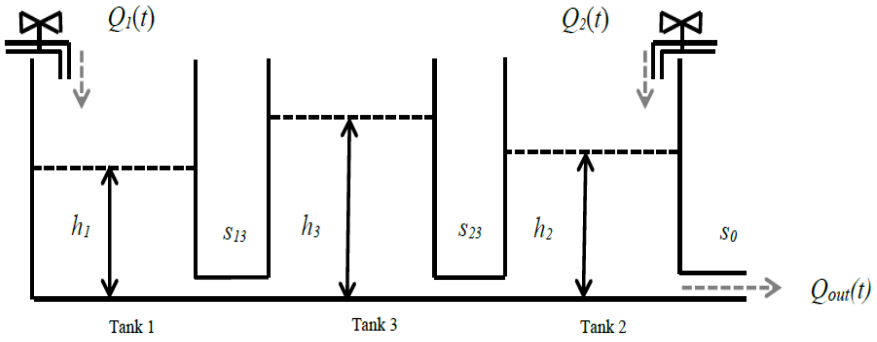


Figure 6.9. Schematic representation of the three-tanks benchmark system.

$$
\begin{aligned}
A\frac{dh_1}{dt} &= -a_1\, s_{13}\, sgn(h_1 - h_3)\sqrt{2g|h_1 - h_3|} + Q_1 \\
A\frac{dh_2}{dt} &= a_3\, s_{23}\, sgn(h_3 - h_2)\sqrt{2g|h_3 - h_2|} - a_2 s_0\sqrt{2gh_2} + Q_2 \\
A\frac{dh_3}{dt} &= a_1\, s_{13}\, sgn(h_1 - h_3)\sqrt{2g|h_1 - h_3|} - a_3 s_{23} sgn(h_3 - h_2)\sqrt{2g|h_3}
\end{aligned}
\tag{6.8}
$$

The system model (in Eqs.(6.8)) describes the dynamic relations among the levels of the tanks, h_1, h_2, h_3, (the process outputs) and the inlet flowrates, Q_1, Q_2, (the control input), whose limiting value is 0.005 m^3/s . The values of the cross section area of the tanks, *A*, the cross section areas of the connecting pipes s_{13}, s_{23}, s_0, and the flow coefficients a_1, a_3, a_0, can be found in (Theilliol, et al., 2002).

A set of multivariate dynamic models is to be constructed, which describes the step-ahead evolution of the tanks levels $h_{1(t+1)}, h_{2(t+1)}, h_{3(t+1)}$, see Eqs.(6.9). The same general procedure described in Section 6.3 and the application details illustrated in Section 6.4.1 are systematically followed in this case, too.

$$
\left.
\begin{aligned}
\hat{h}_{1(t+1)} &= f_1\big(h_{i(t)}, h_{i(t-1)}, .. h_{i(t-L)}, Q_{j(t)}, Q_{j(t-1)}, .. Q_{j(t-L)}\big) \\
\hat{h}_{2(t+1)} &= f_2(h_{i(t)}, h_{i(t-1)}, .. h_{i(t-L)}, Q_{j(t)}, Q_{j(t-1)}, .. Q_{j(t-L)}) \\
\hat{h}_{3(t+1)} &= f_3(h_{i(t)}, h_{i(t-1)}, .. h_{i(t-L)}, Q_{j(t)}, Q_{j(t-1)}, .. Q_{j(t-L)}) \\
&\text{where } i = 1,2,3, \text{and } j = 1,2
\end{aligned}
\right\}
\tag{6.9}
$$

The first training set is obtained by means of the generating input-output signals including 750 instances (Figure 6.10). Thus, piecewise constant signals of the fluid inlet flowrate, Q_1 *and* Q_2, are composed, where the signal amplitude values are randomly selected

along the time in a constant piecewise manner within the ranges of [0.0: 0.005] m^3/s, and each amplitude change holds for 20 sampling periods. The corresponding output signals, h_1, h_2 and h_3, are obtained by the process FPM simulation, where Gaussian noise of the same magnitude described in Section 6.4.1 is added to them.

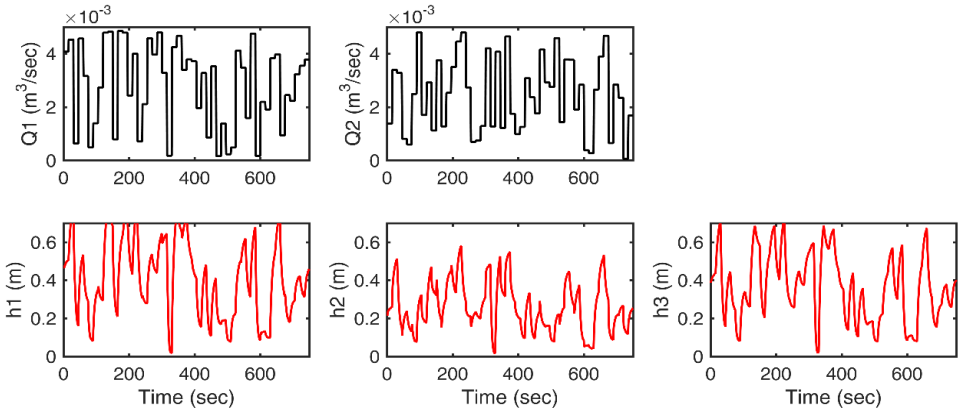


Figure 6.10. Input-output signal of the three-tanks system used for training the set of multivariate dynamic models.

A second training set is again generated following the proposed dynamic DOCE procedure to include 300 samples over the expected variation domain [0: 0.8, 0: 0.8, 0: 0.8, 0: 0.005, 0: 0.005] of the process variables, respectively, h_1, h_2 h_3, Q_1 and Q_2. Gaussian noise with the same mean and standard deviation is added to the process output data and, finally, the input-output training matrices are obtained, $\left[h_{i(t)}, \ldots., h_{i(t-1)}, Q_{j(t)}, \ldots., Q_{j(t-1)}\right]_{300} - \left[h_{i(t+1)}\right]_{300}, i = 1,2,3$ and $j = 1,2$. The set of dynamic models in Eq.(6.9), $[\hat{h}_{1(t+1)}, \hat{h}_{2(t+1)}, \hat{h}_{3(t+1)}]$, is trained using each type of the training datasets, based on the different considered techniques (i.e. OK and ANNs) and different lags. The same setting and guidelines used in Section 6.4.1 for selecting the ANN structure, for customizing its configurations and for tuning the OK models are also considered here.

Table 6.2. NRMSE (%) of the multivariate dynamic metamodels (three-tanks).

Training data type	***Lag***	$\widehat{h}_{1(t+1)}$		$\widehat{h}_{2(t+1)}$		$\widehat{h}_{3(t+1)}$		***Average*** $(\widehat{h}_{1(t+1)}, \widehat{h}_{2(t+1)}, \widehat{h}_{3(t+1)})$	
		OK	***ANN***	***OK***	***ANN***	***OK***	***ANN***	***OK***	***ANN***
Signal	***0***	2.7	1.9	1.8	1.4	2.4	2.0	2.3	1.7
	1	6.6	2.1	2.9	1.9	5.1	2.2	4.9	2.0
	2	5.7	2.8	2.7	2.1	4.4	2.9	4.3	2.6
	3	4.7	2.9	2.5	3.7	3.8	3.0	3.7	3.2
								$\boldsymbol{\mu = 3.8}$, $\boldsymbol{\sigma = 1.1}$	$\boldsymbol{\mu = 2.4}$, $\boldsymbol{\sigma = 0.7}$
DOCE	***0***	1.7	0.4	0.7	0.2	1.6	0.3	1.3	0.3
	1	0.6	0.2	0.3	0.2	0.5	0.2	0.4	0.2
	2	0.9	0.5	0.4	0.4	0.9	0.5	0.7	0.5
	3	0.8	1.6	0.6	1.4	0.7	1.7	0.7	1.6
								$\boldsymbol{\mu = 0.8}$, $\boldsymbol{\sigma = 0.4}$	$\boldsymbol{\mu = 0.7}$, $\boldsymbol{\sigma = 0.6}$

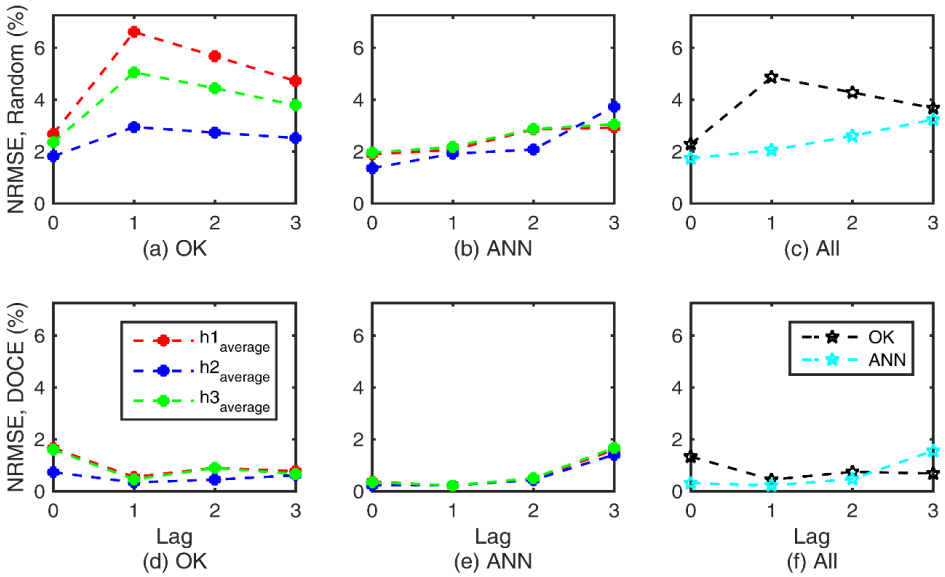


Figure 6.11. NRMSE of the multi-step-ahead predictions of the output variables (h_1, h_2, h_3) of the three-tanks system versus the considered lag of the dynamic models: (a,b,c) training using input-output signals data and (d,e,f) training using DOCE data.

Again, two validation signals, generated as described in Sections 6.3.4 and 6.4.1, are used to assess the fitted dynamic models (Figure 6.12). It deserves to emphasize that the amplitude values of the validation control scenarios (inlet flowrates, Q_1 and Q_2) have been randomly chosen within the specified domain [0, 0.005] m^3/s and the time length over which amplitude values hold has been selected differently for each scenario (see Figure 6.12 top four subplots). Table 6.2 and Figure 6.11 show the low NRMSE of the multivariate dynamic models $\hat{h}_{1(t+1)}$, $\hat{h}_{2(t+1)}$ and $\hat{h}_{3(t+1)}$ when they are trained using the considered techniques,

lags and procedures for training data selection. Also, the evolution of the tanks levels along the time predicted by the multivariate dynamic models sets based on the OK and the ANN techniques are shown in Figure 6.12 and Figure 6.13, respectively.

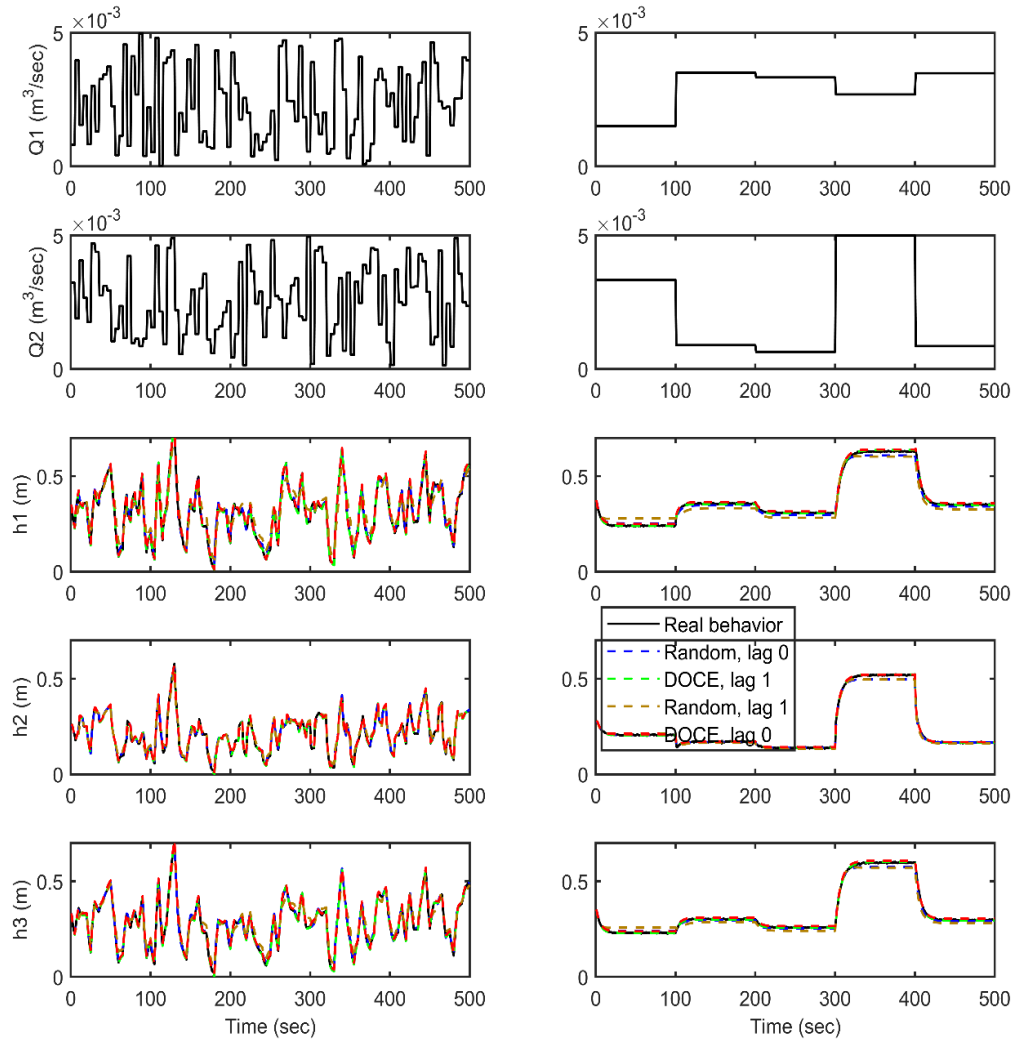


Figure 6.12. Multi-step ahead prediction of the three-tanks system output variables (h_1, h_2, h_3) in two validation scenarios (left and right), predicted by different sets of OK-based dynamic models, trained using different data selection procedures and considering different lags of the dynamic models: solid black line is the exact behavior of the process, blue and brown dashed lines are, respectively, the best and worst predictions of the metamodels set trained using input-output signals and the green and red dashed lines are, respectively, the best and worst predictions of the metamodels set trained using the data selected by the proposed DOCE.

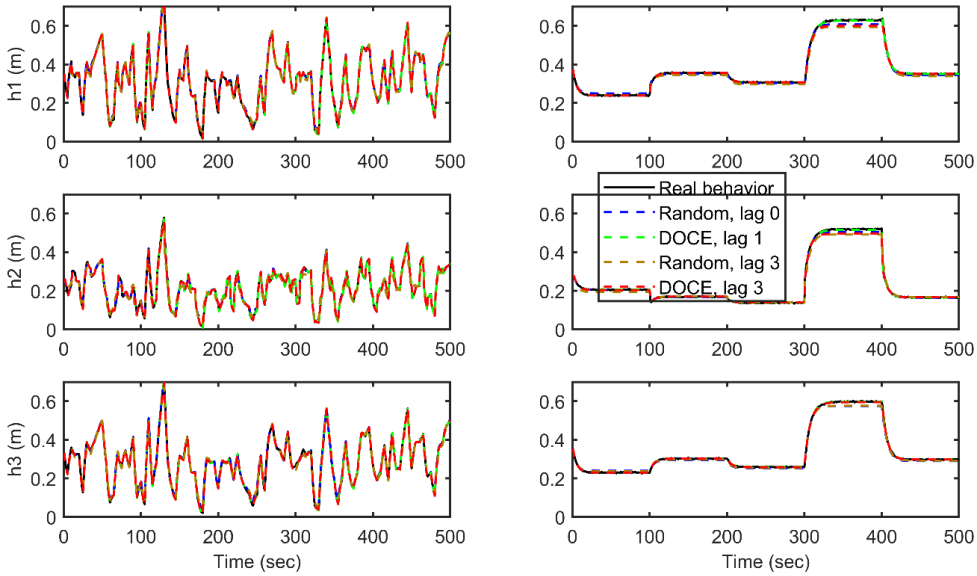


Figure 6.13. Multistep-ahead prediction of the three-tank system output variables (h_1, h_2, h_3) in two validation scenarios (left and right), predicted by different sets of ANN-based dynamic models, trained using different data selection procedures and considering different lags of the dynamic models: solid black line is the exact behavior of the process, blue and brown dashed lines are the best and worst predictions of the metamodel set trained using input−output signals, respectively, and the green and red dashed lines are the best and worst predictions of the metamodel set trained using the data selected by the proposed DOCE, respectively.

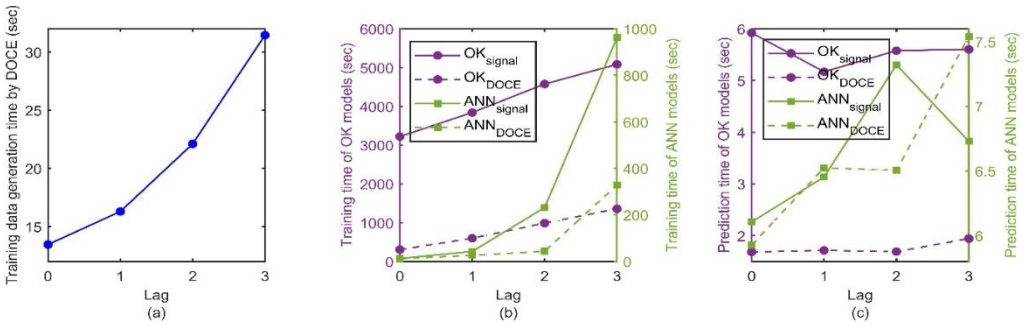


Figure 6.14. Computational times required for the: (a) generation of the training datasets using the proposed DOCE, (b) training of the multivariate dynamic models sets based on OK and ANN and for (c) the prediction of the testing scenarios of the three tanks case study. (Intel core i5-6200U CPU@2.3GHz.)

Figure 6.14-(a) shows the computational effort required for the training data generation using the proposed DOCE procedure. As in the previous case, the time required for generating the other type of training data (input-output signal, see Figure 6.10) is constant (now equals to an average of 9.0 sec.), and the rest of conclusions are also equivalents: Figure 6.14-(b) illustrates the escalation of the training time with the increase of the dynamic models lag and

that the training time of the OK-based dynamic models (mauve color) are much larger relative to the ANN (green color). Figure 6.14-(c) shows the average prediction time of the entire 500 steps ahead of one testing profile. It emphasizes again the capabilities of the dynamic models for real time predictions, requiring an order of magnitude from 10^{-3} to 10^{-2} sec for one-step ahead prediction.

6.4.3 Oil shale pyrolysis

Oil shale pyrolysis is an industrial process that aims at extracting shale oil through the decomposition of the shale. Pyrolysis approximates the natural processing of the organic material in the shale, i.e. kerogen, using higher temperatures to compensate for the geological time frame (Wen & Yen, 1977). Upon heating, kerogen decomposes by consecutive reactions into a benzene-soluble material (pyrolytic bitumen), which, in turn, decomposes to form the final products of oil, gas, and carbonaceous residue on the spent shale (Wen & Yen, 1977):

$$\left.\begin{aligned} Kr &\xrightarrow{k1} Pb \\ Pb &\xrightarrow{k2} Og \\ Kr + Pb &\xrightarrow{k3} Pb + Pb \\ Kr + Pb &\xrightarrow{k4} Og + Pb \\ Kr + Pb &\xrightarrow{k5} Og + Oc \end{aligned}\right\} \tag{6.10}$$

The series of reactions taking place during the process are illustrated in Eqs.(6.10), where Kr is the kerogen, Pb is the pyrolytic bitumen, Og is oil and gas and Oc is the organic carbon residue (Wen & Yen, 1977). The mathematical model in Eqs.(6.11) describes the evolution of the concentrations, $C_{Kr}, C_{Pb}, C_{Og}, C_{Cr}$, where k_i is the specific reaction rate, k_{i0} is its initial value, E_i is the activation energy, R is the gas constant and T is the temperature that can be manipulated within the range of $[698.15 \leq T \leq 748.15]$ (Carrasco & Banga, 1997):

$$\left.\begin{aligned} \frac{dC_{Kr}}{dt} &= -k_1\, C_{Kr} - (k_1 + k_4 + k_5)\, C_{Kr}C_{Pb} \\ \frac{dC_{Pb}}{dt} &= k_1\, C_{Kr} - k_2 C_{Pb} + k_3\, C_{Kr}C_{Pb} \\ \frac{dC_{Og}}{dt} &= k_2 C_{Pb} - k_4\, C_{Kr}C_{Pb} \\ \frac{dC_{Cr}}{dt} &= k_5\, C_{Kr}C_{Pb} \\ k_i &= k_{i0}\, exp^{\left(\frac{E_i}{RT}\right)}, \qquad i = 1,2,3,4,5 \end{aligned}\right\} \tag{6.11}$$

This model has been commonly used for the dynamic optimization of the process (Carrasco & Banga, 1997), aiming at maximizing the pyrolytic bitumen production at the end

of the batch, i.e. $C_{Pb}(t_f)$. With this objective, the optimal batch time, t_f, and the optimal temperature profile over the batch time $[t_0: t_f]$ are to be identified, considering the initial conditions $[C_{Kr}(t_0), C_{Pb}(t_0), C_{Og}(t_0), C_{Cr}(t_0)]$= [1, 0, 0, 0].

In this application, we illustrate the development of a set of dynamic models (Eqs.(6.12)) which is able to accurately approximate the future behavior of the oil shale pyrolysis process. Six different batch runs are simulated, such that each batch corresponds to a different control profile of the temperature, composed as previously mentioned within the known limits $[698.15\,K: 748.15K]$ and, random initial conditions $[C_{Kr}(t_0), C_{Pb}(t_0), C_{Og}(t_0), C_{Cr}(t_0)]$ between the range $[0.95 : 1.05\ ,\ 0 : 0.05,\ 0 : 0.05\ ,\ 0 : 0.05]$. Also, a random Gaussian noise of the aforementioned order of magnitude is added to the output values. It is worthy to mention that the batch time is set to its optimal value identified in the literature (Wen & Yen, 1977), i.e. $t_f = 9.3\ min$, while the sampling period is set to $0.093\ min$.

$$\left.\begin{aligned}
\hat{C}_{Kr(t+1)} &= f_1\left(\begin{matrix}C_{Kr(t)},..,C_{Kr(t-L)},C_{Pb(t)},..,C_{Pb(t-L)},\\ C_{Og(t)},..,C_{Og(t-L)},C_{Cr(t)},..,C_{Cr(t-L)},T_{(t)},..,T_{(t-L)}\end{matrix}\right)\\
\hat{C}_{Pb(t+1)} &= f_2\left(\begin{matrix}C_{Kr(t)},..,C_{Kr(t-L)},C_{Pb(t)},..,C_{Pb(t-L)},\\ C_{Og(t)},..,C_{Og(t-L)},C_{Cr(t)},..,C_{Cr(t-L)},T_{(t)},..,T_{(t-L)}\end{matrix}\right)\\
\hat{C}_{Og(t+1)} &= f_3\left(\begin{matrix}C_{Kr(t)},..,C_{Kr(t-L)},C_{Pb(t)},..,C_{Pb(t-L)},\\ C_{Og(t)},..,C_{Og(t-L)},C_{Cr(t)},..,C_{Cr(t-L)},T_{(t)},..,T_{(t-L)}\end{matrix}\right)\\
\hat{C}_{Cr(t+1)} &= f_4\left(\begin{matrix}C_{Kr(t)},..,C_{Kr(t-L)},C_{Pb(t)},..,C_{Pb(t-L)},\\ C_{Og(t)},..,C_{Og(t-L)},C_{Cr(t)},..,C_{Cr(t-L)},T_{(t)},..,T_{(t-L)}\end{matrix}\right)
\end{aligned}\right\} \quad (6.12)$$

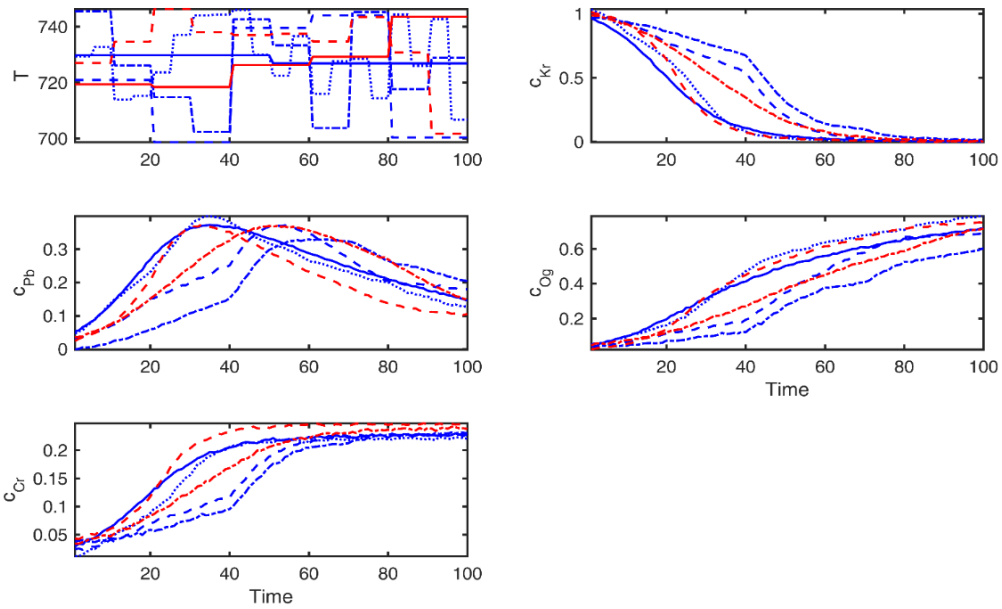


Figure 6.15. Training (blue) and validation batches (red).

Four batches (blue lines in Figure 6.15) are considered as the input-output training set, while two batches (red lines in Figure 6.15) are used for the testing purpose. On the other side, a second training set including 400 samples or instances is generated by the proposed procedure for dynamic DOCE, considering the expected variation domain of the process variables $C_{Kr}, C_{Pb}, C_{Og}, C_{Cr}$, and T: [0 : 1.2, 0 : 0.6, 0 : 1.2, 0 : 0.6, 698.15 : 748.15]

Both types of training data, input-output signals and DOCE, are utilized for fitting the models set in Eqs.(6.12), considering also the different techniques and lags as in the previous sections. The trained sets of models are used to predict the evolution of the process outputs, $C_{Kr}, C_{Pb}, C_{Og}, C_{Cr}$, over 100 time steps, corresponding to each validation scenario of the temperature, T (red lines in Figure 6.15).

The performance of each one of the dynamic models is illustrated in Table 6.3 and Figure 6.16, where the prediction NRMSE is shown for each model independently and for the set of dynamic models. It can be noticed that the multivariate dynamic models possess quite satisfactory level of accuracy (Figure 6.16-(c, f) and the last two columns in Table 6.3), especially taking into account the complex nature of the considered case. This complexity is expressed by the higher dimensionality of the output, the complex reactions mechanisms (see Eqs.(6.10)), the high nonlinear relations in the system (see Eqs.(6.11)) and by the nature of the process as a batch type that often included transient dynamics and sophisticated reaction kinetics and stoichiometry. Besides, the kerogen concentration, C_{Kr}, seems to be the easiest output to be modeled (Figure 6.17, red lines), however, the organic carbon residue, C_{Cr}, represents the most difficult behavior to be captured (Figure 6.17, yellow lines).

Table 6.3. NRMSE (%) of the multivariate dynamic metamodels (oil shale Pyrolysis).

Training data type	***Lag***	$\hat{c}_{Kr(t+1)}$		$\hat{c}_{Pb(t+1)}$		$\hat{c}_{Og(t+1)}$		$\hat{c}_{Cr(t+1)}$		***Average*** $(\hat{c}_{Kr(t+1)}, \hat{c}_{Pb(t+1)}, \hat{c}_{Og(t+1)}, \hat{c}_{Cr(t+1)})$	
		OK	***ANN***	***OK***	***ANN***	***OK***	***ANN***	***OK***	***ANN***	***OK***	***ANN***
Signal	***0***	3.7	2.5	5.4	4.6	3.7	2.5	8.4	6.9	5.3	4.1
	1	2.6	2.6	4.5	8.0	1.6	4.9	8.0	7.4	4.2	5.8
	2	1.7	4.0	3.4	7.4	1.5	4.8	7.9	9.4	3.6	6.4
	3	1.7	1.8	2.9	5.9	1.5	4.0	7.8	6.8	3.5	4.6
										$\mu = 4.2$, $\sigma = 0.8$	$\mu = 5.2$, $\sigma = 1.1$
DOCE	***0***	3.6	1.4	5.6	3.9	4.2	3.0	5.2	5.9	4.6	3.6
	1	4.2	0.4	7.2	0.9	6.0	0.6	8.5	2.9	6.5	1.2
	2	1.4	0.7	3.4	1.1	3.0	0.5	5.2	1.0	3.3	0.8
	3	1.7	0.1	3.0	0.2	3.9	0.2	1.5	0.7	2.5	0.3
										$\mu = 4.2$, $\sigma = 1.7$	$\mu = 1.5$, $\sigma = 1.5$

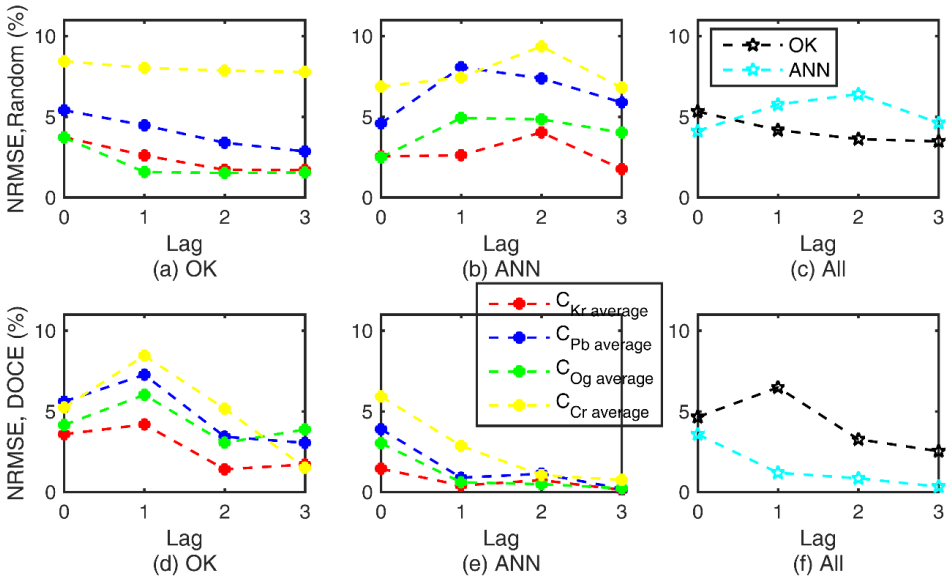


Figure 6.16. NRMSE of the multi-step-ahead predictions of the output variables (C_{Kr}, C_{Pb}, C_{Og}, C_{Cr}) of the oil shale pyrolysis process versus the considered lag of the dynamic models: (a,b,c) training using input-output signals data and (d,e,f) training using DOCE data.

The best performances (input-output training signal) have been achieved by the sets of dynamic models based on ANNs with L=0 and OK with L=3, finding NRMSE of 4.1 %, and 3.5 %, respectively. The dynamic models sets (DOCE training) based on ANNs with L= 3 and OK with L= 3 have provided the best performances, finding NRMSE of 0.3% and 2.5%, respectively. Again, the models trained using data generated by the proposed DOCE procedure exhibit enhanced performance with respect to those trained by the data generated using the input-output signal.

Figure 6.17 shows the evolutions of the kerogen, $C_{Kr(t+1)}$, pyrolytic bitumen, $C_{Pb(t+1)}$, oil and gas, $C_{Og(t+1)}$, and the organic carbon residue, $C_{Cr(t+1)}$, concentrations in two validation batches, predicted by the set of OK-based dynamic models. Similarly, the worst and best performances with respect to each training data type are highlighted by the aforementioned colors. The Figure shows that even in the worst modeling trials (blue and red dotted lines) quite satisfactory levels of accuracy are achieved, especially for the OK and ANN cases. The step-ahead prediction of the ANN-based dynamic models is showed in Figure 6.18.

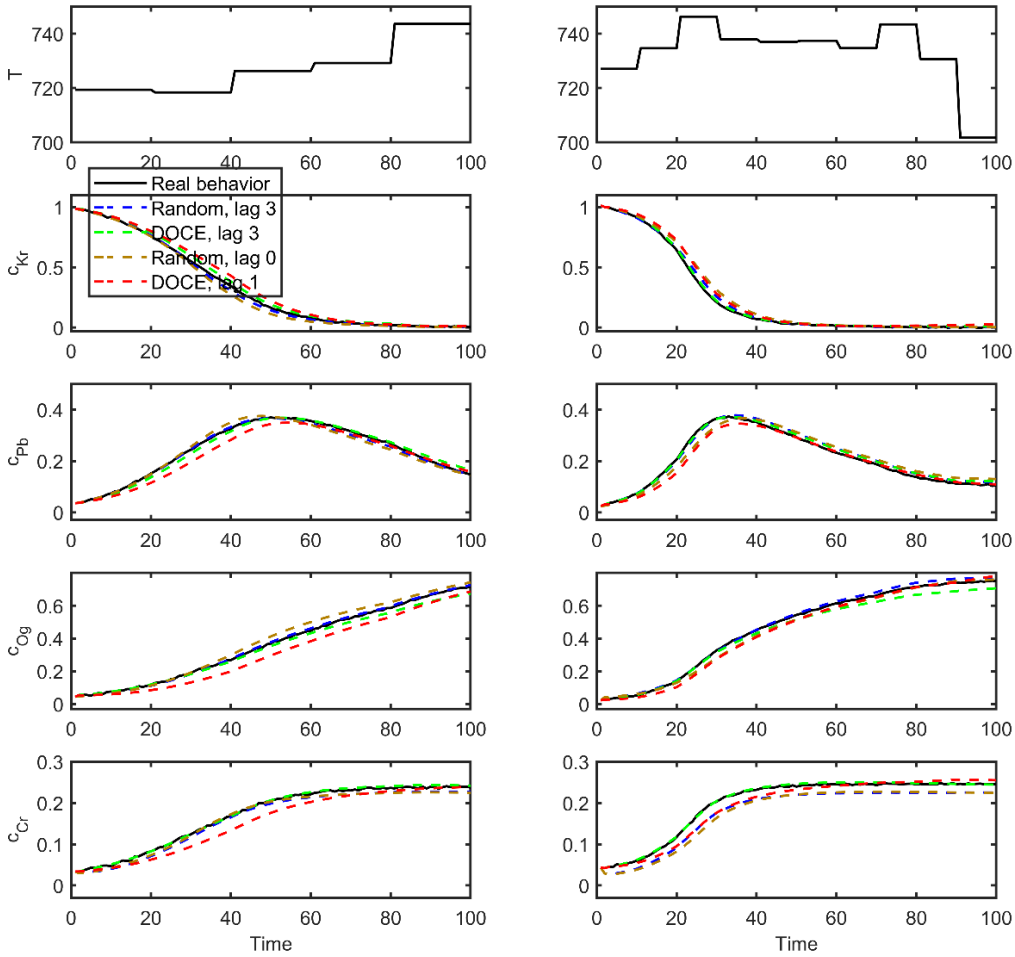


Figure 6.17. Multi-step ahead prediction of the output variables of the oil shale pyrolysis process (C_{Kr}, C_{Pb}, C_{Og}, C_{Cr}) in two validation batches (left and right), predicted by different sets of OK-based dynamic models, trained using different data selection procedures and considering different lags of the dynamic models: solid black line is the exact behavior of the process, blue and brown dashed lines are, respectively, the best and worst predictions of the metamodels sets trained using arbitrary input-output signals, respectively, and the green and red dashed lines are, respectively, the best and worst predictions of the metamodels sets trained using data selected by the proposed DOCE.

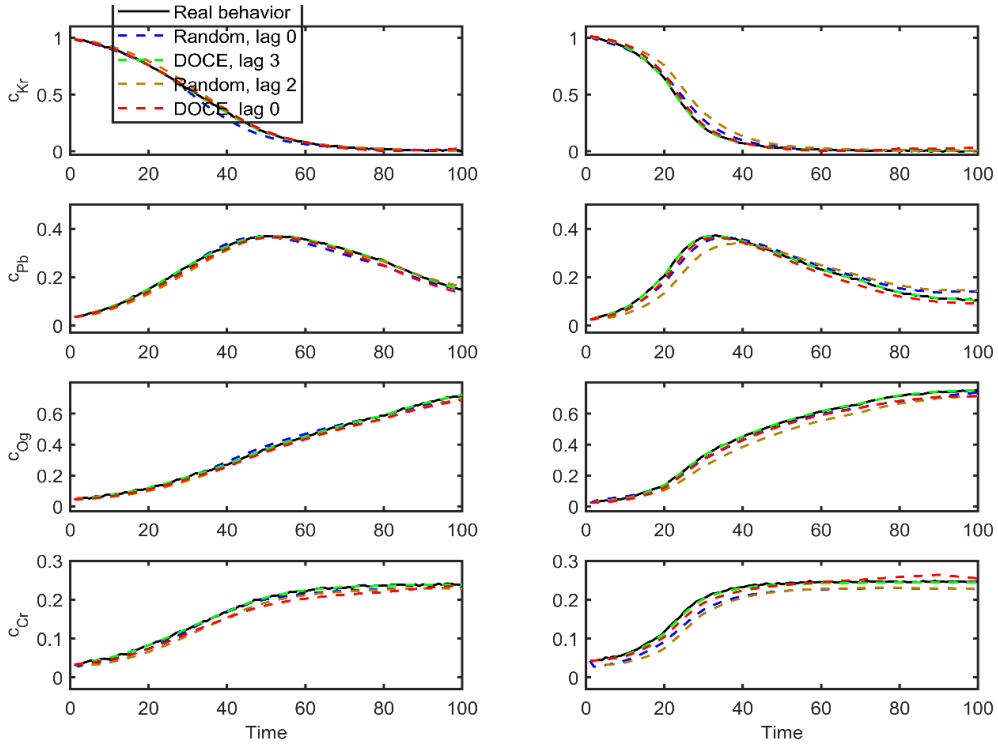


Figure 6.18. Multistep-ahead prediction of the output variables of the oil shale pyrolysis process (C_{Kr}, C_{Pb}, C_{Og}, C_{Cr}) in two validation batches (left and right) predicted by different sets of ANN-based dynamic models, trained using different data selection procedures, and considering different lags of the dynamic models: solid black line is the exact behavior of the process, blue and brown dashed lines are the best and worst predictions of the metamodel sets trained using input−output signals, respectively, and the green and red dashed lines are the best and worst predictions of the metamodel sets trained using data selected by the proposed DOCE, respectively.

Figure 6.19 shows the training data collected by the input-output signal generation (red crosses) and the proposed dynamic DOCE (cyan circles) procedures projected onto some of the metamodels input dimensions. The Figure shows that when the methodology is used for approximating a complex FPM, it is capable of efficiently generating all the possible combinations of the process variables values by the proposed DOCE procedure, in order to collect dataset covering the entire domain of the models input and, consequently, to enhance its prediction accuracy. However, when the methodology is meant to be applied to a real process, the FPM model is considered as a process plant, but with only few input-output datasets available, which have been generated following the procedure in Section 6.3.4 (one/few signal(s) or "profile(s)" evolving through the complete set of feasible situations). See Figure 6.19, where the training data in the latter case (red crosses) represent a small local subset within the entire domain of variability of the model input variables. In this case, these "profiles" have been generated in a random way (see Section 6.3.4) since we do not know the

control mechanism (problem) of each specific process and, moreover, this is the typical procedure used in the literature (Nagy, 2007; Azman & Kocijan, 2007).

For a real situation, where a database of the process variables measurements history is available, the training data selection should cover as much as possible the dynamic conditions of the process, in order to feed the model with sufficient information about the process.

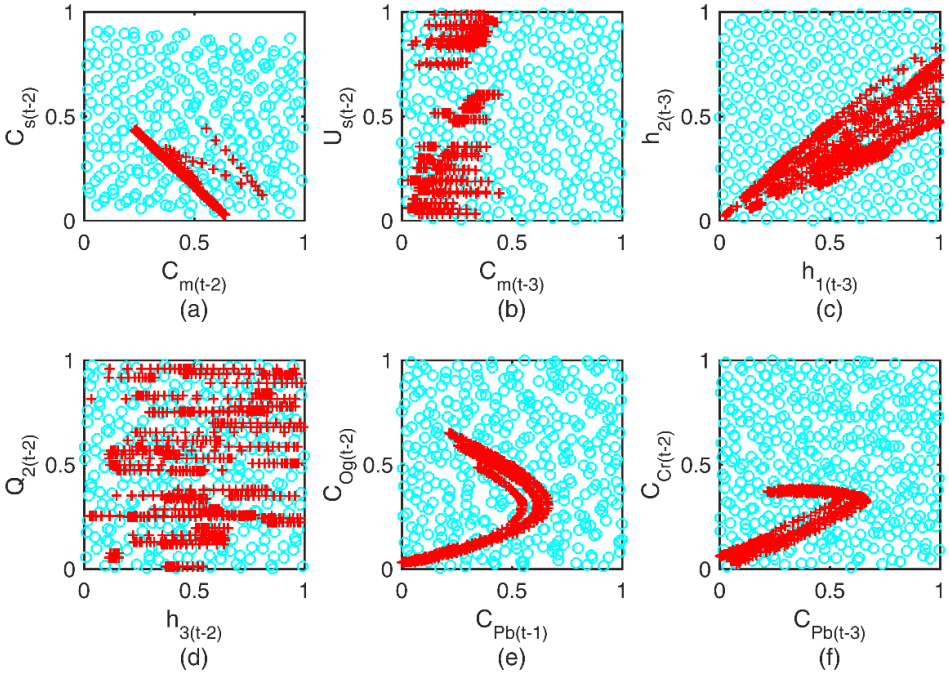


Figure 6.19. Comparison between the training data selected by the proposed DOCE procedure (cyan circles) and the training data in the case of using input-output signals (red crosses), both projected over arbitrary selected pairs of the dynamic models input dimensions: (a,b) bioreactor,(c,d) three-tanks and (e,f) oil shale pyrolysis.

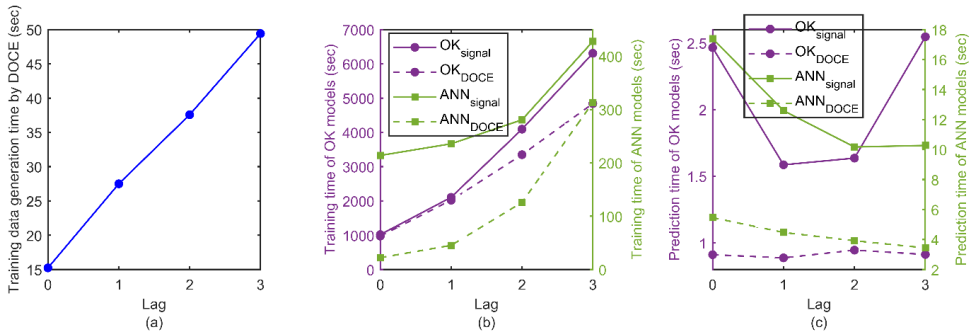


Figure 6.20. Computational times required for the: (a) generation of the training datasets using the proposed DOCE, (b) training of the multivariate dynamic models sets based on OK and ANN and for (c) the prediction of the testing scenarios of the oil shale pyrolysis case study. (Intel core i5-6200U CPU@2.3GHz.)

Figure 6.20-(a) shows the computational time required for the training data generation using the proposed DOCE procedure. The time required for generating the other type of

training data (input-output signal, see Figure 6.15) is (again) constant and equals to an average of 9.0 sec. Figure 6.20-(b) illustrates the escalation of the training time with the increase of the dynamic models lag and that the training times of the OK-based dynamic models (mauve color) are (again) larger relative to the ANN (green color) escalation. Figure 6.20-(c) shows the average prediction time of the entire 100 steps ahead of one testing profile. It emphasizes again the capabilities of the dynamic models for real time predictions, requiring an order of magnitude from 10^{-2} to 10^{-1} sec for one-step ahead prediction.

Finally, it should be mentioned that in all the analyzed cases but, especially, in situations where only few input-output signals are available for the training and/or they may represent a biased or partial view of the process (as in the last case study, see Figure 6.19-(e,f), red crosses), the resulting dynamic models may be very sensible to the eventual evolution of the real process behavior through the time, which may drive it to unexpected/unexplored conditions, either due to the natural evolution of the process (e.g.: heat exchanger fouling, process aging, drifting, etc.), or because a wrong/incomplete selection of the training dataset. In such situations, the dynamic models can perform poorly, because they are going to be applied outside the domain of knowledge/information on which they have been trained. An online updating mechanism that continuously feeds/updates the dynamic models with new data (information) collected from the process would be the solution for such problem.

In this sense, the practical application of the proposed methodology needs to account for the uncertainty or confidence about the model prediction, which should be more reliable when the model is to be used for control and optimization (e.g.: in order to assess how the control actions will tolerate the model predictions errors, or in order to detect that the process is evolving into a new or not well described working area). Ensemble and Monte-Carlo-based methods are suitable for the uncertainty quantification of data-driven models.

6.5 CONCLUSIONS

This work presents a robust and generic methodology for data-driven multivariate dynamic modelling and multi-step ahead prediction of nonlinear chemical processes using surrogate models. The proposed methodology utilizes surrogate models for building a group of NARX models, each of them able to predict the evolution of one output as a function of the other inputs and outputs of the process. The set of multivariate dynamic models are, then, used to forecast the process outputs along larger time intervals, through a recursive and inter-coordinated prediction scheme. The methodology also offers a new procedure for training data selection for dynamic modeling, based on the "design of computer experiments" technique when a FPM of the process is available.

The application of the proposed methodology is illustrated through three case-studies of nonlinear dynamic processes selected from the process engineering literature, including a bioreactor, three-tanks and an oil-shale pyrolysis batch reactor. The results make explicit the promising capabilities of the developed multivariate dynamic models in terms of: 1) a high prediction accuracy, 2) the capability of simulating complex dynamic profiles over large prediction time horizons, and 3) the generality and robustness required to handle cases of different nature (biological, industrial and chemical systems), integrating different metamodel types (ANN and OK), managing situations based on either FPM approximations or where only a limited set of process input-output signals are available, exhibiting very good behavior with respect to the sensitivity against the training data size and the noise in the training data.

The proposal extends the capabilities of the OK techniques (until now only proposed in simpler dynamic situations) and efficiently introduces them to full dynamic scenarios, showing very competitive characteristics with respect to other leading techniques such as ANNs, in terms of accuracy and, more significantly, in terms of flexibility and systematic tuning of parameters. The only disadvantage is the relatively high computational effort required for fitting.

The sets of multivariate dynamic models provided by the methodology fit very well with the requirements and needs of different engineering applications as model predictive control, dynamic optimization, monitoring, etc., where the future values of many process outputs must be accurately and rapidly predicted.

The good results obtained with models trained with a limited quantity of input-output data justify the generalization of the message and the potential applicability of the proposed procedure to situations when no FPM is available or the conditions from the training data may significantly change, although this is to be further investigated. On the other hand, the main issues which main appear during the application of the proposed methodology, associated to the availability, representativeness and quality of the training data, and common to the application of machine-learning techniques, represent potential lines of future research, such as the development of online updating method to overcome the process evolution, or the development of prediction assessment methods in dynamic environments.

Chapter 7: Dynamic Optimization of Batch Processes Based on Multivariate Dynamic Surrogate Models

This Chapter presents a novel data-driven sequential dynamic optimization methodology applicable to solve optimal control problems of complex highly nonlinear processes. The methodology is based on the use of kriging surrogate models to obtain accurate, robust and computationally inexpensive multivariate dynamic models (as presented in Chapter 6), built using input-output training data eventually generated from the simulation of the complex First Principle Model (FPM) of the process (mathematical or analytical model), or collected from the real system. Then these data-driven multivariate dynamic models take the place of the complex FPM of the process in a well-tailored computational scheme of sequential dynamic optimization. The results of applying this approach to three well-known problems from the process systems engineering area are compared with the ones obtained using the corresponding FPMs, showing how the proposed approach significantly reduces the computational effort required to get very accurate solutions, and so enables the use of dynamic optimization procedures in applications where robustness and immediacy are essential practical constraints.

7.1 INTRODUCTION

A key element to improve system performance in the process industry (e.g., to reduce the operating cost, to increase the production yield, or to ensure product quality) is the fast and reliable identification of the optimal time profiles of the process control variables to be followed/applied (e.g., equipment feed rates, cooling temperature profiles, etc.). In most cases, the optimum profiles are scenario-dependent (i.e., the profile must be adapted according to the quality requirements, the characteristics of the raw materials, the economic conditions, etc.); additionally, a large number of uncertain variables must be often contemplated, and the relations between the control variables and the performance are usually difficult to model in detail. In such cases, a priori calculations (steady-state optimization) are not helpful and the process engineer must periodically (even continuously) solve the associated model-based control problem (open loop optimal control (Banga, et al., 2005)), which requires going through a complex mathematical procedure (dynamic optimization) that involves a dynamic

model (usually in the form of differential equations), a multifaceted objective (usually based on the final state of the system, but, also, on its evolution), and a set of control variables which eventually can be changed along the time (Diehl, et al., 2006; Wang, et al., 2017).

State-of-the-art methods for solving dynamic optimization problems of industrial relevance rely on the application of the so-called direct methods (Banga, et al., 2005), based on the discretization of the time domain and the transformation of the original infinite continuous optimal control problem into a finite constrained NonLinear Programming (NLP) problem, which is, then, solved by appropriate numerical nonlinear optimization tools (e.g., Sequential Quadratic Programming (SQP), etc.). Alternatively, indirect methods use the analytical necessary conditions from the calculus of variations to formulate a boundary value problem, which is usually very difficult to solve (Banga, et al., 2005) and requires a deep a priori knowledge of the nature of the problem (initialization, constraints structure, etc.), so they are usually inapplicable to the industrial practice.

Direct methods are further classified according to the elements finally discretized: sequential approaches (also known as Control Vector Parameterization (CVP) approaches) discretize only the control variables in the form of piecewise low order polynomials, and then a NLP optimization problem is carried out in the space of the discretized control variables, which requires the successive evaluation (simulation runs) of the nonlinear process model during its solution. On the contrary, simultaneous approaches discretize both control and state variables by approximating them by a family of polynomials on finite elements (Biegler, 2007), so they avoid the inner evaluation of the differential process model, although they result in a NLP problem of a very large-scale (due to the presence of state variables together with the control variables as optimization variables (Carrasco & Banga, 1997; Banga, et al., 2005)) and require the introduction of extra constraints to enforce the continuity of the discretized state variables (Diehl, et al., 2006; Wang, et al., 2017).

The sequential strategy is straightforward and relatively easy to construct and to apply, and results in a NLP optimization problem of a much reduced size (Carrasco & Banga, 1997; Banga, et al., 2005; Diehl, et al., 2006; Biegler, 2007). However, a major challenge that faces the sequential approach is the required huge computational effort, associated to a large number of evaluations of the nonlinear process FPM, since each evaluation includes the expensive integration of this differential model using complicated integration techniques (Diehl, et al., 2006; Biegler, 2007). This challenge is amplified in case of complex, highly nonlinear and/or large-scale problems (e.g., chemical processes) (Srinivasana, et al., 2003), and the computational cost becomes unaffordable if a fast identification of the process control profiles is required, which is the case in many industrial applications (e.g., transitions between desired

operating conditions, response to sudden disturbances or unexpected events, online optimization - model based control, etc.) (Nagy, 2007).

Recently, the Surrogate Based Optimization (SBO) approach is receiving a great deal of interest for facing similar challenging situations, when the optimization problem involves a complex FPM of the process (Caballero & Grossmann, 2008; Quirante & Caballero, 2016). The SBO approach performs the optimization task relying on simplified, but accurate, data-driven models (also called surrogate models or metamodels, e.g., artificial neural networks), which are built using input-output data obtained from the simulation of the complex FPM of the process (Nagy, 2007; Forrester & Keane, 2009). However, in the chemical engineering literature, almost all SBO methodologies are limited to steady-state optimization problems (Kajero, et al., 2017). Additionally, most of these SBO methodologies used "static" kriging surrogate models to approximate a complex steady-state FPM of the process (Quirante, et al., 2018).

Kriging models are very competitive to many other data-driven model types in many engineering fields (Forrester & Keane, 2009), because of their specific properties, as high prediction accuracy with a relatively small number of training data, and, specially, their ability to estimate a prediction variance (or error) that represents the prediction uncertainty. However, as mentioned before, the majority of the kriging usage and developments are concentrated in emulating complex "static" models (Fang, et al., 2005; Forrester & Keane, 2009), while engineering systems are of dynamic nature, and so, the use of dynamic models is a must in any control application.

This Chapter presents a novel and efficient data-driven dynamic optimization methodology to solve optimal control problems of complex highly nonlinear chemical processes. The methodology is based on two stages, which include:

- development of simple and accurate MultiVariate Dynamic Kriging (MVDK) models (as presented in Chapter 6), which are able to predict the future values of the process outputs over long time horizons, and
- building an efficient sequential dynamic optimization procedure able to integrate these MVDK models that represent the process model.

The rest of this Chapter is organized as follows. Section 7.2 shows the tools and techniques used to build the methodology. Section 7.3 describes the detailed steps of the proposed methodology. Section 7.4 presents the application of the methodology to benchmark case studies that include the dynamic optimization of different batch reactors. Finally, Section 7.5 concludes the Chapter.

7.2 TOOLS AND TECHNIQUES

A dynamic system is characterized through a set of state variables y_t, evolving from their initial values y_0 over time horizon $[t_0: t_f]$, and being affected by the system inherent dynamics, and a set of control variables u_t, which can be externally manipulated within a certain range during this time horizon, affecting the inherent system dynamics. So, the objective is to find the profile of the control variables to obtain the best value of a certain objective function.

7.2.1 Sequential dynamic optimization

The direct sequential approach to solve dynamic optimization problems is based on the discretization or parameterization of the control variables u_t as piecewise polynomials $[u_1, u_2, u_3 \ \dots, u_{N-1}, u_N]$ (Banga, et al., 2005; Diehl, et al., 2006) by dividing the total time domain $[t_0: t_f]$ into a grid of N equally sized intervals Δt, where $\Delta t = \left(t_j - t_{j-1}\right) = (t_0 - t_f)/N \ \ j = 1,2, \dots N$, where $[t_0 < t_1 < t_2 < t_3 < \dots < t_j < \dots < t_{N-1} < t_N = t_f]$. For simplicity, these polynomials are usually assumed to follow piecewise constant profiles. This has also been the choice in this work. The optimization is carried out in the space of the parameterized control variables u_j only, which became decision variables. In each iteration, the NLP solver updates the values of the discretized control variables u_j, the differential FPM of the process (Eq.(7.2)) is integrated using standard integration algorithms (e.g., Runge-Kutta), the state variables y_t are calculated departing from the known initial conditions y_0, and, finally, the objective function J (Eq.(7.1)) and the constraints g (Eq.(7.3) and Eq.(7.4)) are evaluated (Banga, et al., 2005; Diehl, et al., 2006; Biegler, 2007).

$$\underset{y(t),u_j(t)}{\text{Min}} J = \Phi\, y\left(t_f\right) + \int_{t_0}^{t_f} \varphi[y(t), u_j(t), t]\ dt \tag{7.1}$$

$$\text{S.T.:} \quad \frac{\partial y}{\partial t} = F(y(t), u_j(t))\,, \qquad y(t_0) = y_0 \tag{7.2}$$

$$g(y(t), u_j(t)) \leq 0 \tag{7.3}$$

$$u_{min} \leq\leq u_j(t) \leq u_{max} \leq \tag{7.4}$$

7.2.2 Kriging model construction

The construction of an accurate surrogate model relies on the representativeness of the available input-output training data points. Whenever it is feasible, training points should be selected in such a way that the best representation of the original model behavior is obtained. This selection task is performed using Design Of Computer Experiments (DOCE) techniques

called the "sampling plan design" (Fang, et al., 2005), and it results in a set of input combinations (sampling plan $[X]_{n\times k}$, where n is the number of sample points, and k is the number of input variables x) at which the corresponding output data (response variables data matrix $[W]_{n\times M}$) are obtained. The Space-Filling Latin Hypercube Sampling design (SLHS) has been used in this work as sample plan design technique (Forrester, et al., 2008). More details about DOCE techniques can be found in Section 2.1.

Given a set of n input-output training data $[x_i, w_i]$, $i = 1,2,..n$, $x \in R^k, w \in R$ (considering only one output, for the description simplicity), the OK assumes the predictor $\widehat{w}(x) = \mu_{ok} + Z(x)$, where the constant term μ_{ok} represents the main trend of the system to be approximated, and $Z(x)$ is a deviation from that trend. The deviation $Z(x)$ is modeled as a stochastic Gaussian process with expected value $E(Z(x)) = 0$, and a covariance between two residuals $cov(Z(x_i), Z(x_j))$ that only depends on their corresponding inputs x_i, x_j. Thus it can be calculated as: $cov\left(Z(x_i), Z(x_j)\right) = \sigma_{ok}^2 R(x_i, x_j)$, being σ_{ok}^2 the process variance and $R\left(x_i, x_j\right) = exp\left(-\sum_{l=1}^{k} \xi_l \left|w_{i,l-} w_{j,l-}\right|^{p_l}\right) + \delta_{ij}\lambda$ a correlation function, where, $\xi_l, l = 1, \ldots k$ are the model hyper-parameters, δ_{ij} is the Kronecker delta, p_l are smoothing parameters and λ is a regularization constant that enables the kriging predictor to regress noisy data (Azman & Kocijan, 2007). The kriging predictor is given by Eq.(7.5), where (x^*) is a new interpolating point (different from the training data). In Eq.(7.5), $[r]_{n\times 1}$ is the vector of correlations between the point to be predicted x^* and the original training data points and calculated as $R(x_i, w^*)$, $[R]_{n\times n}$ is the correlation matrix between the training inputs, $[W]_{n\times 1}$ is the vector of the training outputs and $[\mathbf{1}]_{n\times 1}$ is the identity vector.

$$\widehat{w}(x^*) = \mu_{ok} + r^T R^{-1}(W - \mathbf{1}\mu_{ok}) \tag{7.5}$$

This work considers the OK implementation developed by Forrester et al. (2008), because of its high efficiency and applicability. Besides, the "*fmincon*" algorithm included in the Matlab optimization toolbox is used for the maximization (nonlinear optimization) of the concentrated likelihood function. More details about the OK model can be found in Section 2.2.1.

Finally, the surrogate model must be assessed in order to make sure that it exhibits a satisfactory level of accuracy over its input domain (Meckesheimer, et al., 2002). Cross-validation allows the characterization of the surrogate model error without any additional data rather than the original set of sample points (Caballero & Grossmann, 2008). Various techniques of cross-validation have been developed, as the "K-fold cross-validation" and "leave-p-out cross-validation". In this Chapter the "Leave-One-Out Cross-Validation" (LOOCV) is used: in each iteration, one sample point is held out for validation, and the

remaining points are used to fit the surrogate model; the cross-validation error at each iteration is calculated, and the average root mean square error of the cross-validation of the surrogate model is calculated from Eq.(7.6), where *n* is the total number of sample points, and $\hat{x}_i$, x_i are the estimated and the real value- respectively -of the held out point (w_i), (Meckesheimer, et al., 2002).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2} \tag{7.6}$$

7.2.3 Multivariate dynamic kriging models

In the case of a dynamic system, the kriging model(s) must be trained to mimic the incremental evolution of the dynamic system behavior (state variables) over a relatively small and constant time-step Δt. In this case, the surrogate model output $\hat{w}$ in Eq. (7.5) will be composed by the state variables y_j at the time t_j (i.e., $\hat{w} = y_i$) and the surrogate model input variables x will include both the control variables u_j and the state variables at the previous time y_{j-1} (i.e., $x = [u_j, y_{j-1}]$). Hence, the resulting MVDK models will be given by Eq.(7.7).

$$\hat{y}_j = f(y_{j-1}, u_j) \tag{7.7}$$

Assuming that an accurate (but complex) FPM model of the system is available, MVDK models can be easily derived following the idea of using "computer experiments" to generate the training data: As a first step, the range (domain) within which the input variables (the state variables y_{j-1} and the control variables u_j) are expected to change is estimated $[y_{t-min}: y_{t-max}, u_{t-min}: u_{t-max}]$. Then a SLHS sampling plan $[X]_{n\times k}$ is designed over this expected domain. At each row (point) of the sampling plan, a computer experiment (simulation run) is carried out using the original complex FPM over a fixed and relatively small time-step Δt, to obtain the corresponding outputs $[W]_{n\times M}$, where M is the number of outputs (state variables). After fitting the M dynamic kriging models (one model for each state variable), the MVDK models are validated using the LOOCV, to ensure that they possess acceptable range of prediction accuracy.

Once obtained accurate MVDK models, they can be then used in a recursive way to predict or interpolate the entire time series/sequence of outputs $[\hat{y}_1, \hat{y}_2, \hat{y}_3, \ldots., \hat{y}_{N-1}, \hat{y}_N]$ (the dash on the y means it is an estimated value). The recursive dynamic interpolation or emulation starts using the given input values (y_0, u_1) to interpolate or predict $\hat{y}_1$, then with $(\hat{y}_1, u_2)$ to predict $\hat{y}_2$, then with $(\hat{y}_2, u_3)$ to predict $\hat{y}_3$, and so on, until the last interpolation step using $(\hat{y}_{N-1}, u_N)$ to predict $\hat{y}_N$.

To illustrate the capability of the MVDK models to emulate a nonlinear dynamic process for large time domain, we have applied the technique to a tank draining system, which is a well-known example that has been commonly used in control textbooks and software. The system model, Eq.(7.8), has one state variable $y(t)$, which represents the water height inside the tank, and one control variable $u(t)$ which corresponds to the inlet flowrate entering the tank. The water leaves the tank through the bottom part, under the gravity effect. b and a are constants related to the inlet and outlet flowrates respectively, and A represents the cross-section area of the tank. Then, the FPM representing the systems dynamics can be expressed as indicated by Eq.(7.8).

$$\frac{\partial y}{\partial t} = \frac{1}{A}(b\,u(t) - a\,y(t)^{0.5}) \tag{7.8}$$

Since there is one state variable, only one MVDK model is required to emulate the system. The input variable (y_{j-1}, u_j) is expected to vary within the limits of [0: 4, 0: 1.5]. Over this domain, a SLHS plan is designed with 65 sample points. The original model in Eq. (7.8) is used to generate the output matrices $[y_j]$, Δt=*0.5 min*. The surrogate model fitting is achieved through the maximization of likelihood of the observed data, and the obtained values of the surrogate model parameters are $(\mu_{ok}, \sigma_{ok}^2\ [\xi_l])$ = (2.24, 0.48, [8.1, 1.4]). The accuracy of the fitted MVDK model is assessed by the RMSE of the LOOCV technique, through Eq.(7.6). The cross-validation results using the LOOCV method is shown in Figure 7.1-(a), and the root mean square error of the cross-validation is obtained (RMSE = 8.13×10^{-4}). The extremely low value of the RMSE (almost 4 orders of magnitude less than the expected values range) indicates that the fitted MVDK model exhibits a very high accuracy, and it is ready to be used for dynamic emulation.

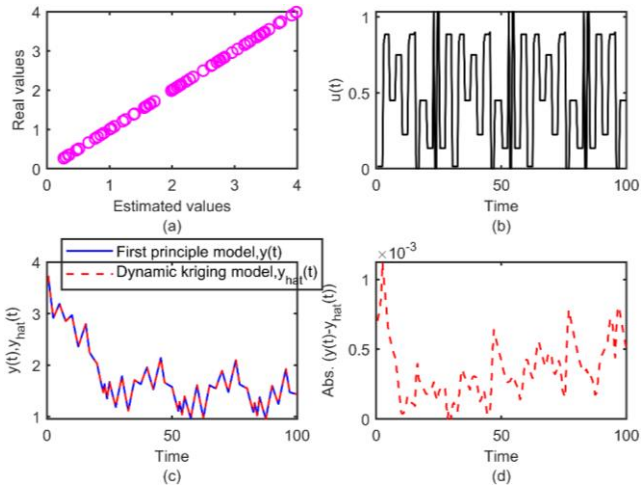


Figure 7.1. (a) LOOCV of the MVDK model, (b) validation control profile, (c) exact dynamic behavior $y(t)$ (blue) and MVDK prediction $\hat{y}(t)$ (red dashed line), (d) Absolute error $(|y(t) - \hat{y}(t)|)$.

Figure 7.1-(c) shows the comparison between the dynamic simulation of a randomly generated control profile (200 time-step (200×Δt =100 *min*) in Figure 7.1-(b)), using the fitted MVDK model (red dashed line), and the model based on first principles (solid blue line), both departing from an initial value of the state variable of y_0=3.96 *m*. Both lines correspond to virtually identical values, so it is clear that the MVDK is able to accurately estimate the same behavior of the dynamic system. Figure 7.1-(d) plots the absolute error between the kriging prediction and the values calculated by the first principles model.

7.3 MVDKS-BASED CONTROL VECTOR PARAMETERIZATION

The previous steps and techniques have been used to construct a robust computational framework for the application of CVP based dynamic optimization:

1. Explore the process behavior and identify the state variables y_t to be modeled and the control variables u_t.
2. Discretize the time domain $[t_0: t_f]$ into a grid of equal time-steps $[t_0 < t_1 < t_2 < t_3 <\ldots< t_j < \cdots < t_{N-1} < t_N = t_f]$, and discretize the control variables u_t as piecewise constants $[u_1, u_2, u_3, \ldots., u_{N-1}, u_N]$
3. Estimate the range of the state and control variables $[y_{t-min}: y_{t-max}, u_{t-min}: u_{t-max}]$.
4. Design a sampling plan $[X]_{n\times k}$ over the surrogate models domain, using the SLHS technique.
5. Carry out a simulation run at each point of the sampling plan using the original (complex) model in order to obtain the corresponding response states $[W]_{n\times M}$.
6. Fit *M* MVDK models and validate them as described (LOOCV),
7. Integrate the MVDK models in the CVP dynamic optimization scheme:
    a. Determine an initial guess for the decision variables $u^0 = [u_1^0, u_2^0, u_3^0, \ldots., u_{N-1}^0, u_N^0]$.
    b. Integrate the dynamic system state variables until the final time horizon, using the MVDK models, and compute the performance index *J* and the constraints g.
    c. Use a NLP optimizer (e.g., SQP) to update the values of the parameterized control variables, until the objective function is minimized, and the optimal control $u^* = [u_1^*, u_2^*, u_3^*, \ldots., u_{N-1}^*, u_N^*]$. policy is obtained.

7.4 APPLICATION AND RESULTS

The proposed kriging based CVP methodology is applied to three benchmark problems, which are commonly used in dynamic optimization studies. The methodology is applied to each case study two times, each one with a different discretization of the total time domain (15 and 20 time-steps discretization).

In each discretization, the methodology results are compared with the use of the classical or standard CVP technique, with the same problem adjustment (optimization domain, time-steps, initial guess of the control profile, NLP optimizer, etc.), but using the first principles mathematical model, integrated using the Matlab ODE algorithm "*ode15s"*. The first case is explained in detail; the application to the other two cases is straightforward so just the main results are commented. Finally, it is worthy to note that, in order to facilitate the comparison of results among the different case-studies, the control policies obtained in the three examples are scaled between [0, 1].

7.4.1 Case 1: Plug flow reactor catalyst blend problem

In this problem (Dadebo & Mcauley, 1995), a plug flow reactor is to be packed with a mixture of two different types of catalysts (type 1 and type 2). The kinetic sequence is given by: $A \leftrightarrow B \rightarrow C$ and the problem is to find the optimal profile of catalyst of type 1 ($u(z)$) along the reactor (in this problem, the independent variable z represents the reactor length, rather than time), to maximize the production of component C (J, Eq.(7.9)).

$$\underset{u}{\text{Max}}\ J = 1 - y_A(z_f) - y_B(z_f) \tag{7.9}$$

S.T.:

$$\left.\begin{aligned} \frac{dy_A}{dt} &= u(z)[10\, y_B(z) - y_A(z)] \\ \frac{dy_B}{dt} &= -u(z)[10\, y_B(z) - y_A(z)] - [1 - u(z)]\, y_B(z) \end{aligned}\right\} \tag{7.10}$$

where, y_A and y_B are the mole fractions of substances A and B respectively, $[y_A(0),\ y_B(0)] = [1,0]$, $z_f = 12$, $0 \le u(z) \le 1$. First, the proposed methodology is applied using a 15 step discretization of the length domain [0: 12], hence each step $\Delta z = 0.8$. Since the process has two state variables, two MVDK models (Eq.(7.11) and Eq.(7.12)) are fitted to mimic the system. The input variables $[y_{Aj-1},\ y_{Bj-1}, u_j]$ are expected to vary within the limits (surrogate model/optimization domain) of $[0:1.2, 0:0.1, 0:1]$. Over this domain, a SLHS sampling plan is designed with 91 sample points. The original model, Eq.(7.10), is used to generate the output matrix $[y_{Aj},\ y_{Bj}]$.

$$y_{Aj} = f_1(y_{Aj-1},\, y_{Bj-1}, u_j), \qquad \Delta z = z_j - z_{j-1} \tag{7.11}$$

$$y_{Bj} = f_2(y_{Aj-1},\, y_{Bj-1}, u_j), \qquad \Delta z = z_j - z_{j-1} \tag{7.12}$$

The accuracy of the fitted MVDK models is evaluated using the LOOCV technique, and the RMSE of the cross-validation are listed in Table 7.1. The obtained MVDK models have been used in the kriging based CVP, in which a sequential quadratic optimizer updates the discretized control profile until obtaining the optimal solution. The same methodology has been applied again with a different discretization of 20 step, hence $\Delta z = 0.6$, using the same generated sampling plan (the only difference is that the response variables are generated by carrying out the computer experiments with $\Delta z = 0.6$). Finally, both problem instances have been solved using the standard CVP technique in which the original first principles mathematical model is used with the same two discretization configurations (15, 20 times steps). The optimization results and computational effort of the proposed methodology and the standard CVP technique are compared in Table 7.1, and visualized in Figure 7.2.

Table 7.1. Case study 1(*)*: optimization results and MVDK models accuracy.*

		***Optimization results* (**)**			***MVDK models accuracy***	
		***Time-step* Δz**	***Objective value***	***CPU time* (s)**	***RMSE* (y_A)**	***RMSE* (y_B)**
15 steps	***FPM***	12/15=0.8	0.4745	70.60		
	MVDK models	12/15=0.8	0.4739	16.72	7.5×10^{-5}	8.9×10^{-5}
20 steps	***FPM***	12/20=0.6	0.4753	172.9		
	MVDK models	12/20=0.6	0.4746	36.00	3.42×10^{-5}	1.3×10^{-5}

(*) An optimal objective value of 0.477 is reported in the literature (Dadebo & Mcauley, 1995) *without specific indication of the required computational effort.*
(**) Optimization results of the MVDK models have been finally assessed using the FPM to ensure a fair comparison among both methods.

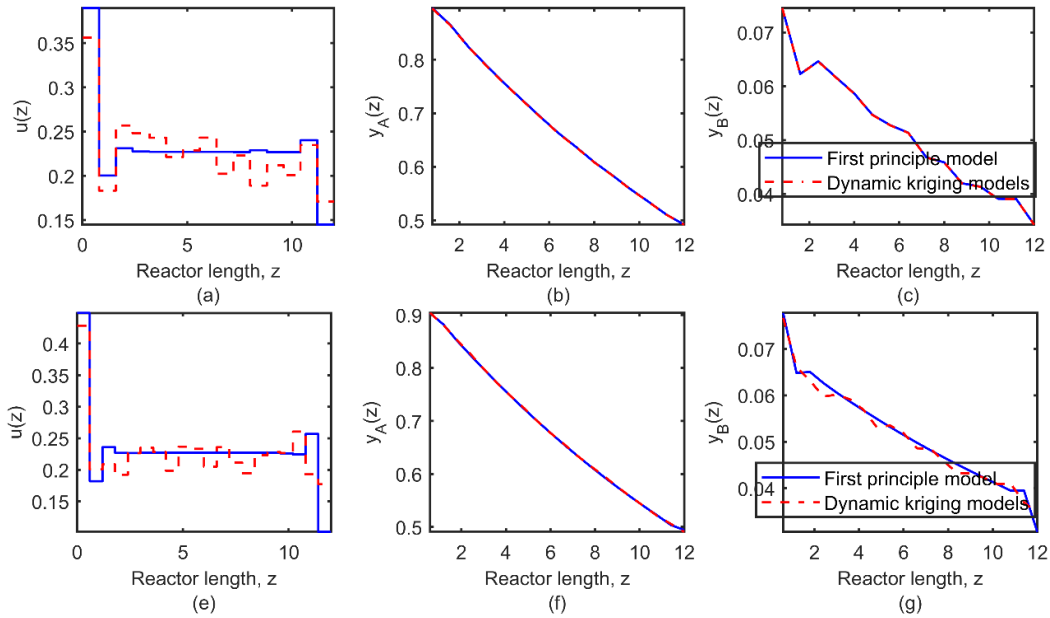


Figure 7.2. Optimal control profile and state variables case (1) obtained by CVP based on the FPM (blue lines) and on the MVDK models (red lines), using15 step (a,b,c), and 20 step (e,f,g) discretization.

For both discretization settings, the proposed methodology was able to obtain very similar control policies (Figure 7.2-(a,e) red dotted line) to the ones obtained using the first principles process model (Figure 7.2-(a,e) blue dashed line), both leading to almost identical process dynamics (Figure 7.2-(b,c,f,g)). Moreover, the methodology has obtained approximately the same global optimal value of the objective (0.13 % difference in case of 15 time-steps; 0.14 % in case of 20 time-steps), with relative significant reduction in the computational effort (more than 75% savings).

7.4.2 Case 2: Batch reactor

In a batch reactor (Luus, 1994), a reversible reaction A↔B is taking place. The problem is to find the best temperature control policy $u(t)$, that maximizes the performance index J in Eq.(7.13),

$$\underset{u}{\text{Max}}\ J = y_2(t_f) \tag{7.13}$$

S.T.:

$$\left.\begin{aligned} \frac{dy_1}{dt} &= \left(1 - y_1(t)\right) k_1 - y_1(t)\, k_2 \\ \frac{dy_2}{dt} &= 300\left[\left(1 - y_1(t)\right) k_1 - y_1(t)\, k_2\right] - u(t)(y_2(t) - 290) \\ k_1 = 1.753 \times 10^5 \exp\left(\frac{-1.1374 \times 10^4}{1.9872 \times y_2(t)}\right), & \; k_2 = 2.488 \times 10^{10} \exp\left(\frac{-2.2748 \times 10^4}{1.9872 \times y_2(t)}\right) \\ [\,y_1(0),\ y_2(0)] = [0{,}380], & \;\; t_f = 5\ min, \;\; 0 \le u(t) \le 0.36 \end{aligned}\right\} \tag{7.14}$$

The time domain [0: 5] is discretized ($\Delta t = 0.333, \Delta t = 0.25$) and, in each case, two MVDK models are fitted within the expected limits (surrogate model/optimization domain) [0: 1, 290: 490, 0: 0.36]. Over this domain, a SLHS sampling plan is designed with 123 sample points. The original model (Eq.(7.14)) is used to generate the output matrix $[y_{1j}, y_{2j}]$ and the accuracy of the fitted MVDK models is evaluated using the LOOCV (Eq.(7.6)). The obtained MVDK models have been used in the kriging based CVP. The results of the proposed methodology and the standard CVP technique are compared in Table 7.2, and in Figure 7.3.

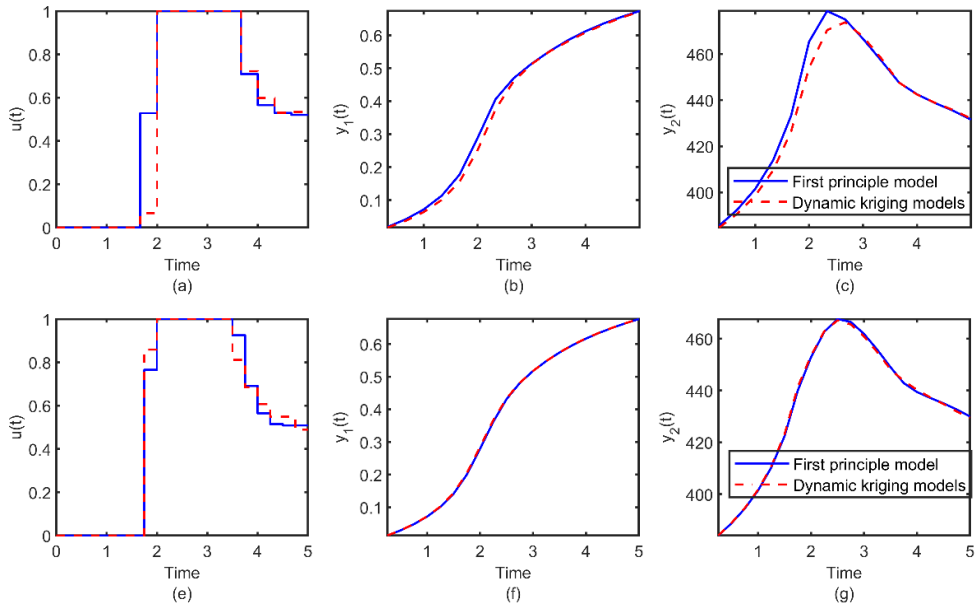


Figure 7.3. *Optimal control profile and state variables of case (2) obtained by CVP based on the FPM (blue lines) and on the MVDKs models (red lines), using15 step (a,b,c), and 20 step (e,f,g) discretization.*

Table 7.2. Case study 2(*): optimization results and MVDK models accuracy.

		Optimization results (**)			***MVDK models accuracy***	
		Time-step Δt	***Objective value***	***CPU time* (s)**	***RMSE* (y_1)**	***RMSE* (y_2)**
15 steps	***FPM***	5/15=0.3333	0.6744	222		
	MVDK models	5/15=0.3333	0.6704	25.5	0.01	3.05
20 steps	***FPM***	5/20=0.25	0.6749	458		
	MVDK models	5/20=0.25	0.6748	43	0.0031	0.0031

(*) The reported optimal objective value for this problem in the literature is 0.6753 (Luus, 1994).
(**) Optimization results of the MVDK models have been finally assessed using the FPM to ensure a fair comparison among both methods.

For both discretization settings, the proposed methodology is able to obtain a very similar control policy (Figure 7.3-(a,e), red dotted line) to the optimal one obtained from using the real process model (Figure 7.3-(a,e) blue dashed line) and leads to so similar process dynamics (Figure 7.3-(b,c,f,g)). Moreover, the methodology has obtained approximately the same objective value (less than 0.6% difference in case of 15 time-steps; 0.02 % difference in case of 20 time-steps), with a significant reduction in the required computational effort (about 90 % of the computation time saved in both cases).

7.4.3 Case 3: parallel reaction problem

In this third problem (Dadebo & Mcauley, 1995), a tubular reactor is considered to produce two substances according to the parallel reaction $A \rightarrow B, A \rightarrow C$, with rate constants k_1 and k_2 respectively. The objective is to maximize the yield of *B* at the final state, finding the most adequate profile of the control variable $u(t) = k_1\, l/v$ (l represents the reactor length and v the plug flow velocity). The dimensionless model describing the system dynamic is given in Eq.(7.16), where y_1, y_2 are the dimensionless concentrations of reactants A and B respectively.

$$\underset{u}{\text{Max}}\ J = y_2(t_f) \tag{7.15}$$

S.T.:

$$\left.\begin{aligned} \frac{dy_1}{dt} &= -(u(t) + 0.5\, u(t)^2) y_1(t) \\ \frac{dy_2}{dt} &= u(t) y_1(t) \\ [\, y_1(0),\, y_2(0)] &= [1, 0],\ \ t_f = 1\,,\ \ 0 \le u(t) \le 5 \end{aligned}\right\} \tag{7.16}$$

Again, the dimensionless domain [0: 1] is discretized ($\Delta t = 0.0667\ and\ \Delta t = 0.05$) and the proposed methodology is applied and compared with the standard CVP technique. Once the accuracy of the fitted MVDK models is ensured, they replace the complex first principles process model in the CVP. For space limitation, the results of the proposed methodology and the standard CVP technique are compared in Figure 7.4 and Table 7.3 only for the 20 step case.

Table 7.3. Case study 3(*): optimization results and MVDK models accuracy.

		Optimization results (**)			***MVDK models accuracy***	
		Time-step Δt	***Objective value***	***CPU time* (s)**	***RMSE* (y_1)**	***RMSE* (y_2)**
20 steps	***FPM***	5/20=0.05	0.5733	254.7		
	MVDK models	5/20=0.05	0.5732	34	1.147×10^{-6}	2.14×10^{-6}

(*) The reported optimal objective value of the problem is 0.5735 (Dadebo & Mcauley, 1995).
(**) Optimization results of the MVDK models have been finally assessed using the FPM to ensure a fair comparison among both methods.

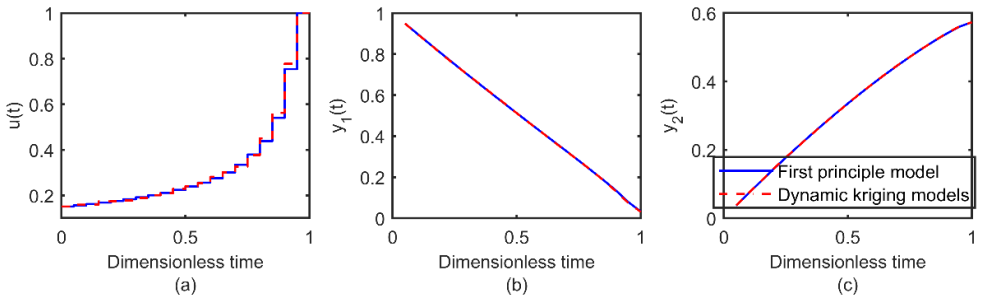


Figure 7.4. Optimal control profile and state variables of case (3) obtained by CVP based on the FPM (blue lines) and on the MVDK models (red lines), using 20 step discretization.

The methodology obtained approximately the same global optimal value of the objective with a very small relative error (1.6% and 0.0102%), with relative significant reduction in the computational time (about 85%). In the three case studies, the proposed CVP framework was able to obtain very accurate results. As well as the standard CVP method, the proposed methodology improves the solution using the finer grid (20 time-steps), but still saving a huge quantity of computational effort.

7.5 CONCLUSION

The potential of dynamic kriging stems from its capacity to replace complex integration rules with simple successive interpolations. This potential has been exploited in this Chapter to develop a sequential dynamic optimization strategy based on such type of surrogate models able to solve the optimal control problem of complex processes, like the ones which usually appear in the chemical or petrochemical sectors, with significant advantages over the use of traditional FPMs. Specifically, relatively simple processes, frequently used as reference in the process systems engineering literature, have been used to assess the eventual benefits of the proposed kriging based CVP strategy in terms of accuracy, robustness and computational cost.

The kriging models have shown high accuracy to capture the nonlinear dynamic nature of these highly nonlinear systems, resulting in an outstanding capacity to predict the system dynamics over large time domains. The integration of such modeling technique with other complementary techniques, like sampling design for computer experiments, cross-validation methods and sequential dynamic optimization, to build a comprehensive dynamic optimization framework where the expensive integration of a complex model is replaced by simpler recursive or successive interpolation (MVDK models) has been straightforward. The resulting characteristics and advantages of the proposed framework, and specially its accuracy and the significant reduction of the computational effort, are evident even on its application to three case studies of moderate complexity.

These results confirm that the proposed framework constitutes a significant step forward to solve one of the biggest challenges that face the standard CVP techniques, associated to the significant computational effort required by the repeated process model integration tasks. The methodology becomes also a unique solution when a mathematical process model is missing, and only experimental process data are available. In this sense, it looks like a promising way to allow a more universal application of nonlinear model-based control techniques, where the optimal control problem must be repeatedly solved.

Chapter 8: FDD of Nonlinear Dynamic Processes Based on MVDK

This Chapter presents a hybrid approach to improve data-based Fault Detection and Diagnosis (FDD). It is applicable to nonlinear dynamic noisy processes, operated under time-varying inputs. The method is based on the combination of kriging models and Classification Techniques. A set of MultiVariate Dynamic Kriging-based predictors (MVDKs) is built and used to estimate the process dynamic behavior, while static kriging models are used to smooth the eventually noisy process outputs. The estimated and the actual smoothed outputs are compared, taking advantage of the higher capacity of the residual patterns generated in this way to characterize the process state. The performance of the method is illustrated through its application to a well-known benchmark case study, for which the FDD performance has been significantly improved. This improvement is consistently maintained in different dynamic operating conditions and faulty situations, including scenarios with modified fault severities and fault styles.

8.1 INTRODUCTION

A fault is an unexpected change of a system behavior with respect to its normal operation. Although it may not lead to immediate physical failure or breakdown, a fault hampers or disturbs the normal system operation, thus causing an unacceptable deterioration of the system performance, which may even lead to dangerous operating conditions (Patton, et al., 1995; Calado, et al., 2001). A FDD system should be able to perform two main functions: first detecting the existence of the fault, as opposite to the normal behavior, and second diagnosing the fault type or characteristics (Patton, et al., 1995; Narasimhan, et al., 2008). Early FDD plays an essential role in the safety and reliability of industrial process operations because of its ability to discover the root cause of abnormal situations, averting sudden shutdowns, breakdowns or even catastrophic events, which finally lead to economic losses due to production stop and/or replacement of spare parts (Amozeghar & Khorasani, 2016).

General FDD methods can be classified into three main groups: knowledge-based, model-based and data-based methods (Calado, et al., 2001; Venkatasubramanian, et al., 2003a; Venkatasubramanian, et al., 2003b). Knowledge-based methods rely on the development of some diagnostic rules, the establishment of rule-based expert system, necessitate a deep knowledge about process structure and components under the normal (fault-free) and the

different possible faulty situations (Calado, et al., 2001). However, the knowledge acquisition is generally a challenging task, beside that the response or the performance of the knowledge-based systems to events outside their domain of knowledge or expertise cannot be reliable (Calado, et al., 2001). Model-based methods rely on what is named "analytical redundancy" (Patton, et al., 1994; Qin, 2012), through the monitoring of the extent of matching between the actual process measured features (e.g. state variables, outputs signals, coefficients or parameters) and the corresponding features calculated by means of an analytical model of the process representing the normal, or fault-free, features.

Therefore, the model-based approach allows the generation of error or residual signals, resulting from the differences between the model-estimated features and the actual process-measured features (Patan & Parisini, 2005; Isermann, 2005). These error signals indicate the extent of the process malfunctioning, i.e. the residuals should be close to zero when no fault occurs, while showing considerable values when any fault affects the system. Thus, error signals are used to detect and diagnose faults, simply by comparing them to threshold values for the errors, or using a more elaborated statistical analysis (Patton, et al., 1995; Narasimhan, et al., 2008; Caccavale, et al., 2010; Elhsoumi, et al., 2011).

Model-based methods mainly include approaches such as observer-based, parity space-based and parameter estimation-based. In the first approach, residuals are generated by comparing process-measured outputs to the corresponding outputs estimated by an observer (e.g., Leunberger observers in deterministic settings or Kalman filter(s) in a stochastic setting). On the other hand, the parity space approach is based on rearranging or transforming the input-output or state-space process model, in order to obtain what is called parity equations or relation. The obtained relations are used to assess the parity (consistency) of the process model with sensor outputs and known process inputs; hence, the unbalance term is used as a residual signal. Meanwhile, the parameter estimation-based FDD approach is based on the assumption that faults can be interpreted as changes on physical process parameters (density, viscosity, specific heat, etc.). Thus, residuals can be generated as the difference between the repeatedly estimated parameters of the actual process, and the parameters of the analytical or reference model obtained under fault-free conditions.

Model-based methods show great advantages when dealing with dynamic systems, where inputs and outputs of the monitored system are fed into a processor (e.g. diagnostic observer) that represents the knowledge about the process dynamics, in order to generate a fault indicator /residual (Patton, et al., 1994; Elhsoumi, et al., 2011). However, they are associated with many shortcomings that complicate their implementation (Venkatasubramanian, et al., 2003a). First of all, the difficulties to create an accurate dynamic model of the process should be considered. Besides the sensitivity of such models to modelling

errors, parameter variations and uncontrolled disturbances, etc., it becomes significantly difficult to develop analytical models for complex, highly nonlinear and/or large-scale (high dimensional) processes (Ardakani, et al., 2016a; Ardakani, et al., 2016c; Banu & Umab, 2011). Second, most of these approaches are based on linear state space models, which effectiveness is reduced when applied to highly nonlinear complex processes (poor linear approximation) (Venkatasubramanian, et al., 2003a; Serdio, et al., 2014); alternatively, some model-based approaches are nonlinear, but they are very specific to certain classes of processes (Venkatasubramanian, et al., 2003a). Finally, applications addressing large-scale processes would result in a high number of observers, which end up with solutions requiring an unaffordable computational effort if they must be used on-line (Venkatasubramanian, et al., 2003a).

Alternatively, data-based methods, especially Classification Techniques (CTs) (e.g. Support Vector Machines (SVMs), Gaussian Naïve Bayes classifiers (GNBs), Decision Tree (DT), Artificial Neural Networks (ANNs), etc.), have shown a great flexibility and robustness for the FDD of nonlinear chemical processes (Askarian, et al., 2016; Ardakani, et al., 2016c). Without requiring any process mathematical model, they are trained based on pattern recognition principles from process historical data, including information about normal and different faulty situations (Patton, et al., 1994). The learning process enables these CTs to extract knowledge from data, via optimization/adjustment of their parameters. Hence, features in the process variables that correspond to each class/situation (normal situation or different faults) can be recognized (Askarian, et al., 2016). Then, the trained CTs can be used for the process supervision in order to detect and diagnose possible faults from the process outputs measurements.

However, these CTs also suffer from serious limitations (Patton, et al., 1994; Caccavale, et al., 2010). The first one is that the classification of faults is based only on the current measurement of the process outputs (features), which, in contrast to model-based approaches, disregards any knowledge about the system dynamics (i.e. relations between the system inputs and outputs). As a result, they are mostly used for FDD of steady state processes, where the process is expected to operate under constant conditions/controls (Patton, et al., 1994; Ardakani, et al., 2016a). Consequently, the process state/output variables are also expected to follow a constant/specific behavior (set points). This constant or steady-state behavior represents a definite or specific pattern that is relatively easy to be recognized by these CTs under normal or faulty conditions as well. However, in many situations the process is to be operated under changeable / manipulated operating conditions (e.g. during the transition between different process set points, or because of the changes in the manipulated inputs to overcome some external disturbances in order to retain the process to its optimal set point). In

these cases, CTs could easily produce false alarms by diagnosing the changes in the processes features as faults. This is due to the lack of information about the dynamics governing the relation between the process inputs and outputs. Thus, a second limitation is that the measurement errors that very often contaminate the measurements can create false diagnosis and alarms. These usual errors may be random (e.g.: sensors white noise) or not (outliers / biases due to instruments malfunctioning, miss-calibration or poor sampling) (Patton, et al., 1994; Amozeghar & Khorasani, 2016).

Complementary to these two basic methodologies (model-based and data-based FDD), some works have proposed the usage of ANNs and other related methods to mimic the system dynamic behavior, identifying the underlying mapping between the system inputs and outputs (Patton, et al., 1994; Calado, et al., 2001; Caccavale, et al., 2010; Honggui, et al., 2014; Smarsly & Petryna, 2014; Serdio, et al., 2014; Tayarani-B. & Khorasani, 2015; Banu & Umab, 2011; Amozeghar & Khorasani, 2016). In these approaches, the ANN is employed as a model (predictor), in order to generate a residual vector between the ANN estimated outputs and the process measured outputs. These residuals are then used to detect and isolate faults using a threshold value for each residual component or applying some statistical analysis. ANNs predictors have been proven to be very robust and capable of approximating the dynamic behavior of a very wide range of complex linear and nonlinear systems, which overcome the previously mentioned difficulties associated to the development of analytical model-based approaches. Due to its flexible structure of neurons and powerful generalization properties, ANNs are capable of learning from input-output data, capturing complex nonlinear dynamic behaviors and filtering out the system noise and disturbances.

Few works (Patton, et al., 1995; Tayarani-B. & Khorasani, 2015; Amozeghar & Khorasani, 2016) have combined these data-based predictors (and the generated residuals) with CTs to automate and improve FDD. However, in most of these works, CTs are trained to isolate each fault type when the residual component of a specific output exceeds a specific threshold value. This approach neglects the basic and most important characteristic of any CT, which is its ability to identify a certain pattern in the features, regardless of the specific values of the residuals. Furthermore, the identification of a specific threshold value for each residual component as a fault indicator is not a trivial task, as it requires prior knowledge about the process behavior besides its behavior under the effects of the fault, and may be even infeasible if scenarios with time-varying inputs are considered. Finally, it should be noted that the use of ANNs for data-based modeling frequently shows drawbacks as the curse of dimensionality and the difficulty to specify the network structure (Boukouvala, et al., 2011). Thus, it would be worth investigating the effectiveness of using other alternative modeling techniques.

This Chapter presents a hybrid data-based approach for FDD that combines data-based predictors and CTs. The objective is to enhance the performance of the data-based CTs used for FDD of nonlinear, dynamic, noisy processes, emphasizing that the dynamics may be associated to the fact that the process is running under changeable operating conditions. This enhancement is based on the following: the use of static kriging models to smooth noise; the use of efficient kriging-based process dynamic predictors; the use of residuals to characterize the different faults; the proper identification of a classification method; and the full exploitation of the potential capabilities of all these techniques. The proposed approach is illustrated through its application to the FDD of a three-tank benchmark problem. Results show not only a high enhancement in the performance of the CTs, but also high robustness under different profiles of the process control inputs and different faulty behaviors, even in situations not included in the training datasets of these CTs.

8.2 METHODOLOGY

The proposed system involves two stages: the first one is the offline stage, in which the predictor and the CT are trained using the process history data, while the second stage (Figure 8.1) involves the online monitoring of the process by means of a smoothing procedure or filters, and of the already fitted predictor and CT.

Assuming a process with inputs $U(t)$ and outputs $X(t)$, $U_t \in R^{k_u}, X_t \in R^{k_x}$, the first step of the offline stage is the construction of the MVDKs based predictor $\hat{X}^{prd}(t+1) = F_{prd}\big(X(t), U(t)\big)$. This predictor is trained with the available process input-output historical data (U_{prd}^{trn}/ X_{prd}^{trn}) under normal (fault-free) conditions. In this way, the MVDKs is capable to estimate the process future outputs $\hat{X}^{prd}(t+1)$ –at the next time step, i.e. next or future sampling period- as a function of the current inputs and outputs (Section 8.2.3). Before its online usage, this predictor has to be validated to ensure that it has satisfactory estimation accuracy beyond the training conditions. This validation can be accomplished through the harnessing of the already fitted predictor to estimate the process outputs of a different set of data (validation data). Hence, the predictor estimations of the outputs are quantitatively compared to their corresponding real values by using some performance indicator (e.g., the Normalized Root Mean Square Error (NRMSE)).

In order to generate the signal U_{prd}^{trn}/ X_{prd}^{trn} used for the predictor training, a random signal of the inputs U_{prd}^{trn} is composed in such a way that it includes several step changes along the time. The amplitudes of these step changes are selected randomly within the allowed limits of the inputs. Additionally, the length (i.e. the time) of the step change should be long enough to capture the whole dynamic response of the system to this step change, and at the same time

it should be short enough to avoid the excess of the steady state samples. After the composition of the input signal U_{prd}^{trn}, a process first-principle model is used to simulate the corresponding outputs X_{prd}^{trn}.

In the cases where there is no first-principle model of the process and only process history data is available, the input-output training signal ($U_{prd}^{trn}/X_{prd}^{trn}$) is selected from the process history data, so that it includes as many changes/fluctuations as possible within the whole variability domain of inputs and outputs. Consequently, this allows the collection of all possible information about the system dynamic behavior along most of its sub-domains or local domains. The size of the required training dataset (i.e. the signal $U_{prd}^{trn}/X_{prd}^{trn}$ length) is a case-dependent factor, since it mainly relies on the number of the system inputs and outputs, the degree of the system behavior nonlinearity and the size of the inputs variability domain.

The second task of the offline stage is the construction and the training of a CT based on the residuals $e(t)$, $e_t \in R^{k_x}$ (Section 8.2.4). These training error signals can be generated as follows: the available measurements signal(s) of the process inputs and outputs under normal and different faulty situations are collected (U_{PRT}^{trn} / X_{PRT}^{trn}); then static filters are used to smooth the outputs, filtering out the noise in order to obtain $\hat{X}_{PRT}^{trn-sm}$. Generally, these filters are very simple static models $\hat{X}^{sm}(t) = F_{sm}(t, X(t))$. They are trained using the measured output data $X(t)$, in order to define static relations able to describe the smoothed values of these outputs $\hat{X}^{sm}(t)$ as a function of their noisy values $X_{(t)}$ and their measurements time t. The black-box functions F_{sm} represent the noise-free underlying behaviors of the outputs, i.e. the filters (Section 8.2.2). Analogously, as in this offline case, the inputs for the static filters are X_{PRT}^{trn}; thus, the outputs of the filters are their corresponding smoothed values $\hat{X}_{PRT}^{trn-sm}$. On the other hand, the predictor is used to estimate the normal outputs $\hat{X}_{PRT}^{trn-prd}$ corresponding to the input scenario $U_{PRT}^{trn}(t)$. The residuals are then calculated as the difference between the smoothed actual outputs and the predictor estimated outputs $e_{PRT}^{trn} = \hat{X}_{PRT}^{trn-prd} - \hat{X}_{PRT}^{trn-sm}$. It is worthy to mention that half of the residual data e_{PRT}^{trn} are used for the training of the CT, while the other half is used for its validation before its usage in the second stage of the framework.

For the generation or the collection (either if a first principle model is available or only the process history data is available respectively) of the input-output signal U_{PRT}^{trn} / X_{PRT}^{trn}, the same principle previously mentioned should be also regarded. Thus, this signal(s) should include -as much as possible- all the potential combinations of the process normal and faulty situations under different/changeable dynamic behavior of the inputs (see the part 3.2). Similarly, the size of the required training dataset for the CT is a case-dependent factor, which

also depends on the process dimensionality, nonlinearity and additionally the expected number of process behavior classes (i.e. number of the faulty situations).

Once the predictor and the classifier are trained and validated, they are linked to monitor and supervise the process during its operation (the second stage of the framework –Figure 8.1). Hence, under a certain time profile of the inputs $U(t)$, the predictor is used to estimate the process behavior $\hat{X}^{prd}(t)$ corresponding to this profile. In parallel, the static filters are used to smooth the noise from the process actual output measurements. The error signal is then calculated as $e(t) = \hat{X}^{prd}(t) - \hat{X}^{sm}(t)$, where $\hat{X}^{sm}(t)$ are the actual smoothed outputs. This error signal enables the CT (classifier) to have information about the process dynamics, along with possible faults, so the CT can discern if the outputs change is normal (i.e. the change is due only to the change of inputs or it is caused by an eventual fault).

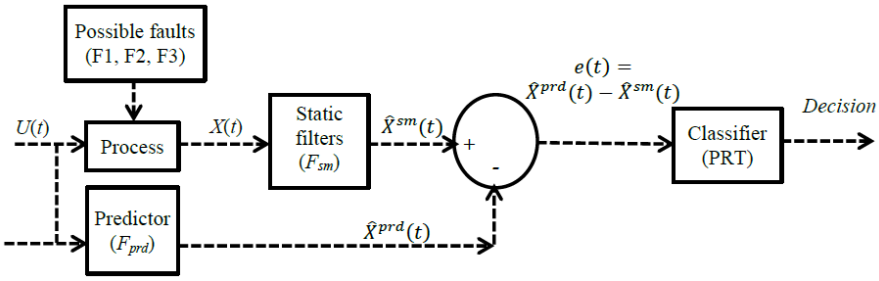


Figure 8.1. The second stage of the proposed FDD framework.

8.2.1 Ordinary kriging

Since both of the dynamic predictor and the static filters are based on the use of the OK model, therefore, this part presents a brief summary about the OK basics, while the subsequent parts (Sections 8.2.2 and 8.2.3) explain how the OK is employed as a dynamic predictor and a static filter.

Given a set of n input-output training data $[w_i, y_i]$, $i = 1,2,..n$, $w \in R^k, y \in R$, the OK assumes the predictor $\hat{y}(w) = \mu_{ok} + Z(w)$, where the constant term μ_{ok} represents the main trend of the system to be approximated, and $Z(w)$ is a deviation from that trend. The deviation $Z(w)$ is modeled as a stochastic Gaussian process with expected value $E(Z(w)) = 0$, and a covariance between two residuals $cov(Z(w_i), Z(w_j))$ that only depends on their corresponding inputs w_i, w_j. Thus it can be calculated as: $cov\left(Z(w_i), Z(w_j)\right) = \sigma_{ok}^2 R(w_i, w_j)$, being σ_{ok}^2 the process variance and $R\left(w_i, w_j\right) = exp\left(-\sum_{l=1}^{k} \xi_l \left|w_{i,l} - w_{j,l}\right|^{p_l}\right) + \delta_{ij}\,\lambda$ a correlation function, where, $\xi_l, l = 1, \ldots k$ are the model hyper-parameters, δ_{ij} is the Kronecker delta, p_l are smoothing parameters and λ is a regularization constant that enables the kriging predictor to regress noisy data (Azman & Kocijan, 2007). The kriging predictor

and its estimated error are given by Eq. (8.1) and Eq. (8.2), respectively, where (w^*) is a new interpolating point (different from the training data). In Eq. (8.1), $[r]_{n\times 1}$ is the vector of correlations between the point to be predicted w^* and the original training data points and calculated as $R(w_i, w^*)$, $[R]_{n\times n}$ is the correlation matrix between the training inputs, $[Y]_{n\times 1}$ is the vector of the training outputs and $[\mathbf{1}]_{n\times 1}$ is the identity vector.

$$\hat{y}(w^*) = \mu_{ok} + r^T R^{-1}(Y - \mathbf{1}\mu_{ok}) \tag{8.1}$$

$$\hat{s}^2(w^*) = \sigma_{ok}^2(1 + \lambda - r^T R^{-1} r + (1 - \mathbf{1}^T R^{-1} r)^{-1}/(\mathbf{1}^T R^{-1}\mathbf{1})) \tag{8.2}$$

This work considers the OK implementation developed by Forrester, et al., (2008), because of its high efficiency and applicability. Besides, the "*fmincon*" algorithm included in the Matlab optimization toolbox is used for the maximization (nonlinear optimization) of the concentrated likelihood function. More details about the OK model can be found in Section 2.2.1.

8.2.2 Static filter

In order to smooth the data of the measured outputs $X(t)$, a simple filtering step based on the use of static OK models is proposed. In summary, for each process output $x_i(t)$, i=1,2,..k_x, an OK model is trained using the measured noisy data of this output (Figure 8.2, black point), in order to approximate a relation $\hat{x}_i^{sm}(t) = f_{sm,i}(t, x_i(t))$ (Figure 8.2, dotted green line) describing the underlying/ smoothed behavior of the output $\hat{x}_i^{sm}(t)$ as a function of the time and the noisy measurements. This relation represents the OK filter that is used to interpolate at the different time instances (Figure 8.2, vertical dotted grey lines) in order to predict the corresponding smoothed output (Figure 8.2 , green circles).

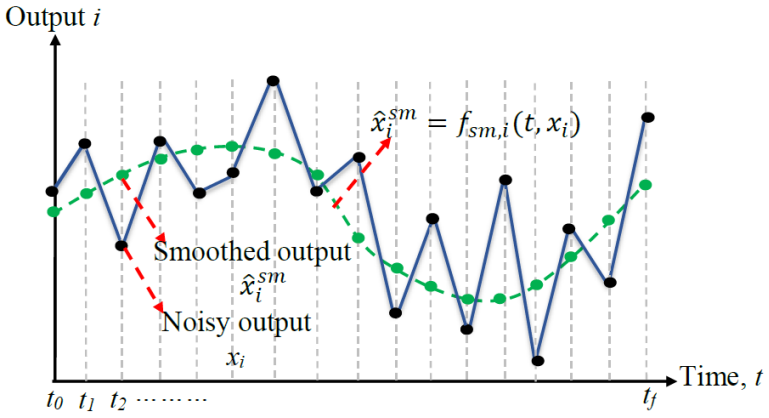


Figure 8.2. OK filter illustration.

The proposed filters $\hat{x}_i^{sm}(t) = f_{sm,i}(t, x_i(t))$ are straightforward applications of the previously described OK model, where the smoothed output $\hat{x}_i^{sm}$, the measurement time *t* and the noisy output data x are analogous to $\hat{y}$, w and Y in Eq. (8.1), respectively. Recently, Ardakani et al. (2016b) have proposed the usage of these OK filters for smoothing noisy data of chemical processes history, in order to enhance the training of different CTs. They have shown that OK has very good capabilities in front of other techniques such as ANNs and polynomial regressions models.

8.2.3 Predictor: multivariate dynamic kriging models

Although the high modeling capabilities offered by the kriging approach have been extensively shown in the process systems engineering area, its usage has been concentrated in the modelling of complex but static systems (Davis & Ierapetritou, 2007; Caballero & Grossmann, 2008). This Chapter considers the MVDKs, which have been proposed (in Chapter 6) for data-driven modelling of multivariate dynamic systems, showing outperforming capabilities over many state-of-the-art techniques (e.g. ANNs, and Gaussian models) (Boukouvala, et al., 2011).

This MVDKs method will be used in this work. It is based on the construction and training (using measured or simulated data) of a number of k_x OK models that are trained to capture the incremental evolution of the system, i.e. the system future state/output variables over one time-step. In more details, each OK model is trained to approximate the mapping between the future value of one state/output variable at the next time step $\hat{x}_i^{prd}(t+1)$ as a function of the system previous state and control variables values [X_t, X_{t-1} ,… $X_{t-L,}$ U_t, U_{t-1},… U_{t-L}] considering a specific time lag or delay L=0, 1, 2….or L. This is given by Eq. (8.3), where $U(t) \in R^{k_u}$ represents the control/input variables, and $X \in R^{k_x}$ corresponds to the state/output, which are recorded at discrete time instances of equal intervals Δt between them, being k_u and k_x the number of control and state variables respectively.

$$\left.\begin{array}{l} \hat{x}_1^{prd}(t+1) = f_{prd,1}\left[\hat{X}(t),.. \hat{X}(t-L), U(t),..U(t-L)\right] \\ \hat{x}_2^{prd}(t+1) = f_{prd,2}\left[\hat{X}(t),.. \hat{X}(t-L),\ U(t),..U(t-L)\right] \\ \ldots\ldots \\ \hat{x}_i^{prd}(t+1) = f_{prd,i}\left[\hat{X}(t),.. \hat{X}(t-L),\ U(t),..U(t-L)\right] \\ \ldots\ldots\ldots\ldots \\ \hat{x}_{k_x}^{prd}(t+1) = f_{prd,kx}\left[\hat{X}(t),.. \hat{X}(t-L),\ U(t),..U(t-L)\right] \end{array}\right\} \quad (8.3)$$

The time intervals Δt are always conditioned by the sampling periods of the system to be modelled. On the other hand, the lag L - which is also the dynamic model order- is often determined via a cut-and-try approach in order to select the lag value that achieves the best model prediction accuracy. At the same time, it is favorable to keep it as low as possible in

order to obtain simple dynamic models, since the increase of the lag value increases the number of the dynamic model inputs (see Eq.(8.3)), which consequently increases the complexity of the models fitting and usage. These sets of single-step emulators (Eq.(8.3)) are also considered as Nonlinear AutoRegressive models with Exogenous inputs (NARX), which are able to predict the system outputs over one time-step ahead. Additionally, they can be also used via recursive interpolation to predict the outputs over several time steps. Thus, at each time step, the predicted values of the state variable are fed back to the model representing its input for the next time step estimation, together with the new value of the control variables. For more details about the dynamic kriging models, multivariate dynamic prediction via recursive interpolation and their applications to other case studies, the interested reader is referred to (Boukouvala, et al., 2011; Biegler, 2007).

8.2.4 Classification techniques

Classification techniques may be based on a priori knowledge or on statistical information obtained from process data (García-Laencina, et al., 2010). In this work, statistical information-based methods have been adopted because of their flexibility and ease of implementation, especially for complex nonlinear processes. Each of them presents relative strengths and weaknesses, so the combination of some of them in hybrid systems has been suggested as a practical way for exploiting their advantages and covering their individual shortcomings (Venkatasubramanian, et al., 2003a).

Different classification methods are adopted and compared in this study, representing the most common types of CTs. They are: ANNs, as a machine learning method, Support SVMs, as hyperplane-based methods (margin-based the latter), GNB, as a probabilistic method and finally DT, as a rule-based method. For the sake of generalization, but also in order to limit the number of tests, the performance of the proposed approach is quantified in this work for a reduced but significant set of classification methods. These methods are:

- **Artificial Neural Networks** are non-statistical methods that have been imported from the machine learning area, and have been used as a fault diagnostic tool in the process engineering field (Venkatasubramanian, et al., 2003a; Venkatasubramanian, et al., 2003b). Its learning and prediction potential make them attractive in many areas. However, ANNs application in FDD could be limited because of its high computational load for complex systems.

- **Support Vector Machine** are margin based classification approaches initially introduced by Vanpik et al. (Boser, et al., 1992) and have been commonly used by researchers as reliable tools for a wide range of purposes (Monroy, et al., 2010; Akram, et al., 2014).

They establish the classification space based on the maximization of the margin between the training patterns and the decision boundaries, which is a way to reduce the structural risk of misclassification (Monroy, et al., 2010).

- **Gaussian Naïve Bayes classifiers** are probabilistic models based on applying Bayes' theorem. These classifiers exhibit two main benefits: First, their easy construction and the absence of a learning procedure. Second, the independency assumption of the features, leading to a very efficient classification process (Addin, et al., 2011; Atoui, et al., 2015).

- **Decision Trees** are simple algorithms based on the formulation of diverse classification rules. These rules are extracted through a recursive approach (Dash & Venkatasubramanian, 2000). Many decision tree structures can be considered, some of them already standardized (Özyurt, et al., 1998).

Finally, the performance of the CT and the resulting FDD system will be assessed in this work based on the $f1$-score measure, which is calculated as in Eq.(8.4) (based on a test dataset) and represents the harmonic mean of classifier precision and recall. More details can be found in Section 2.3.2.

$$f1 - score = \frac{2 \times Precision \times Recall}{Percision + Recall} \tag{8.4}$$

The $f1$-score ranges from 0.0 (worst value) to 1.0 and (best value), and facilitates the comparison between methods and summarizing various concepts. However, it obviously implies a loss of information which may be relevant in particular situations in which precision and recall need to be discriminated or weighted to model the actual consequences of the misdiagnosis (i.e., when the consequences for false fault identification and false negative identification are different).

8.3 APPLICATION AND DISCUSSION

The three-tank system in Figure 8.3 has been used to illustrate the application and characteristics of the proposed approach. This is a well-known benchmark used in monitoring, control and FDD studies (Frank & Ding, 1997; Kouadri, et al., 2012; Sarailo, et al., 2015); it includes the basic characteristics of a fluid distribution network typically found in the chemical industry (Patton, et al., 1994), and its evolution can be described through a simple mathematical model, so it can be easily used to develop and reproduce faulty scenarios of different diagnosis difficulty to objectively test data-based (and model-based) the performance of FDD systems in scenarios including all the different elements motivating this work: nonlinear dynamic noisy processes, operated under time-varying inputs.

For comparative purposes, the same design and characteristics used in the original work (laboratory scale) have been maintained in this study: The system consists of three identical cylindrical tanks of cross section area $A=0.0154\ m^2$, which are serially interconnected by three cylindrical pipes of cross section area $s_{13}=s_{23}=s_0=0.005\ m^2$, and flow coefficients ($a_{13}=$ 0.6836, $a_{23}=$ 0.4819, $a_0=$ 0.4819. Two pumps are delivering the liquid to the system with flowrates Q_1, Q_2, where the maximum allowed flowrates are limited to 0.003 m^3/s. The tank levels (h_1, h_2 and h_3 respectively) are the measurable process outputs to be used for FDD.

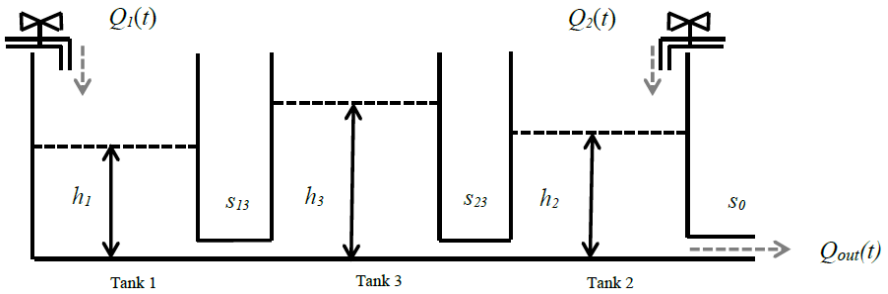


Figure 8.3. Three tanks benchmark system

The process is described by the set of ordinary differential equations illustrated in Eq.(8.5).

$$
\begin{aligned}
A\frac{dh_1}{dt} &= -a_1\, s_{13}\, sgn(h_1 - h_3)\sqrt{2g|h_1 - h_3|} + Q_1 + Q_{f1} \\
A\frac{dh_2}{dt} &= a_3\, s_{23}\, sgn(h_3 - h_2)\sqrt{2g|h_3 - h_2|} - a_2 s_0\sqrt{2gh_2} + Q_2 + Q_{f2} \\
A\frac{dh_3}{dt} &= a_1\, s_{13}\, sgn(h_1 - h_3)\sqrt{2g|h_1 - h_3|} - a_3 s_{23} sgn(h_3 - h_2)\sqrt{2gh_3 - h_2} +
\end{aligned}
\tag{8.5}
$$

The process is subjected to three possible faults: the first fault (F1) is the leaking in tank 1 (Q_{f1}*=-0.0007 m3/s*), the second (F2) is the plugging in tank 2 (Q_{f2}*=+0.0007 m3/s*), and the third fault (F3) is the leaking in tank 3 (Q_{f3}*=-0.0007 m3/s*). These values have been selected to be between 10% and 25% of the inlet flow, based on the literature of this case study. Additionally, a Gaussian error, $\mathcal{N}(\mu = 0, \sigma = 0.010)$, has been added to the model output in order to represent the noise introduced by the different sensors; besides, outliers following a normal distribution $\mathcal{N}(\mu = 0, \sigma = 0.01)$ have been also added to the 7% of the measurements. A sampling time of one second has been selected.

The objective is to design a data-driven FDD system for the detection and diagnosis of possible faults, considering scenarios which should consider arbitrary changes in the manipulated inputs (Q_1, Q_2). It is worthy to mention that many studies exploiting this case study have used different values of the operating conditions/ adjustments (e.g. faults magnitudes, maximum limits of the input flowrates, etc.); however, the order of magnitude of

these values is always the same. During the subsequent parts of the methodology applications to the addressed case study, the process first principle model in Eq.(8.5) is only used to generate an input-output database (that imitates the real process history database), which is used for the training and the validation of both MVDKs and CTs.

8.3.1 Predictor construction

As previously mentioned, the first task in the offline stage is the MVDKs predictor construction. In this case, the task includes the development of three dynamic kriging models, Eq.(8.6), where *i*=1,2,3 and *j*=1,2. Each of the three models is approximating the future value of each tank level as a function of the pervious values of the system the levels $h_1(t)$, $h_2(t)$, $h_3(t)$ and inlets $Q_1(t)$, $Q_2(t)$ considering a certain time lag *L*.

$$\left.\begin{aligned}\hat{h}_1^{prd}(t+1) &= f_{prd,1}\left[h_i(t), h_i(t-1), .. h_i(t-L), Q_j(t)\ , Q_j(t-1), .. Q_j(t-L)\right]\\ \hat{h}_2^{prd}(t+1) &= f_{prd,2}\left[h_i(t), h_i(t-1), .. h_i(t-L), Q_j(t)\ , Q_j(t-1), .. Q_j(t-L)\right]\\ \hat{h}_3^{prd}(t+1) &= f_{prd,3}\left[h_i(t), h_i(t-1), .. h_i(t-L), Q_j(t)\ , Q_j(t-1), .. Q_j(t-L)\right]\end{aligned}\right\} \quad (8.6)$$

As previously described in Section 8.2.3, different lag values can be tested, however, it is also obvious to assume first the simplest case when no lag is introduced to the models (i.e. L=0). Hence, the models take the form in Eq.(8.7).

$$\left.\begin{aligned}\hat{h}_1^{prd}(t+1) &= f_{prd,1}[h_1(t), h_2(t), h_3(t), Q_1(t), Q_2(t)\ ]\\ \hat{h}_2^{prd}(t+1) &= f_{prd,2}[h_1(t), h_2(t), h_3(t), Q_1(t), Q_2(t)]\\ \hat{h}_3^{prd}(t+1) &= f_{prd,3}[h_1(t), h_2(t), h_3(t), Q_1(t), Q_2(t)]\end{aligned}\right\} \quad (8.7)$$

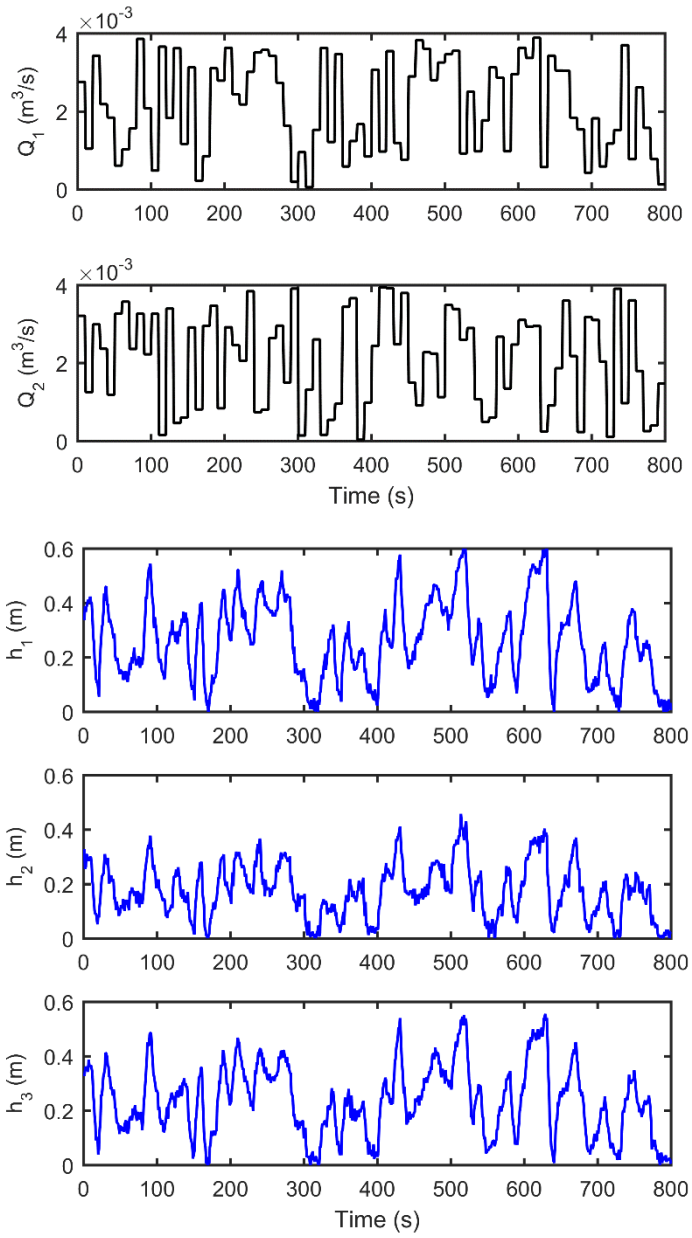


Figure 8.4. Training data of the MVDKs predictor.

To train the MVDKs predictor, a fault-free random signal set of process inputs $U(t) = [Q_1(t), Q_2(t)]$ and output measurements $X(t) = [h_1(t), h_2(t), h_3(t)]$ has been used (Figure 8.4). The training of these dynamic models has been achieved via determining the values of the parameters $[\xi_l]$ that maximize the concentrated log-likelihood of the training data. This task involves a computationally expensive nonlinear unconstrained optimization problem due to the iterative inversion of the correlation matrix $[R]_{n\times n}$. Since the MVDKs are programmed and implemented in Matlab subroutines, the aforementioned nonlinear optimization problem has been solved using the "*fmincon*" algorithm included in the Matlab optimization toolbox library.

A nontrivial challenge that often faces OK training is the selection of the appropriate initial values required to start this optimization: a local search optimizer can be easily trapped in local optima, due to the complexity of the likelihood function. Different optimization trials (starting from different initial parameter values) are recommended in order to guarantee a successful fitting task. Table 8.1 illustrates the MVDKs optimal parameters that resulted from the training task. The training of the predictor (in this case, the three OK dynamic models) required a relatively high computational effort (Table 8.4), although it should be emphasized that this task is performed offline. Yet, the main element to assess system performance is the prediction time, which is small enough.

Table 8.1. Parameters for the three OK dynamic models.

	μ_{ok}	σ_{ok}^2	λ	ξ_{h_1}	ξ_{h_2}	ξ_{h_3}	ξ_{Q_1}	ξ_{Q_2}
$\widehat{\boldsymbol{h}}_{\boldsymbol{1}}^{\boldsymbol{est}}{}_{(t+1)}$	0.572	0.159	0.0020	0.166 0	0.0416	0.0681	0.0398	0.0044
$\widehat{\boldsymbol{h}}_{\boldsymbol{2}}^{\boldsymbol{est}}{}_{(t+1)}$	0.143	0.095	0.0029	0.0948	0.0500	0.0476	0.0296	0.0717
$\widehat{\boldsymbol{h}}_{\boldsymbol{3}}^{\boldsymbol{est}}{}_{(t+1)}$	0.518	0.161	0.0019	0.1201	0.0354	0.0599	0.0124	0.0066

The predictor is validated by using it to estimate the process outputs (tank levels) corresponding to different inlet profiles. Figure 8.5 shows the validation inlet scenarios, and the predicted tank levels (dotted red lines) compared to the ideal outputs (solid black lines) and the process measured outputs (solid blue lines). The figure reveals the high accuracy of the predictor, and its efficient ability to identify the real underlying behavior of the outputs, achieving a very small NRMSE values for each model (1.05%, 1.1%, 1.02 %, respectively). The figures and the results also highlight the high potential capabilities of MVDKs to predict a multivariate behavior over relatively large time horizons (in this case, 800 steps ahead).

The zero-lag models (Eq.(8.7)) can approximate the system behavior with high accuracy. Therefore, there is no need to introduce any lagged behavior into the models (i.e. testing other lag values where $L>0$) and no reason to assume any extra cost in terms of the computational effort of the training and the prediction times, and the complexity of the resulting models structure.

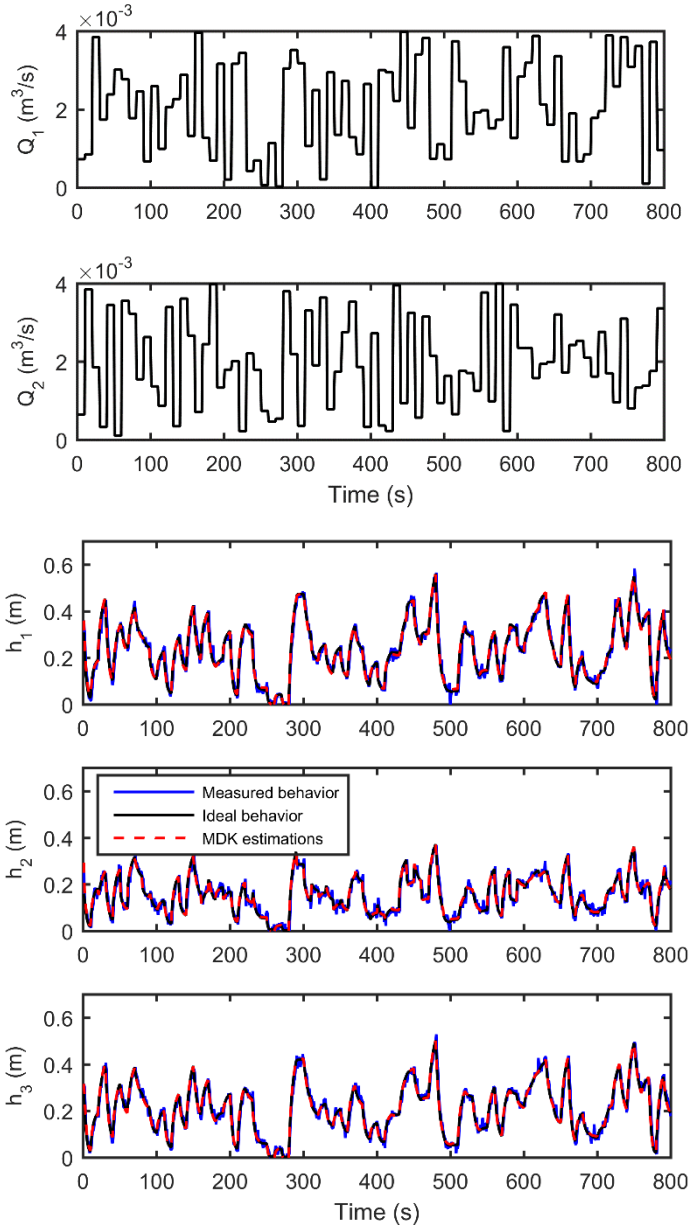


Figure 8.5. Validation of the MVDKs predictor.

8.3.2 Classifier construction

The second task in the offline stage is the CT construction using the residuals. For this goal, process historical data, including the inlets ($U_{PRT}^{trn} = [Q_1(t), Q_2(t)]$) and the tank levels ($X_{PRT}^{trn} = [h_1(t), h_2(t), h_3(t)]$), are collected under many process conditions, including normal and faulty situations [Nr, F1, F2, F3]. Figure 8.6-(a) shows the profiles of the inlets, Figure 8.6-(b) shows the faulty scenario, and Figure 8.6-(c) shows the measured tank levels (solid blue lines). In Figure 8.6-(b), the same sequence of faults [Nr, F1, F2, F3] is repeated three times, where at each time, the faulty scenario is consistent with a different dynamic behavior of the inlets (Figure 8.6-(a)).

Three dynamic modes of the inlets are selected, including sinusoidal, linear decreasing and linear increasing profiles. The objective is to gather -as much as possible- information about the effects of the faults (patterns) on the process under different modes of the control inputs (inlets), in order to obtain an accurate classifier. Conversely, the usage of a large number of training data could complicate the training of the classifier due to the required computational effort.

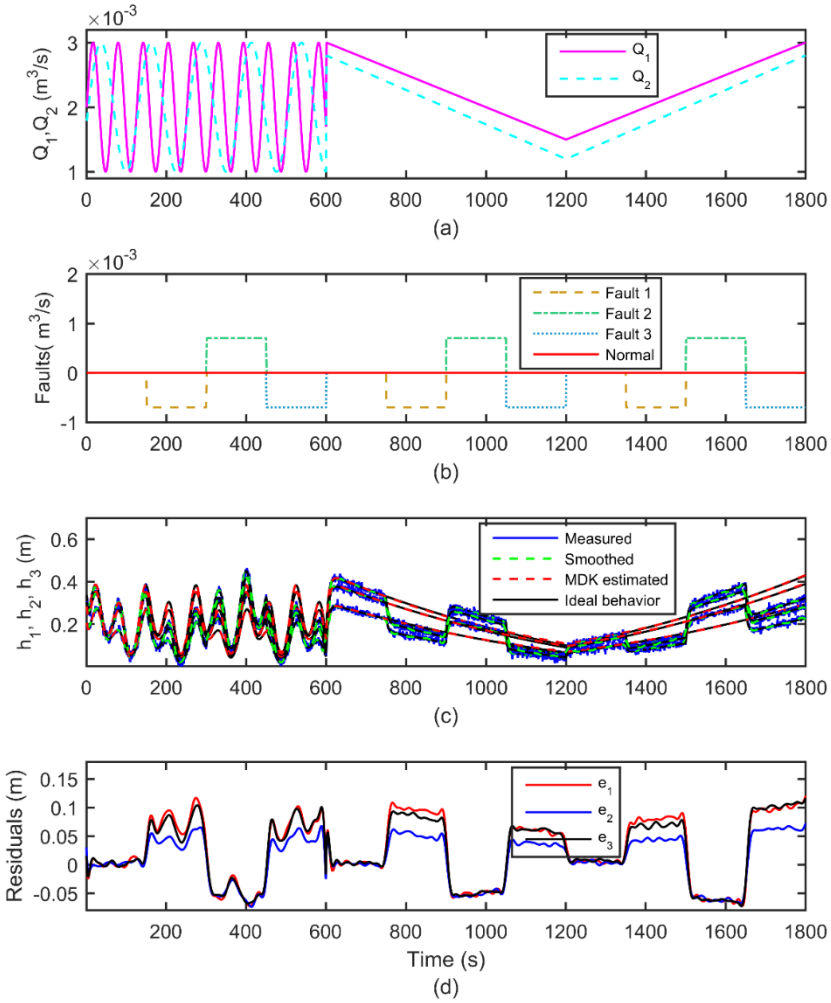


Figure 8.6. Training and validation data of the CT: Process inputs (a), faults scenario (b), outputs (c) and residuals (d).

Three kriging filters $\hat{h}_{PRT,i}^{trn-sm} = f_{sm,i}(t,\ h_{PRT,i}^{trn}),\ i = 1,2,3$ are fitted and used - as proposed in Section 8.2.2 - to filter out the noise from the measured outputs (Figure 8.6-(c), solid blue lines) in order to obtain the corresponding smoothed ones (Figure 8.6-(c), dotted green lines). Table 8.2 shows the parameter values of the OK-based filters, and the NRMSE of the values estimated by the filters (smoothed output values) compared to the ideal known behavior of the tank levels (Figure 8.6-(c), solid black lines). Using the information of the inlet profiles and the initial values of the tank levels $(h_1(0), h_2(0), h_3(0))$, the MVDKs predictor is used to predict the normal (fault free) behavior of the tank levels (Figure 8.6-(c), dotted red

lines): $\hat{X}_{PRT}^{trn-prd} = \hat{h}_{PRT,i}^{trn-prd}, i = 1,2,3$. Again, the MVDKs estimations of the tank levels are very accurate and close to the known ideal behavior given by the simulation (Figure 8.6-(c), black lines).

Table 8.2. Parameter values and NRMSE for the three OK static filters.

	μ_{ok}	σ_{ok}^2	λ	ξ_t	***NRMSE (%)***
$\hat{h}_{1,PRT}^{trn-sm}(t)$	0.2125	0.0012	0.2000	0.0036	1.41
$\hat{h}_{2,PRT}^{trn-sm}(t)$	0.1527	0.0008	0.2000	0.0029	1.20
$\hat{h}_{3,PRT}^{trn-sm}(t)$	0.1937	0.0010	0.2000	0.0033	1.40

Finally, the training residuals (Figure 8.6-(d)) are generated as the difference between the estimated tank levels (Figure 8.6-(c), dotted red lines) and the corresponding smoothed measured levels (Figure 8.6-(c), dotted green lines); $e_{PRT,i}^{trn} = \hat{h}_{PRT,i}^{trn-prd} - \hat{h}_{PRT,i}^{trn-sm}, i = 1,2,3$. The total set of 1800 data (Figure 8.6-(d)) is randomly separated into a training set (900 data) used to train the CT, and a validation set (900 data) used to assess the trained CT.

Table 8.3. Offline validation accuracy of the CT.

Validation accuracy (f1-score %)										
Classifiers	***Variable-based CT***					***Residual-based CT***				
	Nr	**F1**	**F2**	**F3**	**total**	**Nr**	**F1**	**F2**	**F3**	**total**
ANN	90.4	83.4	97.9	82.4	88.3	96.5	92.4	97.4	89.9	94.0
SVM	79.0	74.5	81.8	81.8	79.2	96.6	93.9	97.8	91.1	94.9
GNB	11.2	63.2	56.3	39.3	43.1	95.2	75.3	97.4	65.2	83.3
DT	87.5	79.3	91.5	79.2	84.4	97.9	89.3	98.0	88.5	93.4

In order to allow fair comparisons, the CTs are also trained in the classical way, using the inlets and the smoothed measured variables$[Q_1, Q_2, \hat{h}_{PRT,1}^{trn-sm}, \hat{h}_{PRT,2}^{trn-sm}, \hat{h}_{PRT,3}^{trn-sm}]$, since the residuals are also generated by comparing the smoothed actual outputs to the estimation of the predictor. Additionally, the training and validation data-sets are kept the same for both the residual-based and the variable-based classifiers. Table 8.3 shows the accuracy of several classifiers including the ANN, SVM, GNB and DT in terms of the $f1$-score, when they are trained and validated using the residuals and the process variables, and how the residuals are able to isolate the effect of the inputs from the effect of faults. The results also indicate that fault F2 (plugging in tank 2) is the most easy-to-detect fault, as it will be further explained in the following parts of this section. The table also indicates that among the four classifiers, ANN and SVM show the best accuracy in these cases, while GNB shows the lowest.

In this work, the MVDKs predictor, the static filters and the OK model are programmed and implemented in a Matlab subroutine. On the other hand, the ANN toolbox of Matlab is used to construct the ANN classifier through the "*patternnet*" function. The ANN structure is chosen to have two hidden layers, each one including seven neurons, and the function "*trainscg*", based on the "Scaled Conjugate Gradient" algorithm, is used for training the network. The ANN structure and the training algorithm are selected using a cut-and-try approach, in order to balance the network simplicity and its prediction accuracy. The SVM, GNB and DT classifiers are built using the Python 3.3.2-2013/05/15 libraries: A SVM classifier having a radial basis function kernel type is constructed using the Python SVM and grid search libraries, where the grid search library is employed for tuning the kernel parameters. Regarding the GNB and the DT classifiers, the "*GaussianNB*" library and the "*DecisionTreeClassifier*" library are used, respectively. Lastly, the computational effort required by the predictor, the static filter and the CTs for training and execution (i.e. to predict, filter or diagnose) are also reported in Table 8.4. It is important to note that the given execution times are per one prediction/filtering/diagnosis step (i.e., during one sampling period).

Table 8.4. Computational effort (training and execution) for the MVDKs predictor, the OK-based filters, and the CTs.

	Predictor	***Filter***	***Classifiers***			
			ANN	***SVM***	***GNB***	***DT***
Training (*)	2.08×10^3	53.45	1.650	0.009	<0.001	0.002
Execution (*)	0.009	0.003	0.003	<0.001	<0.001	<0.001

(*) CPU seconds, Intel(R) Core (TM) i7-4710HQ CPU @ 2.5GHz

8.3.3 Application

After the training and the validation of the MVDKs predictor and the CTs, the proposed framework is ready to supervise the process and to detect and diagnose eventual faults, through its application to several tests. The results obtained are next presented, discussed and classified according to the three main elements that characterize the robustness and flexibility of the proposed solution: the analysis of the performance of the proposed FDD method under different dynamic profiles of the process inputs, the assessment of this performance under different faulty scenarios, and the analysis of the sensitivity of the proposed method toward changes in the magnitude of the faults. In all cases, different faulty and normal situations will be combined. It is worth noting that this situation is not realistic, since after the detection of the earliest fault (whichever the type), some corrective actions will be taken in order to remedy this process malfunctioning as soon as possible. However, the objective of this study is to determine the prompt reaction and diagnosis consistency of the proposed FDD system, which

in a dynamic scenario can be better assessed by maintaining the faulty situation during a significant time period.

8.3.3.1 Robustness against changes in the dynamic inputs

For the sake of illustration, the first test involves a very simple scenario in which both inlets are assumed to be constant (Figure 8.7-(a)) while different step/abrupt faults occur (Figure 8.7-(b)), starting from normal conditions and following a sequence of faults [Nr, F1,F2,F3] during the whole time horizon (100 time intervals). Figure 8.7 shows the emulated process measurements (actual outputs, (c), solid blue lines) which are first smoothed ((c), green dotted lines) and compared to the MVDKs estimation of the outputs corresponding to the normal behavior ((c), red dotted lines); finally, this comparison is given by the residual values ((d)). In order to visualize the accuracy of the static kriging-based filters and the MVDKs predictor, the simulated data have been also represented in both cases ((c), solid black lines, showing almost identical behavior in all cases).

Figure 8.8 shows the classification labels (FDD results) estimated using the residual-based ANN (red stars) and the variable-based ANN (green stars) compared to the real scenario (black stars), where label "0" corresponds to the normal conditions, label "1" corresponds to fault F1, etc. Table 8.5 presents the performance of these CTs in terms of the $f1$-score. The proposed methodology detected and isolated the test faults with total $f1$-score of 98.4%, in front of a $f1$-score of 73.6% for the best result obtained (ANN results) using the classical variable-based CT. This conventional approach is severely affected by the slight difference between the patterns associated to faults F1 and F3. On the contrary, the use of the residuals (Figure 8.7-(d)) leads to more differentiated patterns for these faults. For the rest of faults (normal conditions and F2) the resulting patterns are quite different, and the FDD shows no problems.

A remarkable issue is the low failure rate of the proposed FDD method, and the fact that most failures correspond to transitions between the different faults/conditions: In some of these cases, the delay includes a wrong detection of a condition that does not coincide with any of the states associated to this transition, typical result of the application of a filter to smooth the process information. The same behavior can be also identified at the beginning of the test, when the static kriging filters show low prediction performance because of the lack of data. In any case, all these effects are shown to be amplified when a variable-based classifier is used.

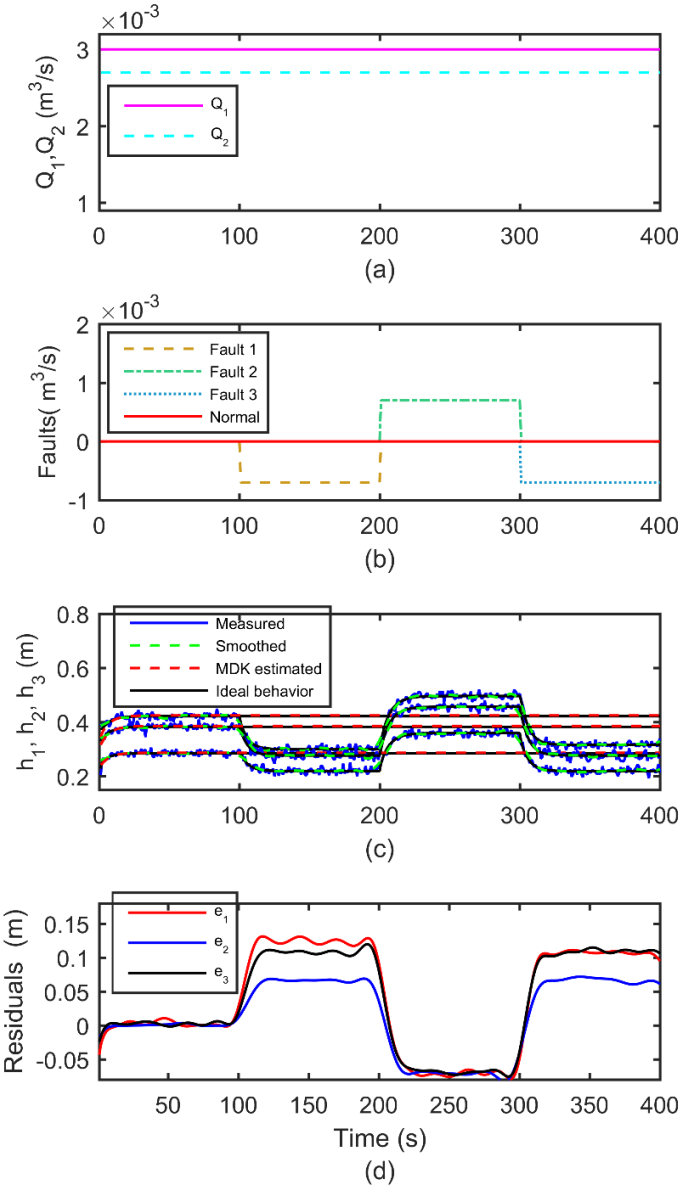


Figure 8.7. Test 1: Process inputs (a), faulty scenario (b), outputs (c) and residual values (d).

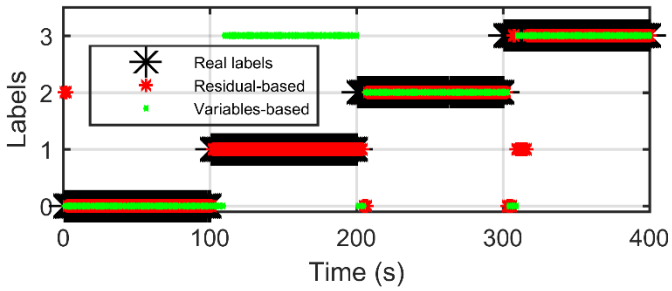


Figure 8.8. Test 1: Data-based Diagnosis: Simulated situation, proposed error-based method and variable-based CT.

Additional tests are next carried out, each one including different inlet profiles, while the faulty scenario [Nr, F1, F2, F3] is kept constant. The inlet profiles present increasing

complexity, from the same simple behavior in test 2 (Figure 8.9-(a)) to profile of the fifth test (Figure 8.9-(d)), which includes a linear increase of the first inlet parallel to a linear decrease of the second inlet, as well as flowrate values higher than those used for the CT training. In any case, it is worth to emphasize that this different behavior of the variable-based CT in front of the residual-based CT is very difficult to be detected during the training (Table 8.3): although both the training and the testing sets include more complex input profiles, compared to test 1, training results seem correct.

Results for all the tests are illustrated in Figure 8.11 and Table 8.5. Middle subplots show the smoothed actual outputs and the MVDKs estimations, which are compared in order to generate the residuals (bottom subplots). The MVDKs models produce accurate estimations, as well as they show robustness and flexibility to predict the system behavior under changing inlet profiles. The figures also show that the resulting FDD isolates the effect of changing the forcing inputs, generating distinct patterns for each fault type even with the change of the control inlet profiles, which is extremely important to the CTs in order to easily detect and diagnose the faults. The complete test results in Table 8.5 reveal that, for all CTs, the accuracy attained by the residual-based approach is always higher than the accuracy produced by the variable-based approach. ANN and SVM appear to be very competitive, showing the best accuracies among the classifiers, even when they are used as variable-based CTs, while GNB shows the lowest FDD accuracies.

It is worth noting that, in the second test, the variable-based approach performs in a similar way to the residual-based method, especially for the ANN and DT classifiers. This can be explained because the inlet profiles in this case were included along the training data for the offline construction of the CTs (see Figure 8.6). On the contrary, in the fifth test, the accuracy of the variable-based approach is about 50 % of that obtained by the residual-based methodology, which is maintained even when the control inputs violated the limits of the offline training. This confirms the high robustness of the proposed methodology.

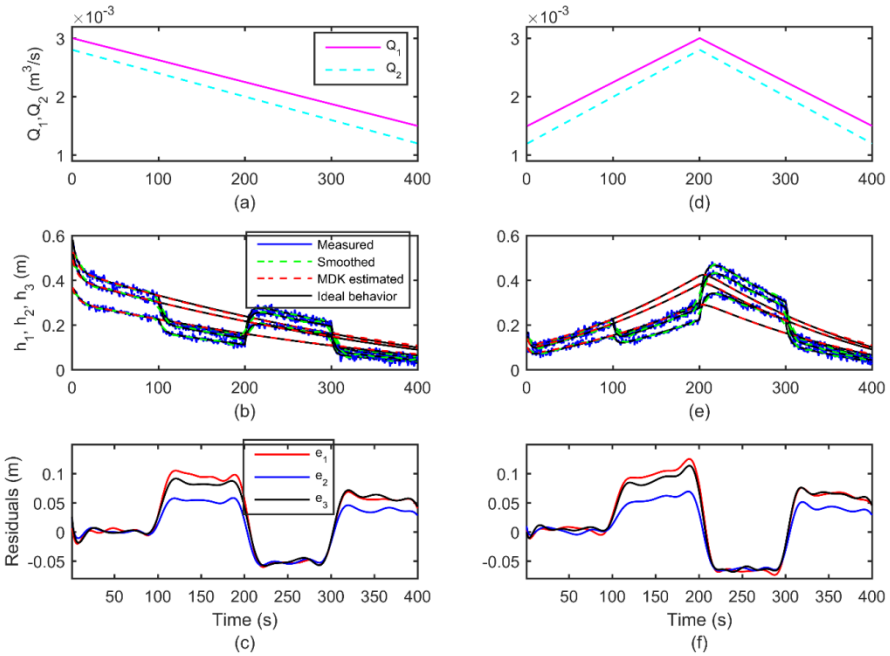


Figure 8.9. Test 2 (a,b,c) and test 3 (d,e,f).

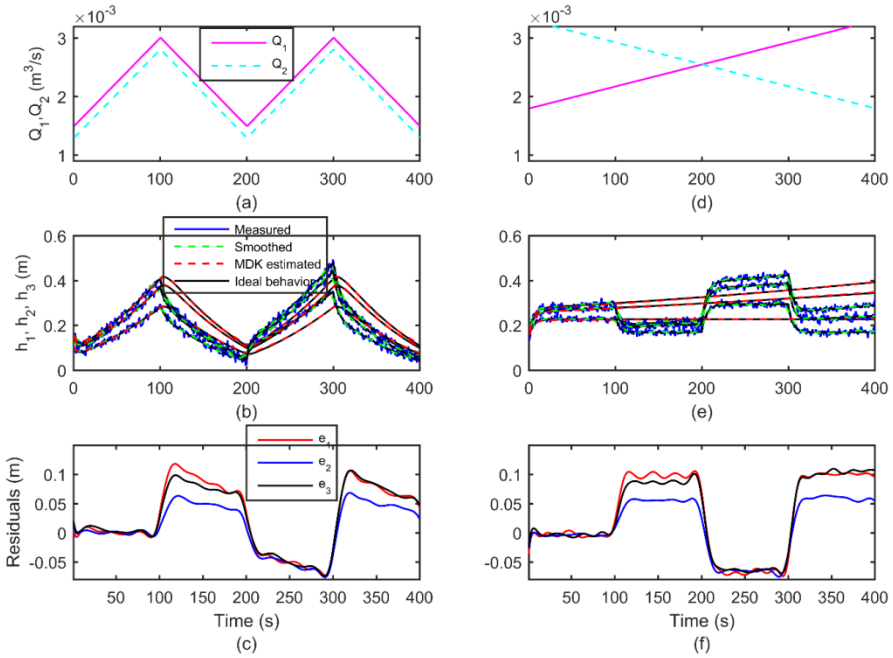


Figure 8.10. Test 4 (a,b,c) and test 5 (d,e,f).

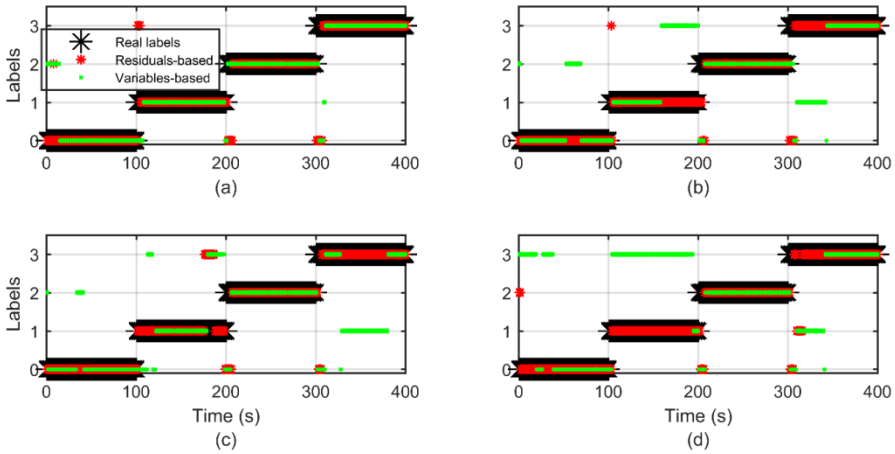


Figure 8.11. The ANN classifiers estimated labels compared to the ideal labels of: Test 2 (a), test 3 (b), test 4 (c) and test 5 (d).

Table 8.5. FDD accuracy based on ANN, SVM, GNB and DT classifiers under different inlet scenarios.

		Residual-based CT				*Variable-based CT*			
		ANN	SVM	GNB	DT	ANN	SVM	GNB	DT
Test 1	Nr	98.4	97.5	96.0	98.9	96.10	92.5	62.0	3.8
	F1	98.8	94.7	15.5	52.1	00.0	0.0	0.0	0.0
	F2	98.2	95.3	94.4	96.4	98.2	66.0	3.2	64.7
	F3	98.2	93.1	62.8	73.5	65.8	84.7	0.0	59.9
	F all	98.4	95.2	67.2	80.2	73.6	60.8	16.3	32.
Test 2	Nr	94.7	95.2	92.7	92.9	86.9	92.9	40.0	82.4
	F1	97.0	96.9	86.6	87.8	95.4	84.6	71.3	86.5
	F2	97.0	96.3	94.8	96.9	92.2	84.2	63.5	83.3
	F3	94.4	94.4	80.2	78.7	94.7	94.7	72.6	95.3
	F all	95.8	95.7	88.6	89.1	92.3	89.1	61.8	86.9
Test 3	Nr	97.1	96.1	96.1	95.8	84.4	62.5	0.0	94.1
	F1	96.0	95.4	48.6	70.1	58.5	47.7	33.3	64.7
	F2	96.4	95.8	95.3	92.2	85.9	76.3	58.2	94.6
	F3	95.4	94.4	33.1	71.8	57.7	79.5	47.3	54.9
	F all	96.3	95.4	68.3	82.5	72.0	66.5	34.7	77.1
Test 4	Nr	95.2	94.7	95.6	98.4	85.0	51.3	0.0	60.5
	F1	93.0	88.2	61.4	84.1	56.3	36.1	36.5	48.3
	F2	96.0	96.4	95.9	94.6	92.2	61.7	39.3	66.9
	F3	90.8	87.5	34.4	83.5	45.0	56.0	19.0	37.8
	F all	93.8	91.7	71.8	90.2	70.8	51.3	23.7	53.4
	Nr	96.1	95.6	95.6	90.9	53.6	0.0	17.2	92.6
	F1	94.2	98.0	85.0	89.5	08.6	3.9	56.1	14.8
Test 5	F2	95.0	95.9	94.9	91.5	95.4	32.1	43.2	68.0
	F3	93.6	96.4	80.0	87.0	38.3	96.9	0.0	91.1
	F all	94.8	96.5	88.8	89.7	50.0	33.2	29.1	66.6

8.3.3.2 Performance under different faulty scenarios

In order to analyze the effect of a more complex sequence of faults on the diagnosis capabilities, different faulty scenarios have been incorporated considering the same inlet profiles. Figure 8.12 summarizes the scenario and the results of the first of these applications (test 6): the inlet profiles are shown in part (a), and the first faulty scenario [Nr, F1, F2, F3] is shown in part (b); Figure 8.13 illustrates the conditions of the second and third applications, corresponding to different sequences of faults: [F2, Nr, F3, F1] and [F3, F1, Nr, F2]. The figures show again the accuracy and the flexibility of the methodology, since the residuals are able to establish the fault patterns regardless of their arrangement or sequence. Table 8.6 compares the FDD accuracy of the proposed method to the FDD accuracy of a variable-based CT along these three tests. The results prove that the sequence of the faults does not significantly affect the performance of the proposed method. The comparative performance of the different CT methods is maintained, and a significant enhancement of the proposed residual-based method in front of the equivalent variable-based methods is evident.

Again, the limitations of the static kriging filter appear only at the early stages of the changing behavior. Although test 8 evidences a more complex situation (Figure 8.13-(b)), this does not significantly affect the diagnostic performance in the residual-based proposed procedure. It is worth noting that in this case changing the sequence significantly affects the residual patterns, in part because the hold-up of the tanks have been affected by the previous faults. In this sense, good results when changing the sequence assure that the classifiers do not over fit the system incorporating the fault sequences included in the training data in the patterns to be recognized; in this specific case, this also assures reliability, confirming that the diagnosis system is really considering the dynamics of the process.

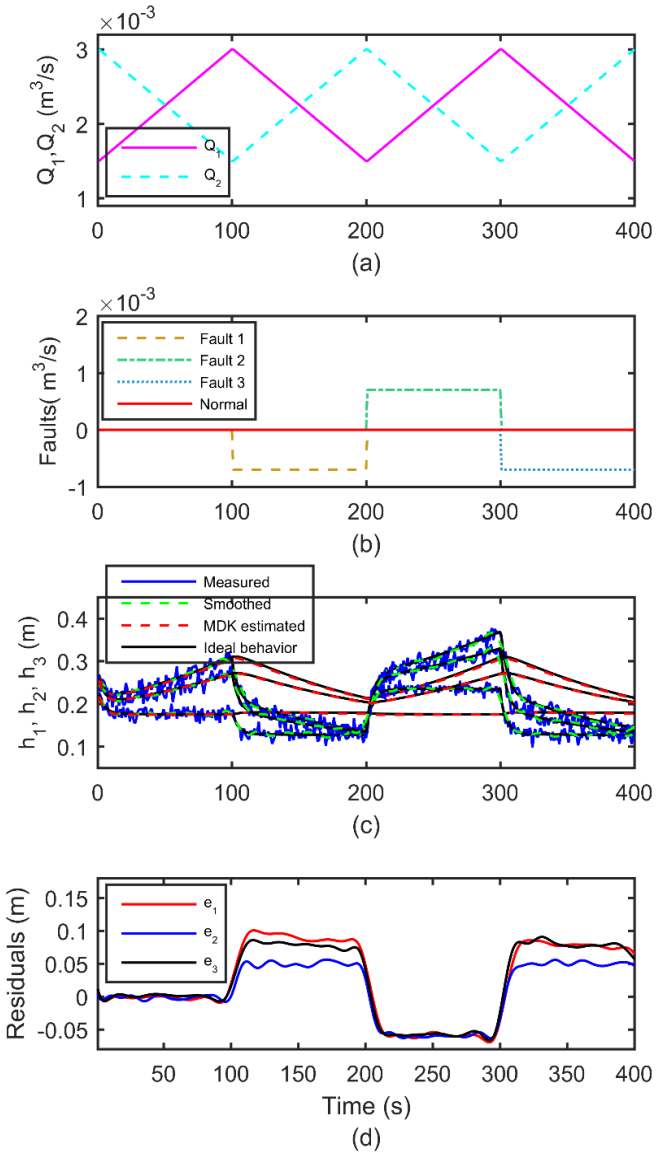


Figure 8.12. The application results of the methodology to test 6.

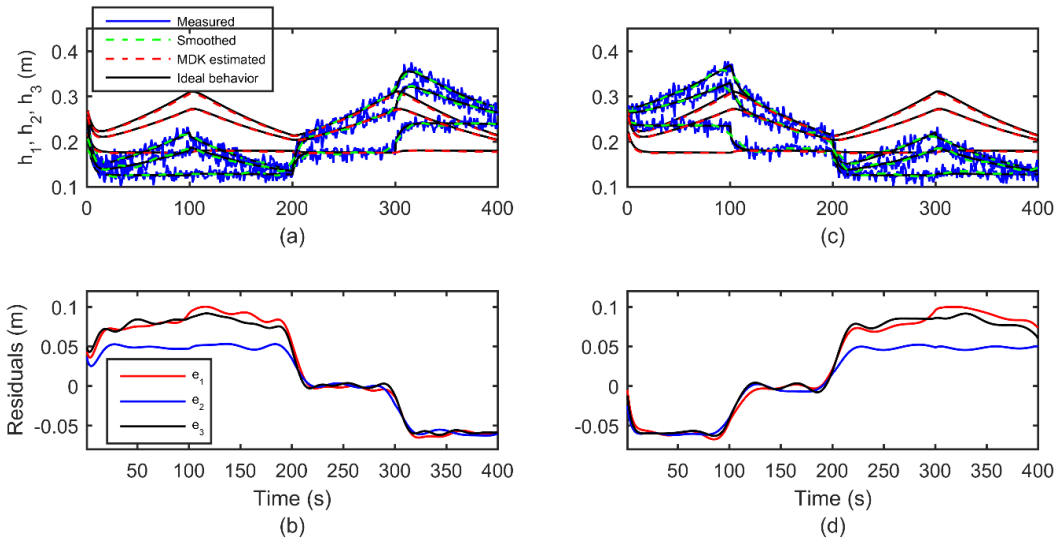


Figure 8.13. The application results of the methodology to: (a) test 7, (b) test 8.

Table 8.6. FDD accuracy (f1-score) based on ANN, SVM, GNB and DT classifiers under different faulty scenarios.

		Residual-based CT				***Variable-based CT***			
		ANN	***SVM***	***GNB***	***DT***	***ANN***	***SVM***	***GNB***	***DT***
Test 6	Nr	96.2	96.1	95.6	96.4	94.8	31.2	14.2	29.7
	F1	93.0	93.0	65.2	84.7	65.7	70.4	64.5	45.4
	F2	97.0	97.4	96.9	96.5	95.8	63.3	63.0	63.2
	F3	89.6	90.1	08.8	79.7	20.0	42.3	5.5	17.6
	F all	94.0	94.2	66.6	89.3	73.8	51.8	36.8	39.0
Test 7	Nr	93.3	95.3	93.6	79.2	93.6	30.1	9.5	12.5
	F1	96.6	97.5	69.4	96.0	65.9	76.0	66.2	50.2
	F2	95.2	96.6	94.7	89.2	96.2	63.2	69.4	60.7
	F3	94.7	96.3	19.8	90.7	34.3	50.6	3.6	16.7
	F all	95.0	96.4	69.4	88.8	75.0	55.0	37.1	35.0
Test 8	Nr	96.4	97.4	95.8	95.2	67.1	32.4	05.7	23.3
	F1	96.1	98.0	68.5	78.5	56.3	73.0	65.3	48.6
	F2	99.0	99.5	99.0	98.0	84.7	59.2	70.4	61.9
	F3	93.5	96.0	20.0	73.2	26.6	52.8	10.8	26.4
	F all	96.3	97.7	70.8	86.2	61.3	54.4	38.0	40.1

8.3.3.3 Sensitivity to the magnitude of the faults

Most of the studies that have been performed in the area of FDD methods have been illustrated considering a fixed threshold value to characterize each fault. However, in real situations the faults may occur with different magnitude or intensity degrees. Hence, an important characteristic that is worthy to be examined is the robustness of the method toward different magnitudes of the fault(s), i.e.: to which extent the method is able to handle a range of fault magnitudes with acceptable FDD accuracy. So, in this section, the analysis will include the application of the proposed method to the simplest studied scenario (test-1; Figure 8.7) but manipulating the absolute magnitudes of the faults. Also, for simplicity, and in order to facilitate the comparison between the different studied situations, it will be considered that, in the faulty scenarios, the magnitude/intensity of all the faults [F1, F2 and F3] will take place with the same absolute magnitude/intensity. The absolute values for these faults will be then fixed within a specific range [0.0002: 0.0012 m^3/s] around their nominal values (0.0007 m^3/s). Only the results obtained by the ANN and the SVM methods for CT are shown, since these have been the ones consistently showing the most accurate results in all previous cases.

Figure 8.14 shows the overall $f1$-score (diagnosis accuracy) along each scenario, for the normal and faulty situations, given the different intensities of each fault. The performance of both, the proposed residual-based methodology (red colors) and the equivalent results using

variable-based CT (green colors) can be compared, as well as the two selected methods for CT. In general, the increase of the fault magnitude above its nominal value (0.007 m^3/s - vertical dotted black line) leads to slight improvements of the FDD accuracy, and the reduction in the fault magnitude leads to gradual reduction in the accuracy of all the proposed methods, but only until a certain threshold is reached, after which a rapid decrease of the accuracy starts. So, in general, when the fault magnitude is large enough, the performance of any CT method becomes good enough, highlighting the need of a good selection in order to build flexible and sensible FDD systems. The results also clearly confirm the superiority of the proposed method, since it is able to efficiently detect and diagnose smaller magnitudes of the same faults with good accuracy. On the contrary, the classical CTs show significant decays in their performance when dealing with small fault magnitudes and, in the extreme cases, they even become unable to detect some faults at all (as F1 and F3, see Figure 8.14-(b) and (d) respectively) while the proposed procedure still shows good performance.

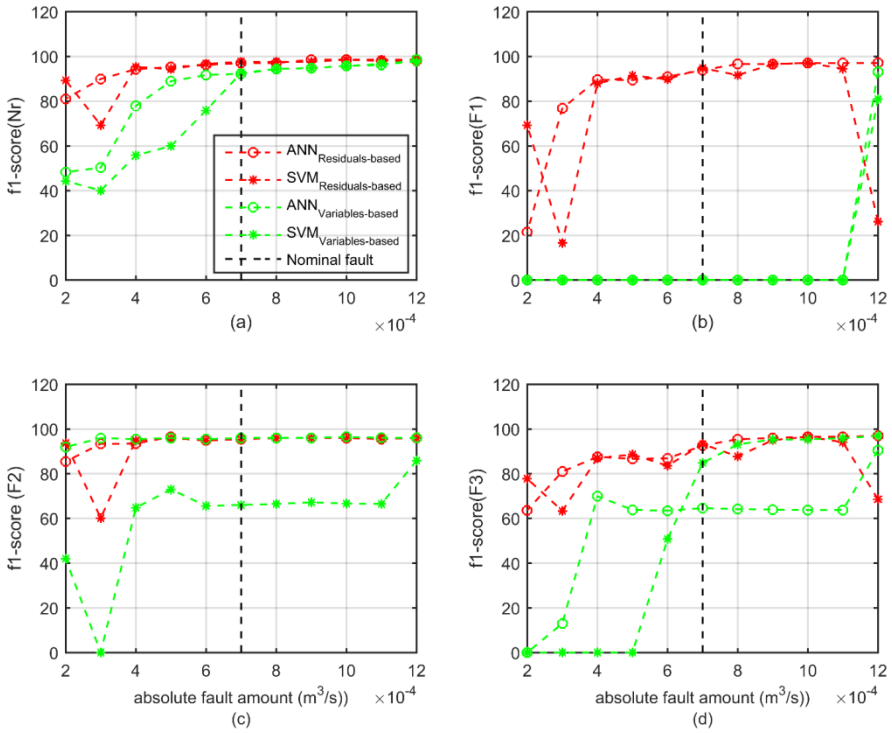


Figure 8.14. f1-score of the four classes for different values of the faults (absolute magnitudes).

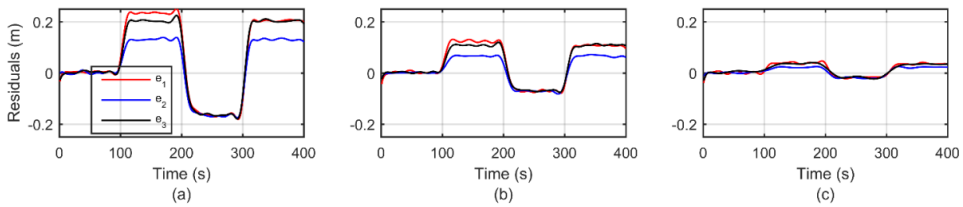


Figure 8.15. Residual signals for test 1, using different magnitudes of the faults: (a) 0.0012 m^3/s (b) 0.0007m^3/s, and (c) 0.0002 m^3/s.

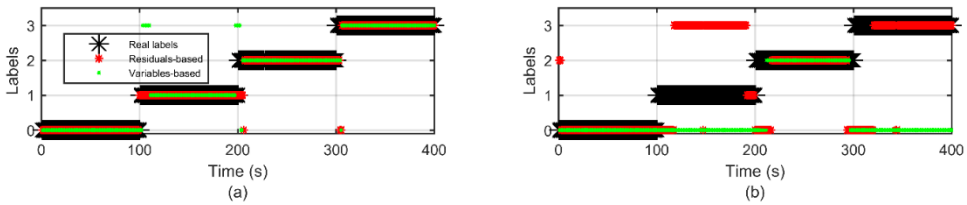


Figure 8.16. ANN classification labels of two extreme cases of the fault magnitudes (test 1): (a) 0.0012, (b) 0.0002 m3/s.

This generic correlation between the FDD accuracy of the proposed method and the fault magnitudes could be further explained by the results shown in Figure 8.15, where the residual signals for different absolute magnitudes of the faults are shown. It is clear how the patterns of the faults included in the residual signals change, facilitating or complicating the mission of the CT which has not been trained for these scenarios, to correctly classify them. Figure 8.16 shows the detail of the obtained diagnosis using the residual-based ANN and the variable-based ANN classifier.

The previous analyses prove that the proposed methodology is able to detect and diagnose the process state under different profiles of the control inputs dynamics and different faulty scenarios, in an accurate flexible and robust way, allowing a significant enhancement of the performances of the different classifiers. Additionally, the method also proved its efficiency and significant enhancement in the FDD performance in cases where the faults affect the process with magnitudes or degrees different than the ones included in the training data. But in all these analyses, the faulty scenarios were always taking the same “step” style. This systematization is required to present a fair comparison between the performance of different classifiers under different inlet scenarios or different magnitudes or intensities of the faults and obtaining clear conclusions and a deeper understanding of the method behavior, but it is usually far from realistic scenarios. Thus, in order to confirm the previous conclusions, the method has been finally applied to a more complex test consisting of a complicated piecewise constant profile of inlets (Figure 8.17-(a)), and also a complicated sequence of faults with different changing magnitudes, totally different than the sequence and magnitudes included in the training data (Figure 8.17-(b)). Some of them can be considered as “incipient faults”, which gradually (nonlinearly) increase with a pattern that is also far away from what has been considered during the training of CTs.

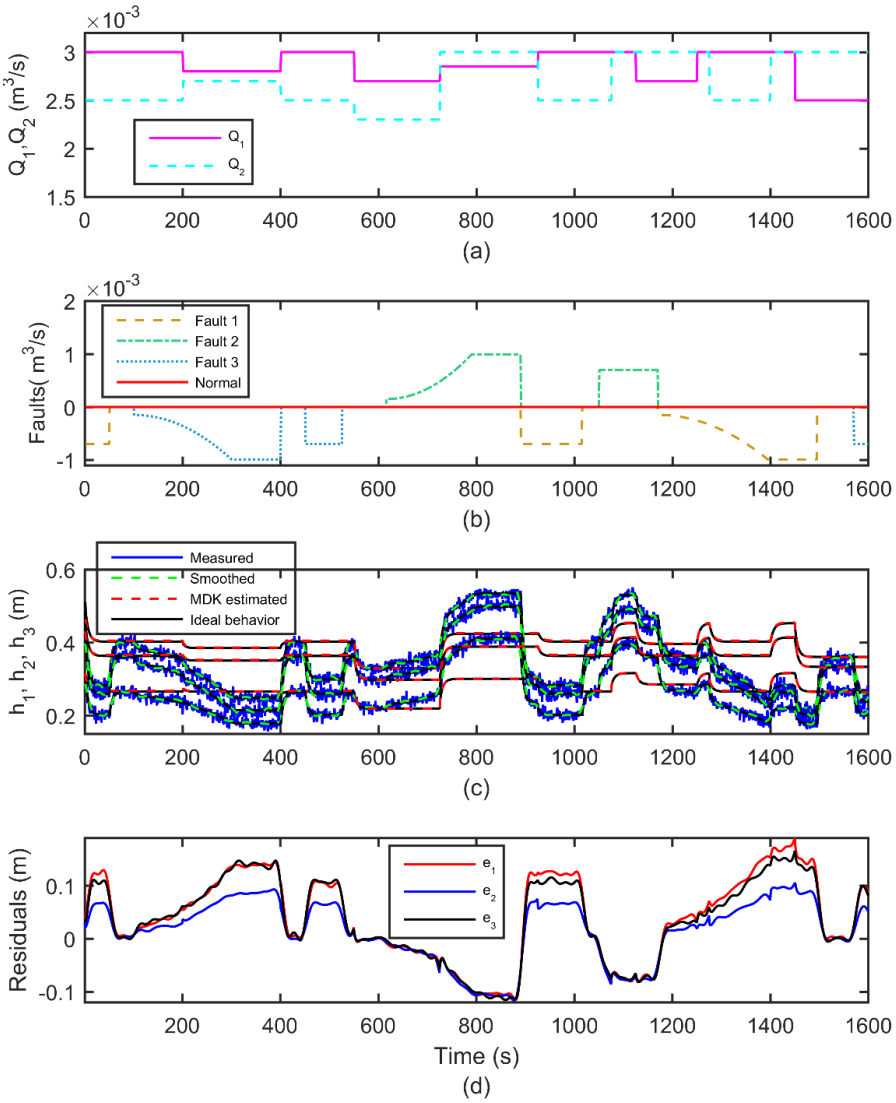


Figure 8.17. The application results of the methodology to test 9.

Figure 8.17-(c) shows the measured information, leading to the obtained residuals (Figure 8.17-(d)), while Table 8.7 shows the accuracy of the overall FDD obtained using the proposed method and its equivalent using a variable-based FDD. These overall results (FDD performance) are also represented in Figure 8.18. The CTs based on ANNs and the SVMs confirm their robustness although, when compared with the previous reported results, in all cases the overall FDD accuracies have been reduced (as it was obviously expected). Table 8.7 also quantifies the enhancement obtained by the use of a residual-based CT, which is always significant. Also, the problem of the misclassification of F1 and F3 appears for all the variable-based classifiers. Figure 8.18 also illustrates how the differences are more evident when trying to detect and diagnose incipient faults, since in all cases the pattern of the residuals at the early moments of the incipient fault is much clearer.

Table 8.7. FDD accuracy of test 9 based on ANN, SVM, GNB and DT classifiers.

		Residual-based CT				***Variable-based CT***			
		ANN	***SVM***	***GNB***	***DT***	***ANN***	***SVM***	***GNB***	***DT***
Test 9	**Nr**	76.7	77.4	79.9	79.2	62.9	48.2	32.5	50.7
	F1	85.5	87.1	22.4	54.9	0.4	0.0	0.0	15.5
	F2	95.2	97.3	96.0	89.0	91.7	28.7	18.9	88.5
	F3	83.0	87.0	54.8	65.7	55.1	72.9	0.0	31.0
	F all	85.4	87.0	59.6	70.6	57.3	34.6	10.7	44.0

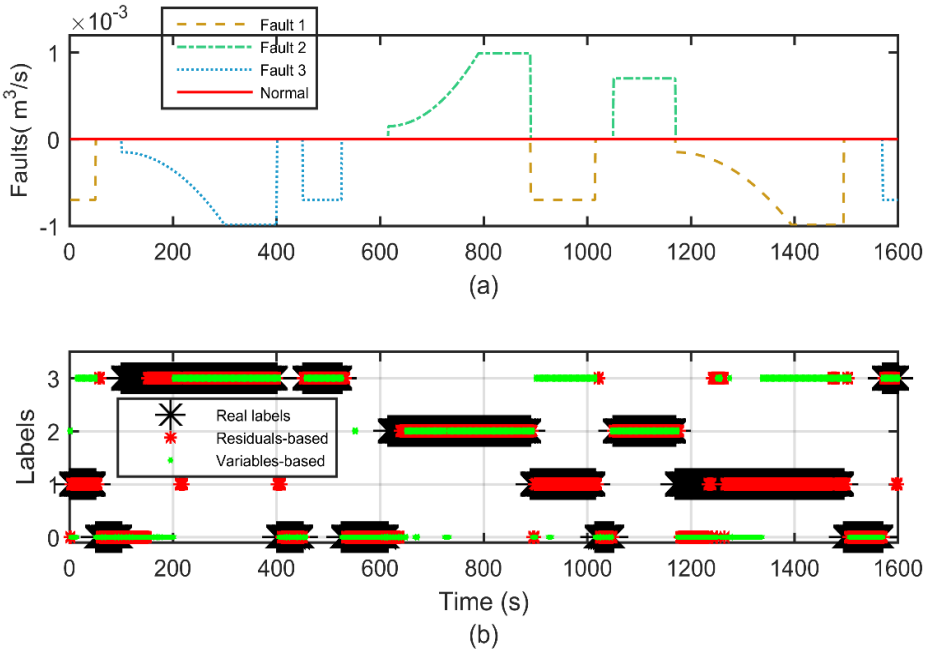


Figure 8.18. ANN labels, compared to the ideal labels (test 9).

8.3.3.4 Sensitivity to the measurement noise

In order to confirm the robustness of the selected signal smoothing procedure to compensate for the effect of sensor noise, two additional types of artificial sensor noise patterns have been tested: uniform and logistic probability distributions. The parameters of both distributions have been chosen in such way that the ranges of the corresponding deviations are similar to the previously used one (Gaussian distribution: ($\mu = 0, \sigma = 0.01$); uniform distribution with $(a = -0.0175, b = 0.0175)$, and logistic distribution with $(\mu = 0, s = 0.01)$). Figure 8.19 shows the noisy measured tank levels (in yellow, blue and green solid lines) and their smoothed values (dashed red lines) using the kriging filters, as well as their ideal behavior (solid black lines). The results over the FDD accuracy (Table 8.8 and

Table 8.9) confirm that the proposed procedure is still able to maintain a consistent high level of accuracy in all cases; in this sense, the effects of the noise appear to be significantly more important in the variable-based CT, although there is not any specific pattern on the found differences. Additionally, Table 8.10 shows the NRMSE of the OK-based filters with respect to the known ideal behavior, also confirming the capabilities of the OK-based filters for smoothing different random noise types, achieving very low normalized root mean square error values.

Table 8.8. FDD accuracy of test 3 using the ANN classifier under different types of the artificial sensor noise.

		Residual- based ANN CT			***Variable-based ANN CT***		
		Gaussian	***Uniform***	***Logistic***	***Gaussian***	***Uniform***	***Logistic***
Test 3	**Nr**	97.0	96.6	96.6	84.4	75.5	81.0
	F1	96.0	95.5	96.0	58.5	62.1	61.8
	F2	96.4	96.4	96.4	85.9	81.0	84.3
	F3	95.4	95.4	94.9	57.7	59.4	58.0
	F all	96.2	96.0	96.0	72.0	70.0	71.7

Table 8.9. FDD accuracy of test 3 using the SVM classifier under different types of the artificial sensor noise.

		Residual-based SVM CT			***Variable-based SVM CT***		
		Gaussian	***Uniform***	***Logistic***	***Gaussian***	***Uniform***	***Logistic***
Test 3	**Nr**	96.1	97.0	96.6	62.5	80.0	72.6
	F1	95.4	95.5	96.4	47.7	53.0	49.7
	F2	95.8	96.4	95.8	76.3	77.8	78.4
	F3	94.4	94.9	93.0	79.5	80.2	73.4
	F all	95.4	95.9	95.5	66.5	72.7	68.5

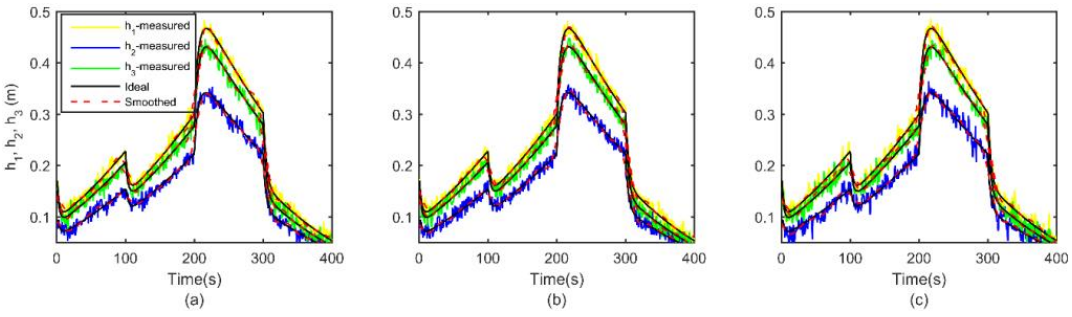


Figure 8.19. Filter performance in test 3, using three different random sensor noises: Gaussian (a), uniform distribution (b) and logistic distribution (c).

Table 8.10. NRMSE (%) of the filters prediction in test 3.

Noise type	$\widehat{\boldsymbol{h}}_1^{sm}(\boldsymbol{t}) = \boldsymbol{f}_{sm,1}(\boldsymbol{t}, \boldsymbol{h}_1(\boldsymbol{t}))$	$\widehat{\boldsymbol{h}}_2^{sm}(\boldsymbol{t}) = \boldsymbol{f}_{sm,2}(\boldsymbol{t}, \boldsymbol{h}_2(\boldsymbol{t}))$	$\widehat{\boldsymbol{h}}_3^{sm}(\boldsymbol{t}) = \boldsymbol{f}_{sm,2}(\boldsymbol{t}, \boldsymbol{h}_3(\boldsymbol{t}))$
Gaussian	1.64	1.31	1.46
Uniform	1.54	1.29	1.49
logistic	1.77	1.44	1.80

8.4 CONCLUSIONS

This work proposes a hybrid approach for data-based FDD of nonlinear noisy dynamic processes, affected by changes in the manipulated inputs. Typical applications include situations where the control system reacts in order to compensate for uncontrolled disturbances, transitions between different operating conditions, batch and fed-batch processes, or plant start-up processes, when the failure rate is likely to be higher. The proposed method combines three different techniques, namely data-based dynamic models capable to estimate the expected state of the system (predictor), static models smoothing plant data to reduce the noise effects, and data-based classification techniques trained with the patterns created with the residuals from the comparison between the predictor results and the process smoothed information.

The use of a data-based predictor system is a necessary step of the proposed approach, transforming data measurements into highly sensitive residual values without requiring an analytical model of the system. The proposed predictor system, based on Ordinary Kriging, has shown very high accuracy (NRMSE) to approximate process nonlinear dynamic behaviors even though it is trained with noisy process data.

The use of residuals (error signals) in addition to process variables is also a key element which enables a better identification of the process state, since residuals compensate the effects of the manipulated inputs on the process outputs. Unlike other works combining CTs and data-based predictors, the proposed approach takes the maximum advantage of the generated residuals, by exploiting their patterns in addition to their values. This significantly improves the FDD efficiency, gives the flexibility required to deal with different fault severities and allows the system to better identify incipient faults. The use of the proposed approach has been illustrated through its application to a FDD benchmark problem. In this case, the analytical model of the system is known, so the theoretical/ideal behavior of the system may be calculated, allowing a fair assessment, clear analysis and credible comparison of the obtained results in different scenarios.

Different CTs implementations, based on usual classification techniques such as ANN, SVM, GNB and DT, have been developed and tested for this case study, using the proposed approach and other state of the art methods. All these implementations established the

enhanced performance of the residual-based approach over the variable-based approach, as well as the better performance of ANN and SVM over GNB and DT.

When an analytical or mechanistic model-based FDD is unavailable or unaffordable, the proposed approach is revealed as an efficient alternative for data-based FDD. The approach developed in this work has shown to accurately detect and diagnose faults under different input profiles and in different faulty scenarios. Moreover, a significant enhancement in the classification performance has been obtained in the cases where the faults affect the process in a way significantly different from the situations included in the training data.

These encouraging results open some issues for future investigations, most of them related to the exploration of the performance of the system in more realistic working conditions (closer to the real industrial challenges) to check if the advantages now reported are maintained, for example, when the input disturbances affect both normal and abnormal conditions, when a closed-loop control system is in operation (thus compensating the effects of the faults), or when the non-linearities of the system are stronger, complicating the task of the prediction system.

Chapter 9: Soft-Sensors for Batch Processes with Different Initial Conditions

In this Chapter, soft-sensing methodologies applicable to batch processes operated under changeable initial conditions are presented. These cases appear when the raw materials specifications differ from batch to batch, different production scenarios should be managed, etc. The proposal exploits the capabilities of the machine learning techniques to provide practical soft-sensing approaches with minimum tuning effort in spite of the fact that the inherent dynamic behavior of batch systems are tracked through other online indirect measurements. Current data modelling techniques have been also tested within the proposed methodologies to demonstrate their advantages. Simulation case-studies and a pilot-plant case-study involving a complex batch process for wastewater treatment are used to illustrate the problem, to assess the modelling approaches and to compare the modelling techniques. The results reflect a promising accuracy even when the training information is scarce, allowing significant reductions in the cost associated to batch processes monitoring and control.

9.1 INTRODUCTION

Competitive and rapidly changing market environments bear many sources of uncertainty and variability such as product demands, material availability, prices, product specifications and environmental restrictions. This has favored a continuous and growing interest in batch processes due to their high flexibility and adaptability, which also allow a quicker development of new products. These abilities stem from the relative independence of each equipment/unit and the possibility to reassign them and develop new production schemes. Thus, a wide range of important low-volume and high-value-added products are manufactured in a batch mode, including specialty chemicals, materials for microelectronics, pharmaceutical, agricultural and biochemical products, etc. (Jin, et al., 2014; Moreno-Benito, 2014).

However, batch processes typically exhibit challenging operational problems (high inherent nonlinearity, transient dynamic behavior with no steady-state operating point, complex reaction kinetics, mechanisms and stoichiometry, etc.) that hamper their optimal management (Bonne & Jorgensen, 2004). A key challenge that commonly complicates their

monitoring, supervision and/or control is the unavailability of online measurements of the process Quality Indicator Variables (QIV), which are often obtained through expensive and time-consuming offline sampling and processing (Zamprogna, et al., 2005; Desai, et al., 2006). Moreover, a large laboratory delay also hinders a reliable process monitoring and supervision (Liu, et al., 2012). Thus, soft-sensor techniques have been proposed as a promising solution that has proven its effectiveness in many situations (Hoskins & Himmelblau, 1988; Kadlec, et al., 2009).

Soft-sensors are computational techniques that provide online estimations of process variables (including QIVs) that cannot be measured online in a continuous and/or reliable way due to technological and/or economic reasons (Lin, et al., 2007). These techniques exploit the process variables that are reliably measured and recorded online with minimum cost by means of available physical sensors. Soft-sensors can be used for different purposes, but their basic application field is the online prediction of QIVs, so they could be further integrated in a monitoring and/or a control system (Kadlec, et al., 2009; Jin, et al., 2015).

Soft-sensor techniques can be categorized in two main classes: analytical model-based soft-sensors and data-based soft-sensors. Analytical model-based soft-sensors rely on First Principle Models (FPMs) that provide a detailed process description based on phenomenological knowledge (Lin, et al., 2007; Jin, et al., 2014); these FPMs are used to predict/monitor the process behavior, either solely or using the information provided by physical sensors (e.g. for continuously adjusting their parameters). However, accurate and reliable FPMs of chemical processes are often unobtainable, especially for complex highly nonlinear ones (Jain, et al., 2007): in many cases, the details required to build the models needed to describe such processes are limited, because of the involved highly nonlinear behaviors, sophisticated mechanisms and complex phenomena as reaction kinetics, thermodynamics etc. (Caballero & Grossmann, 2008). Even more, the existing FPMs of many processes have been developed under the assumption of the most favorable/ideal experimental and laboratory conditions, which make them sensitive to parameter variations, uncontrolled disturbances and distinct reactors geometries (Kadlec, et al., 2009; Jin, et al., 2015). Moreover, in real industrial or pilot plant scale applications, many other factors that affect or interact the process (e.g. mechanical and electrical components and connection etc.), usually are not considered by the FPMs, which badly affects their prediction accuracy (Kadlec, et al., 2009; Jin, et al., 2014).

As an alternative, data-based soft-sensors are gaining wide interest in the process industry, because of its practicability, robustness and flexibility to be developed and applied to a wide range of processes, in addition to their independence from a process mathematical model (Hoskins & Himmelblau, 1988). They are based on the construction of a data-driven

model able to accurately approximate the relation between the QIV and other online variables (Bonne & Jorgensen, 2004; Facco, et al., 2009). In this line, many machine learning techniques for regression have been used to identify this relation, as Artificial Neural Networks (ANNs) (Gonzaga, et al., 2009; Banu & Umab, 2011), Support Vector Regression (SVR) (Yan, et al., 2004; Desai, et al., 2006), and recently Gaussian Process (GP) models (Jin, et al., 2015). These machine learning models are built relying on the use of input (online variables)-output (QIV) data from the process history, in order to find a black-box relation that describes the underlying mapping between the inputs and the output (training step). Then the constructed soft-sensor is used for the online prediction of the QIVs once the other online variables (online inputs) are measured (Jin, et al., 2015). In this framework, measures of the QIV, when available, can be easily integrated in the system, either offline (e.g. Kaneko and Funatsu, (2013), to update/improve the soft-sensor) or even online (e.g. Jin, et al., (2015), to confirm estimations and eventually reset them to better match the current situation).

Data-based soft-sensors have been vastly applied to continuous processes, in order to predict the process steady state behavior, although they have shown limitations dealing with the transient states of the process (e.g. start-up and shut-down) (Facco, et al., 2009; Wang, et al., 2016). Comparatively, the development and application of data-based soft-sensors to batch processes, which are always in transient state, have been found to be relatively more complicated (Bonne & Jorgensen, 2004; Liu, et al., 2012).

The combination of Principal Component Regression (PCR) and Partial Least-Squares (PLS) techniques is the most common method for building data-based soft-sensors. PCR is used to reduce the input space via transforming it into a lower dimensional space in which the importance of each new feature/input is determined (Zamprogna, et al., 2005; Kadlec, et al., 2009; Jin, et al., 2014). PLS is then used to find a regression model that correlates the transformed input variables or features with the output variables (QIVs) (Zamprogna, et al., 2005; Lin, et al., 2007; Facco, et al., 2009). However, these methods are basically developed for linear modeling and they may oversimplify the description of complex nonlinear behaviors (Nagy, 2007; Jin, et al., 2015). Although some versions of these techniques have been developed to inexpensively handle nonlinearities (as kernel PCR and kernel PLS), more advanced and efficient techniques have been proposed for soft-sensing highly nonlinear processes (Kadlec, et al., 2009).

In this scope, Artificial Neural Networks (ANNs) approaches (Masters, 1993) have been often selected for soft-sensing, due to their universal approximation and efficient generalization performance, additional to their flexible structure of nonlinear neurons that enable ANNs to capture sophisticated behaviors (Kadlec, et al., 2009; Yan, et al., 2004) . Several types of ANNs have been efficiently used for soft-sensing, as feed-forward multi-

layer, radial basis and Fuzzy ANNs (Kadlec, et al., 2009). Nevertheless, ANNs particularly suffer from the curse of dimensionality and require laborious tuning (selection of the network structure and configuration; as number of layers, number neurons in each layer, transfer function type, training method etc.) to achieve a reliable model fitting (Azman & Kocijan, 2007; Davis & Ierapetritou, 2007).

Models based on Support Vector Regressions (SVR) have been also proposed for soft-sensing in batch processes (Desai, et al., 2006). The SVR model (Vapnik, 1995) consists of a subset of specific training data (support vector) that compose the margins of the simplest functional shape within which the prediction error for all training data is acceptable. SVR techniques have very good generalization properties, and rapidity of tuning (associated to the optimization problem solution time for the support vectors selection) (Forrester, et al., 2008). However, the effort and the time required to select the parameters of the SVR model –prior to the optimization-, as the penalty cost, the error margin and the variance become a major limitation.

In the Bayesian statistics and inference area, the Gaussian Process (GP) models are initially proposed by O'Hagan et al. (1978; 1999) in order to represent a general class of non-parametric probabilistic models. Later on, they have gained a wide popularity within the machine learning community through the works of Rasmussen and Williams (2006). Recently, GP models are attracting huge attention in the soft-sensing area, and have been applied either for continuous (Grbić, et al., 2013; Wang, et al., 2016; Liu, et al., 2016) or batch processes (Jin, et al., 2015), offering high prediction accuracy and tuning flexibility while requiring a relatively small set of the training data. Pioneering GP metamodels were developed in the nineties (O'Hagan, et al., 1999), mainly for complex computer code emulation (O'Hagan, 2001), and afterward they became popular for dynamic modeling (Azman & Kocijan, 2007). But the computational effort and capabilities required for its parameter tuning could be a serious shortage, especially for high dimensional cases and/or large training datasets.

The techniques based on Ordinary Kriging (OK) (Cressi, 1993) may be considered as specific forms/applications of the GP models. Their use was first proposed in the field of mining industry and geo-statistics by Dr. Danie Krige (1951) and Matheron (1963), where their predictor derivations and parameters estimations are sought as the "Best linear Unbiased Predictor" of the realization of a spatial process. After the works of Sacks et al. (1989) and Jones et al. (1998), the OK became very popular in the context of the modelling, simulation and optimization of complex nonlinear systems in many engineering fields (Queipo, et al., 2005; Forrester & Keane, 2009). Ordinary kriging has shown outperforming characteristics, as high prediction accuracy with relatively small number of training data, and relatively high tuning flexibility (Forrester, et al., 2008). The use of OK metamodels was introduced to the

chemical process engineering area by Davis and Ierapetritou (2007), and Caballero and Grossmann (2008), and since that time it is gaining growing interest mainly for surrogate based optimization and analysis of complex nonlinear chemical systems (Quirante, et al., 2018), and later on for multivariate dynamic modelling (Boukouvala, et al., 2011). However, the use of OK metamodels - as a specific implantation/formulation of Gaussian Process models - has never been introduced to the area of the soft-sensing of batch chemical processes yet.

In the literature, the studies that have addressed data-based soft-sensing of batch processes consider a single operating phase/mode along the batch run, which is also the scope of this work. Consequently, the machine learning techniques are harnessed to provide a single global data-based model that describes the relation between the online variables and the QIV along the whole batch (Yan, et al., 2004; Desai, et al., 2006; Grbić, et al., 2013; Gustavsson, et al., 2015; Banu & Umab, 2011). However, for alternate batch process conditions, these global models may produce inaccurate predictions in specific input regions, and in these cases, the process should be described through a multiphase/multimode model, based on the construction of several local models, each one responsible for predicting the process behavior over a specific local input domain (Liu, et al., 2012; Jin, et al., 2014; Wang, et al., 2016). Further research addresses the development of adaptive data-based soft-sensors to be used online in processes showing time-varying behavior, e.g. due to process fouling, aging etc. (Grbić, et al., 2013; Kaneko & Funatsu, 2013). Different approaches have been proposed as the moving window approach and the recursive adaptation methods that automatically manage online data to update the soft-sensor in order to maintain its prediction performance (Jin, et al., 2015).

Most of the data-driven soft-sensing approaches for batch processes proposed in the literature have not considered the initial conditions of the batches in their design, because they have been tailored for batch processes operated under fixed initial conditions. In these cases, the data-based soft-sensing approach have addressed the batch-to-batch data variability -due to a very slight change in the initial condition- from the uncertainty and noise perspectives: input-output training data from different batch runs are assumed to have random errors due to undesired disturbances, which are expected to be representative of a population of batches that are swarming around the mean behavior of the process or what is called the “reference batch” or the “golden batch” (Kadlec, et al., 2009). Then, the correct underlying process behavior can be identified, thanks to the regularization abilities of the employed machine learning models, which enable them to learn from this uncertain and perturbed data, and to filter out the assumed noise.

Considering this overview, the novelty of this work relies on three main contributions:

1. To address the development of a soft-sensing approaches for a special type of batch processes that is rarely explored in the area of soft-sensing: those batch processes that show a characterized variability in their initial settings or conditions, as a result of a specific purpose decision (i.e.: different from a small uncertain noise or an uncontrolled variation). These cases are usual in such batch processes aiming to manage raw materials whose specifications or properties differ from one batch to another (e.g. waste treatment systems), or when different product qualities/quantities are to be generated. Hence, the objective is to develop a soft-sensor able to estimate the QIVs along the batch run under any set of initial conditions in the expected operating range.

2. To explore the advantages of the OK metamodeling technique –as a kind of GP metamodels- for soft-sensing in the chemical engineering area: the kriging method is compared to the most common data-based modeling techniques used for soft-sensing as SVR and ANN, in order to assess and compare its capabilities.

3. To provide efficient soft-sensors able to track an advanced oxidation process (AOPs) based on the photo-Fenton reaction, which in the same time can be used as data-based process models to help understanding the complex behavior usually associated to these processes. Nowadays, AOPs are receiving a huge interest, because of their capacity to manage Contaminants of Emerging Concern present in the water, which cannot be degraded using conventional water treatments technologies (biological, physical or chemical). However, due to the complexity and high nonlinearity of these processes, the best way to address their analytical or phenomenological modeling is still under debate in the scientific and research community; while many data-based modelling studies of these processes have been accomplished from the point of view of experimental design in laboratory scale, their monitoring and control have been never addressed from a soft-sensing perspective, i.e. at industrial or pilot plant scale.

The Chapter is organized as follows:

- Section 9.2 presents the theoretical basis of the modeling techniques utilized and the proposed training and validation procedures.
- Section 9.3 illustrates the first proposed modelling approach for building soft-sensors for the online prediction of the QIV “current” values (Section 9.3.1), and their application details (Section 9.3.2) to three case studies, including 1) a simulated simple first-order batch reactor, 2) a simulated fed-batch fermenter and 3) a real pilot plant of a batch process for wastewater treatment.
- Section 9.4 shows the second proposed modelling approach for building soft-sensors for the online “step-ahead” prediction of the QIV “future” values

(Section 9.4.1), and their application to two case studies, including 1) a simulated batch reactor and 2) a real pilot plant of a batch process for wastewater treatment.

- Section 9.5, finally, summarizes the main conclusions.

9.2 DATA-DRIVEN MODELLING TECHNIQUES

Data-based soft-sensors rely on building machine learning or data-driven models to define accurate black-box relations between the QIV and the online variables. This part spots the light on the most common nonlinear data-driven modeling techniques that have been used in the soft-sensing area as OK, ANN and SVR. More details about their mathematical basis, available software and details of implantations can be found in Chapter 2 (Tools and Techniques).

9.2.1 Ordinary kriging

Given a set of *n* input-output training data $[x_i, y_i]$, $i = 1,2,..n$, $x \in R^k, y \in R$, the OK assumes the predictor $\hat{y}(x) = \mu_{ok} + Z(x)$, where the constant term μ_{ok} represents the main trend of the system to be approximated, and $Z(x)$ is a deviation from that trend. The deviation $Z(x)$ is modeled as a stochastic Gaussian process with expected value $E(Z(x)) = 0$, and a covariance between two residuals $cov(Z(x_i), Z(x_j))$ that only depends on their corresponding inputs x_i, x_j. Thus it can be calculated as: $cov\left(Z(x_i), Z(x_j)\right) = \sigma_{ok}^2 R(x_i, x_j)$, being σ_{ok}^2 the process variance and $R\left(x_i, x_j\right) = exp\left(-\sum_{l=1}^{k} \xi_l \left|x_{i,l-} x_{j,l-}\right|^{p_l}\right) + \delta_{ij}\,\lambda$ a correlation function, where, $\xi_l, l = 1, \ldots k$ are the model hyper-parameters, δ_{ij} is the Kronecker delta, p_l are smoothing parameters and λ is a regularization constant that enables the kriging predictor to regress noisy data (Azman & Kocijan, 2007). The kriging predictor and its estimated error are given by Eq.(9.1) and Eq.(9.2), respectively, where (x^*) is a new interpolating point (different from the training data). In Eq. (9.1), $[r]_{n\times 1}$ is the vector of correlations between the point to be predicted x^* and the original training data points and calculated as $R(x_i, x^*)$, $[R]_{n\times n}$ is the correlation matrix between the training inputs, $[Y]_{n\times 1}$ is the vector of the training outputs and $[\mathbf{1}]_{n\times 1}$ is the identity vector.

$$\hat{y}(x^*) = \mu_{ok} + r^T R^{-1}(Y - \mathbf{1}\mu_{ok}) \tag{9.1}$$

$$\hat{s}^2(x^*) = \sigma_{ok}^2(1 + \lambda - r^T R^{-1} r + (1 - \mathbf{1}^T R^{-1} r)^{-1}/(\mathbf{1}^T R^{-1}\mathbf{1})) \tag{9.2}$$

This work considers the OK implementation developed by Forrester, et al., (2008), because of its high efficiency and applicability. Besides, the "*fmincon*" algorithm included in

the Matlab optimization toolbox is used for the maximization (nonlinear optimization) of the concentrated likelihood function, see Section 2.2.1. The work, also, considers another software implementation for the GP model construction: the GP-Regression (GPR) algorithm based on the function "*fitrgp*" included in the Matlab statistics and machine learning toolbox. Here, it is worthy to emphasize that the objective of this work is not to compare different specific implementations of the GP models but to explore the robustness and flexibility of the proposed soft-sensing methodology by handling different data-based modelling techniques and software.

9.2.2 Artificial neural networks

The ANNs are very well-known efficient machine learning models, which are widely used for data-driven modelling of nonlinear systems. In this work, the Matlab ANN toolbox and the function "*feedforwardnet*" have been used to create multilayer feedforward ANNs. In each of the following application cases, the number of layers, number of neurons and the training algorithm were selected based on a trial and error procedure in order to balance the ANN structure simplicity and its prediction accuracy. More details about ANNs can be found in Section 2.2.2.

9.2.3 Support vector regression

Given a set of *n* input-output training data $[x_i, y_i]$, $i = 1,2,..n$, $x \in R^k, y \in R$, SVR (Vapnik, 1995) maps the input data original space into a high-dimensional feature space, often through a basis or kernel function $\Phi(x_i, x_j)$ that may be represented by different styles as linear, polynomial, Gaussian, etc. Then, the modeling problem becomes the determinations of the optimal (flattest) surface $\hat{y}(x) = b + \sum_{i=1}^{n} w_i \Phi(x_i, x_j)$ in this feature space that fits the data, through the minimization of the weights vector norm $|w|^2, w \in R^n$, where $b = \mu_{svr}$ is a base or bias (Forrester & Keane, 2009). The final predictor of the SVR is given by Eq.(9.3), where α_j^+, α_j^- are Lagrange multipliers resulting from the solution of the aforementioned minimization problem.

$$\hat{y}(x^*) = b + \sum_{i=1}^{n} (\alpha_j^+ - \alpha_j^-)\, \Phi(x^*, x_i) \tag{9.3}$$

A seriousdraw back of the SVR is the huge time and effort required to select the kernel function type and the values of its parameters (e.g., the value of the parameter σ_{svr}, in a Gaussian kernel $\Phi(x_i, x_j) = exp(-\|x_i, x_j\|^2 / 2{\sigma_{svr}}^2)$), which are case dependent (Forrester & Keane, 2009). The detailed mathematical description and derivations can be found in

Section 2.2.3. This work uses the SVR algorithm based on the function "*fitrsvm*" included in the Matlab statistics and machine learning toolbox.

9.2.4 Metamodel validation

A common way to validate a metamodel is to use a different input-output dataset, usually known as validation set: $[x^v, y^v]_{n_v}$. Then, the metamodel is used to predict the outputs $\hat{y}_i^v$, and the prediction is compared by the known output y_i^v. The Root Mean Square Error (RMSE), the Normalized Root Mean Square Error (NRMSE), and the Correlation Coefficient (CC) are calculated as accuracy measures of the prediction. The RMSE (Eq.(9.4)) and the NRMSE (Eq.(9.5)) are direct measures of accuracy: they are reporting an average deviation measure of predicted values from the actual values, where the RMSE is an absolute quantity, and the NRMSE is a relative quantity to the variability range ($y_{i,max}^v - y_{i,min}^v$) of the output y. However, the CC (Eq.(9.6)) is an indicator of the matching or the trend between the overall predictions and the real output values.

$$RMSE = \sqrt{\frac{1}{n_v}\sum_{i=1}^{n_v}(y_i^v - \hat{y}_i^v)^2} \tag{9.4}$$

$$NRMSE = 100\frac{RMSE}{(y_{i,max}^v - y_{i,min}^v)} \tag{9.5}$$

$$CC = r_{y^v,\hat{y}^v} = \frac{\sum_{i=1}^{n_v}(\hat{y}_i^v - \bar{\hat{y}}^v)(y_i^v - \bar{y}^v)}{\sqrt{\sum_{i=1}^{n_v}(\hat{y}_i^v - \bar{\hat{y}}^v)^2}\sqrt{\sum_{i=1}^{n_v}(y_i^v - \bar{y}^v)^2}}, \quad \bar{y}^v = \frac{\sum_{i=1}^{n_v} y_i^v}{n_v}, \quad \bar{\hat{y}}^v = \frac{\sum_{i=1}^{n_v} \hat{y}_i^v}{n_v} \tag{9.6}$$

9.3 SOFT-SENSING FOR ONLINE PREDICTION

This section addresses the development and application of novel soft-sensors able to, cheaply and continuously, predict the value of the QIV at the "current" time instance.

9.3.1 Soft-sensor modelling approach

Consider a batch process in which a significant QIV $y(t)$ (e.g.: reaction progress) may be only expensively measured and/or requires offline sampling and analysis over relatively large sampling time periods. This variable coexists along with other variables $x(t)$ that are measured and recorded online in an automatic and continuous way (over very small sampling periods) probably with a more reduced cost. Additionally and more importantly, the process

may run under different combination of initial conditions $[x(t=0)\ y(t=0)]$, which vary within a known range or bounds $[x_{min}^{t=0}{:}\,x_{max}^{t=0}\,, y_{min}^{t=0}{:}\,y_{max}^{t=0})]$.

The objective is to develop a soft-sensor that is able to predict the QIV as a function of the cheaply measured online variables, over the whole batch run and departing from any combination of the initial conditions over their known variation range. This soft-sensor might be further used to monitor the process and to predict the reaction progress at any time along the batch run, saving the cost and the time of the offline experimental sampling and analysis. The proposed design/modeling approach of such soft-sensor is based on the approximation of the current measurements of the offline QIV, $y(t)$, as a function of the current values of the online variables $x(t)$ and the initial values of both of the offline $y\ (t=0)$ and the online $x(t=0)$ variables (Eq.(9.7)). Thus, the initial conditions $[x\ (t=0), y\ (t=0)]$ provide or identify the overall effect on the main path or trajectory of the batch QIV, while the current values of the online variables $x(t)$ provide the instantaneous or temporal effect.

$$y(t) = f[x(t), y(t=0), x(t=0)] \tag{9.7}$$

The methodology steps include:

1. The selection of a proper set of training batches whose ICs $[x(t=0)\ y(t=0)]$ cover or span –as much as possible- the known variation domain of the ICs $[x_{min}^{t=0}{:}\,x_{max}^{t=0}\,, y_{min}^{t=0}{:}\,y_{max}^{t=0})]$. In this way, the training data includes more information about the system different dynamic behaviors corresponding to different ICs.
    a. In the case of data generation from a complex FPM simulation, different design of computer experiments techniques can be efficiently used for the definition of a representative set of the training data. Typical situations are when the complexity of the FPM hinders its practical usage for the online monitoring of the process.
    b. In the case of a real process without an available accurate FPM, the process history data should be exploited as much as possible
2. The online data $x(t)$ of each training batch is smoothed using a moving average technique –a proper time window should be selected- to reduce Gaussian noise of the sensors.

3. The input-output (online-offline) data are collected from each training batch. Thus, the number of the input-output training data extracted from each batch equals to the number of the QIV measurements along the batch run.
4. The data collected from each training batch is unfolded to compose the overall training set, noticing that the overall training dataset does not include any information neither about the batches identification nor about the temporal sequence of the measurements. Then the soft-sensor (Eq.(9.7)) is trained according to the requirements of the selected modelling technique (OK, GPR, SVR or ANN).
5. The trained soft-sensor is then used along a series of validation batches, in order to predict the values of the offline QIV($y(t)$) along the whole run of each batch, using only the initial values [$x_{t=0}$; $y_{t=0}$] and the online measured variables $x(t)$.
    a. The accuracy measures (Eq. (9.4), (9.5) and (9.6)) can be calculated through comparing the estimated QIV values by the soft-sensor to their known real values.

It is worth noting that the data used to train the soft-sensor will be obviously affected by random or Gaussian errors, due to the nature of the real-sensor(s) and the experimental instruments providing these data. Thus, the target accuracy of a soft-sensor cannot be higher than the accuracy offered by the real sensors and instruments providing the training data. In addition, this maximum accuracy will be also affected (reduced) by the accuracy of the sensors (real-sensors) used to provide the online information required by the already fitted soft-sensor to perform the required online estimations. As a result, it can be concluded that the accuracy of a soft-sensor is expected to be within the same order of magnitude of (but lower than) the accuracy offered by the real-sensor(s) and instruments providing the information required to train and also to use the soft-sensor.

9.3.2 Applications

In these sections, the application of the proposed soft-sensing approach to two simulated case-studies based on the modeling techniques previously introduced in Section 9.2 (OK, SVR and ANN), will be illustrated.

9.3.2.1 Batch reactor

This simulation case-study was proposed by Ruppen, et al., (1995) and involves a common batch reactor model (Eqs.(9.8)) running the reactions $2A \overset{k1}{\rightarrow} B \overset{k2}{\rightarrow} C$, being A the

reactant, B the undesired product and C the desired product. It is assumed that the process depends only on the initial conditions (no perturbations/external inputs are considered along the batch run).

$$\frac{dc_A}{dt} = -2k_1\, c_A^2\ , \qquad \frac{dc_B}{dt} = k_1 c_A^2 - k_2 c_B c_C\,, \qquad \frac{dc_C}{dt} = k_2 c_B c_C, \\ k_i = k_{i,0} exp\left(\frac{E_i}{RT}\right) \tag{9.8}$$

The concentration C_C of product C is considered as the offline QIV, while the concentrations C_A and C_B of products A and B are treated as the online variables. Thus, during each batch run (1 hr), C_C is calculated at 7 specific sampling times (every ten minutes), simulating its expensive offline sampling and analysis. On the contrary, C_A, C_B are calculated every second, simulating the automatic recording of data by the physical sensors. Small white or Gaussian noise $\mathcal{N} \approx (\mu = 0, \sigma = 0.018)$ is added to the simulated online variables values (C_A, C_B), while a higher error $\mathcal{N} \approx (\mu = 0, \sigma = 0.16)$ is also added to the simulated offline variable data (C_C) in order to mimic the experimental error. The white noise and the experimental errors in this case-study and the next ones are estimated in a heuristic manner with respect to the variability domain of each variable. Hence, the experimental error variance σ has been set to 1.5 % of the variability range for the offline variables, while the white noise variance σ for the online variables has been set to 0.5 % of their respective variability ranges.

In order to generate the training data, 24 batches are simulated in the previously described way, using different initial concentrations values $[C_A(t=0), C_B(t=0), C_C(t=0)]$ selected by a Hammersley sampling procedure within the limits $\left[C_{A,min}^{t=0}: C_{A,max}^{t=0}, C_{B,min}^{t=0}: C_{B,max}^{t=0}, C_{C,min}^{t=0}: C_{C,max}^{t=0},\right]$ = [1 : 4, 0 : 2, 0 : 3] (Mol/L). Additionally, another set of 100 batches with different initial concentrations (but within the same domain) are simulated and used as the validation set, see Figure 9.1. The whole online/offline data set generated for 4 specific training batches is illustrated in Figure 9.2. The initial conditions of these 4 training batches are highlighted in Figure 9.1 with their corresponding colors.

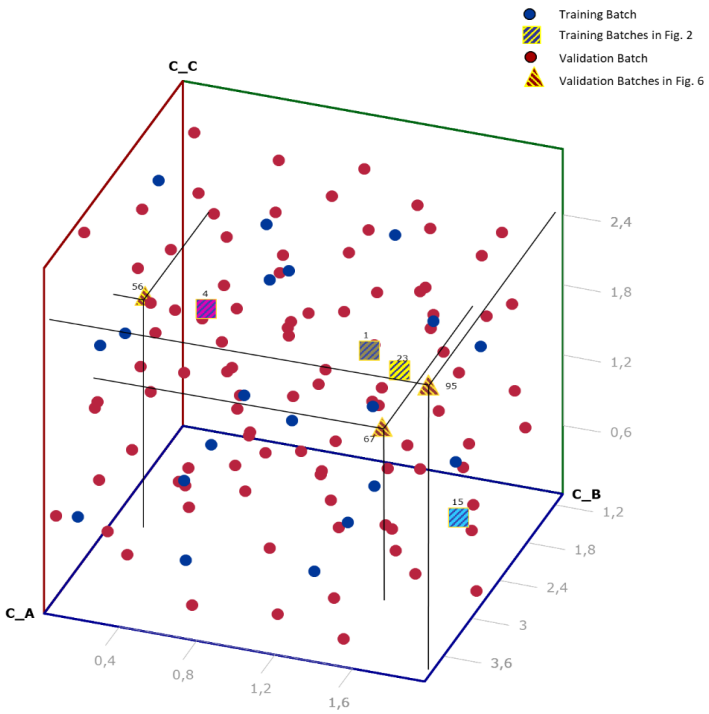


Figure 9.1. Initial conditions of the training and validation batches.

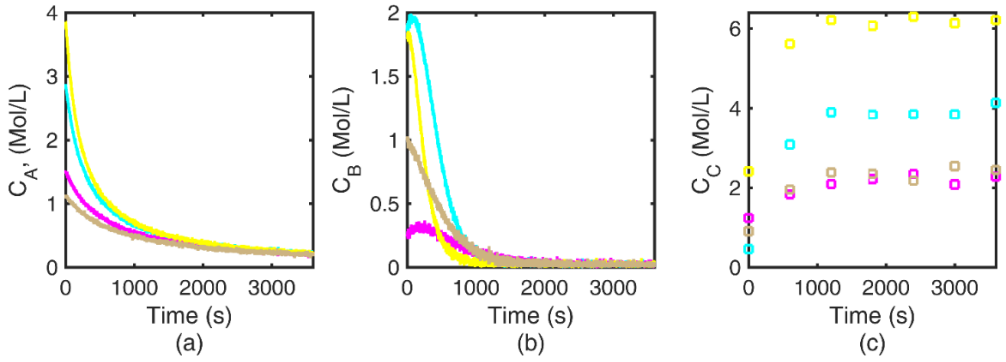


Figure 9.2. Subset of 4 training batches: (a, b) measured noisy online data, and (c) offline data.

The Hammersley sequence is a sampling technique for computer experiments which is used in this example to find a suitable set of the initial values combinations that uniformly cover the modeling space, in order to collect -as much as possible- information about the process dynamics when it is initiated departing from different initial settings. However, in the real situations (e.g. the pilot plant illustrating, section 4), this welfare of selecting the training data might not be feasible. So, it will be required to take the maximum advantage of the available historical data recorded and measured from the batch process (previous runs).

A moving average technique with a time window of 50 seconds has been used to smooth the online data (C_A, C_B) through all the batches (training and validation) in order to diminish

the effect of the artificial (simulated) white noise introduced in the measures, see the zoom-out view in Figure 9.3-(c). After that, 7 input-output (online-offline) training points have been collected from each one of the 24 training batches. Figure 9.3-(a, b) shows the collected input-output (online-offline) data from two training batches: the solid blue and red lines are the measured online data (C_A, C_B) and the dashed black lines correspond to their smoothed values, while the green squares represent the values of the measured offline variable C_C. Hence, the circles, diamonds and squares are the collected input-output data from each batch.

The number of the training batches is selected based on a try and cut procedure, in order to roughly find the minimum number of batches that is able to provide a high prediction accuracy of the soft-sensor. During this try and cut procedure, two main principles are considered: first, the number of the training batches should emulate real situations, where often few information about such kind of processes is available; second, the number of the training batches should be related to the initial conditions variation domain, and also to the complexity of the process behavior. On the contrary, a much higher number of validation batches is considered in this case-study since their generation is feasible (simulation runs) and this allows to get a precise evaluation of the developed soft-sensors, through assessing their prediction accuracy at each sub-region of the initial condition variation domain.

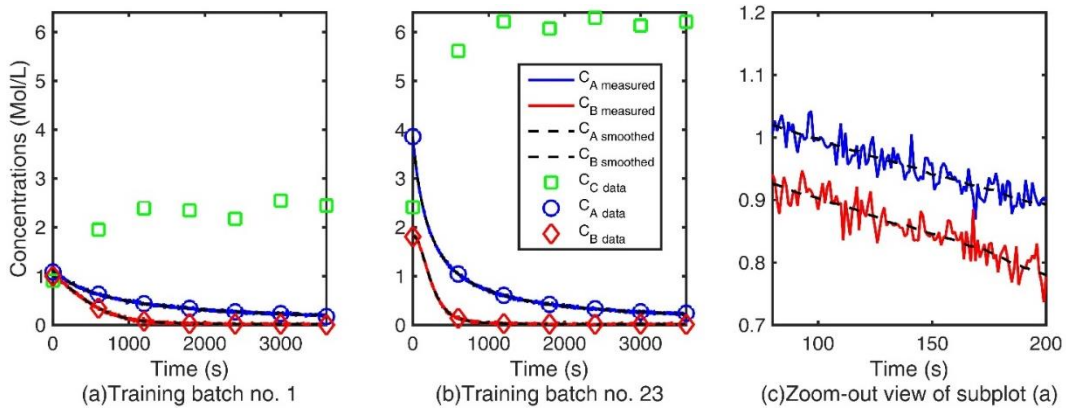


Figure 9.3. (a, b) online data smoothing and input-output data collections from training batches no. 1 and no. 23, respectively, (c) Zoom-out view of batch 1.

The collected 168 (7 ×24) input-output data are used to train the soft-sensor in Eq.(9.9) based on the four metamodel implementations considered (OK, GPR, SVR and ANN). It should be emphasized that the training data does not include any information neither about the batches order/identifications nor the temporal sequence of each specific sample. The training task is accomplished according to the requirements of each metamodel type as previously described in Section9.2. As previously mentioned in Section 9.2, a traditional try and cut procedure has been used to select the best ANN configuration. So, different ANN adjustments have been trained with the 24 training batches and validated with the 100 validation batches,

and the adjustment that achieved the best prediction accuracy for the validation batches is selected. The different configurations were in terms of the number of layers (from one to two layers), the number of neurons in each layer (from 3 to 10 neurons) and the training algorithm (Levenberg-Marquardt back-propagation or Bayesian regularization back-propagation). After several trials, an ANN with one hidden layer of six neurons, trained using the Levenberg-Marquardt back-propagation method (via the Matlab algorithm "*trainlm*"), is found to exhibit the best tradeoff between the ANN structure simplicity and prediction accuracy. On another side, a linear kernel function $\Phi\left(x_i, x_j\right) = x'x$ has been shown to be the best kernel type for the SVR-based soft-sensor.

$$C_C(t) = f[C_A(t), C_B(t), C_A(t=0), C_B(t=0), C_C(t=0)\,] \tag{9.9}$$

After the training of the soft-sensors, they are used to predict the C_C behavior of the 100 validation batches, using only the noisy initial conditions and the values of the smoothed online variables of each batch. It is also worth to emphasize that, although the soft-sensors are trained using noisy measurements of the offline variable C_C (Figure 9.3-(a, b)), their performance assessment is achieved via comparing their predictions with the ideal exact/theoretical behavior of C_C.

The quality of the soft-sensors predictions is evaluated numerically through the calculation of several accuracy measures including the RMSE (Eq.(9.4)), the NRMSE (Eq.(9.5)) and the CC (Eq.(9.6)). It's worthy to mention that usually the aforesaid accuracy measures are calculated by comparing the predictions with the available noisy measurements. But then, a conceptual dilemma arises, as these data involve random deviations accumulated from human operation and analysis apparatus errors. Hence, the metamodel prediction is intended to match the experimental data but also to compensate these errors so avoiding overfitting. In real situations, the aforementioned manner is the only possible way to calculate an accuracy measure, since the real behavior of the system is unknown. But since we are dealing with a simulation case, we may take advantage of our knowledge about the process theoretical/exact behavior. So, the corresponding accuracy measures are calculated by comparing the metamodels prediction with the process theoretical/ideal behavior, as well with the C_C data emulating the noisy measurements.

The calculation of the accuracy measures with respect to theoretical behavior is much more credible to express the soft-sensor accuracy. However, calculating them with respect to the noisy measurement allows extracting some conclusions that can help to evaluate the soft-sensor performance when applied to a real case-study, where only noisy measurements are

available. Table 9.1 shows the accuracy measures of the soft-sensors predictions including the average the RMSE, NRMSE and the CC. In the case when calculating these measures with respect to the data emulating noisy measurements, the number of evaluation data n_v in Eq. (9.4), (9.5) and (9.6) is 700 (100 batches × 7 measurements). However, when calculating them relative to the system theoretical behavior - that is known from the simulation with one second time step-, so n_v is 360000 data points (100 batches × 3600 measurements).

The results in Table 9.1 illustrate how the soft-sensor based on any of the metamodels is able to predict the C_C concentration with high accuracy, as it was able to achieve in the worst case (the soft-sensor based on ANN) a NRMSE of 3.89 % of the total variation range of the C_C [0 : 6.3], and a correlation of 0.983. More importantly, the table reflects that the accuracy measures calculated with respect to the system theoretical behavior are better than those calculated with respect to the noisy experimental data. This indicates that the developed soft-sensors are able to identify the process underlying behavior, although they have been trained using noisy measurements, and these soft-sensors are not over-fitting the training data. It is worth to refer that the soft-sensors based on the SVR and the OK have achieved the best prediction accuracy (average NRMSE of 2.56 % and 2.39 % respectively).

Table 9.1. Average RMSE, NRMSE and CC of the batch reactor soft-sensors.

	W.R.T. the noisy measurements			***W.R.T. the known exact behavior***		
	RMSE	**NRMSE (%)**	**CC**	**RMSE**	**NRMSE (%)**	**CC**
OK	0.21	3.34	0.988	0.16	2.56	0.992
GPR	0.27	4.39	0.980	0.23	3.80	0.984
SVR	0.20	3.22	0.989	0.15	2.39	0.994
ANN	0.28	4.52	0.978	0.24	3.89	0.983

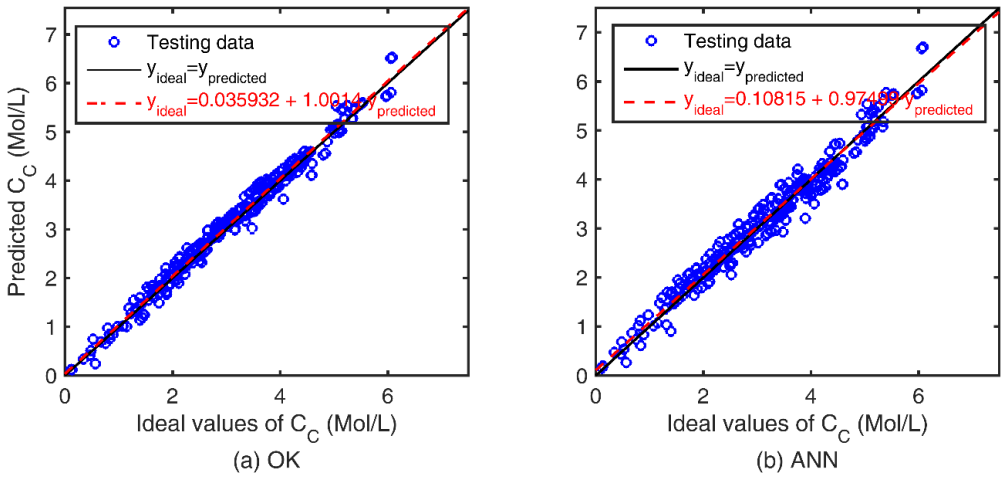


Figure 9.4. Exact versus predicted values of Cc of a random subset of the 100 validation batches using the soft-sensors based on (a) OK and (b) ANN metamodels.

Figure 9.4 shows a randomly selected validation subset (300 data points) of the estimated offline variable (C_C) using the OK and the ANN based soft-sensors (best and worst cases), compared to their corresponding theoretical/ideal behavior. The displayed qualitative assessment demonstrates the high quality of the soft-sensors predictions: the prediction linear fit (red dashed line calculated considering the whole validation set - 360000 data points) is very close to the ideal fit ($y_{ideal} = y_{predicted}$; solid black line).

Another qualitative evaluation of the soft-sensors performance is presented in Figure 9.5, which shows the predictions normalized error distribution for the best (OK) and worst (ANN) soft-sensors; it is clear that the soft-sensors based on the four metamodels behave normally without any biased behavior/ predictions, since the majority (mean) of the predictions exhibit very small error values along with some outliers.

The values in Table 9.1 represent average or gross measures of the soft-sensors accuracy, so it is necessary to give more deep or detailed sight to the performance of the soft-sensors. Figure 9.6 shows the estimations of the offline variable (C_C) for the batches with the highest (Figure 9.6-(a)), average (Figure 9.6-(b)) and lowest (Figure 9.6-(c)) prediction accuracy, using the OK (dashed red line) and the ANN (dashed mauve line) soft-sensors. The three batches are selected through the calculation of the NRMSE for each batch independently, using average values with respect to the four soft-sensor types.

Figure 9.6 demonstrates that the four soft-sensors are able to continuously predict the offline variable C_C with high accuracy under different initial settings within the known bounds. It also illustrates that soft-sensors predictions are able to capture the ideal/theoretical process behavior (continuous black line), without overfitting the noisy measured behavior/data (green squares), in spite of the deviations introduced to the training data in order to emulate the

measurement errors. Much more significant, the figure reflects the ability of the proposed soft-sensor to predict the offline variable along different batches operated under different initial settings within the known bounds: the initial conditions $[C_A(t=0), C_B(t=0), C_C(t=0)]$ of the selected three batches are [3.01, 1.52, 1.52], [2.68, 0.22, 2.07] and [3.85, 1.95, 2.46] for batch no. 67 (Figure 9.6-(a)), batch no. 56 (Figure 9.6-(b)) and batch no. 95 (Figure 9.6-(c)), respectively.

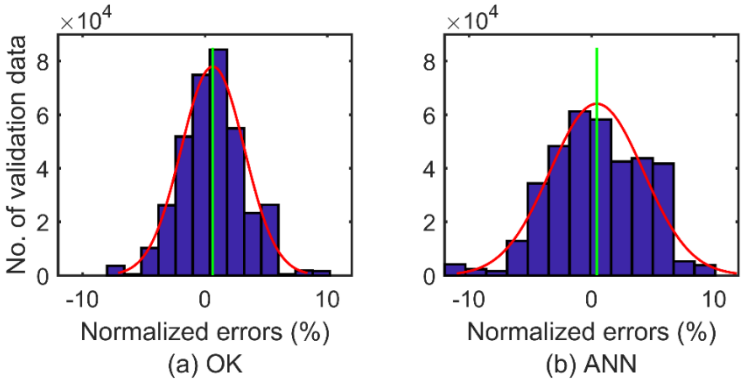


Figure 9.5. Normalized errors distributions of the OK and ANN based soft-sensors predictions.

Correlating Figure 9.1 and Figure 9.6 highlights the effect of the location of the batches initial conditions on the soft-sensor accuracy. For example, the initial conditions of the most accurate predicted batch (no. 67, Figure 9.1) lay within the middle of the training batches initial conditions, so soft-sensors have sufficient knowledge about the behavior of the process dynamics associated to this local area of the initial conditions. On the contrary, the initial conditions of the batch with least accurate predictions (no. 95, Figure 9.6) fall far from the main bulk of initial conditions of the training batches (i.e. on the limits of the initial conditions domain). Hence, the soft-sensor includes less information about the process behavior when it departs from this local area of the initial conditions. These conclusions and remarks match with the OK and GPR metamodels main assumptions that distinguish them from any other metamodel type. This assumption considers that the prediction uncertainty/error increases as the point to be predicted moves far from the training data, and vice versa.

Additionally, the OK (and GPR) estimated errors can be exploited to construct a confidence area that can be used as an uncertainty measure about the predictions (Figure 9.6, grey lines). So, in the cases where the OK predictions behave a significant deviation from the theoretical behavior, these deviations are often accommodated within an estimated confidence area $(-/+ 5\,\sigma)$. The reliability of this estimation (which would correspond to a confidence interval of 99.999% in the case of a normal distribution) depends on the quality of the approach and the application, so other elections are also feasible according to the case characteristics

and modeler preferences (consequences of a wrong estimation, risk aversion, knowledge about the system, etc.). In very few batches, the theoretical behavior may fall outside this uncertainty area (Figure 9.6-(c)).

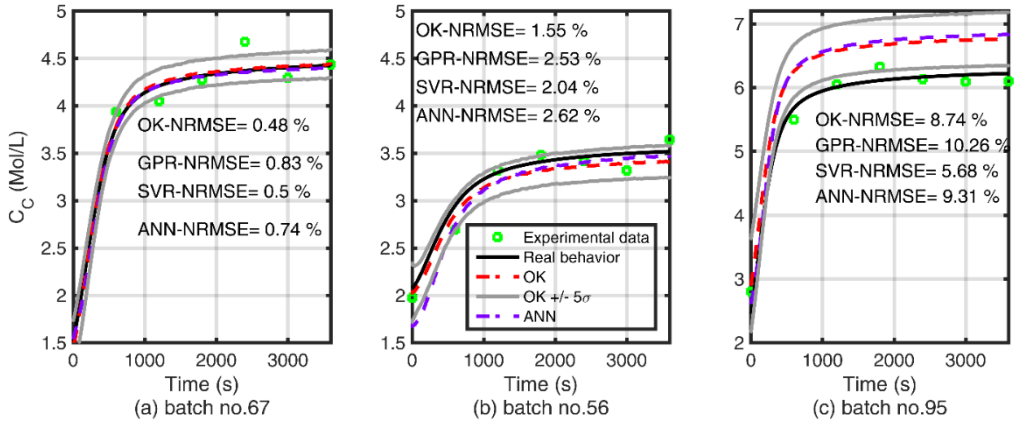


Figure 9.6. Predictions of Cc for three validation batches.

Finally, it is worth to note that, in this case, both of the online variables and the QIV are concentrations closing the mass balance, so one straightforward procedure to estimate the missing information ($C_C(t)$) in this case would be to exploit this knowledge about the system (analytical model-based soft-sensor). Since the proposed soft-sensors have been trained in a behavior which accomplishes the mass conservation principle, the results would be finally equivalent, but it should be emphasized that the proposed soft-sensors are not based on the exploitation of any "a priori" knowledge, but on the approximation of the latent relationships between the offline variable and other online variables regardless of the nature of this knowledge and the involved variables (i.e. concentrations, temperatures, pressures, etc.).

Hence, a modified version of this case-study is addressed, which is exactly the same as the original one but assuming that no information is available about the online concentration $C_B(t)$. Thus, the soft-sensors should infer the offline values $C_C(t)$ only from the online variable $C_A(t)$ and the initial conditions of all the variables, i.e. $C_C(t) = f[C_A(t), C_A(t=0), C_B(t=0), C_C(t=0)]$. In this way, no mass balance can be closed to predict/validate the required offline concentration $C_C(t)$.

Table 9.2 displays the validation results of the soft-sensors resulting from this modified version of the case-study. Although the comparison with Table 9.1 reveals a slight reduction in the soft-sensors accuracy (reduced information about the process behavior), the results still indicate a very good accuracy even when the concentration $C_B(t)$ is not available. The same subset of the off-line variable predictions previously represented in Figure 9.4 are now displayed in Figure 9.7 for the new soft-sensors, confirming the same previously drawn

conclusions (again, the linear fit equation has been calculated considering the entire validation set).

Table 9.2. Average RMSE, NRMSE and CC of the batch reactor soft-sensors (modified case).

	W.R.T. the noisy measurements			***W.R.T. the known exact behavior***		
	RMSE	**NRMSE (%)**	**CC**	**RMSE**	**NRMSE (%)**	**CC**
OK	0.33	5.27	0.9700	0.31	5.09	0.9701
GPR	0.33	5.28	0.9699	0.31	5.09	0.9701
SVR	0.45	7.23	0.9458	0.37	7.26	0.9419
ANN	0.34	5.48	0.9690	0.32	5.18	0.9706

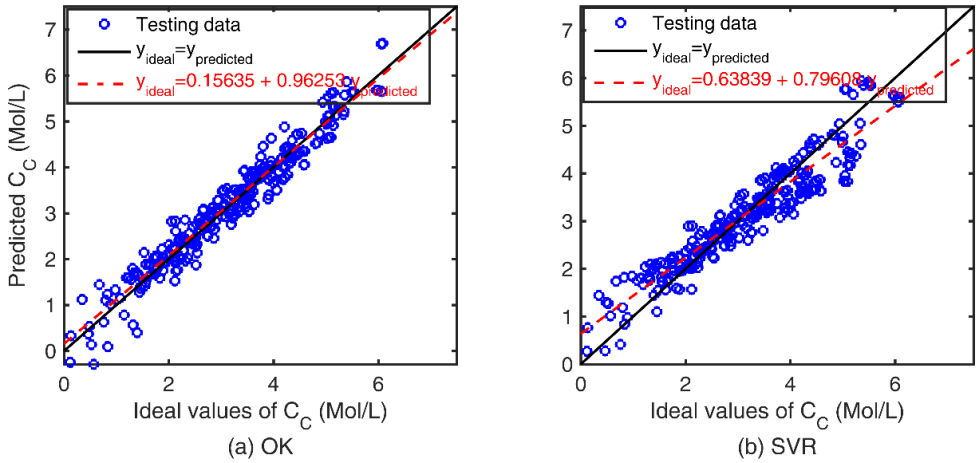


Figure 9.7. Exact versus predicted values of Cc of a random subset of the 100 validation batches using the soft-sensors based on (a) OK and (b) SVR metamodels (modified case).

The application of the proposed soft-sensing methodology to the modified version of the case-study also highlights the robustness and tuning flexibility of the OK compared to the other considered techniques. The comparison between Table 9.1 and Table 9.2 reveals that the OK has been able to maintain its high prediction accuracy level, while this is not the case of the SVR soft-sensors. This is due to the challenging task of selecting the best SVR kernel through a try and error procedure each application time, which does not guaranty the optimal selection, while in the case of using OK –as well as the GPR- parameters are automatically optimized, which ensures an optimal fitting through repetitive applications considering different soft-sensor structures and also different training data. As it will be seen in the next case-study, a similar problem affects the ANN based soft-sensors, although in this first example the best ANN structure found has allowed to maintain a performance similar to the one shown by the soft-sensors based on OK and GPR.

9.3.2.2 Fed-batch fermenter for Penicillin production

A fed-batch fermentation process for the production of Penicillin is used as a second simulation case-study. The process mechanistic model had been initially developed by Bajpai and Reul, (1980), and since this time, it has become a very popular benchmark case-study for the dynamic optimization and control studies, due to its high nonlinearity (Cuthrell & Biegler, 1989; Dadebo & Mcauley, 1995; Banga, et al., 2005; Wang, et al., 2017). The analytical model in Eqs.(9.10) describes the relations among the process variables, including the concentrations (g/L) of the biomass B, penicillin P and substrate S inside the reactor, the reactants volume V (L) and the substrate inlet flowrate F (g/h). This case represents relative further challenges to the proposed soft-sensing method in terms of the higher dimensionality, higher nonlinearity (see the process model in Eqs.(9.10)) and, more important, the process kind (fed-batch), since the existence of the external input (F) generates a more complicated dynamic behavior of the system.

$$\begin{aligned}
\frac{dB}{dt} &= h_1 B - F\left(\frac{B}{500\,V}\right) \\
\frac{dP}{dt} &= h_2 B - 0.01P - F\left(\frac{P}{500V}\right) \\
\frac{dS}{dt} &= -\frac{h_1 B}{0.47} - h_2\frac{B}{1.2} - B\left(\frac{0.029S}{0.0001+S}\right) + \frac{F}{V}\left(1-\frac{S}{500}\right) \\
\frac{dV}{dt} &= \frac{F}{500}
\end{aligned} \tag{9.10}$$

Where $\quad h_1 = 0.11\left(\frac{S}{0.006B+S}\right), \qquad h_2 = 0.0055\left(\frac{S}{0.0001+S(1+10S)}\right)$

During each batch run of 150 minutes, the product (penicillin) concentration P is treated as the offline variable that is measured via expensive offline sampling and analysis (one sample every 15 minutes). Meanwhile, the biomass concentration B, substrate concentration S, the reactants volume V, and the substrate inlet flowrate F are assumed to be the online variables, measured and registered by the sensors every second. Also, white noise distributed according $\mathcal{N}(\mu=0,\sigma=0.2)$, $\mathcal{N}(\mu=0,\sigma=0.15)$, $\mathcal{N}(\mu=0,\sigma=0.05)$ and $\mathcal{N}(\mu=0,\sigma=0.05)$ are added to the simulated measurements of the online variables B, S, V, and F, respectively. Besides, a higher experimental error of $\mathcal{N}(\mu=0,\sigma=0.2)$ is added to the simulated measurement of the offline variable P. The noise amounts are calculated as mentioned in the previous case. The nominal initial conditions often used in the literature for this case are $B(t=0), P(t=0), S(t=0), B(t=0) = [1.5,\ 0,\ 0,\ 7]$ but, for the purpose of this study, the initial conditions are allowed to vary within the limits

$[B_{min}^{t=0}: B_{max}^{t=0},\ P_{min}^{t=0}: P_{max}^{t=0},\ S_{min}^{t=0}: S_{max}^{t=0},\ V_{min}^{t=0}: V_{max}^{t=0}] =$ [0.5 10; 0 3; 0 10; 5.0 8.5] to simulate a situation in which several settings of the batch are feasible, assuming the unavailability of a first principle model.

A set of 24 noisy batches are simulated for training of the soft-sensors, while other 100 batches are also simulated for validation purposes. As in the first studied case (Section 9.3.2.1), the noisy data (a subset of them are shown in Figure 9.9) are used to train the soft-sensor, which is then used to predict the Penicillin concentration *P* of the validation batches. Then, the predicted *P* values are compared to their corresponding ideal ones.

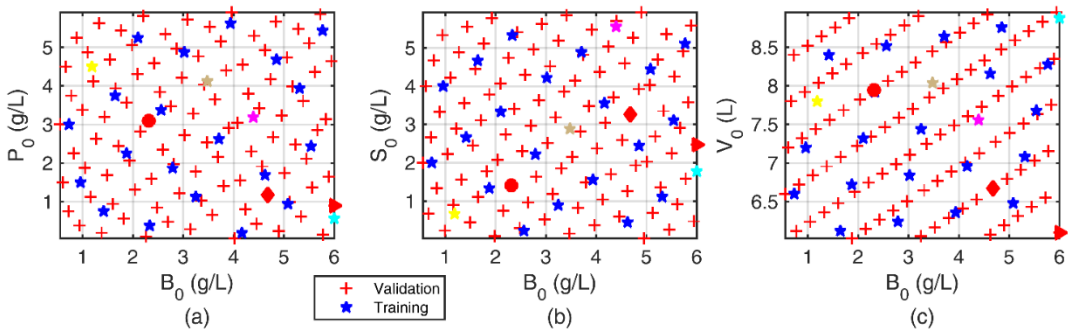


Figure 9.8. Initial conditions of the training (stars) and validation (crosses) batches of the penicillin production case-study.

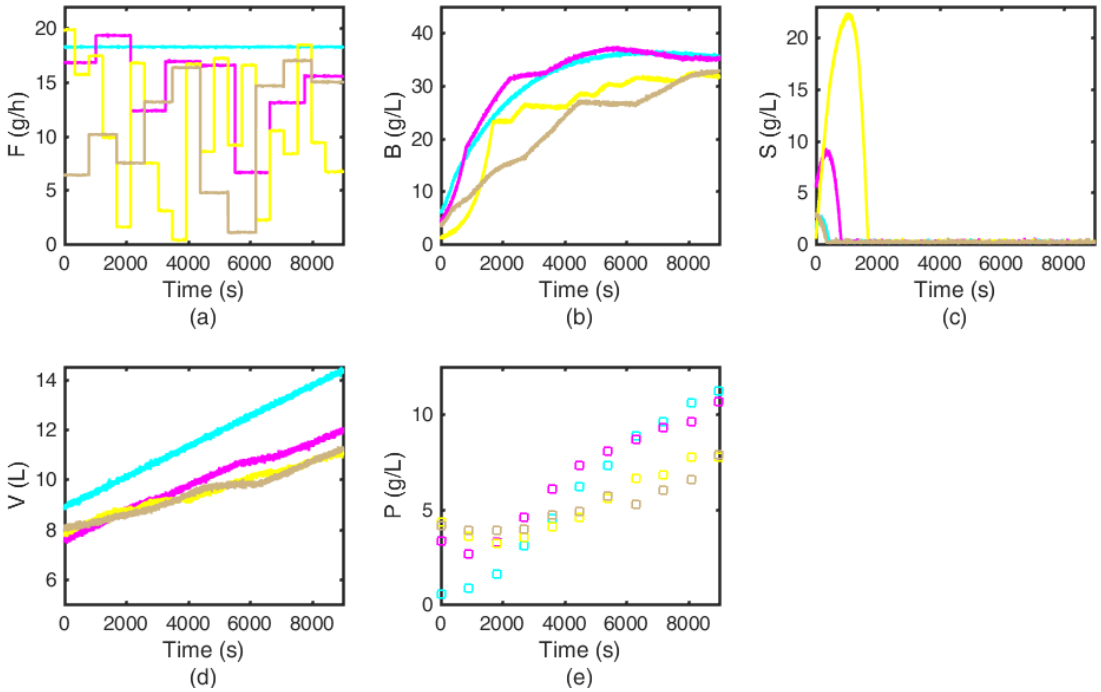


Figure 9.9. Case 2: Subset of 4 training batches: (a, c, d, e) measured noisy online data, and (b) offline data.

The inlet flowrate profiles (F) associated to the training and validation batches are also synthesized in a random manner in order to ensure that the training will include as much information as possible about the dynamic behavior of the process. Thus, each flowrate profile is characterized by random step changes (piecewise constant) along the batch time. A subset of the training batches are represented in Figure 9.9, where the effect of the variations in the initial conditions and the inlet flowrate profile F can be observed in terms of significant changes in the process behavior from one batch to another. The initial conditions of this subset of the training batches are also highlighted in Figure 9.8 with the corresponding colors. Figure 9.10 illustrates the inlet flowrate profiles associated to the three specific validation batches previously marked in Figure 9.8 with a red solid circle, diamond and triangle, respectively.

A moving average technique with a time window of 120 seconds is used to manage the (artificial) white noise associated to the inputs (simulated physical sensors measurements), see Figure 9.11 Then, ten input-output training points have been collected from each one of the 24 training batches (Figure 9.11). The soft-sensors based on the four metamodel implementations considered (OK, GPR, SVR and ANN) are trained using the collected 264 (24×11) input-output training data. Again, no knowledge about the batch identifications or the temporal sequence of the measurements is introduced during the training. The same SVR kernel type and ANN configurations – but with a Bayesian regularization back-propagation algorithm, via the Matlab algorithm "*trainbr*"- used in the previous example have been found to be the most suitable customizations in this case too.

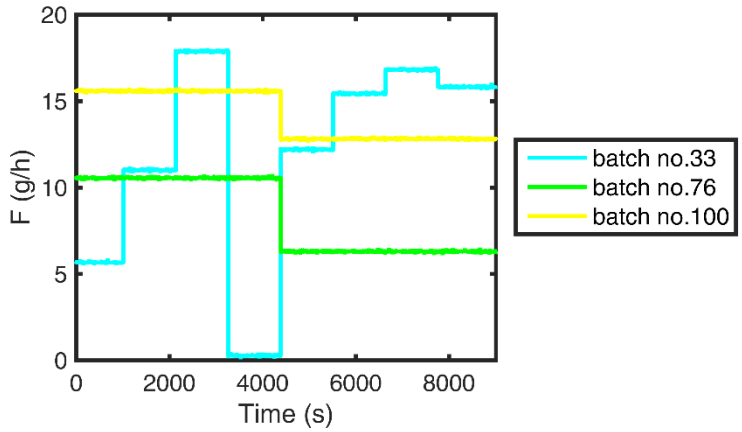


Figure 9.10. Case 2: Profiles of the inlet flowrate F for the three validation batches.

Table 9.3 displays the numerical/quantitative assessment of the Penicillin soft-sensors prediction accuracy in terms of the RMSE (Eq.(9.4)), the NRMSE (Eq.(9.5)) and the CC (Eq.(9.6)), as in the previous case (n_v=1100 samples (100 batches × 11 measurements) when compared to noisy measurements, and n_v=900000 samples (100 batches × 9000 measurements) when ideal behavior is used as reference).

$$P(t) = f[B(t), S(t), V(t), F(t), B(t=0), S(t=0), V(t=0), F(t=0), P(t=0)\,] \tag{9.11}$$

Again, the results in Table 9.3 reflect the high accuracy of the Penicillin soft-sensors, since they were able to achieve in the worst cases (the soft-sensor based on ANN) a NRMSE of 5.9 % of the total variation range of the P (0 : 13.75), and a CC of 0.94. These accuracy measures are not as good as the ones reported in the previous example (Table 9.1) because of the relatively higher nonlinearity and dimensionality of the Penicillin process/model (Eqs.(9.10)). Besides, this case poses an additional significant challenge to the proposed method compared to the previous example, due to its nature as a fed-batch process with external forcing input (flowrate F).

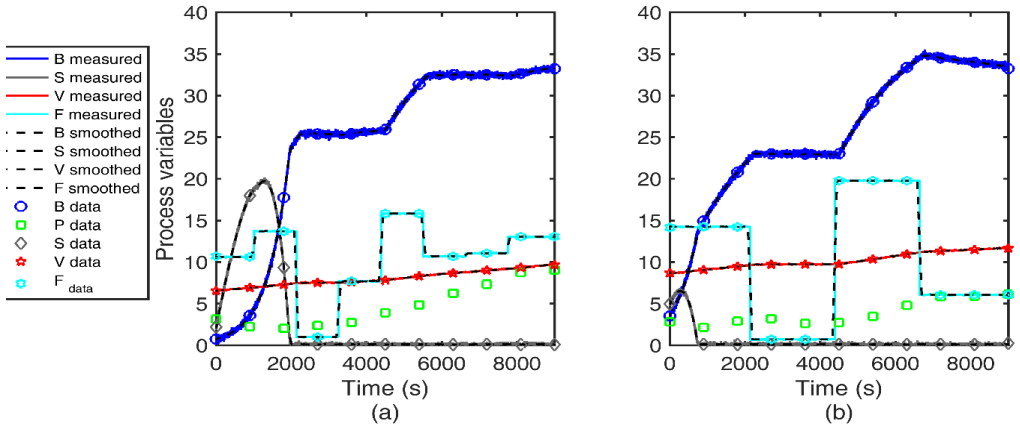


Figure 9.11. Input-output data collection and smoothing for two different training batches (Penicillin case).

The conclusions drawn from the previous example regarding the capabilities of the proposed soft-sensors to identify the process underlying behavior, even though they have been trained using noisy measurements, are also confirmed in this case. Besides, the results indicate the ability of the proposed soft-sensor to handle fed-batch processes. Table 9.3 also highlights the performance of the OK based soft-sensor, which is able to achieve higher prediction accuracy than the SVR and ANN based systems.

Figure 9.12 displays a subset (300 data point) of the estimated Penicillin concentrations compared to their ideal values, using the soft-sensors based on the OK (most accurate soft-sensor) and ANN (least accurate soft-sensor), and represents a qualitative assessment of the soft-sensors accuracy. Also, the prediction linear fit is calculated considering the whole validation set (900000 data points). On another side, Figure 9.13 shows the predictions normalized error distribution of the validation set; again, homogenous normal distributions of

the soft-sensors prediction errors can be clearly identified, which reflects the soundness and unbiasedness of the soft-sensors performances.

Table 9.3. Average RMSE, NRMSE and CC of the Penicillin concentration soft-sensors.

	W.R.T. the noisy measurements			*W.R.T. the known exact behavior*		
	RMSE	NRMSE (%)	CC	RMSE	NRMSE (%)	CC
OK	0.51	3.91	0.9773	0.49	3.69	0.9778
GPR	0.52	4.00	0.9762	0.50	3.76	0.9771
SVR	0.77	5.84	0.9480	0.71	5.33	0.9527
ANN	0.81	6.16	0.9445	0.81	5.99	0.9417

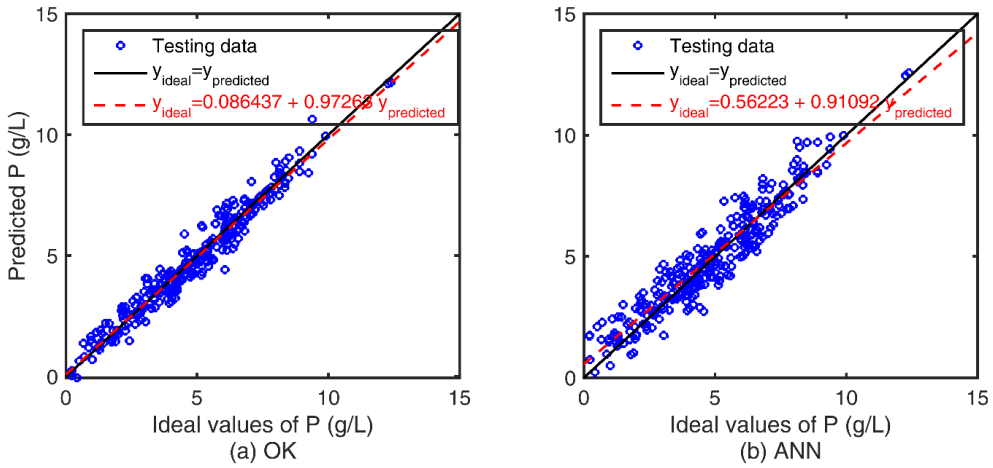


Figure 9.12. Predictions of a subset of the 100 validation batches (Penicillin case).

Figure 9.14 shows the predictions of the Penicillin concentration (P) for three validation batches with the highest (a), average (b) and lowest (c) prediction accuracy, using the OK and the ANN soft-sensors. The three batches are selected in the same way as the previous case in Section 9.3.2.1. The figure also indicates the high accuracy of the soft-sensors and their ability to continuously (each second) capture the underlying behavior of the Penicillin concentration (continuous black line). Additionally, it confirms the high capability and flexibility of the soft-sensors to predict different dynamic behaviors of the Penicillin concentration, associated to the changes in the initial conditions $[B(t=0), P(t=0), S(t=0), V(t=0)]$ (see Figure 9.8 red solid circle, diamond and triangle, respectively), and -more importantly- different forcing input F profiles, see Figure 9.10-(b).

Again, when the initial conditions of a validation batch lie very near to or within the main bulk of the training batches initial conditions (batch no. 33, red circle in Figure 9.8) the results exhibit the highest prediction accuracy, as the soft-sensor is trained with sufficient

information or knowledge about the process dynamics associated to this local area of the initial conditions. In contrast, the initial conditions of batch no. 100 lay far from the main bulk of the training batches initial conditions –on the limits of their domain- (Figure 9.8, red triangle). Therefore, the soft-sensor has less data/information/knowledge about the process behavior when departing from this local area of the initial conditions. These observations match the OK and GPR metamodels main principle: as the predicted point moves far from the training data, the prediction error increases.

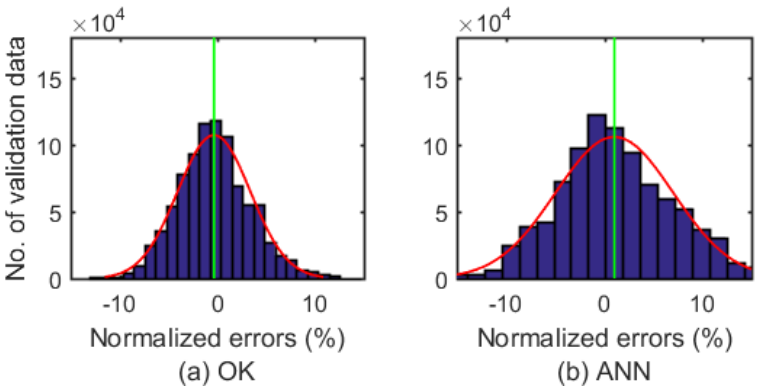


Figure 9.13. Normalized errors distributions of the Penicillin soft-sensors predictions: (a) OK and (b) ANN.

The figure also emphasizes the merit of the OK confidence area (grey lines) that is established around its prediction thanks to its estimated error. Again, even in the cases where the OK predictions –as well the GPR ones- behave a significant deviation from the theoretical or ideal behavior, these deviations often fall within the confidence area $(-/+ 5\,\sigma)$. Similar to the previous case (Section 9.3.2.1), Figure 9.8 and Figure 9.14 also show that the OK uncertainty area– i.e. distance between the two grey lines– of a batch increases as its initial conditions goes far from the main bulk of the training batches initial conditions.

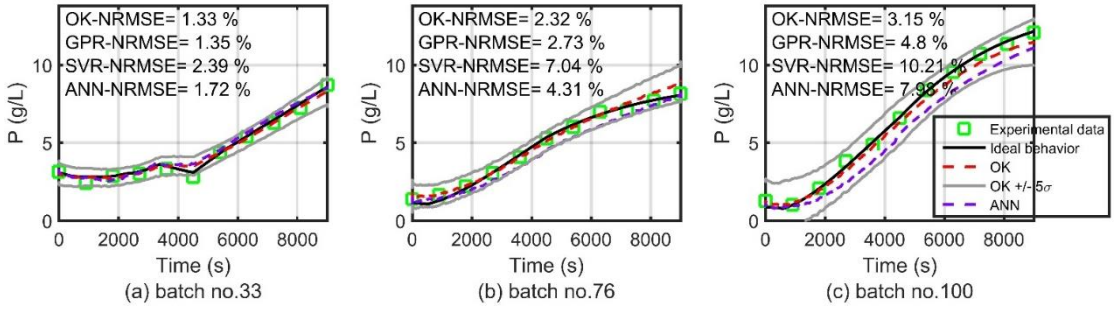


Figure 9.14. Prediction of the Penicillin concentrations for three validation batches: (a) highest (b) average, and (c) lowest prediction accuracies.

Another very important point that has been raised by the nature of this case-study (fed-batch process) is that the soft-sensor (Eq.(9.11)) is able to accurately predict the Penicillin

concentration $P(t)$ just considering the current values of the control variable -substrate inlet flowrate- $F(t)$ although, in fact, the Penicillin concentration $P(t)$ depends also on the dynamic evolution of the inlet flowrate (i.e. $F(t-1), F(t-2), F(t-3)$ etc.). This can be explained by the fact that the soft-sensor approximates the Penicillin concentration $P(t)$ not only as a function of the control variable $F(t)$, but also as a function of other measured online state variables $[S(t), B(t), V(t)]$ whose current values are also dynamically affected by the inlet flowrate. In other words, the values of these online state variables include enough knowledge about the accumulation/history of the inlet flowrate from the beginning of the batch time to build good estimations of the QIV.

In order to better illustrate this aspect, a modified form of this case has been studied, where the online variables $F(t)$ and $S(t)$ have been assumed to be unmeasured and the initial values $F(t=0)$ and $S(t=0)$ are also assumed to be unknown. Consequently, the modified soft-sensors take the form: $P(t) = f[B(t), V(t), B(t=0), V(t=0), P(t=0)]$.Table 9.4 and Figure 9.15 show how the soft-sensors built for the modified case still maintain very high prediction accuracy, even with this dramatic reduction of the available information about the process. Also, the results underline again the consistent and robust performance of the OK approach in front of the oscillating accuracy of the ANN and SVR methods (Table 9.3 vs. Table 9.4).

Table 9.4. Average RMSE, NRMSE and CC of the modified soft-sensors of the Penicillin concentration.

	W.R.T. the noisy measurements			*W.R.T. the known exact behavior*		
	RMSE	NRMSE (%)	CC	RMSE	NRMSE (%)	CC
OK	0.58	4.46	0.9719	0.55	4.19	0.9733
GPR	0.68	5.24	0.9607	0.66	5.03	0.9604
SVR	0.76	5.82	0.9483	0.69	5.23	0.9542
ANN	0.60	4.62	0.9703	0.58	4.40	0.9708

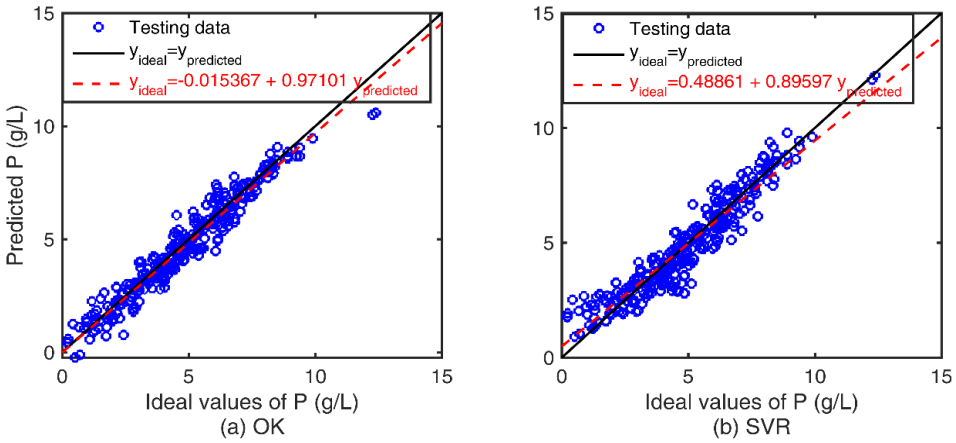


Figure 9.15. Predictions of a subset of the 100 validation batches (Penicillin case, modified soft-sensors).

9.3.2.3 Accuracy assessment

Excluding just one outlier case (SVR soft-sensor in Table 9.2), the accuracies (NRMSE) of the soft-sensors proposed in this section are in the range [2.5% - 6.0%], and the corresponding Correlation Coefficients are in the range [0.94 - 0.99] (Table 9.1, Table 9.2, Table 9.3 and Table 9.4). As previously indicated, these values are to be compared with the accuracy of the information used to train the soft-sensors, which in these cases is known, since the disturbances in this information have been simulated by introducing white noise in the information provided by the mathematical models: the variance of such noise has been fixed to be 1.5% of the variability range of the offline variables, and 0.5% of the variability range of the online variables. Additionally, the NRMSE values have been calculated in front of 2 references: the exact values (provided by the mathematical model), and the disturbed values, again affected by the corresponding white noise (1.5%).

So, the obtained NRMSE values (2.5% - 6.0%) are in accordance with the target accuracy that can be expected from a soft-sensor, as it has been previously discussed in this section: lower but within the same order of magnitude of the accuracy of the offline information used to train these soft-sensors (1.5%), further reduced by the accuracy of the online information available to use them (0.5%).

Besides, the proposed soft-sensors offer better or equivalent accuracies than the ones offered by other soft-sensing approaches recently proposed in the literature for continuous and batch processes: although the available information is presented in different forms and usually the resulting accuracies are not referred to the accuracy of the raw information, common CC results are in the range [0.84 – 0.95] (Grbić, et al., 2013; Jin, et al., 2015), and values of RMSE

of 3.35 (Mota, et al., 2014) and average relative errors of 15% (Duran, et al., 2006) are reported.

9.3.2.4 Application to a photo-Fenton batch process

Previous studies have been devoted to model wastewater treatments based on the photo-Fenton processes. Some of these works have proposed the use of analytical or First Principle Models (FPMs) for the characterization of these processes in particular experimental conditions, or for specific reactor geometries (Farias, et al., 2009; Reina, et al., 2012). However, the complex and nonlinear natures of these processes make such FPMs unable to include all the involved mechanisms in a general exhaustive way.

Alternatively, many studies have been carried out focusing on the data-based modeling of different types of treatments based on photo-Fenton processes, mainly using ANNs. However, they have been oriented to address the kinetic modeling from experimental design perspectives: the ANN is used to model a measure of the treatment performance (the total removal of contaminant, the final H_2O_2 concentration, etc.) at the final time of the treatment as a function of the working conditions (Nascimento, et al., 1994; Duran, et al., 2006; Guimarães, et al., 2007; Jaafarzadeh, et al., 2012; Khataee, et al., 2014; Hassani, et al., 2015; De Tuesta, et al., 2015; Expósito, et al., 2017; Belkacem, et al., 2017). Thus, the obtained ANN model can be only used for the identification and analysis of the impact of the experiment initial settings on this final measure of performance.

A different class of data-based modeling studies has been focused on the prediction of the contaminant degradation dynamics during the treatment time (Göb, et al., 1999; Salari, et al., 2005; Elmolla, et al., 2010; Ayodele, et al., 2012; Mota, et al., 2014; Mustafa, et al., 2014; Gazi, et al., 2017; Sebti, et al., 2017). To do so, the degradation evolution is assumed to be a function of the initial experiment parameters (e.g. initial concentrations of H_2O_2, Fe^{2+}, etc.) and the time. But this approach does not allow monitoring an online process, since any change or disturbance affecting the process cannot be tracked or captured by the model, which only considers the time and not any dynamics nor the rest of the online measurements. Besides, the modeling approaches proposed in the majority of these works have not been systematically validated, neither through the comparison of different modelling techniques nor by using other test case(s). Finally, the data used in most of these works have been collected from the experimentations on a laboratory scale (reactors or a flasks of 500 to 2000 mL), which makes the obtained information/data much smoother/more ideal than in an industrial-size situation (it is easier to control the random error within this scale than within a pilot plant/ industrial scale).

9.3.2.4.1 Materials and methods

In this work, a pilot plant case-study is addressed, consisting of a photochemical pilot plant built to study water treatment processes based on the photo-Fenton reaction, working in a batch mode and considering paracetamol as reference contaminant. Paracetamol (acetaminophen or 4-amidophenol, PCT from this point) is a widely used analgesic, anti-inflammatory and antipyretic, reported as the most popular non-opioid analgesic sold in Spain in recent years (Yamal-Turbay, et al., 2015). Due to this, and also due to the difficulties to eliminate it through conventional wastewater treatment techniques, it has been widely used to investigate the efficiency of non-conventional wastewater treatment processes, like the photo-Fenton technologies. Experiments were carried out using 98 % purity PCT to prepare samples in distilled water. Fenton reagents of H_2O_2 33 % w/v and $Fe_2SO_4 \cdot 7H_2O$ were used (Yamal-Turbay, et al., 2015). The process performance (reaction progress) must be evaluated through an off-line procedure consisting on withdrawing aliquots from the pilot plant reactor and measuring the Total Organic Carbon (TOC) concentration by means of a TOC analyzer, which offers an accuracy of 1% of the measurement range, although the overall analysis procedure includes several manual steps which obviously affect the accuracy of the final obtained information.

9.3.2.4.2 Pilot plant

The photochemical pilot plant (Yamal-Turbay, et al., 2015) (Figure 9.16) consists of an annular photo-reactor equipped with a Philips Actinic BL TL-DK 36W/10 1SL lamp and a pumping system set to keep a constant recirculation. The total volume of the system was 15 L, pumped at 12 $L\ min^{-1}$ to guarantee proper mixing. The processing conditions that were kept constant for all the experiments were: 90 minutes process time, irradiation, 3±0.2 pH, 10 $mg \cdot L^{-1}$ iron(II) salt. During every batch run (each of 90 minutes), samples of the reaction mixture were taken out at regular time intervals (every 15 minutes). Then, the reaction progress is measured by expensive offline analysis of the extracted samples, which resulted in only seven measurements (TOC and H_2O_2 evolution) available for each batch run. Additionally, the SCADA system automatically registers the Temperature (T) and Redox potential (R), which are also expected to be related to the reactions progress. Both variables were measured online and recorded every second at a minimum cost, providing 5400 measurements along each batch run. The accuracy of the respective sensors is of 0.5% of the measuring ranges. It should be noticed that the use of Redox potential as online variable implicitly allows to take into consideration one of the most significant factors in photo-Fenton process, such as the Fenton reagent ratio with respect to the amount of contaminant ($TOC/H_2O_2/Fe^{2+}$).

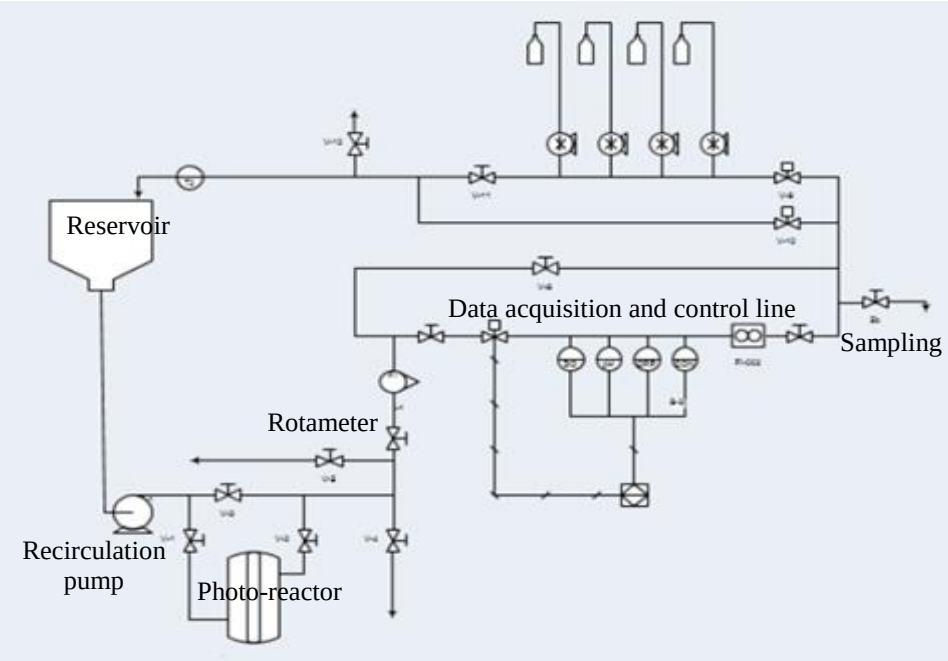


Figure 9.16: Photo-Fenton pilot plant.

Several batch runs have been carried out under different initial concentrations of the contaminant (TOC) and Hydrogen peroxide [H_2O_2], in order to characterize the process behavior and to determine the significant process factors (Figure 9.17). Some of these batches are carried out considering the same initial conditions (highlighted by dotted gray circles), in order to minimize the noise effects resulted from human errors in the training set, and also to assess the sensitivity /robustness of the metamodel in the validation set for confirmation. 7 batches have been used as training batches, while the soft-sensor performance will be confirmed using other 4 validation batches (Figure 9.17); the whole data recorded (online) and measured (offline) for the eleven batches are displayed in Figure 9.18 where each color express a different batch.

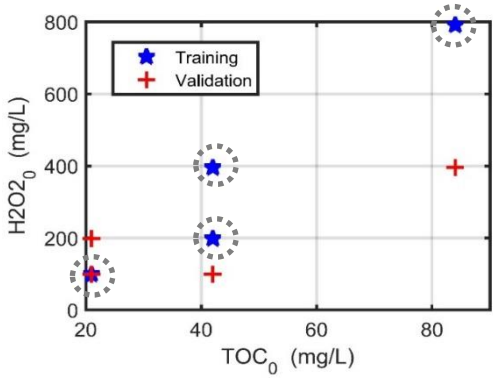


Figure 9.17. Initial concentrations of the H_2O_2 and TOC of the eleven batches.

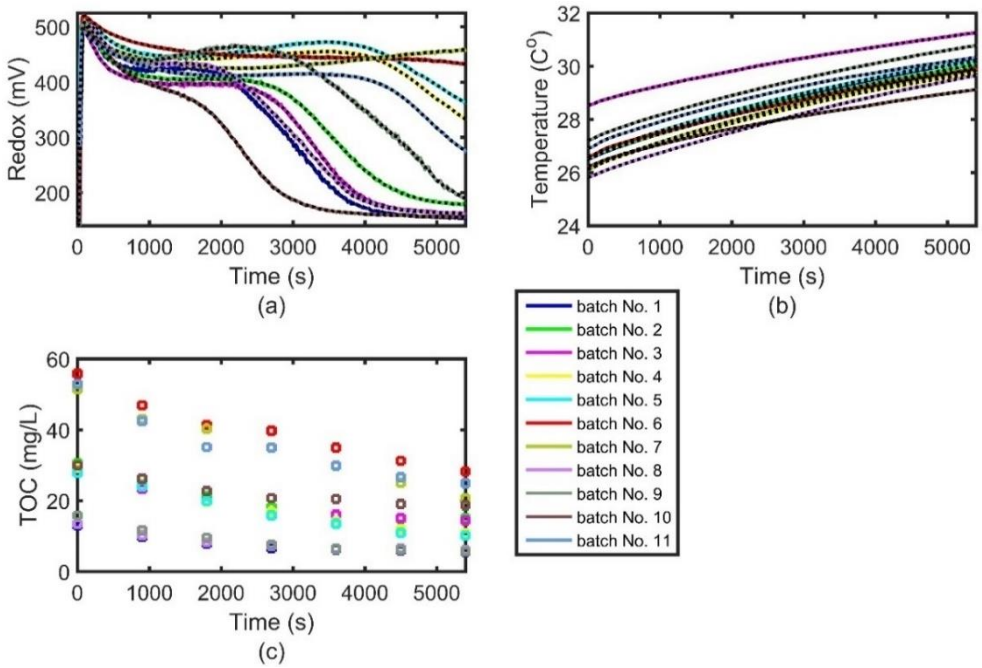


Figure 9.18. Online and offline data of all the batches.

The same procedure that has been presented and validated in section 3 is directly applied here. Thus, the soft-sensor (Eq.(9.12)) is designed to model and predict the expensively measured offline TOC for any initial batch settings of the TOC and H_2O_2 as a function of the initial values of the all variables ($TOC\ (t=0), T(t=0), R(t=0)$) and the timely measured values of the available online variables ($T(t), R(t)$). The online data (T, R) are smoothed using a moving average technique with a time window of 60 seconds. Then, seven input-output training points are collected from each batch of the training group, which results in a training set of 49 input-output data. The soft-sensor (Eq.(9.12)) is then trained based on each of the four metamodels (OK. GPR, SVR, ANN) according to their requirements, see Section 9.2. Using the same try and cut approach previously described, an ANN with one hidden layer of nine neurons, trained using the algorithm "*trainbr*" is selected. Besides, a linear kernel function is selected for the SVR-based soft-sensor.

Each of the constructed soft-sensors is validated by employing it to predict the TOC of the four validation batches. So, with the known initial concentrations [$TOC\ (t=0), T(t=0), R(t=0)$], together with the timely measured values of the online variables, the soft-sensors will enable the TOC monitoring along the whole time of each batch.

$$TOC(t) = f[T(t), R(t), TOC(t=0), T(t=0), R(t=0)\ ] \qquad (9.12)$$

The numerical accuracy of the soft-sensors predictions can be assessed through the results shown in Table 9.5. The NRMSE values are again in the range of the accuracy of the

available information used to train the system and affected by the accuracy of the online information available to make the required predictions. In this case, it must be noted the limited number of data available (49 samples), which probably underestimates the real performance according to the conclusions drawn from the simulation case-studies results (the accuracy measures calculated relative to the noisy measurement often underestimate the real soft-sensor performance –Table 9.1 and Table 9.3).

Table 9.5. Average accuracy measures (RMSE, NRMSE, CC) of the TOC soft-sensors.

	RMSE	***NRMSE (%)***	***CC***
OK	1.19	2.37	0.9956
GPR	1.50	2.98	0.9942
SVR	2.84	5.66	0.9730
ANN	1.90	3.84	0.9901

Figure 9.19 and Figure 9.20 show the qualitative assessment of the TOC soft-sensors: Figure 9.19 displays the TOC estimations of the validation batches set using the fitted soft-sensors compared to their corresponding real measurements, resulting in a well distributed correlation along the ideal diagonal of the graphic; Figure 9.20 shows the distributions of the prediction normalized errors for the soft-sensors based on the OK (a) and the SVR (b), which are quite close to normal type.

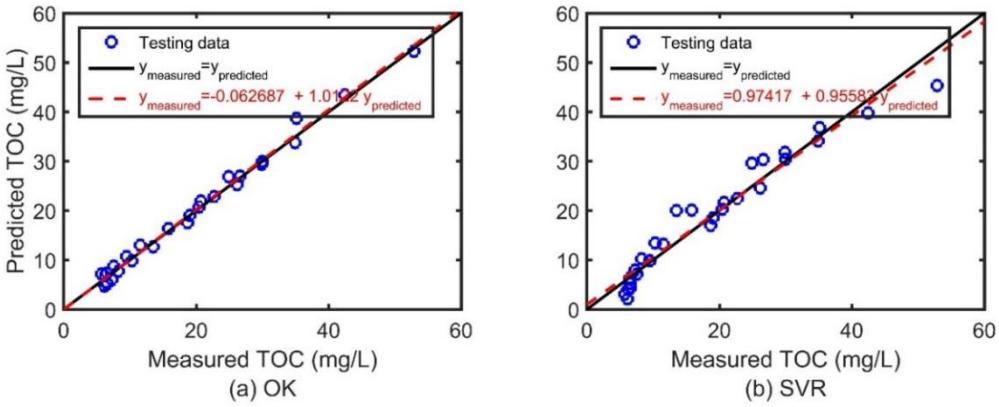


Figure 9.19. Prediction of the validation batches data versus their experimental measurements.

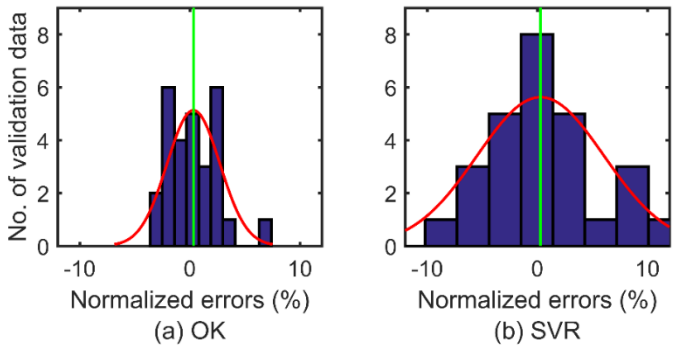


Figure 9.20. Normalized errors distributions of the TOC soft-sensors predictions: (a) OK and (b) ANN.

Both of the qualitative assessments confirm that the four soft-sensors do not reflect any bias, and also the better performance of the OK based approach. But the distributions of the prediction errors of the photo-Fenton soft-sensors (Figure 9.20) do not show the same very high quality compared to the soft-sensors of the simulation case-studies (Figure 9.5 and Figure 9.13). This is explained by three main reasons: first, the very low size of the training and validation datasets in this case (49 data for training and 28 data for validation), while relatively larger datasets are used in the previously presented simulation cases. Second, the uniformity of the training and validation data relative to the initial conditions domain, which in this case was forced by the need to take the maximum advantage of the available expensive data, in front of the use of a well-designed experimentation plan that spans/covers the whole working domain (Hammersley sampling technique - see Figure 9.1 and Figure 9.8 vs. Figure 9.17). Third and finally, the hypothesis that the experimental errors will follow a normal distribution, which may not be true in this case (while this was obviously true in the simulated cases).

Table 9.6 shows the detailed accuracy measures for each of the four validation batches independently. It should be noticed that batch no. 9 is showing the lowest prediction accuracy (maximum RMSE and NRMSE) in any case. This is associated to the fact that the initial conditions of this batch are far from the initial conditions of the training batches set (see Figure 9.17), which makes the trained soft-sensors having relatively less knowledge about the process behavior in this region of the initial operating condition than the other batches (no. 1, 3, 4), whose initial conditions are relatively closer to the initial conditions of the training batches.

Table 9.6. Accuracy measures for each of the four validation batches.

	Batch No. 1		**Batch No. 3**		**Batch No.4**		**Batch No.9**	
	RMSE	***NRMSE***	***RMSE***	***NRMSE***	***RMSE***	***NRMSE***	***RMSE***	***NRMSE***
OK	1.05	2.09	1.10	2.20	0.74	1.48	1.68	3.35
GPR	1.61	3.21	0.82	1.63	1.14	2.27	2.09	4.18
SVR	3.33	6.63	2.22	4.43	1.00	1.99	3.90	7.78
ANN	1.38	2.76	0.73	1.45	2.12	4.23	2.81	5.60

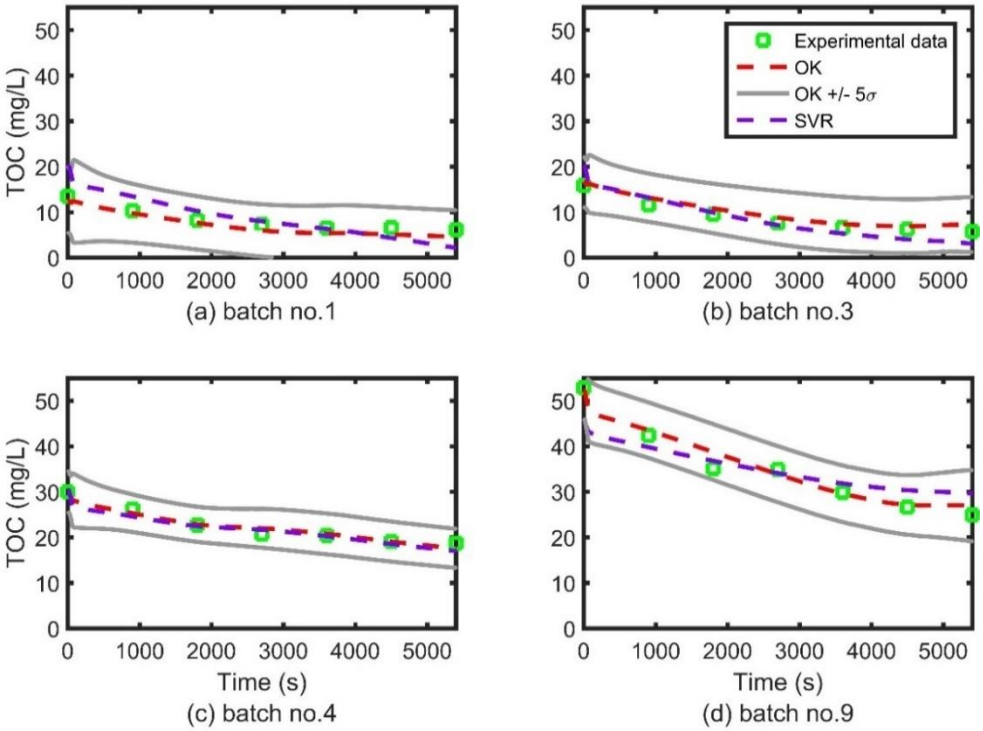


Figure 9.21. TOC prediction of the four validation batches.

Figure 9.21 shows the TOC prediction in the four validation batches using the OK (most accurate) and the SVR (least accurate) soft-sensors, proving the high capabilities of the soft-sensors to continuously (each second) predict the TOC along different batch runs with very high accuracy and flexibility in terms of the initial conditions. The same figure confirms how the OK estimated error can be very useful in practical situations where no knowledge about the exact process behavior (i.e. FPM) is available. Hence, this estimated error is used to establish a confidence interval about the soft-sensor prediction (Figure 9.21, solid gray lines, -/+ 5 sigma). So, even in the cases where the OK –or the GPR- predictions behave a significant deviation from the measured data (green squares), these deviations are still within the confidence area.

9.4 SOFT-SENSING FOR STEP-AHEAD PREDICTION

The soft-sensors developed and applied in Section 9.3 (as well as most of data-driven soft-sensing techniques developed for batch processes in the literature) are able to predict only the "current" value of the offline variable (QIV) as a function of the current and previous values of the online variables (Kadlec, et al., 2009; De-Canete, et al., 2016). Although these soft-sensors are very efficient from the points of view of reducing the experimental cost and enhancing the process monitoring by providing continuous estimates of the QIV values, for the process control point of view, a dynamic soft-sensor able to predict in advance the "future" values of the QIV would be much more rewarding (Liu, et al., 2016).

Therefore, this section proposes the development of novel soft-sensors able to predict the "future" values of the QIV at the "next" time instances for the same class of challenging batch processes (those which are expected to operate under different IC and include significant differences in the sampling rates among their measured variables). The proposed dynamic soft-sensor is based on the use of Nonlinear AutoRegressive (NAR) models (as presented in Chapter 6), which has been customized to suit this class of processes, through their integration with a suitable data collection procedure/guideline and an imputation step for the missing data.

9.4.1 Soft-sensor modeling approach

The dynamic soft-sensor modeling approach is based on building a NAR model (Eq.(9.13)) to approximate the future behavior of the offline QIV (y_{t+1}) as a function of the previous state of the batch, including the offline and online variables (y and x) over a specific time lag or delay L.

$$y_{t+1} = f[y_t, y_{t-1}, \dots . y_{t-L}, x_t, x_{t-1}, \dots . x_{t-L}] \tag{9.13}$$

The application of the NAR models, as presented in Chapter 6, to the targeted class of processes poses several challenges. Thus, the proposed dynamic soft-sensing methodology has been tailored to tackle these challenges. This includes a first step for adjusting the collection of training data: data from different batch runs with different ICs $[y_{t=0}, x_{t=0}]$ are collected through the whole operation domain $[y_{t=0}^{min}: y_{t=0}^{max}, x_{t=0}^{min}: x_{t=0}^{max}]$, in order to get the maximum information about process dynamics.

Having all the system variables available at the same time step is a numerical requirement for the application of NAR models, due to their recursive prediction nature. So, in a second step, a simple linear interpolation is used to estimate the missing data of the offline QIV for each training batch. Advanced techniques for interpolation (e.g., cubic, splines) may

be also used, but they are not recommended, in order to avoid forcing any incorrect behavior to the imputed data. Then, the data of each batch (including the imputed data) are unfolded in the form of input-output set (see Eq.(9.13)), and used for training the NAR model according to the requirements of the adopted data-modeling technique (GP, OK or ANN).

The resulting NAR model(s) are then used along a series of validation batches in a recursive interpolation manner, to predict the future values of the offline QIV (y_{t+1}) along the whole batch run, knowing only the initial values [$x_{t=0}$; $y_{t=0}$] and the online measured variables x_t. The time grid at which the offline variables are imputed is selected via cut-and-try procedure, in order to achieve the best prediction accuracy of the fitted NAR model, but avoiding excessive redundancy of training data, which would severely complicate the training task. Similarly, a try and cut procedure is also used to choose the dynamic soft-sensor model lag L taking into account both accuracy and model simplicity.

9.4.2 Applications

9.4.2.1 Simulation-based case study

The proposed procedure is first illustrated through its application to a simple simulation case (Tieu, et al., 1995). A batch process running the reactions $A \rightarrow B \rightarrow C$ is considered, where A is the reactant, B is the undesired product and C is the desired product. During a 30-minute batch, the concentration of C is assumed to be determined at 7 time instants, simulating an expensive offline QIV determination, while the concentrations of A and B are recorded every second, simulating the case of online measured variables. 24 training batches are simulated in this way, using different initial values $\left[C_{A(t=0)}, C_{B(t=0)}, C_{C(t=0)}\right]$ selected by a Hammersley sampling procedure within the limits [14:20, 0:2, 0:2]. Small white noise $\simeq \mathcal{N}(\mu = 0, \sigma = 0.03)$ is added to the recorded online values (C_A, C_B), while a higher error $\simeq \mathcal{N}(\mu = 0, \sigma = 0.3)$ is also added to the simulated offline data (C_C). 100 additional batches with different initial concentrations (within the same range [14:20, 0:2, 0:2]) are also simulated, in order to be used as a validation set.

$$C_{C(t+1)} = f\left[C_{A(t)} \dots C_{A(t-L)}, C_{B(t)} \dots . C_{B(t-L)}, C_{C(t)} \dots . . C_{C(t-L)}\right] \qquad (9.14)$$

The data from the 24 training batches are smoothed using a moving average method. Then the missing C_C data of each training batch are imputed at a specific time grid. Hence, the data from the training batches (real and imputed) are collected, unfolded in one input-output dataset, and finally three NAR models (Eq.(9.14)) are trained, based on the ANN, GP and OK techniques.

Figure *9.22*-(a) shows two of these training batches, including the simulated measurements of C_C and the imputed values. Different imputation grids have been tested, and the one that achieved the best averaged prediction accuracy for the 100 validation batches has been selected.

Figure *9.22*-(b,c) and Table 9.7 show the enhancement obtained by the proposed approach (NAR models trained with real and imputed values of C_C using a suitable lag for each model) respect to the direct/classical applications of the NAR models (real C_C data without imputation and considering zero lag). The improvements are evident disregarding the specific technique (GP, OK, ANN). Table 9.7 shows the average NRMSE and CC for the prediction of the 100 test batches, also considering both scenarios: classical method (unshaded side), and proposed method (shaded side). All errors are relative to the underlying behavior described by the system FPM.

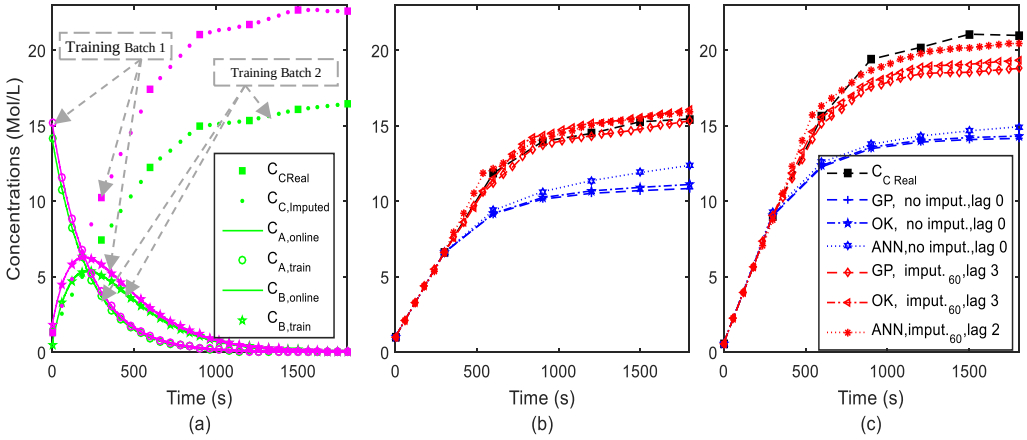


Figure 9.22. (a) Data of two training batches, (b) and (c) Prediction of two validation batches.

Table 9.7. Average NRMSE (%) and Pearson coefficient for the 100 validation batches.

Model	***NRMSE***	***PC***	***Model***	***Imputation Grid (sec)***	***Lag (L)***	***NRMSE***	***Pearson Coefficient***
GP	19.2	0.9816	***GP***	60	3	3.9	0.9991
ANN	17.2	0.9869	***OK***	60	3	3.5	0.9990
OK	18.7	0.9831	***ANN***	60	2	2.6	0.9986

9.4.2.2 Application to a photo-Fenton pilot plant

The second application considers the same photo-Fenton pilot plant for wastewater treatment running in a batch mode, which is already presented in section 9.4.2.2. Also, the same data used in section 9.4.2.2 (11 batches) and its divisions to training and validation subsets are exactly maintained.

The training subset (7 batches) are used to fit the TOC dynamic soft-sensor based on the NAR model illustrated by Eq.(9.15), while the validation subset (4 batches) are used to assess the soft-sensor performance. For comparison, the TOC soft-sensor in Eq.(9.15) is also trained using only the measured TOC values, without any imputation. Table 9.8 and Figure 9.23 show the predicted TOC accuracy for the validation batches, with imputation and best Lag (shaded part of the table) and without imputing the missing data and $L = 0$, confirming the advantages of the proposed approach.

$$TOC_{t+1} = f[TOC_t, TOC_{t-1}, \ldots TOC_{t-L}, T_t, T_{t-1}, \ldots T_{t-L}, R_t, R_{t-1}, \ldots R_{t-L}] \quad (9.15)$$

Table 9.8. Average accuracy measures (NRMSE, CC) of the TOC dynamic soft-sensors.

	NRMSE	CC		Imputation Grid (sec)	Lag (L)	NRMSE	CC
GP	5.2	0.9862	***GP***	180	3	4.4	0.9889
OK	5.2	0.9869	***OK***	180	4	3.0	0.9891
ANN	6.1	0.9681	***ANN***	180	3	3.8	0.9827

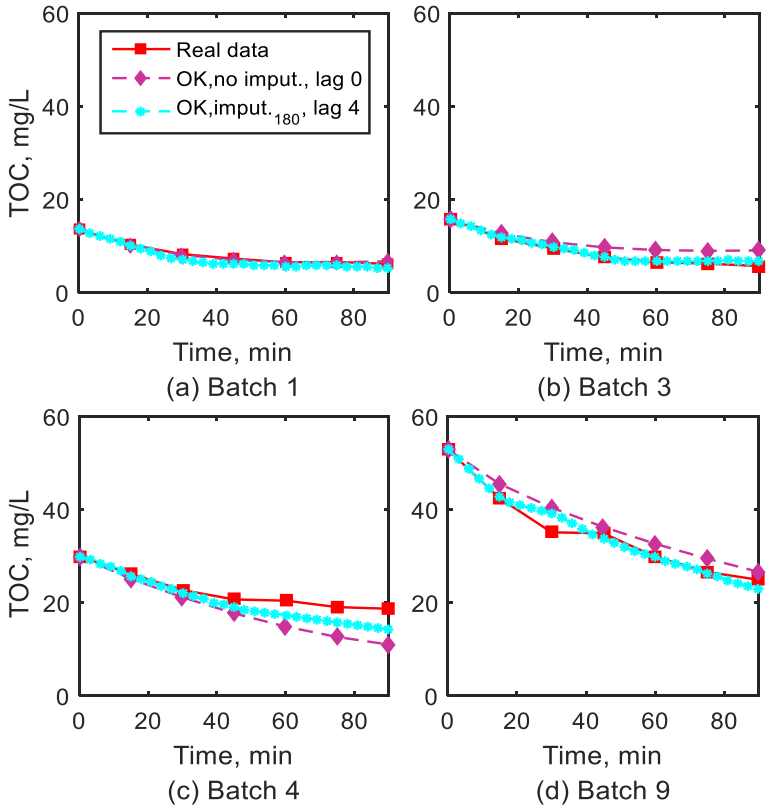


Figure 9.23. TOC prediction (OK-based dynamic soft-sensor for the validation batches).

Table 9.9 shows the NRMSE for each of the four validation batches independently. It should be noticed that batches no. 9 and no. 4 are showing the lowest prediction accuracy (maximum NRMSE).

Table 9.9. Average NRMSE for each of the four validation batches.

	Imput. grid	***Lag L***	***Batch No.***			
			1	**3**	**4**	**9**
GP	180	3	1.5	2.1	8.0	6.1
OK	180	4	1.8	1.0	5.2	3.8
ANN	180	3	2.2	2.5	5.5	5.2

The obtained soft-sensor may be also integrated in a monitoring/supervision system, where the measured process output (TOC) is compared to the dynamic soft-sensor prediction. Residuals can be used to check process malfunctioning, through comparing its value to a specific threshold that is often specified based on the knowledge about the processes and the soft-sensor accuracy. A faulty batch (rather than the 11 batches) illustrates this: the difference between the predicted and the measured TOC value (residual value) is compared with a threshold (five times of the average absolute prediction error of the dynamic soft-sensor; i.e. 5σ). Figure 9.24-(a) shows the predicted TOC of the faulty batch, and the corresponding real measurements, while Figure 9.24-(b) displays the generated residual signals of the healthy batches and a faulty one.

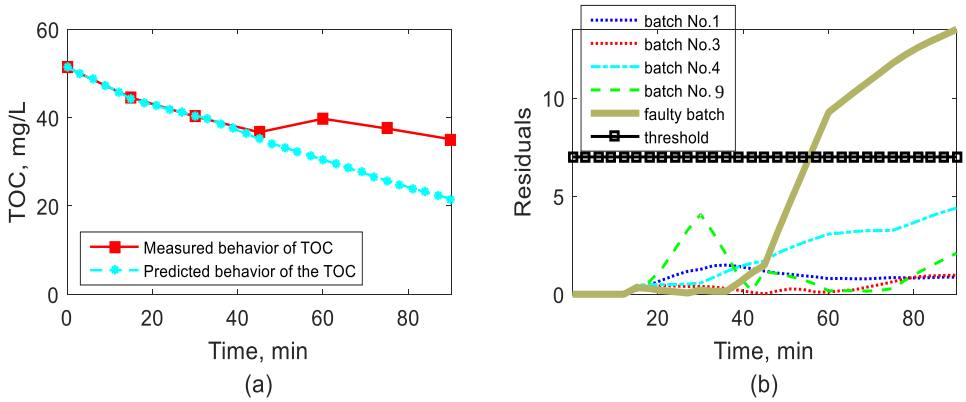


Figure 9.24. (a) TOC prediction of the faulty batch and (b) generated residuals for different batches.

9.5 CONCLUSIONS AND FUTURE WORK

This work proposes soft-sensing methodologies for a specific class of batch processes, those whose dynamic nature is combined with a significant variability in the initial batch conditions. The presented soft-sensors building approaches are found to be appropriate for these cases that usually appear when the process should manage raw materials whose specifications or properties frequently differ from one batch to another (e.g. waste treatment systems), new process conditions are considered (e.g. to incorporate new production resources), or at the

early stages of designing new products, so different alternatives of the initial conditions/settings are explored.

The procedure is tested using the most common data-based modeling architectures in the soft-sensing area, as OK, GP, SVR and ANNs in order to demonstrate its robustness and potential capabilities. The method is applied to simulation case-studies and a pilot-plant situation running the photo-Fenton reaction (an advanced oxidation process) for wastewater treatment in batch mode, in order to online predict the reaction progress.

The results produced reveal promising prediction accuracy even when few input-output training data are available due to large sampling periods. Additionally, they illustrate high capabilities in the approximation of the QIV -that is expensively measured by offline sampling and analyses- along the whole batch, even when the batch initial conditions vary from one batch run to another. Consequently, the application of the proposed methodologies may result in huge savings of time and cost consumed by the expensive offline sampling and experimentations, enhancing the online supervision and monitoring of such complex processes difficult to follow through FPMs, and also to allow flexible exploration of many alternative designs (of the initial conditions) with minimum cost.

In quantitative terms, the accuracy shown by the soft-sensors proposed in this work clearly meets the target expectation of being within the same order of magnitude of the accuracy offered by the real-sensor(s) providing the information used to train the soft-sensor, additionally affected by the accuracy of the available online information used to make the required estimations. In any case, it is worth noting that, when a soft-sensor is needed, this is most likely because no feasible real-sensor is available to work online and, in this sense, the proposed soft-sensors offer better accuracy than other soft-sensors recently proposed in the literature for continuous and batch processes and, even more, they are applicable to situations not supported by other alternatives (batch processes with changing initial conditions).

Considering the presented application cases in their different versions, the OK and the GPR based models have shown the best average performance among all the considered metamodels in terms of their prediction accuracy. Besides, both exhibit very high flexibility and robustness during their tuning -compared to the SVR and ANN-, since all the model parameters can be easily optimized (the modeler just makes a guess about the parameters initial values and then the optimization/training procedure provides their optimal values). Additionally, the OK and GPR metamodels offer an outperforming characteristic: their ability to estimate a prediction error, which has demonstrated to be very useful to construct a confidence area around the soft-sensor predictions. This estimation can be used for process control when the real behavior of

the system is unknown (as in the photo-Fenton case), or when it is hard to estimate the prediction accuracy.

The SVR has also achieved good accuracy, and was very competitive to the OK and GPR in the first simulation cases study. However, the SVR requires higher effort and time to select a suitable kernel function (once the kernel is selected, the SVR training task –the support vectors selection- is usually carried out requiring very low computational effort). Similarly, the ANN requires a significantly higher testing effort to select the best configuration including the number of layers, the number of neurons in each layer and the training algorithm. Nevertheless, it is not the main objective of the work to compare specific metamodeling approaches, but to prove the flexibility and robustness of the soft-sensing approach to work properly with different metamodel types. Thus, any other metamodel type can be employed, and it may achieve better accuracy than the considered ones (e.g. advanced types of ANNs).

Particularly, with respect to the first soft-sensing methodology (Section 9.3), in spite of the inherent dynamic behavior of any batch system, in many cases (as the ones studied in this work) it is not required to introduce any lag or delay to the soft-sensor model (i.e.: explicit dynamics) to provide very high prediction accuracy, so the dynamic nature of the process can be correctly tracked just through the available online indirect measurements. In the proposed approach, lag introduction is feasible, but it would result in additional effort in means of the complication of the soft-sensor model structure through including additional model inputs (i.e. delayed values of the online variables) and, consequently, additional model parameters which, in turn, would increase the required size of training data –which are often scarce for such processes- in order to identify these extra parameters. However, a future work line is the investigation of the ability of the first soft-sensing methodology to handle batch processes involving delayed behavior, where the lagged values of the online variables can be considered as additional model inputs. Hence, the soft-sensor in Eq.(9.7) might be $y(t) = f[x(t), x(t-1), \dots x(t-L), x(t=0), y(t=0)]$, where L=1, 2,... represents the suitable lag to be specifically applied to the model (Espinosa & Vandewalle, 1998a; Espinosa & Vandewalle, 1998b; Nelles, 2001; Cho, et al., 2007).

With respect to the second soft-sensing methodology (Section 9.4), the constructed soft-sensors-based NAR models show promising enhancements in terms of the ability to predict the future values of the QIV along different batch runs. Thus, they can be better used for supporting the process control and supervision tasks. The consideration of lagged input in the dynamic soft-sensors design is essential, since these soft-sensors are not mapping the instantaneous (i.e., at the same shot of time) relation between the QIV and the online variables, but they are approximating the evolution of the QIV over the time, which is more complex.

Also, the consideration of the imputed data allows higher modeling degrees of freedom (i.e., possibilities of trying different model lags) when very limited number of samples measured over wide time intervals along each batch run are available. In contrast, if the NAR models are directly applied to such class of processes introducing higher lags ($L = 4$), five samples of the batch will be required to start the prediction, which makes the dynamic model useless.

Chapter 10: Conclusions and Future Work

In the chemical process engineering area, the growing capacity and efficiency of available hardware and software computational resources can hardly satisfy the also increasing demand for using higher fidelity models, which involve more detailed descriptions of the process behavior, with the purpose of achieving more accurate and realistic operational decisions. As a consequence, the use of FPM models is becoming more and more computationally challenging, especially at the lower levels of the decision-making hierarchy, e.g., online optimization, model predictive control, fault detection and diagnosis etc., where the operational decisions are required in minutes or even seconds, and decision making involves lots of simulation runs using complex and highly nonlinear process models. Surrogate modelling approaches, in which a simple and accurate data-driven model replaces the complex FPM in these applications, represent a potential solution to overcome such challenges. However, their current use in the chemical engineering area is just concentrated on process design and steady-state optimization problems.

10.1 THESIS CONTRIBUTIONS

This Thesis presents a framework for the efficient and systematic use of surrogate models and other artificial intelligence tools (e.g., DOCE, classification, clustering) in different tasks related to the process operation optimization, monitoring and control. The framework includes a set of efficient and flexible methodologies, each supporting the decision making in a specific phase/module of the process operation management domain. These phases are the steady-state optimization, dynamic optimization, model predictive control, multivariate system identification, multistep-ahead prediction, fault detection and diagnosis and soft-sensing. In this sense, it should be noted that the main challenges identified in Chapter 1 have been addressed through the different chapters of this Thesis.

The first challenge addressed in this Thesis is the practical difficulties of using complex, highly nonlinear and/or black box FPMs in the steady-state operation optimization module, and the additional complications associated to the process-inherent and external uncertainties that amplifies this challenge.

- In Chapter 3, an improved implementation of the usual Surrogate-Based Optimization (SBO) methodology is presented and applied to complex nonlinear steady-state chemical processes. The method replaces the entire complex FPM with a set of global

kriging metamodels representing the objective function and the constraints. The search space of the decision variables is explored by an adaptive sampling procedure, based on the Constrained Expected Improvement (CEI) criterion, in which the metamodels are iteratively updated with the obtained optimal solution to refine the search around the candidate solution. The methodology is compared to classical optimization procedures through their applications to three benchmark examples and two case studies including black box models that are built using modular process simulators. The methodology and its proposed implementation provide very accurate solutions, significantly reducing the number of the function evaluations and computational effort required, avoiding local optima, exploring separated feasible regions in constrained optimization, and ensuring the convergence of the optimization problem to global solutions. With these proven capabilities, the methodology demonstrates its reliability for applications requiring hour-to-hour or day-to-day operation optimization in complex processes.

- The performance of the SBO methodology proposed in Chapter 3 might be limited by the frequent and sudden variation of process-inherent and external uncertainty sources/parameters (inlet feed temperatures, demand, etc.). To overcome this limitation, Chapter 4 presents innovative SBO methodologies for the continuous and mixed-integer multiparametric optimization of steady-state processes influenced by traceable uncertainties. The main idea is simple but innovative: the training of surrogate models and classification techniques to capture the underlaying mapping between the traceable Uncertain Parameters (UPs) and the optimal decision variables of the process. The developed methodologies combine DOCE techniques, state-of-art optimization algorithms, metamodeling, classifications and clustering techniques. The proposed methods are able to accurately approximate the multiparametric behavior of the optimal solution with very high accuracy using relatively small number of training data. Moreover, the simplicity, systematism and robustness of these methodologies have been proven through their application to different optimization problem types (linear, bilinear, quadratic, nonlinear, black box) in front of many classical/mathematical MultiParametric Programming (MPP) algorithms, each of them specialized and particularly tailored to solve a type of optimization problems.

Once the optimum steady-state operating conditions (i.e., set points) have been identified, the subsequent task is to maintain the plant operating conditions at their optimum set-points against the eventual disturbances and fluctuations through the use of efficient Model Predictive Control (MPC) technologies, able to handle the complexities of the process.

- Chapter 5 presents a new Data-Based MultiParametric-Model Predictive Control (DBMP-MPC) methodology, which enables simple implementations of explicit MPC for nonlinear processes. The method is established considering the same principles applied to develop the methodologies presented in Chapter 4, where surrogate models are built offline to approximate the relation between the state variables of the process, measured at the current time instant, and the optimal values of the control variables to be applied to the process along the next time step or sampling period. Then, the developed surrogate models, which represent explicit control laws, are used online to control the process by calculating the optimal control decisions through simple interpolation instead of solving the classical MPC problem based on complex FPMs. The results obtained by applying this methodology to benchmark problems proposed in the MPC literature show the very high accuracy of the predicted optimal control decisions, its robustness to handle different dynamic model structures (i.e., discrete and continuous dynamic models) and its capability to overpass complex mathematical formulations associated to the traditional MP-MPC.

Up to the moment, in the Process System Engineering (PSE) area, surrogate modeling techniques have been mainly applied to systems assuming steady-state condition, although in may modules of the process operation (e.g., dynamic optimization, MPC, Fault Detection and Diagnosis (FDD)), dynamic FPMs of the process must be considered, and, in many cases, they are complex.

- Chapter 6 presents the basis for the extension of the proposed approaches to non-steady-state systems, in the form of a robust and generic methodology for data-driven multivariate dynamic modelling and multistep ahead prediction of nonlinear chemical processes. The developed methodology utilizes surrogate models for building a set of Nonlinear AutoRegressive with eXogenous (NARX) models, each of them predicting the evolution of one output as a function of the other inputs and outputs of the process. The resulting set of multivariate dynamic models is, then, used to forecast the process outputs along larger time intervals, through a recursive and inter-coordinated prediction scheme. Additionally, the proposed methodology provides a novel procedure for the DOCE in dynamic modelling, suitable for the approximation of complex and computationally expensive dynamic FPMs. The multivariate dynamic models provided by this methodology showed very promising capabilities such as high prediction accuracy, large prediction time horizons and high applicability and robustness (handling different case studies of different nature, integrating different metamodel types, etc.).

These capabilities efficiently satisfy the requirements and needs of other process operations modules, such as dynamic optimization (Chapter 7) and monitoring and supervision (Chapter 8), where the future values of many outputs of the process must be accurately and rapidly predicted.

- Chapter 7 introduces a novel surrogate-based dynamic optimization methodology for the quick solution of open-loop optimal control problems associated to complex highly nonlinear processes. The methodology is based on building a set of multivariate dynamic surrogate models (as presented in Chapter 6), which are able to accurately and rapidly predict the process outputs with respect to any time-profile of the process control inputs. The resulting set of dynamic surrogate models is then integrated in a sequential dynamic optimization procedure based on a Control Vector Parametrization (CVP) approach. The accuracy of the optimal solutions found and the significant reduction of the required computational effort stem from the fact that the multivariate dynamic surrogate models accurately predict the time-profiles of the process outputs through simple and rapid recursive interpolations, avoiding the expensive integration of a complex dynamic FPMs, as in standard dynamic optimization techniques.
- Chapter 8 presents a novel data-driven methodology for Fault Detection and Diagnosis (FDD) of chemical processes operated under time-varying control inputs (e.g., plant start-ups and shutdowns, changes in set-points, etc.), when the failure rate is likely to be higher. The proposed methodology relies on a dynamic observer (based on multivariate dynamic surrogate models - Chapter 6), which predicts the expected normal behavior of the system, and the use of classification techniques which are trained using the residuals patterns created from the comparison between the process outputs estimated by the dynamic observer (expected normal behavior) and the actual outputs of the process measured under different dynamic conditions and faulty scenarios. The obtained results confirm the methodology capabilities to enhance the performance of classical data-driven approaches for FDD (which are based on the sole use of classification techniques trained using measured process outputs) when used for nonlinear processes operating under time-varying inputs. Thanks to the use of this data-driven observer, valuable information about the process dynamics, in the form of highly sensitive residuals, are efficiently fed to the classification techniques. This enables better identification of the process state (normal one or in one of the faulty conditions) and provides the flexibility required to deal with different fault severities, fault scenarios (sequences of different fault types) and fault styles (abrupt and incipient).

- Finally, Chapter 9 introduces new data-driven soft-sensing techniques applicable to a specific class of batch processes, those whose dynamic nature is combined with a significant variability in the batch initial conditions (e.g., processes managing raw materials whose specifications or properties may differ from one batch to another, or when new process conditions must be considered). The capabilities of the proposed methods are proved by their applications to simulation benchmark case-studies and a real photochemical pilot plant running a water treatment process in a batch mode. The results confirm: *i*) the very high accuracy of the predictions obtained by the proposed soft-sensors, even when few training data (measurements) are available for the Quality Indicator Variable (QIV), *ii*) significant savings (in time and cost) associated to the reduction of expensive offline sampling and laboratory measurements of the QIV, and iii) significant potential improvements in the feasibility and reliability of the process control, supervision and monitoring applicable techniques, which typically require online and continuous measurements of the process QIV variables, whose measurements are typically expensive, time consuming and obtained with very large delay.

10.2 DEPTH AND WIDTH OF THE THESIS DEVELOPMENTS

The Thesis offers an exhaustive library of novel methodologies that exploit the powerful capabilities of machine learning (regression or surrogate models, classification and clustering) and artificial intelligence (e.g., Genetic Algorithms) techniques. The methodologies are aimed at supporting the decision making at the main stages of the process operation optimization in situations when it is not possible to use the available analytical models of the process due to their complexity, or when the measurements data are the only available source of information about the process without the support of a well-founded analytical model. These stages include the steady-state optimization (Chapters 3, 4), model predictive control (Chapters 5), multivariate dynamic modeling and multistep ahead predictions (Chapter 6), dynamic optimization (Chapter 7), fault detection and diagnosis (Chapter 8) and soft-sensing (Chapter 9).

The powerful capabilities of machine learning techniques, in terms of the high potential of universal approximation and high applicability and adaptability, are efficiently employed in this Thesis to approximate i) the steady-state behavior of the process (Chapter 3), ii) the optimal behavior of the process in response to uncertain variations (Chapters 4, 5) and iii) the dynamic behavior of the process (Chapters 6, 7, 8, 9).

Thanks to these three main application scenarios, machine learning techniques have been the core of novel methods performing diverse and different functions. Within the first

application scenario, the developed surrogate models can be used to simplify the steady-state optimization task by not only reducing the required computational effort but also by improving the reliability of the computations/convergence, essential in the case of complex systems (Chapter 3). With respect to the second application scenario, the developed surrogate models are able to simply and accurately reproduce multiparametric relations and control laws to predict/adapt the optimal set-points (Chapter 4) or control inputs (Chapter 5) of a process in response to uncertain variation/fluctuations of the process conditions, alleviating very complex mathematical solution procedures proposed for the same goal in the current literature. Regarding the third application scenario, the developed surrogate models have been trained to act as simple and accurate multivariate dynamic discrete state-space models of the process (Chapter 6), which enhanced the dynamic optimization tasks by a significant reduction in the required computational effort (Chapter 7). Also, these dynamic surrogate models improved the process supervision, by acting as observers of the process that feed relevant information about the process dynamics to the fault classifiers to enhance their detection and diagnosis performance (Chapter 8). Finally, the surrogate models have been proposed as soft-sensors to enable the real-time monitoring of a process thanks to their continuous and real-time estimates of the QIV, which otherwise must be expensively measured through offline laboratory procedures that, additionally, introduce a very large time delay in the control loop (Chapter 8). Finally, it's worth to emphasize, again, that the machine learning models have been proven to be able to efficiently (i.e., accurately and quickly) and reliably perform all these wide range of functions only relying on data collected from the real process (or generated by complex model simulations), without the need of knowledge neither about the first principles governing the process nor about complex mathematical optimization formulations and techniques which may fail to perform the required tasks.

Besides the diversity of the functions (e.g., optimization, control, system identification, supervision and soft-sensing) that the surrogate models have been proven to efficiently perform (through the developed methodologies in this Thesis), it's important to spot the light on the diversity of the applications themselves. This Thesis has not considered a certain case-study or even a certain class of them. On the contrary, it handled a relatively large number of examples and case studies (reached to 24) possessing very diverse characteristics, including

- fed-batch (Chapters 6, 7, 9) and continuous (Chapters 3, 4, 5, 6, 8) processes,
- single-unit (Chapters 5, 6, 7,8, 9) and multi-units (Chapters 3, 4) processes,
- a wide range of domains such as energy engineering (utility systems in Chapters 3, 4), reaction engineering (fed-batch reactors in Chapters 4, 8, CSTR reactor in Chapter 5, Oil-shale pyrolysis reactor in Chapter 6), biochemical engineering (bioreactor in

Chapter 6, Penciling production reactor in Chapter 9) fluid dynamics (three-tank system in Chapter 6 and 8), and environmental engineering (batch Photo-Fenton reactor for water treatment in Chapter 9).

This, from one side, confirms the high potential of these machine learning or surrogate models in terms of their applicability and robustness and, from the other side, indicates the credible and exhaustive validation of the developed methodologies.

Within the developed methodologies, different machine learning techniques for regression have been compared (Artificial Neural Network (ANN) Ordinary Kriging (OK), Support Vector Regression (SVR)), as well as for classification (ANN, Support Vector Machine (SVM)). For regression, OK and ANN have shown the best performances. In general, the OK showed the best approximation accuracies, especially when the size of the training set is small (Chapter 9), due to its nonparametric nature that reduces the number of model parameters to be tuned and, consequently, the required size of training data. Another advantage of the OK model is the fact that all of its parameter values are automatically tuned. Also, the OK outstanding ability to estimate an error representing the uncertainty about its prediction, enables the development of efficient SBO procedures with probabilistic and global search mechanism (e.g., CEI in Chapter 3) able to consider the infeasibility of the optimal solution due to the models' approximation errors, as well as the adaptation of the models during the optimization search. However, the OK suffers from a serious limitation, which is the very high computational cost required for its parameters tuning, especially for high dimensional cases and/or large training datasets, because of the associated repetitive and expensive calculations of the inverse of the correlation matrix among the training data samples.

The ANN based approaches, on the other side, also showed very good performance, in terms of the high prediction accuracies which were, in most cases, comparable to those of the OK. In situations where sufficient number of training data are available (e.g., Chapter 5, 6) and/or where the approximated output shows non-smooth behavior (e.g., the stepwise behavior of control laws in Chapter5), the accuracy of the ANN outperforms the results from other models. Also, in the case of high dimensional input and/or large training data size, the ANNs require much lower computational effort than that of the OK. The main drawback of the ANN is the effort required for the selection of its structure (i.e., number of layers, number neurons in each layer, transfer function type, training algorithm, etc.). For the classification techniques (chapter 8), ANN classifiers showed better accuracy and more robust performance than that of the SVM.

10.3 EXTENSIONS AND FURTHER DEVELOPMENTS

One of the main strength points of the Thesis is the high flexibility of the developed methodologies, which makes them a very good basis for future research. In this context, the perspectives include:

The extension of the SBO method, developed in Chapter 3, in order to handle general multi-objective optimization problems, where each objective is to be represented with a kriging model (besides the kriging models of the constraints). This will imply the development of new sequential optimization search procedures (i.e., Multi-objective Constraint Expected Improvement methods) that consider the sequential enhancement of the entire Pareto front.

For the methodologies developed in Chapters 4 and 5, an interesting research line would be the extension of their capabilities to improve the approximation of the multiparametric behavior of continuous decision variables (Chapter 4) or continuous control input (Chapter 5) that show significant/discrete changes over the UPs (Chapter 4) or state variables (Chapter 5) space. This can be accomplished by exploiting adaptive sampling techniques to collect training data from the local subspaces of the UPs at which the behavior of the optimal decision variables or control inputs show sharp changes. Also, the use of deep learning models (e.g., Convolutional Neural Networks) can help to treat the previous problem since their powerful approximation capacities make them good candidates to efficiently handle such discrete behavior, better than classical machine learning models (e.g., kriging and ANN).

With respect to the data-driven explicit MPC methodology presented in Chapter 5, an additional research line would consist of expanding its capabilities so as to handle hybrid MPC problems, where the control inputs to be determined (optimized) are of discrete nature. For this purpose, clustering and classification techniques might be considered to handle and quantify this discrete or integer behavior. It would be also worth to consider MPC problems involving UPs (that often influence the process dynamic model), so the optimal control at the future sampling period will be modeled as a function of the current values of the state variables and the UPs.

Regarding the multivariate dynamic modeling and multistep-ahead prediction methodology developed in Chapter 6, three interesting future research lines are recommended. The first one consists in handling real world problems (i.e., data) where, in spite of the quantity of data, very limited knowledge about the process behavior is known in the form of specific profiles of input-output data signals. The second is the management of the prediction uncertainty in order to propagate it through the multistep-ahead prediction. Finally, the characteristics of the Long-Short Term Memory (LSTM) ANNs for modeling temporal information are probably worth to be exploited in the multivariate dynamic modeling of

chemical processes. These characteristics include the ability of capturing long-term temporal dependencies, the automatic extracting of temporal and spatial patterns in the input feature space, and the capacity of handling nonlinear and complex feature interactions in the data without explicitly defining them and memorizing the occurrence of the distant-past and the near-past and balance out the two when making predictions, all of them contributing to a potential increase of the prediction accuracy.

With respect to Chapter 7, considering more complex application cases (higher dimensional fed-batch processes) to further exploit the capabilities of the data-driven dynamic optimization methodology is a straightforward future work. Also, the improvement of the methodology through a sort of adaptive optimization procedure will be interesting: at each iteration, the data-driven dynamic models of the process might be updated with the input-output data corresponding to the identified optimal profiles of control variables and the corresponding profiles of the state variables.

Regarding Chapter 8, the extension of the methodology capacities in order to handle "unsupervised" FDD is a realistic need that should be investigated, where the labels of faults are not known. In this line, advanced clustering techniques to isolate the monitoring data with respect to the process faults can be leveraged. Also, another line is the exploration the performance of the methodology in more realistic working conditions, for example, when the input disturbances affect both normal and abnormal/fluty conditions, when a closed-loop control system is in operation (thus compensating the effects of the faults), or when the non-linearities of the system are stronger, complicating the task of the observer (prediction).

Finally, for Chapter 9, an interesting research direction is to investigate the development of multivariate soft-sensors, where more than one QIV of the process is to be modeled.

References

Acevedo, J. & Pistikopoulos, E. N., 1997. A Multiparametric Programming Approach for Linear Process Engineering Problems under Uncertainty. *Ind. Eng. Chem. Res.,* Volume 36, pp. 717-728.

Addin, O., Sapuan, S. M., Othman, M. & Ali, B. A. A., 2011. Comparison of Naïve bayes classifier with back propagation neural network classifier based on f - folds feature extraction algorithm for ball bearing fault diagnostic system. *International Journal of Physical Sciences,* Volume 6, pp. 3181-3188.

Adebiyi, O. A. & Corripio, A. B., 2003. Dynamic neural networks partial least squares (DNNPLS)identification of multivariable processes. *Computers and Chemical Engineering,* Volume 27 , pp. 143-155.

Akram, A. N., Isa, D., Rajkumar, R. & Lee, L. H., 2014. Active incremental Support Vector Machine for oil and gas pipeline defects prediction system using long range ultrasonic transducers.. *Ultrasonics,* Volume 54, p. 1534–1544.

Ali, J. M., Hussain, M. A., Tade, M. O. & Zhang, J., 2015. Artificial Intelligence techniques applied as estimator in chemical process systems – A literature survey. *Expert Systems with Applications,* Volume 42, pp. 5915-5931.

Amozeghar, M. & Khorasani, K., 2016. An ensemble of dynamic neural network identifiers for fault detection and isolation of gas turbine engines. *Neural Networks,* Volume 76, p. 106–121.

Ardakani, M. H. et al., 2016c. Optimal Feature Selection for Designing a Fault Diagnosis System. *Computer Aided Chemical Engineering,* Volume 38, pp. 1111-1116.

Ardakani, M. H. et al., 2016a. *A Framework for Unsupervised Fault Detection and Diagnosis Based on Clustering Assisted Kriging Observer.* Barcelona, Spain, IEEE, pp. 183-188.

Ardakani, M. H. et al., 2016b. Imputation of Missing Data with Ordinary Kriging for Enhancing Fault Detection and Diagnosis. *Computer Aided Chemical Engineering,* Volume 38, pp. 1377-1382.

Askarian, M. et al., 2016. Fault diagnosis of chemical processes with incomplete observations: A comparative stud. *Computers & Chemical Engineering,* Volume 84, p. 104–116.

Atoui, M. A., Verron, S. & Kobi, A., 2015. Fault Detection and Diagnosis in a Bayesian Network classifier incorporating probabilistic boundary. *IFAC-PapersOnLine,* Volume 48, p. 670–675.

Ayodele, O. B., Auta, H. S. & Nor, N. M., 2012. Artificial Neural Networks, Optimization and Kinetic Modeling of Amoxicillin Degradation in Photo-Fenton Process Using Aluminum Pillared Montmorillonite-Supported Ferrioxalate Catalyst. *Ind. Eng. Chem. Res.,* Volume 51 , p. 16311–16319.

Azman, K. & Kocijan, J., 2007. Application of Gaussian processes for black-box modelling of biosystems. *ISA Transactions,* Volume 46, pp. 443-457.

Ažman, K. & Kocijan, J., 2011. Dynamical systems identification using Gaussian process models with incorporated local models. *Engineering Applications of Artificial Intelligence,* Volume 398-408, p. 24.

Bajpai, R. K. & Reul, M., 1980. A Mechanistic Model for Penicillin Production. *J. Chem. Tech. Biotechnol.,* Volume 30, pp. 332-344.

Banga, J. et al., 2005. Dynamic optimization of bioprocesses: Efficient and robust numerical strategies. *Journal of Biotechnology,* Volume 117, pp. 407-419.

Banu, U. S. & Umab, G., 2011. ANFIS based sensor fault detection for continuous stirred tank reactor. *Applied Soft Computing,* Volume 11, pp. 2618-2624.

Baraldi, P., Cadini, F., Mangili, F. & Zio, E., 2013. Model-based and data-driven prognostics under different available information. *Probabilistic Engineering Mechanics,* Volume 32, pp. 66-79.

Beck, J., Friedrich, D., Brandani, S. & Frag, E. S., 2015. Multi-objective optimisation using surrogate models for the design of VPSA systems. *Computers & Chemical Engineering,* Volume 82, pp. 318-329.

Belkacem, S., Bouafia, S. & Chabani, M., 2017. Study of oxytetracycline degradation by means of anodic oxidation process using platinized titanium (Ti/Pt) anode and modeling by artificial neural networks. *Process Safety and Environmental Protection,* Volume 111, pp. 170-179.

Bemporad, A., Morari, M., Dua, V. & Pistikopoulos, E. N., 2002. The Explicit Linear Quadratic Regulator for Constrained Systems. *Automatica,* Volume 38, pp. 3-20.

Benardos, P. & Vosniako, G.-C., 2007. Optimizing feedforward artificial neural network architecture. *Engineering Applications of Artificial Intelligence,* Volume 20, pp. 365-382.

Biegler, L., 2007. An overview of simultaneous strategies for dynamic optimization. *Chemical Engineering and Processing: Process Intensification,* Volume 46, pp. 1043-1053.

Biegler, L., 2010. *Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes.* Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Bojarski, A., Pérez-Fortes, M., Velo, E. & Puigjaner, L., 2010. *Life Cycle Assessment of Integrated Gasification Power Plants: Conceptual Design and Techno-Economic Evaluation.* Lausanne, International Conference on Efficiency, Cost, Optimization, Simulation and Environmental Impact of Energy Systems, pp. Book of Abstracts 1493-1501.

Bonne, D. & Jorgensen, S. B., 2004. *Data-driven modeling of batch processes.* s.l., s.n.

Boser, B., Guyon, I. & Vapnik, V., 1992. A training algorithm for optimal margin Classifiers. *in Proceedings of the workshop on computational learning theory, COLT.*

Boukouvala, F., Muzzio, F. J. & Ierapetritou, M. G., 2011. Dynamic Data-Driven Modeling of Pharmaceutical Processes. *Ind. Eng. Chem. Res,* Volume 50, p. 6743–6754.

Bradford, E. et al., 2018. Dynamic modeling and optimization of sustainable algal production with uncertainty using multivariate Gaussian processes. *Computers & Chemical Engineering,* Volume 118, pp. 143-158.

Bruno, J. C., Fernandez, F., Castells, F. & Grossmann, I., 1998. A Rigorous MINLP Model for The Optimal Synthesis and Operation of Utility Plants. *Chemical Engineering Research and Design,* Volume 76, pp. 246-258.

Caballero, J. A. & Grossmann, I. E., 2008. An algorithm for the use of surrogate models in modular flowsheet optimization. *AIChE Journal,* Volume 54, p. 2633–2650.

Caccavale, F. et al., 2010. A neural network approach for on-line fault detection of nitrogen sensors in alternated active sludge treatment plants. *Water Science & Technology,* Volume 62, pp. 2760-2768.

Calado, J. et al., 2001. Soft Computing Approaches to Fault Diagnosis for Dynamic Systems. *European Journal of Control,* pp. 248-286.

Carrasco, E. & Banga, J., 1997. Dynamic Optimization of Batch Reactors Using Adaptive Stochastic Algorithms. *Ind. Eng. Chem. Res.,* Volume 36, pp. 2252-2261.

Chaudhary, M. N. R., 2009. *Real Time Optimization of Chemical Processes. Master thesis.* Bentley, Western Australia: Curtin University of Technology.

Chia, G., Hua, S., Yanga, Y. & Chen, T., 2012. Response surface methodology with prediction uncertainty:A multi-objective optimisation approach. *Chemical Engineering Research and Design,* Volume 90, p. 1235–1244.

Cho, J., Principe, J. C., Erdogmus, D. & Motter, M. A., 2007. Quasi-sliding mode control strategy based on multiple-linear models. *Neurocomputing,* Volume 10, pp. 960-974.

Christianini, N. & Shawe, T., 2000. *An introduction to Support Vector Machines and Other Kernel-Based Learning Methods.* s.l.:Cambridge University Press.

Conn, A. R., Scheinberg, K. & Vicente, L. N., 2009. *Introduction to Derivative-Free Optimization.* Philadelphia, USA: MPS-SIAM Book Series on Optimization.

Conti, S., Gosling, J. P., Oakley, J. & O'Hagan, A., 2009. Gaussian process emulation of dynamic computer codes. *Biometrika,* Volume 96, pp. 663-676.

Cressi, N. A., 1993. *Statistics for spatial data.* New York: Wiley & Sons.

Cuthrell, J. & Biegler, L., 1989. Simultaneous optimization and solution methods for batch reactor control profiles. *Computers & Chemical Engineering,* Volume 13, pp. 49-62.

Dadebo, S. A. & Mcauley, K. B., 1995. Dynamic optimization of constrained chemical engineering problems using dynamic programming. *Computers & Chemical Engineering,* Volume 19, pp. 513-525.

Dantzig, G. & Thapa, M., 1997. *Linear Programming 1: Introduction.* Stanford, USA: Springer Series in Operations.

Dash, S. & Venkatasubramanian, V., 2000. challenges in the industerial applications of fault diagnostic systems.. *Computers & Chemical Engineering,* Volume 24, p. 785–791.

Davis, E. & Ierapetritou, M., 2007. A kriging method for the solution of nonlinear programs with black-box functions. *AIChE,* Volume 53, pp. 2001-2012.

Davis, E. & Ierapetritou, M., 2008. A Kriging-Based Approach to MINLP Containing Black-Box Models and Noise. *Ind. Eng. Chem. Res.,* Volume 47, pp. 6101-6125.

De Tuesta, j. L. D. et al., 2015. Application of high-temperature Fenton oxidation for the treatment of sulfonation plant wastewater. *J Chem Technol Biotechnol,* Volume 90, p. 1839–1846.

De-Canete, J. F. et al., 2016. Soft-sensing estimation of plant effluent concentrations in a biological wastewater treatment plant using an optimal neural network. *EXPERT SYST APPL,* Volume 63, pp. 8-19.

Deisenroth, M. P., Rasmussen, C. E. & Peters, J., 2009. Gaussian process dynamic programming. *Neurocomputing,* Volume 72, pp. 1508-1524.

Desai, K., Badhe, Y., Tambe, S. S. & Kulkarni, D., 2006. Soft-sensor development for fed-batch bioreactors using support vector regression. *Biochemical Engineering Journal,* Volume 27, p. 225–239.

Diangelakis, N. A., Burnak, B., Katz, J. & Pistikopoulos, E. N., 2017. Process design and control optimization: A simultaneous approach by multi-parametric programming. *AIChE Journal,* Volume 63, pp. 4827-4846.

Diehl, M., Bock, H., Diedam, H. & Wiebe, P.-B., 2006. Fast Direct Multiple Shooting Algorithms for Optimal Robot Control. *Lecture Notes in Control and Information Sciences ,* Volume 340, pp. 65-93.

Domínguez, L. F., Narciso, D. A. & Pistikopoulos, E. N., 2010. Recent advances in multiparametric nonlinear programming. *Computers and Chemical Engineering,* Volume 34 , pp. 707-716.

Dua, V., 2010. A mixed-integer programming approach for optimal configuration of artificial neural networks. *Chemical Engineering Research and Design,* Volume 88, pp. 55-60.

Dua, V., Bozinis, N. A. & Pistikopoulos, E. N., 2002. A multiparametric programming approach for mixed-integer quadratic engineering problems. *Computers and Chemical Engineering,* Volume 26, pp. 715-733.

Dua, V. & Pistikopoulos, E., 1999. Algorithms for the Solution of Multiparametric Mixed-Integer Nonlinear Optimization Problems. *Ind. Eng. Chem. Res.,* Volume 38, pp. 3976-3987.

Duran, A., Monteagudo, J. & Mohedano, M., 2006. Neural networks simulation of photo-Fenton degradation of Reactive Blue 4. *Applied Catalysis B: Environmental,* Volume 65, p. 127–134.

Duran, M. a. G. I., 1986. An outer-approximation algorithm for a class of mixed-integer nonlinearprograms. *Mathematical Programming,* Volumen 36, p. 307–339.

Durantin, C., Marzat, J. & Balesdent, M., 2016. Analysis of multi-objective Kriging-based methods for constrained global optimization. *Computational Optimization and Applications,* Volume 63, p. 903–926.

Egea, J. A., Vries, D., Alonso, A. A. & Banga, J. R., 2007. Global Optimization for Integrated Design and Control of Computationally Expensive Process Models. *Ind. Eng. Chem. Res.,* Volume 46, pp. 9148-9157.

Elhsoumi, A. et al., 2011. Monitoring of Chemical Processes Using Model-Based Approach. In: *Numerical Simulations of Physical and Engineering Processes.* s.l.:InTech, pp. 413-435.

Elmolla, E. S., Chaudhuri, M. & Eltoukhy, M. M., 2010. The use of artificial neural network (ANN) for modeling of COD removal from antibiotic aqueous solution by the Fenton process. *Journal of Hazardous Materials,* Volume 179, p. 127–134.

Espinosa, J. J. & Vandewalle, J., 1998a. Fuzzy Modeling and Identification, A guide for the user.

Espinosa, J. J. & Vandewalle, J., 1998b. Predictive control using fuzzy models applied to a steam generating unit. *Proceedings of the 3rd International Workshop on Fuzzy Logic and Intelligent Technologies for Nuclear Science and Industry.*

Expósito, A., Monteagudo, J., Durán, A. & Fernández, A., 2017. Dynamic behavior of hydroxyl radical in sono-photo-Fenton mineralization of synthetic municipal wastewater effluent containing antipyrine. *Ultrasonics Sonochemistry,* Volume 35,A, pp. 185-195.

Facco, P., Doplicher, F., Bezzo, F. & Barolo, M., 2009. Moving average PLS soft sensor for online product quality estimation in an industrial batch polymerization process.. *Journal of Process Control,* Volume 19, p. 520–529.

Fadda, G., 2017. *SUPERVISION AND DIAGNOSIS OF INDUSTRIAL SYSTEMS. PhD.* Cagliari: Universit`a degli Studi Di Cagliari.

Fang, K.-T., Li, R. & Sudjianto, A., 2005. *Design and modelling for computer experiment.* New York: Chapman and Hall/CRC.

Farias, J., Albizzati, E. & Alfano, O., 2009. Kinetic study of the photo-Fenton degradation of formic acid: Combined effects of temperature and iron concentration. *Catalysis Today,* Volume Catalysis Today, pp. 117-123.

Fisher, B. J., 1980. R.A. Fisher and the design of experiments (1922-1929). *The American Statistician,* Volume 34, pp. 1-7.

Fisher, R. A., 1971. *The Design of Experiments.* 9 ed. London: Macmillan Pub. Co..

Flemming, T., Bartl, M. & Li, P., 2007. Set-Point Optimization for Closed-Loop Control Systems under Uncertainty. *Ind. Eng. Chem. Res.,* Volume 46, pp. 4930-4942.

Fletcher, R., 1987. *Practical Methods of Optimization.* 2 ed. Chichester, UK: Wiley.

Forrester, A. & Keane, A., 2009. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences,* Volume 45, pp. 50-79.

Forrester, A., Sobester, A. & Keane, A., 2008. *Engineering Design via Surrogate Modelling: A Practical Guide.* Southampton, UK: John Wiley and Sons.

Frank, P. & Ding, X., 1997. Survey of robust residual generation and evaluation methods in observer-based fault detection systems. *Journal of Process Control,* Volume 17, pp. 403-424.

García-Laencina, P. J., Sancho-Gómez, J.-L. & Figueiras-Vidal, A. R., 2010. Pattern classification with missing data: a review. *Neural Computing and Applications,* Volume 19, p. 263–282.

Garud, S. S., Karimi, I. A. & Kraftb, M., 2017. Design of computer experiments: A review. *Computers & Chemical Engineering,* Volume 106, pp. 71-95.

Gauthier, J. P., Hammouri, H. & Othman, S., 1992. A simple observer for nonlinear systems applications to bioreactors. *IEEE Transactions on Automatic Control,* Volume 37, pp. 875-880.

Gazi, M., Oladipo, A. A., Ojoro, Z. E. & Ozan, H., 2017. High-Performance Nanocatalyst for Adsorptive and Photoassisted Fenton-Like Degradation of Phenol: Modeling Using Artificial Neural Networks. *Chemical Engineering Communications,* Volume 204, pp. 729-738.

Girard, A., Rasmussen, C. E. R. & Murray-Smith, R., 2002. Gaussian Process priors with Uncertain Inputs:Multiple-Step-Ahead Prediction. *Technical Report DCS TR-2002-119, University of Glasgow, Glasgow.*

Göb, S. et al., 1999. Modeling the kinetics of a photochemical water treatment process by means of artificial neural networks. *Chemical Engineering and Processing: Process Intensification*, Volume 38, p. 373–382.

Godarzi, A. A., Amiri, R. M., Talaei, A. & Jamasb, T., 2014. Predicting oil price movements: A dynamic Artificial Neural Network approach. *Energy Policy,* Volume 371-382, p. 68.

Gonzaga, J., Meleiro, L., Kiang, C. & Filhoc, R. M., 2009. ANN-based soft-sensor for real-time process monitoring and control of an industrial polymerization process. *Computers & Chemical Engineering,* Volume 33, p. 43–49.

Grbić, R., Slišković, D. & Kadlec, P., 2013. Adaptive soft sensor for online prediction and process monitoring based on a mixture of Gaussian process models. *Computers and Chemical Engineering,* Volume 58, p. 84–97.

Grossmann, I. E. et al., 2016. Recent advances in mathematical programming techniques for the optimization of process systems under uncertainty. *Computers & Chemical Engineering,* Volume 91, pp. 3-14.

Guimarães, O. L. C. et al., 2007. Prediction via Neural Networks of the Residual Hydrogen Peroxide used in Photo-Fenton Processes for Effluent Treatment. *Chemical Engineering & Technology,* Volume 30, p. 1134–1139.

Gustavsson, R., Lukasser, C. & Mandenius, C.-F., 2015. Control of specific carbon dioxide production in a fed-batch culture producing recombinant protein using a soft sensor. *Journal of Biotechnology,* Volume 200, p. 44–51.

Hale, E. T. & Qin, S. J., 2004. Multi-Parametric Nonlinear Programming and the Evaluation of Implicit Optimization Model Adequacy. *IFAC Proceedings Volumes,* Volume 37, pp. 449-454.

Hassani, A., Khataee, A. & Karaca, S., 2015. Photocatalytic degradation of ciprofloxacin by synthesized TiO2 nanoparticles on montmorillonite: Effect of operation parameters andartificial neural network modeling. *Journal of Molecular Catalysis A: Chemical,* Volume 409, pp. 149-161.

Hauptman, B. & Jovan, V., 2004. An approach to process production reactive scheduling. *ISA Transactions,* Volume 43, pp. 305-318.

Henao, C. & Maravelias, C., 2011. Surrogate-based superstructure optimization framework. *AIChE Journal,* Volume 57, pp. 1216-1232.

Hernandez, A. F. & Grover, M. A., 2010. Stochastic dynamic predictions using Gaussian process models for nanoparticle synthesis. *Computers & Chemical Engineering,* Volume 34, pp. 1953-1961.

Himmelblau, D. M., 2000. Applications of artificial neural networks in chemical engineering. *Korean Journal of Chemical Engineering,* Volume 17, p. 373–392.

Honggui, H., Ying, L. & Junfei, Q., 2014. A fuzzy neural network approach for online fault detection in waste water treatment process. *Computers and Electrical Engineering,* Volume 40, p. 2216–2226.

Hoskins, J. & Himmelblau, D., 1988. Artificial neural network models of knowledge representation in chemical engineering. *Computers & Chemical Engineering,* Volume 12, pp. 881-890.

Ibrahim, M. et al., 2019. Impact of Sampling Technique on the Performance of Surrogate Models Generated with Artificial Neural Network (ANN): A Case Study for a Natural Gas Stabilization Unit. *Energies,* Volume 12, pp. 1-12.

Ichihara, H. & Anai, H., 2012. *A symbolic-numeric approach to multi-parametric programming for control design.* Orlando, proceeding of the 50th IEEE Conference on Decision and Control and European Control Conference, pp. 7152-7157.

Isermann, R., 2005. Model-based fault-detection and diagnosis-status and applications. *Annual Reviews in Control,* Volume 29, p. 71–85.

Jaafarzadeh, N. et al., 2012. Predicting Fenton modification of solid waste vegetable oil industry for arsenic removal using artificial neural networks. *Journal of the Taiwan Institute of Chemical Engineers,* Volume 43, p. 873–878.

Jain, P., Rahman, I. & . Kulkarni, B. D., 2007. Development of a Soft Sensor for a Batch Distillation Column Using Support Vector Regression Techniques. *Chem.Eng.Res.Des. 85,* pp. 283-287.

Jiao, Y., Su, H., Hou, W. & Liao, Z., 2012. Optimization of refinery hydrogen network based on chance constrained programming. *Chemical Engineering Research and Design,* Volume 90, pp. 1553-1567.

Jin, H., Chen, X., Wang, L. & Yang, K., 2015. Adaptive Soft Sensor Development Based on Online Ensemble Gaussian Process Regression for Nonlinear Time-Varying Batch Processes. *Ind. Eng. Chem. Res.,* Volume 54, p. 7320−7345.

Jin, H., Chen, X., Yang, J. & Wu, L., 2014. Adaptive soft sensor modeling framework based on just-in-time learning and kernel partial least squares regression for nonlinear multiphase batch processes. *Computers & Chemical Engineering,* Volume 71, pp. 77-93.

Jones, D. R., 2001. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization,* Volume 21, pp. 345-383.

Jones, D. R., Schonlau, M. & Welch, W. J., 1998. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization,* Volume 13, pp. 455-492.

Joseph, V. R., 2016. Space-filling designs for computer experiments: A review. *Quality Engineering,* Volume 28, pp. 28-35.

Jurecka, F., 2007. *Robust Design OptimizationBased on Metamodeling Techniques.* München: PhD thesis, Technische Universität München.

Kadlec, P., Gabrys, B. & Strandt, S., 2009. Data-driven Soft Sensors in the process industry. *Computers and Chemical Engineering,* Volume 33, p. 795–814.

Kajero, O. et al., 2017. Meta-modelling in chemical process system engineering. *Journal of the Taiwan Institute of Chemical Engineers,* Volume 73, pp. 135-145.

Kaneko, H. & Funatsu, K., 2013. Classification of the Degradation of Soft Sensor Models and Discussion on Adaptive Models. *AIChE Journal,* Volume 59, p. 2339–2347.

Katz, J., Burnak, B. & Pistikopoulos, E. N., 2018 . The impact of model approximation in multiparametric model predictive control. *Chemical Engineering Research and Design,* Volume 139, p. 211–223.

Katz, J., Pappas, I., Avraamidou, S. & Pistikopoulos, E. N., 2020b. *Integrating Deep Learning and Explicit MPC for Advanced Process Control.* Denver, CO, USA, Published in: 2020 American Control Conference, pp. 3559-3564.

Katz, J., Pappas, I., Avraamidou, S. & Pistikopoulos, E. N., 2020. Integrating deep learning models and multiparametric programming. *Computers and Chemical Engineering,* Volumen 136, p. 106801.

Kaufman, L. & Rousseeuw, P. J., 1990. *Finding Groups in Data: An Introduction to Cluster Analysis.* Hoboken, New Jersey: John Wiley & Sons.

Kelly, J. D. & Zyngier, D., 2017. Unit-operation nonlinear modeling for planning and scheduling applications. *Optimization and Engineering,* Volume 18, pp. 133-154.

Kempf, S., Forget, B. & Hu, L.-W., 2012. Kriging-based algorithm for nuclear reactor neutronic design optimization. *Nuclear Engineering and Design,* Volume 247, pp. 248-253.

Khataee, A. et al., 2014. Modeling and optimization of photocatalytic/photoassisted-electro-Fenton like degradation of phenol using a neural network coupled with genetic algorithm. *Journal of Industrial and Engineering Chemistry,* Volume 20, pp. 1852-1860.

Kleijnen, J. P., 2017. Regression and Kriging metamodels with their experimental designs in simulation: A review. *European Journal of Operational Research,* Volume 256, p. 1–16.

Kline, D. M. & Berardi, V. L., 2005. Revisiting squared-error and cross-entropy functions for training neural network classifiers. *Neural Comput & Applic,* Volume 14, p. 310–318.

Kocijan, J., Girard, A., Banko, B. & Murray-Smith, R., 2005. Dynamic systems identification with Gaussian processes. *Mathematical and Computer Modelling of Dynamical Systems,* Volume 11, pp. 411-424 .

Kohavi, R., 1995. *A study of cross-validation and bootstrap for accuracy estimation and model selection.* Montreal, Quebec, Canada, Proceedings of the 14th international joint conference on Artificial intelligence, pp. 1137-1143.

Kouadri, A., Aitouche, M. A. & Zelmat, M., 2012. Variogram-based fault diagnosis in an interconnected tank system. *ISA Transactions,* Volume 51, p. 471–476.

Kouramas, K., Faisca, N. P., Panos, C. & Pistikopoulos, E. N., 2011. Explicit/multi-parametric model predictive control (MPC) of linear discrete-time systems by dynamic and multi-parametric programming. *Automatica,* Volume 47, p. 1638–1645.

Krige, D. A., 1951. statistical approach to some mine valuations and allied problems at the Witwatersrand. *Master's thesis of the University of Witwatersrand.*

Lee, W. J. et al., 2018. NARX modeling for real-time optimization of air and gas compression systems in chemical processes. *Computers & Chemical Engineering,* Volume 115, pp. 262-274.

Leperi, K. T., Yancy-Caballero, D., Snurr, R. Q. & You, F., 2019. 110th Anniversary : Surrogate Models Based on Artificial Neural Networks To Simulate and Optimize Pressure Swing Adsorption Cycles for CO 2 Capture. *Ind. Eng. Chem. Res.,* Volume 58, 39, pp. 18241-18252.

Lin, B., Recke, B., Knudsen, J. . K. & Jørgensen, S. B., 2007. A systematic approach for soft sensor development. *Computers and Chemical Engineering,* Volume 31, p. 419–425.

Li, S. & Li, Y., 2015. Neural network based nonlinear model predictive control for an intensified continuous reactor. *Chemical Engineering and Processing: Process Intensification,* Volume 96, pp. 14-27.

Liu, Y., Gao, Z., Li, P. & Wang, H., 2012. Just-in-Time Kernel Learning with Adaptive Parameter Selection for Soft Sensor Modeling of Batch Processes. *Ind. Eng. Chem. Res.,* Volume 51, p. 4313–4327.

Liu, Y. et al., 2016. Development of multiple-step soft-sensors using a Gaussian process model with application for fault prognosis. *Chemometrics and Intelligent Laboratory Systems,* Volume 157, p. 85–95.

Li, X. & Wang, W., 2020. Learning discriminative features via weights-biased softmax loss. *Pattern Recognition,* Volume 107, p. 107405.

Li, X. et al., 2019. Dual Cross-Entropy Loss for Small-Sample Fine-Grained Vehicle Classification. *IEEE Transactions on Vehicular Technology ,* Volume 68, pp. 4204-4212.

Li, Z., 2010. *PROCESS OPERATIONS WITH UNCERTAINTY AND INTEGRATION CONSIDERATIONS.* New Brunswick, New Jersey: Rutgers, The State University of New Jersey.

Lloyd, S., 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory,* Volume 28, p. 129–137.

Ludermir, T. B., Yamazaki, A. & Zanchettin, C., 2006. An Optimization Methodology for Neural Network Weights and Architectures. *IEEE Transactions on Neural Networks,* Volume 17 , pp. 1452-1459.

Lupera, G., Shokry, A., Campanya, G. & Espuña, A., 2016. Application of the Meta-MultiParametric Methodology to the Control of Emissions in the Industry under Continuous and Discrete Uncertain Parameters. *Chemical Engineering Research and Design,* Volume 155, pp. 365-373.

Lupera, G., Shokry, A., M.Kopanos, G. & Espuña, A., 2018. Mixed-integer multiparametric Metamodeling: A machine learning tool applied to reactive scheduling. *Computer Aided Chemical Engineering ,* Volume 43, pp. 163-168.

Lupera, G. et al., 2018. Ordinary Kriging: A machine learning tool applied to mixed-integer multiparametric approach. *Computer Aided Chemical Engineering,* Volume 43, pp. 531-536.

Luus, R., 1994. Optimal control of batch reactors by iterative dynamic programming. *Journal of Process Control,* Volume 4, pp. 218-226.

Maldonado, S., Weber, R. & Famili, F., 2014. *Feature selection for high-dimensional class-imbalanced data sets using Support Vector Machines.* s.l.:Information Sciences. Volume: 286. Pages 228-246.

Marchetti, A. G., Ferramosca, A. & González, A., 2014. Steady-state target optimization designs for integrating real-time optimization and model predictive control. *Journal of Process Control,* Volume 24, pp. 129-145.

Masters, T., 1993. *Practical Neural Network Recipies in C++.* San Diego New York: Academic Press.

Matheron, G., 1963. Principles of geostatistics. *Economic Geology,* Volume 58, p. 1246–1266.

Matlab, 2018. *Mathworks.* [Online]
Available at: www.mathworks.com

Mattosa, C. L. C. et al., 2017. Deep recurrent Gaussian processes for outlier-robust system identification. *Journal of Process Control,* Volume 60, pp. 82-94.

Meckesheimer, M., Booker, A. J., B. R. & Simpson, T. W., 2002. Computationally Inexpensive Metamodel Assessment Strategies. *AIAA Journal,* Volume 40, pp. 2053-2060.

Medina-González, S., 2019. *A contribution to support decision making in energy/water sypply chain optimization.* Barcelona, Spain: Thesis, Universitat Politècnica de Catalunya .

Medina-González, S. et al., 2020. Optimal management of bio-based energy supply chains under parametric uncertainty through a data-driven decision-support framework. *Computers & Industrial Engineering,* Volume 139, p. 105561.

Mesfin, G. & Shuhaimi, M., 2010. A chance constrained approach for a gas processing plant with uncertain feed conditions. *Computers & Chemical Engineering,* Volume 34, pp. 1256-1267.

Mitchell, M., 1996. *An Introduction to Genetic Algorithms.* Cambridge, MA: MIT Press.

Monroy, I., Benitez, R., Escudero, G. & Graells, M., 2010. A semi-supervised approach to fault diagnosis for chemical processes. *Computers & Chemical Engineering,* Volume 34, p. 631–642..

Moreno-Benito, M., 2014. *Integrated Batch Process Development based on Mixed-Logic Dynamic Optimization.* Barcelona, Spain: PhD thesis.

Mota, A. L. N. et al., 2014. Application of artificial neural network for modeling of phenol mineralization by photo-Fenton process using a multi-lamp reactor. *Water Sci Technol.,* Issue 69, pp. 768-774.

Muller, D. et al., 2017. Real-Time Optimization in the Chemical Processing Industry. *Chem. Ing. Tech.,* Volume 89, pp. 1464-1470.

Murray-Smith, R., Sbarbaro, D., Rasmussen, C. E. & Girard, A., 2003. Adaptive, cautious, predictive control with Gaussian process priors. *IFAC Proceedings Volumes,* Volume 36, pp. 1155-1160.

Mustafa, Y. A., Jaid, G. M., Alwared, A. I. & Ebrahim, M., 2014. The use of artificial neural network (ANN) for the prediction and simulation of oil degradation in wastewater by AOP. *Environ Sci Pollut Res,* Volume 21, p. 7530–7537.

Nagy, Z. K., 2007. Model based control of a yeast fermentation bioreactor using optimally designed artificial neural networks. *Chemical Engineering Journal,* Volume 127, pp. 95-109.

Narasimhan, S., Vachhani, P. & Rengaswamy, R., 2008. New nonlinear residual feedback observer for fault diagnosis in nonlinear systems. *Automatica,* Volume 44, p. 2222–2229.

Nascimento, C. A. D., Oliveros, E. & Braun, A. M., 1994. Neural network modelling for photochemical processes. *Chemical Engineering and Processing: Process Intensification,* Volume 33, pp. 319-324.

Nelles, O., 2001. *Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models.* Berlin: Springer.

Norbert, A. et al., 2017. Simulation and Multi-criteria Optimization under Uncertain Model Parameters of a Cumene Process. *Chemie Ingenieur Technik,* Volume 89, pp. 665-674.

O'Hagan, M. C. K. a. A., 2001. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology),* Volume 63, p. 425–464.

Ochoa-Estopier, L. et al., 2018. Industrial Application of Surrogate Models to Optimize Crude Oil Distillation Units. *CHEMICAL ENGINEERING TRANSACTIONS,* Volume 69, pp. 289-294.

Ochoa-Estopier, L. & Jobson, M., 2015. Optimization of Heat-Integrated Crude Oil Distillation Systems. Part I: The Distillation Model. *Ind. Eng. Chem. Res.,* Volume 54, pp. 4988-5000.

O'Hagan, A., Kennedy, M. C. & Oakley, J. E., 1999. Uncertainty analysis and other inference tools for complex computer codes. *In Bayesian Statistics 6, (J. M. Bernardo et al (eds.)), Oxford University Press,* pp. 503-524.

O'Hagan, A. & Kingman, J. F. C., 1978. Curve Fitting and Optimal Design for Prediction. *Journal of the Royal Statistical Society. Series B (Methodological) ,* Volume 40, pp. 1-42 .

Özyurt, B. et al., 1998. Chemical plant fault diagnosis through a hybrid symbolic-connectionist machine learning approach. *Computers & Chemical Engineering,* Volume 22, p. 299–321.

Palmer, K. & Realef, M., 2002. OPTIMIZATION AND VALIDATION OF STEADY-STATE FLOWSHEET SIMULATION METAMODELS. *Trans IChemE,* Volume 80, pp. 773-782.

Panapakidis, I. P. & Dagoumas, A. S., 2016. Day-ahead electricity price forecasting via the application of artificial neural network based models. *Applied Energy,* Volume 172, pp. 132-151.

Papageorgiou, L. & Fraga, E., 2007. A mixed integer quadratic programming formulation for the economic dispatch of generators with prohibited operating zones. *Electric Power Systems Research,* Volume 77, p. 1292–1296.

Papathanasiou, M. M. y otros, 2019. Control of a dual mode separation process via multi-parametric Model Predictive Control. *IFAC PapersOnLine,* Volumen 52, pp. 988-993.

Park, Y.-J., Fan, S.-K. S. & Hsu, C.-Y., 2020. A Review on Fault Detection and Process Diagnostics in Industrial Processes. *processes,* Volume 8, pp. 1123,1-26.

Parr, J. M., Holden, C., Forrester, A. I. J. & Keane, A. J., 2010. *Review of Efficient Surrogate Infill Sampling Criteria With Constraint Handling.* Lisbon, Portugal, 2nd International Conference on Engineering Optimization.

Parr, J. M., Keane, A. J., Forrester, A. I. & Holden, C. M., 2012. Infill sampling criteria for surrogate-based optimization with constraint handling. *Engineering Optimization,* Volume 44, pp. 1147-1166.

Patan, K. & Parisini, T., 2005. Identification of neural dynamic models for fault detection and isolation: the case of a real sugar evaporation process. *Journal of Process Control,* Volume 15, p. 67–79.

Patton, R., Chen, J. & Siew, T., 1994. Fault diagnosis in nonlinear dynamic systems via neural networks. *Proc. of IEE Int. Conf. on Control,* pp. 1346-1351.

Patton, R. J., J.Chen & Nielsen, S. B., 1995. Model-Based Methods for Fault Diagnosis: Some Guide-Lines. *Transactions of the Institute of Measurement and Control,* Volume 17, pp. 73-83.

Pistikopoulos, E., 1995. Uncertainty in process design and operations. *Computers & Chemical Engineering,* Volume 19, pp. 553-563.

Pistikopoulos, E. N., 2008. Perspectives in Multiparametric Programming and Explicit Model Predictive Control. *AIChE Journal,* Volume 55, pp. 1918-1925.

Pistikopoulos, E. N. et al., 2002. On-line optimization via off-line parametric optimization tools. *Computers and Chemical Engineering,* Volume 26, pp. 175-185.

Pistikopoulos, E. N., Georgiadis, M. C. & Dua, V., 2007. *Multi-Parametric Programming: Volume 1: Theory, Algorithms, and Applications.* Weinheim: WILEY-VCH Verlag GmbH & Co.

Platt, J., 1999. *Fast training of support vector machines using sequential minimal optimization.* Cambridge: In: Scholkopf, B., et al. (Eds.), Advances in Kernel Methods - Support Vector LEarning, MIT Press.

Poznyak, A., Chairez, I. & Poznyak, T., 2019. A survey on artificial neural networks application for identification and control in environmental engineering: Biological and chemical systems with uncertain models. *Annual Reviews in Control (in press https://doi.org/10.1016/j.arcontrol.2019.07.003).*

Qian, J. et al., 2019. A sequential constraints updating approach for Kriging surrogate model-assisted engineering optimization design problem. *Engineering with Computers,* pp. https://doi.org/10.1007/s00366-019-00745-w.

Qin, S. J., 2012. Survey on data-driven industrial process monitoring and diagnosis. *Annual Reviews in Control,* Volume 36, pp. 220-234.

Queipo, N. V. et al., 2005. Surrogate-based analysis and optimization. *Progress in Aerospace Sciences,* Volume 41, pp. 1-28.

Quirante, N. & Caballero, J., 2016. Large scale optimization of a sour water stripping plant usingsurrogate models. *Computers and Chemical Engineering,* Volume 92 , pp. 143-162.

Quirante, N., Javaloyes-Antón, J. & Caballero, J. A., 2018. Hybrid simulation-equation based synthesis ofchemical processes. *Chemical Engineering Research and Design,* Volume 132, pp. 766-784.

Rasmussen, C. E. & Deisenroth, M. P., 2008. *Probabilistic Inference for Fast Learning in Control.* 229-242, Recent Advances in Reinforcement Learning , pp. 229-242.

Rasmussen, C. E. & Williams, C. K. I., 2006. *Gaussian Processes for Machine Learning. Adaptive Computation and Machine Learning series.* Cambridge, Massachusetts: MIT Press.

Regis, R., 2016. Trust regions in Kriging-based optimization with expected improvement. *Engineering Optimization,* Volume 48, pp. 1037-1059.

Rehman, S. u. & Langelaar, M., 2017. Expected improvement based infill sampling for global robust optimization of constrained problems. *Optimization and Engineering,* Volume 18, p. 723–753.

Reina, A. C. et al., 2012. Modelling photo-Fenton process for organic matter mineralization, hydrogen peroxide consumption and dissolved oxygen evolution. *Applied Catalysis B: Environmental,* Volume 119–120, p. 132–138.

Rigamonti, M. et al., 2018. Ensemble of optimized echo state networks for remaining useful life prediction. *Neurocomputing,* Volume 281, pp. 121-138.

Rios, L. M. & Sahinidis, N. V., 2013. Derivative-free optimization: a review of algorithmsand comparison of software implementations. *J Glob Optim,* Volume 56, p. 1247–1293.

Rivotti, P., Lambert, R. S. C. & Pistikopoulos, E. N., 2012. Combined model approximation techniques and multiparametric programming for explicit nonlinear model predictive control. *Computers & Chemical Engineering ,* Volume 42, pp. 277-287.

Rocco, C. M. & Zio, E., 2007 . A support vector machine integrated system for the classification of operation anomalies in nuclear components and systems. *Reliability Engineering & System Safety,* Volume 92, pp. 593-600.

Roffel, B. & Betlem, B., 2004. *Advanced Practical Process Control.* Berlin: Springer-Verlag Berlin Heidelberg.

Ruppen, D., Benthack, C. & Bonvin, D., 1995. Optimization of batch reactor operation under parametric uncertainty - computational aspects. *Journal of Process Control,* Volume 5, pp. 235-240.

Sacks, J., Welch, W. J., Mitchell, T. J. & Wynn, H. P., 1989. Design and Analysis of Computer Experiments. *Statistical Science ,* Volume 4, pp. 409-423 .

Sadeghassadi, M., Macnab, C. J. B., Gopaluni, B. & Westwick, D., 2018. Application of neural networks for optimal-setpoint design and MPC control in biological wastewater treatment. *Computers & Chemical Engineering,* Volume 115, pp. 150-160.

Salari, D., Daneshvar, N., Aghazadeh, F. & Khataee, A., 2005. Application of artificial neural networks for modeling of the treatment of wastewater contaminated with methyltert-butyl ether (MTBE) by UV/H2O2 process. *Journal of Hazardous Materials B,* pp. 205-210.

Salback, D. D., 2004. *A SURROGATE MODEL APPROACH TO REFINERY-WIDE OPTIMIZATION. PhD thesis.* Texas: Texas Tech University.

Sarailo, M., Rahmani, Z. & Rezaie, B., 2015. A novel model predictive control scheme based on bees algorithm in A class of nonlinear systems: application to A three tank system. *Neurocomputing,* Volume 152, p. 294–304.

Schmitt, L., 2001. Theory of genetic algorithms. *Theoretical Computer Science,* Volume 259, pp. 1-61.

Schonlau, M., Welch, W. J. & Jones, D. R., 1998. Global Versus Local Search in Constrained Optimization of Computer Models. *Lecture Notes-Monograph Series,* Volume 34 , pp. 11-25.

Seborg, D. E., Edgar, T. F., Mellichamp, D. A. & Doyle, F. J., 2016. *Process Dynamics and Control.* 4th ed. USA: John Wiley & Sons.

Sebti, A., Souahi, F., Mohellebi, F. & Igoud, S., 2017. Experimental study and artificial neural network modeling of tartrazine removal by photo-catalytic process under solar light. *Water Science & Technology,* Volume 76, pp. 311-322.

Serdio, F. et al., 2014. Residual-based fault detection using soft computing techniques for condition monitoring at rolling mills. *Information Sciences,* Volume 259, p. 304–320.

Shao, G., Hasan, L., Martin-Villalba, C. & Denno, P., 2019. Standards-based integration of advanced process control and optimization. *Journal of Industrial Information Integration,* Volume 13, pp. 1-12.

Slowik, A., 2020. *Swarm Intelligence Algorithms: Modifications and Applications.* Boca Raton, Florida: CRC Press.

Smarsly, K. & Petryna, Y., 2014. A Decentralized Approach towards Autonomous Fault Detection in Wireless Structural Health Monitoring Systems. 7th European Workshop on Structural Health Monitoring. *7th European Workshop on Structural Health Monitoring-EWSHM,* July.

Srinivasana, B., Palankib, S. & Bonvin, D., 2003. Dynamic optimization of batch processes: I. Characterization of the nominal solution. *Computers and Chemical Engineering,* Volume 27, pp. 1-26.

Suykens, J. A., Vandewalle, J. & De Moor, B. L., 1996. *Artificial Neural Networks for Modelling and Control of Non-Linear Systems.* 1 ed. s.l.:Springer US.

Tao, T., Zio, E. & Zhao, W., 2018. A novel support vector regression method for online reliability prediction under multi-state varying operating conditions. *Reliability Engineering & System Safety,* Volume 35-49, p. 177.

Tayarani-B., S. S. & Khorasani, K., 2015. Fault detection and isolation of gas turbine engines using a bank ofneural networks. *Journal of Process Control,* Volume 36, p. 22–41.

Tenny, M. J. & Rawlings, J. B., 2004. Closed-Loop Behavior of Nonlinear ModelPredictive Control. *AIChE Journal,* Volume 50, pp. 2142-2154.

Theilliol, D., Noura, H. & Ponsart, J.-C., 2002. Fault diagnosis and accommodation of a three-tank system based on analytical redundancy. *ISA Transactions,* Volume 41, pp. 365-382.

Tian, N., Fang, H. & Wang, Y., 2020. Real-Time Optimal Lithium-Ion Battery Charging Based on Explicit Model Predictive Control. *IEEE Transactions on Industrial Informatic,* Volume in press.

Tieu, D., Cluett, W. R. & Penlidis, A., 1995. A Comparison of Collocation Methods for Solving Dynamic Optimization Problems. *COMPUT CHEM ENG,* Volume 19, pp. 375-381.

Tsai, C.-S. & Chang, C.-T., 1995. Dynamic process diagnosis via integrated neural networks. *Computers & Chemical Engineering,* Volume 19, pp. 747-752.

Vaccari, M. & Pannocchia, G., 2017. A Modifier-Adaptation Strategy towards Offset-Free Economic MPC. *Processes,* Volume 5, pp. 1-22.

Vapnik, V., 1995. *The Nature of Statistical Learning Theory.* New York: Springer-Verlag.

Venkatasubramanian, V., Rengaswamy, R., Kavuri, S. N. & Yin, K., 2003b. A review of process fault detection and diagnosis Part III: Process history based methods. *Computers and Chemical Engineering,* Volume 27, pp. 327-346.

Venkatasubramanian, V., Rengaswamy, R., Yin, K. & Kavuri, S. N., 2003a. A review of process fault detection and diagnosis Part I: Quantitative model-based methods. *Computers and Chemical Engineering*, Volume 27, pp. 293-311.

Wang, L. et al., 2016. Soft Sensor Development Based on the Hierarchical Ensemble of Gaussian Process Regression Models for Nonlinear and Non-Gaussian Chemical Processes.. *Ind. Eng. Chem. Res.*, Volume 55, pp. 7704-7719.

Wang, L., Liu, X. & Zhang, Z., 2017. A new sensitivity-based adaptive control vector parameterization approach for dynamic optimization of bioprocesses. *Bioprocess Biosyst Eng*, Volume 40, p. 181–189.

Wang, S. et al., 2019. A Surrogate-Assisted Approach for the Optimal Synthesis of Refinery Hydrogen Networks. *Ind. Eng. Chem. Res.*, Volume 58, pp. 16798-16812.

Wang, Z. & Ierapetritou, M., 2017. A Novel Surrogate-Based Optimization Method for Black-Box Simulation with Heteroscedastic Noise. *Ind. Eng. Chem. Res.*, Volume 56, pp. 10720-10732.

Wen, C. & Yen, T., 1977. Optimization of oil shale pyrolysis. *Chemical Engineering Science*, Volume 32, pp. 346-349.

Wright, S., 1996. *Primal-Dual Interior-Point Methods.* Philadelphia, USA: Society for Industrial and AppliedMathematics.

Xu, D., Jiang, B. & Shi, P., 2014. Adaptive Observer Based Data-Driven Control for Nonlinear Discrete-Time Processes. *IEEE Transactions on Automation Science and Engineering*, Volumen 11, pp. 1037-1045.

Xu, G., Zhou, H. & Chen, J., 2018. CNC internal data based incremental cost-sensitive support vector machine method for tool breakage monitoring in end milling. *Engineering Applications of Artificial Intelligence*, Volume 74, pp. 90-103.

Yamal-Turbay, E., E., O., O., C. L. & M., G., 2015. Photonic efficiency of the photodegradation of paracetamol in water by the photo-Fenton process. *Environ Sci Pollut Res*, Volume 22, p. 938–945.

Yan, W., Shao, H. & Wang, X., 2004. Soft sensing modeling based on support vector machine and Bayesian model selection. *Computers and Chemical Engineering*, Volume 28, p. 1489–1498.

Zamprogna, E., Barolo, M. & Seborg, D. E., 2005. Optimal Selection of Soft Sensor Inputs For Batch Distillation Columns Using Principal Component Analysis. *Journal of Process Control*, Volume 15, p. 39–52.

Zhang, P. G., 2000. Neural Networks for Classification: A Survey. *IEEE Transactions on Systems Man and Cybernetics Part C ,* Volume 30, pp. 451- 462.

Zhang, Y., Han, Z.-H. & Zhang, K.-S., 2018. Variable-fidelity expected improvement method for efficient global optimization of expensive functions. *Structural and Multidisciplinary Optimization*, Volume 58, p. 2018.

Zhou, L., Chen, J. & Song, Z., 2015. Recursive Gaussian Process Regression Model for Adaptive Quality Monitoring in Batch Processes. *Mathematical Problems in Engineering*, Volume 2015, pp. 1-9.

Zuhal, L. R., Palar, P. S. & Shimoyama, K., 2019. A comparative study of multi-objective expected improvement for aerodynamic design. *Aerospace Science and Technology*, Volume 91, pp. 548-560.

List of Abbreviations and Acronyms

AOPs	Advanced Oxidation Process
CC	Correlation Coefficient
CEI	Constrained Expected Improvement
CTs	Classification Techniques
CVP	Control Vector Parameterization
DT	Decision Tree
DOE	Design Of Experiments
DBMP-MPC	Data-Based MultiParametric-Model Predictive Control
EGO	Efficient Global Optimization
EI	Expected Improvement
ESO	Ensemble of Single-Output
FDD	Fault Detection and Diagnosis
FN	False Negative
FP	False Positive
FPMs	First Principle Models
GA	Genetic Algorithm
GNBs	Gaussian Naïve Bayes classifiers
GP	Gaussian Process
GPR	GP-Regression
GT	Gas Turbine
LOOCV	Leave-One-Out Cross-Validation
LP	Linear Programming
MILP	Mixed-Integer Linear Programming
MINLP	Mixed-Integer NonLinear Programming
ML	Machine Learning
MO	Multi-Output
MPC	Model-Predictive Control
MPMs	MultiParametric Metamodels
MP-MPC	MultiParametric- Model-Predictive Control
MPP	Multi-Parametric Programming
MSE	Mean Squared Error
MVDK	MultiVariate Dynamic Kriging
NAR	Nonlinear AutoRegressive
NARX	Nonlinear AutoRegressive eXogenous
NLP	NonLinear Programming
NRMSE	Normalized Room Mean Square Error
OK	Ordinary Kriging
PCR	Principal Component Regression
PI	Probability of Improvement
PLS	Partial Least-Squares
QIV	Quality Indicator Variables
QP	Quadratic Programming
RMSE	Root Mean Square Error

RTO	Real Time Optimization
SBO	Surrogate Based Optimization
SLHS	Space-filling Latin Hypercube Sampling
SOA	State-Of-Art
SQP	Sequential Quadratic Programming
SSE	Sum of Squared Error
SVMs	Support Vector Machines
SVRs	Support Vector Regressions
TOC	Total Organic Carbon
TP	True Positive
TN	True Negative
UPs	Uncertain Parameters