UNIVERSITÀ DEGLI STUDI DI PADOVA
DIPARTIMENTO DI INGEGNERIA INDUSTRIALE
CORSO DI LAUREA MAGISTRALE IN INGEGNERIA CHIMICA E DEI PROCESSI INDUSTRIALI

Tesi di Laurea Magistrale in
Ingegneria Chimica e dei Processi Industriali

STEADY-STATE DETECTION, DATA RECONCILIATION AND MACHINE LEARNING FOR HYBRID PROCESS MODELLING

Relatore: Prof. Fabrizio Bezzo
Correlatore: Dr. Maarten Nauta

Laureando: MARCO BASSETTO

ANNO ACCADEMICO 2018 – 2019

A mamma e papà

Al nonno Giuseppe

Abstract

In the process industry, it is possible to encounter systems whose behaviour cannot be mapped through a first principles (white-box) model. Hybrid models aim at integrating data-driven (black-box) elements within white-box process models in order to fill the gap between the white-model model predictions and the actual system response. The goal of this Thesis is to propose and implement a hybrid modelling framework, and to assess its performance with respect to a white-box model. The industrial process for the manufacturing of cumene is taken into account as a case study. Under the assumption that it is not possible to model the separation section following a white-box approach, a steady-state hybrid model of the plant is developed and implemented in the gPROMS process simulator. In the hybrid model, the behaviour of the train of distillation columns is entirely mapped with data-driven elements, while the rest of the equipment is modelled via first principles. The set of steady-state operating points on which the black-box elements are calibrated is derived by performing data reconciliation and steady-state detection on a simulated plant historian. In particular, the identification of steady-state conditions is carried out through a novel steady-state detection algorithm developed in this Thesis. The performance of the hybrid model is tested against that of a fully first principle model considering four steady-state operating points, which were not included in the black-box training dataset. Results demonstrate the hybrid model capability to match the predictions of the first principle model accurately.

Riassunto

Il lavoro di Tesi è stato condotto presso l'azienda Process Systems Enterprise a Londra, Regno Unito, nell'ambito di un tirocinio della durata complessiva di sei mesi.

L'industria chimica fa ampio affidamento sui simulatori di processo per la progettazione, l'ottimizzazione e il controllo operativo degli impianti. I simulatori di processo monitorano la produttività degli impianti sfruttando dei modelli matematici per descrivere il comportamento delle singole unità di processo. Esistono tre principali categorie di modelli: i modelli meccanicistici (o a principi primi), i modelli ibridi e i modelli a scatola nera. I modelli meccanicistici si basano sui principi fondamentali della fisica e della chimica mentre i modelli a scatola nera si affidano esclusivamente ai dati di processo con i quali vengono calibrati. I modelli ibridi combinano al loro interno sia modelli a principi primi che modelli a scatola nera con lo scopo di trarre il massimo vantaggio dai punti di forza di entrambi.
I modelli meccanicistici richiedono solitamente un ingente mole di lavoro per essere sviluppati e sono computazionalmente onerosi da risolvere, in particolare in caso di applicazioni in tempo reale. I modelli a scatola nera, al contrario, sono poco onerosi dal punto di vista computazionale, ma non sono in grado di fornire previsioni affidabili al di fuori del range dei dati con la quale sono stati calibrati. Per questo motivo, tipicamente, si preferisce descrivere il comportamento delle unità di processo utilizzando dei modelli a principi primi.
Quando però, all'interno di un impianto, vi sono apparecchiature il cui funzionamento non può essere descritto accuratamente tramite bilanci di massa, energia e quantità di moto, si deve adottare una soluzione alternativa ai modelli a meccanicistici. Se sono disponibili dati di processo, spesso l'opzione migliore consiste nell'integrare, all'interno del modello meccanicistico dell'impianto, delle correlazioni empiriche che descrivano il funzionamento delle apparecchiature difficilmente modellabili attraverso i principi primi.
Lo scopo del seguente lavoro di Tesi è dimostrare che i modelli a principi primi e i modelli a scatola nera possono essere combinati con successo, generando un modello ibrido di processo affidabile ed in grado di fornire previsioni accurate. Per portare a termine l'obbiettivo è stato preso in considerazione il processo industriale per la produzione di cumene. Ipotizzando di non avere a disposizione un modello a principi primi capace di descrivere il comportamento delle colonne di distillazione, è stato sviluppato un modello ibrido dell'impianto. Lo sviluppo della Tesi si è articolato in cinque fasi.
Nella prima fase sono stati generati dei dati di processo virtuali utilizzando il simulatore di processo gPROMS per rappresentare in modalità dinamica un modello meccanicistico dell'intero impianto. In seguito, il set di dati simulati è stato corrotto con rumore e misurazioni errate al fine di renderlo il più possibile simile ad un vero storico d'impianto. Nella seconda

fase è stato sviluppato un algoritmo per individuare, all'interno dello storico di impianto virtuale, i punti operativi di stato stazionario. Nella terza fase, le misurazioni di portata e frazioni molari relative alla sezione di separazione dell'impianto sono state riconciliate imponendo il rispetto dei bilanci di conservazione della massa. Nella quarta fase sono state generate delle correlazioni empiriche per descrivere il comportamento delle colonne di distillazione. Infine, nell'ultima fase del progetto, le correlazioni empiriche sono state combinate con i modelli a principi primi delle altre unità di processo, generando un modello ibrido dell'impianto. Il modello ibrido ha dimostrato di avere le stesse capacità predittive del modello meccanicistico dell'intero impianto.

Contents

Notation

a_t = white noise sequence

A = total number of latent variables

$AICc(S, \beta)$ = corrected Akaike information criteria

b_a = regression coefficient of the r^{th} latent variable

B = full set of basis functions of the ALAMO model

$\mathrm{cov}(\mathbf{X})$ = covariance matrix of **X**

$e_{i,j}$ = element of row i and column j of the residual matrix **E**

$\mathbf{ex\_var}_j$ = vector of the percentages of variance retained by each PC

$\mathbf{ex\_var}_j{}'$ = normalized vector of the percentages of variance retained by each PC

E_i = Activation energy of reaction i

E = residual matrix of **X**

$f(x)$ = cost function

F_{D1} = overall molar flowrate of the distillate of the first distillation column

F_{R1} = overall molar flowrate of the residue of the first distillation column

F_{F1} = overall molar flowrate of the feed of the first distillation column

F_{D2} = overall molar flowrate of the distillate of the second distillation column

F_{R2} = overall molar flowrate of the residue of the second distillation column

F_{F2} = overall molar flowrate of the feed of the second distillation column

$g(x)$ = set of constraints

$HC1$ = heat duty of the condenser of the first distillation column

$HR1$ = heat duty of the reboiler of the first distillation column

$HC2$ = heat duty of the condenser of the second distillation column

$HR2$ = heat duty of the reboiler of the second distillation column

I = number of samples of **X**

J = number of process variables of **X**

k = parameter used to modulate the noise amplitude

K_i = pre-exponential factor of reaction i

m = slope of the regression line that fits the trajectory of the process signal x_t

$\mathbf{M}_r$ = matrix of rank 1 of the r^{th} latent variable

n = number of sampled values included in the data window

N = total number of training points employed to build the ALAMO model

obj_{fun} = objective function

$\mathbf{p}_a$ = loading of the a^{th} latent variable

$\mathbf{p}_i$ = i^{th} loading vector

$\mathbf{p}_k$ = composite steady-state index

$\mathbf{p}_r$ = loading of the matrix of rank 1 of the r^{th} latent variable

$\mathbf{p}_R$ = loading of the matrix of rank 1 of the R^{th} latent variable

P_{D1} = pressure of the distillate of the first distillation column

P_{R1} = pressure of the residue of the first distillation column

P_{F1} = pressure of the feed of the first distillation column

P_{D2} = pressure of the distillate of the second distillation column

P_{R2} = pressure of the residue of the second distillation column

P_{F2} = pressure of the feed of the second distillation column

$\mathbf{P}$ = loading matrix of $\mathbf{X}$

$\mathbf{P}_a$ = loading matrix referring to the first a latent variables

$\mathbf{q}_a$ = loading vector of the a^{th} latent variable of $\mathbf{Y}$

$\mathbf{q}_r$ = loading vector of the r^{th} latent variable of $\mathbf{Y}$

Q^2 = predictive relevance

$\mathbf{Q}$ = loading matrix of $\mathbf{Y}$

r_1 = reflux ratio of the first distillation column

r_2 = reflux ratio of the second distillation column

r_i = rate of reaction i

R = universal gas constant

R^2 = determination coefficient

$\mathbf{R}_{kj}$ = boolean matrix storing the steadiness predictions for each PC

$\mathbf{s}_k$ = boolean time series storing the steadiness predictions for the whole process

S = subset of basis functions of the ALAMO model

$SF_{C_9H_{12},C1}$ = cumene split fraction in the first distillation column

$SF_{C_6H_6,C1}$ = benzene split fraction in the first distillation column

$SF_{C_9H_{12},C2}$ = cumene split fraction in the second distillation column

$SF_{C_{12}H_{18},C2}$ = *p*-diisopropyllbenzene split fraction in the second distillation column

t = relative time running within the window

t_{crit} = critical value of the Student's t-test

$\mathbf{t}_a$ = score vector of the a^{th} latent variable of **X**

$\mathbf{t}_i$ = i^{th} score vector of **X**

$\mathbf{t}_k$ = uniform time grid

$\mathbf{t}_r$ = score vector of the matrix of rank 1 of the r^{th} latent variable

$\mathbf{t}_R$ = score vector of the matrix of rank 1 of the R^{th} latent variable

T = indicator for the complexity of the ALAMO model

T_{D1} = temperature of the distillate of the first distillation column

T_{R1} = temperature of the residue of the first distillation column

T_{F1} = temperature of the feed of the first distillation column

T_{D2} = temperature of the distillate of the second distillation column

T_{R2} = temperature of the residue of the second distillation column

T_{F2} = temperature of the feed of the second distillation column

Tr = threshold value for the assessment of the process steadiness

Tr_1 = first threshold value of the novel SSD algorithm

Tr_2 = second threshold value of the novel SSD algorithm

T = score matrix of **X**

$\mathbf{u}_a$ = score vector of the a^{th} latent variable of **Y**

$\mathbf{u}_r$ = score vector of the r^{th} latent variable of **Y**

U = score matrix of **Y**

v = minimum percentage of variance explained by the retained PCs

$\mathbf{w}_r$ = weight of the r^{th} latent variable

W = matrix of the weights

x = degree of freedom

x_i = process variable trajectory

$x_{i,noised}$ = noised process variable trajectory

$x_{i,D1}$ = component molar fractions in the distillate of the first distillation column

$x_{i,R1}$ = component molar fractions in the residue of the first distillation column

$x_{i,F1}$ = component molar fractions in the feed of the first distillation column

$x_{i,D2}$ = component molar fractions in the distillate of the second distillation column

$x_{i,R2}$ = component molar fractions in the residue of the second distillation column

$x_{i,F2}$ = component molar fractions in the feed of the second distillation column

x_i^{calc} = value of the variable calculated by the model in gPROMS

x_i^{meas} = value of the variable measured by the sensor

x_{id} = set of input data used to develop the ALAMO model

$x_{i,noised}$ = noised process variable trajectory

$x_{i,j}$ = element of row *i* and column *j* of the matrix **X**

$\bar{x}_j$ = average of the j^{th} variable

x_t = process signal

$X_j(x)$ = basis functions

$\mathbf{X}$ = process data matrix of dimensions $(I \times J)$

$\widehat{\mathbf{X}}$ = projection of the matrix **X** onto the space of the latent variables

$\mathbf{y}$ = column of the matrix **Y**

$\mathbf{Y}$ = matrix of the quality variables

$\mathbf{Y}_{ki}$ = matrix of n measured variables

z_{ik} = set of responses used to develop the ALAMO model

$\hat{z}(x)$ = model developed through ALAMO

Apex

T = transpose of the matrix or of the vector

Greek letters

α = significance level of the Student's t-test

β_j = ALAMO model parameters

θ = total number of sampling instants

λ = eigenvalue of the covariance matrix **X**

λ_a = a^{th} eigenvalue of the covariance matrix **X**

λ_i = i^{th} eigenvalue of the covariance matrix **X**

μ = intercept of the regression line that fits the trajectory of the process signal x_t

σ = standard deviation

σ_a = standard deviation of the white noise shocks

σ_i = standard deviation of the measured value estimated by the SSD algorithm

σ_j = standard deviation of the j^{th} variable

$\Phi(S, \beta)$ = ALAMO model goodness of fit

τ	=	fraction of time within the window in which the process is deemed to be at steady-state
τ_p	=	process time constant

Acronyms

ANN	=	artificial neural network
BB	=	black-box
BDF	=	backward differentiation formulae
DAE	=	differential and algebraic equations
DCS	=	distributed control system
EM	=	empirical model
EMS	=	error maximization sampling
IMC	=	internal model control
I/O	=	input-output
LDPE	=	low density polyethylene
MILP	=	mixed-integer linear problem
NILES	=	nonlinear iterative least squares
NIPALS	=	nonlinear iterative partial least squares
PC	=	principal component
PCA	=	principal component analysis
PLS	=	partial least squares
PRESS	=	predicted residual error sum of squares
VLE	=	vapor-liquid equilibria
RMSECV	=	root-mean-square error of cross validation
RSS	=	residual sum of squares
SSD	=	steady-state detection
TSS	=	total sum of squares
VLE	=	vapor-liquid equilibria
WB	=	white-box

Introduction

Nowadays the chemical industry relies extensively on process models for plant design, control and performance assessment. Process models consist on a set of equations that allow to describe how the output process variables are influenced by the inputs. The inputs commonly fall into two categories: the variables that can be manipulated (for instance the operating conditions or the design decisions) and the variables that cannot be changed arbitrarily (for instance the market prices or the atmospheric conditions).

Process models are typically divided into three classes:

1. The first principles (or white-box) models;
2. The hybrid (or grey-box) models;
3. The data-driven (or black-box) models.

This classification is based on the extent of model reliability on process knowledge, input-output (I/O) data structure or a combination of both.

White-box models rely strongly on process mechanism, providing a deep understanding of the behaviour of the system which is under investigation. Through the first principles models, furthermore, the effect of the inputs on the output process variables is analysed extensively from the physical and chemical point of view. The black-box models, on the other hand, map the process behaviour exploiting its I/O data structure. The white-box models can be developed even before the start-up of the process and include extrapolation capabilities. The black-box models, instead, despite ensuring a higher computational speed, can be built only after process data are available and their performances are restricted to the range of data they have been calibrated on. The grey-box (or hybrid) models combine the white-box and black-box approaches with the objective to take advantages from the strength of both.

The aim of this Thesis is to demonstrate the potential of hybrid modelling. Taking into account the industrial process for the production of cumene as a case study, a methodology for the hybrid models development was proposed. What is expected is to prove that data-driven elements can be successfully integrated within white-box process models to make up for the presence of poorly understood systems whose behaviour cannot be mapped through first principles.

The Thesis is organized as follows.

In the first Chapter, an overview of the mathematical and statistical background of the methods adopted in this Thesis is provided. In particular, the principal component analysis, the partial least squares regression and the ALAMO model building methodology are presented.

The second Chapter deals with the issue of the identification of steady-state operating points within industrial plant historians. Firstly, the state-of-the art scientific literature concerning the

approaches to the detection of the steady-state is reviewed. Then, the novel algorithm developed in this Thesis is presented.

The third Chapter describes the industrial process that was taken under investigation (i.e. the process for the production of cumene). The flowsheet and the control strategy are discussed and the technique through which process data were collected is described.

Finally, in the fourth and last Chapter the hybrid model of the industrial plant for the production of cumene is presented. Firstly, the procedure through which the grey-box model was developed is reviewed. Then, the model implementation in the process simulator gPROMS is described. At last, the results of the tests carried out to evaluate the model predictive capabilities are discussed.

Some final remarks conclude the Thesis.

Chapter 1

Motivation and mathematical background

This Chapter overviews the fundamentals of hybrid modelling, addresses the objectives and the motivations of the Thesis and presents the mathematical and statistical techniques that have been adopted for the hybrid model development. First, a background on hybrid modelling is given. Second, the possible structures of the hybrid models are described. Third, a literature review of the hybrid model applications in the chemical industry is provided. Fourth, the motivations and the objectives of the project are presented. Then, details about the multivariate statistical techniques and about the model building methodology ALAMO are discussed and presented. In particular, an insight on the principal component analysis and on the projection on latent structures is provided. At last, a comprehensive description of the theory and the algorithms behind the ALAMO model building approach is given.

1.1 Hybrid modelling

As can be seen in Figure 1.1, the process models are developed to describe the influence of the input variables on the plant performances. Examples of output variables that are commonly monitored and controlled are the product flowrate, the product purity and the total energy consumption. The inputs, instead, commonly fall into two different categories. The first category includes those variables that can be manipulated; for instance, the equipment sizes or the operating conditions. The equipment sizes are specified before the start-up of the process while the operating conditions are continuously reassigned throughout the entire life span of the plant to compensate for the disturbances, to match the production goals and to meet the safety and environmental regulations. The second inputs category includes those variables that cannot be controlled; for instance, the atmospheric conditions or the prices of the raw materials.

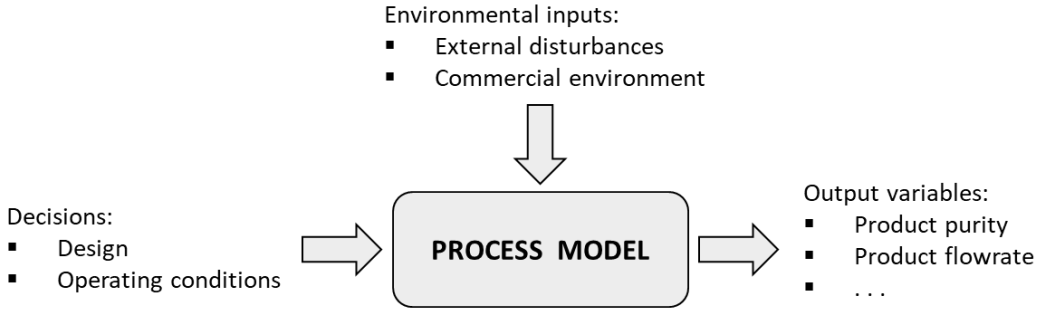


Figure 1.1. *Schematic representation of a process model.*

The chemical industry relies on process modelling for many different purposes including process design, optimization and control. Process models are commonly divided into three categories:

1. The white-box (or first principles) models;
2. The hybrid (or grey box) models;
3. The black-box (or data-driven) models.

This classification is based on the extent of model reliability on process mechanism (or knowledge), input-output (I/O) data inference, or/and a combination of both (Zendehboudi *et al.*, 2018). The white-box models describe the behaviour of the processes exploiting the science and engineering laws that govern their mechanism while the black-box models rely entirely on sets of experimental values. Therefore, the first principles models present extrapolation capabilities while the reliability of the predictions of the black-box models is only ensured within the range of data they have been trained on.

The white-box models describe the behaviour of the systems applying the fundamental laws of conservation of the mass, energy and momentum. Therefore, since they rely only to a minor extent on empirical data, they can be developed before the start-up of the process and can be used to evaluate the pros and cons of different design solutions (Zendehboudi *et al.*, 2018).

The prerequisite for the development of a white-box model is a deep understanding of process mechanism. Hence, building a first principles model is a time and assets consuming task. Furthermore, since they commonly employ a large number of equations, the first principles models have a high computational burden, which makes them unsuitable for on-line implementations. Differently from the white-box models, the data-driven models map the behaviour of the process units through empirical correlations based on a set of experimental observations. Therefore, their development requires only process data to be carried out. The main advantage of the black-box models is their high computational speed while the most important drawback is the limited extrapolation capability they show outside the region covered by the experimental data used in their calibration (De Prada *et al*., 2018). Since they do not require a deep understanding of the underlying mechanism of the process, moreover, they are usually very fast to develop.

As it can be seen in Figure 1.2, the grey-box (or hybrid) models combine the white-box and the black-box approaches with the aim to compensate for the respective shortcomings. The hybrid models are easier and faster to develop with respect to the first principles models and require fewer experiments than the black-box models to be calibrated. When the white-box and the black-box models are combined, the structure of the resulting hybrid model present an adaptive nature, meaning that it can be re-trained whenever new process data are available. Furthermore, the presence of the white-box sub-model ensures the physical significance of certain model parameters and allows to take control decisions which are reasonable and in accordance with the underlying process mechanism (Zendehboudi *et al.*, 2018).

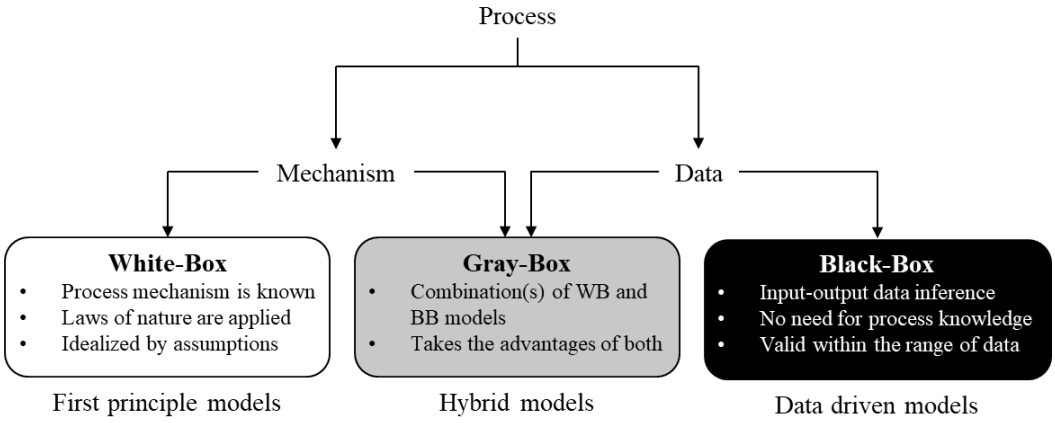


Figure 1.2 *Schematic representation of the differences between the white-box, the grey-box and the black-box models. From: Zendehboudi et al. (2018).*

In the following section, first the possible structures of the hybrid models are discussed. Then, the latest hybrid model applications in the chemical industry are reviewed.

1.1.1 Architecture of the hybrid models

The possible hybrid model architectures are summarized in Figure 1.3.

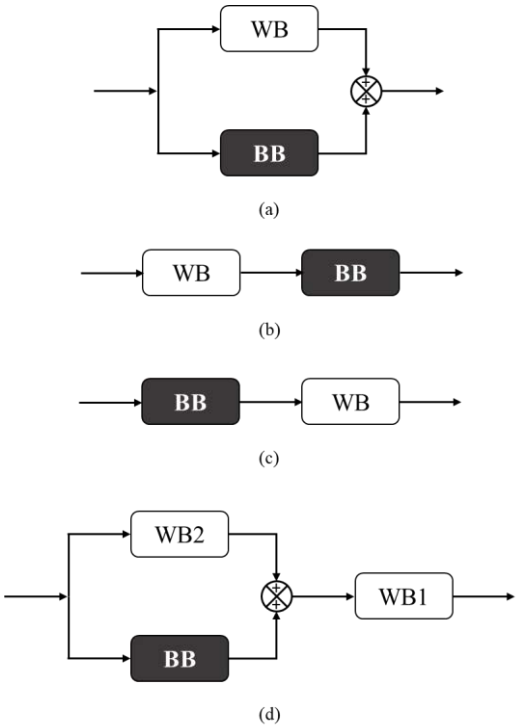


Figure 1.3 *Graphical representation of (a) the parallel hybrid model structure, (b) the serial WB/BB hybrid model structure, (c) the serial BB/WB hybrid model structure and (d) the parallel/serial mixed hybrid model structure.*

The parallel hybrid model structure is employed when a comprehensive white-box model of the process is available, but its predictions do not match the reality satisfactorily. In those cases,

the black-box sub-model is exploited to fill the gaps existing between the hybrid model predictions and the measured process outputs. In the petrochemical industry, for instance, hybrid models with a parallel architecture are used to model the behaviour of the heat exchangers that are heavily affected by fouling. In the scientific literature, indeed, there are plenty of white-box models for the prediction of the heat exchangers overall heat transfer coefficient. These white-box models, however, commonly fail to represent the reality when the equipment start getting fouled. For this reason, a data-driven correlation is employed within the parallel architecture to account specifically for the time dependent performance degradation.

The second type of hybrid model architecture is the serial structure. The serial structure is employed when it is not possible to model all the plant equipment via first principles. In those cases, the behaviour of the poorly understood systems is mapped through a black-box model. In the chemical industry, for instance, empirical correlations are often developed to model the behaviour of the reactors within which reactions with an unknown mechanism occur. In addition, the WB/BB serial structure can also be exploited when no empirical data are available to calibrate the data-driven elements. In those cases, an accurate white-box model can generate a suitable training dataset from simulation results (Zendehboudi *et al.*, 2018).

The last type of hybrid model structure is the mixed serial/parallel architecture. As can be seen in Figure 1.3, this structure is used when only a part (the sub-model WB2) of the white-box process model (WB1) does not provide accurate predictions. In those cases, a data-driven element is developed to compensate for the mismatch with reality caused by the poor predictive capabilities of the sub-model WB2. The mixed structure, moreover, can be exploited to improve the performance of a BB/WB serial architecture. In those cases, the sub-model WB2 is introduced in parallel with the black-box sub-model to support the empirical correlations when dealing with process data heavily corrupted by measurement noise.

1.1.1.1 Common black-box sub-models

Hereinafter, three common types of black-box sub-models that will be mentioned in the hybrid model literature review are briefly overviewed. The first type of black-box models that is taken into account in this section are the empirical models (EMs). The EMs consist on straightforward empirical correlations that relate the value of some measured inputs to the value of some key output variables. Once a training dataset is available, this kind of black-box models are easy and fast to be developed. Unfortunately, however, they commonly suffer from bad extrapolation capabilities. The second common type of black-box models are the artificial neural networks (ANNs). The ANNs are the mathematical equivalent of the human biological neural system (Zendehboudi *et al.*, 2018). Due to their dynamic nature, flexibility and adaptivity, the ANNs are particularly suitable for the description of non-linear complex systems. Finally, the last kind of sub-models are the molecular dynamic simulations. The molecular dynamic simulations have

implication in thermodynamics, nano and micro fluid mechanics and material science. On its simplest form, the technique considers the molecules as rigid hard spheres interacting with each other without having intermolecular flexibility (Zendehboudi *et al.*, 2018).

1.1.2 Literature review

Hybrid modelling is widely employed in several areas of chemical and process engineering. In this section, hybrid model implementations are reviewed considering the following application fields (Zendehboudi *et al.*, 2018):

- Chemical reaction engineering;
- Separation unit operations;
- Transport phenomena.

In the chemical reaction engineering area, the hybrid models are commonly employed to map the behaviour of those reactors within which reactions with a poorly understood mechanism occur. Xiong and Jutan (2002), for instance, developed and successfully implemented a hybrid model based control strategy for an exothermic batch reactor. Employing a parallel architecture, in particular, they first modelled the broad process behaviour with a white-box model. Then, they implemented an ANN to patch the gaps between the model predictions and the reality. Furthermore, the hybrid model developed by Xiong and Jutan (2002) proved to be able to estimate the heat release within the reactor.

Later, Chen *et al.* (2004) developed a hybrid model for a continuous stirred reactor in which the linear equations are solved within a first principles model while the nonlinearities are modelled with a black-box approach. Their model, in addition, was successfully implemented in an internal model control (IMC) scheme for an industrial reactive distillation column employed by Solvay S.A. in the manufacturing process of the epichloroydrin.

For what concerns the separation processes, instead, Engell and Dadhe (2001) exploited hybrid modelling to control the operation of a batch distillation column. In their hybrid model, in particular, ANNs were integrated within a white-box equipment model to estimate the vapor-liquid equilibria (VLE) relationships. The combination of the first principles and the black-box models reduced the computational burden of the overall model, making it suitable for on-line applications. Another application in the field of separation unit operations was proposed by Safavi *et al.* (1999). The goal of their work was to simplify the first principle model of a distillation column for the purpose of on-line optimization of the unit. They accomplished the task integrating within the white-box model of the equipment (which solved the mass and energy balances) a wavelet-based neural network for the estimation of the column separation factor (Safavi *et al.*, 1999).

In the transport phenomena domain, Mjalli and Al-Mfargi (2009) mapped the behaviour of a fluidized catalytic bed reactor used to produce low density polyethylene (LDPE). In their model, ANNs were employed to predict the heat and mass transfer coefficient. Finally, to

conclude this literature review, Liu *et al.* (2007), proposed a hybrid atomistic-continuum scheme to simulate micro and nano flows with heat transfer. In their model molecular dynamics simulations were employed to model the regions where atomistic detail was crucial, while classical fluid dynamics models were used in the remaining regions.

1.2 Motivation and objective of the project

In the chemical industry it is common to encounter processes that are not easy to model with a white-box approach. For instance, there are still plenty of reactors for which a comprehensive first principles model cannot be developed due to the complexity of the reactions mechanism. Furthermore, sometimes even if a white-box model of a process unit exists, it may fail to represent the reality when deviation from the ideal behaviour arises (i.e. the models for the prediction of the overall heat transfer coefficient in heat exchangers heavily affected by fouling). Lastly, there are some occasion in which the first principles model, despite providing accurate predictions, involves a computational burden that is not compatible with on-line model implementations. In all these cases hybrid modelling offers an effective solution to the problem that ensure to retain, as far as possible, the physical significance of the model parameters. The objectives of this thesis are to demonstrate the potential of hybrid modelling and to propose a methodology for the hybrid model development. In order to meet these goals, the industrial process for the manufacturing of cumene will be considered as a case study.
What is expected from the case study, in particular, is to be able to demonstrate that, if process data are available, data-driven elements that map the behaviour of the distillation columns can be integrated within the white-box model of the rest of the plant without compromising the robustness and the reliability of the overall process model.

1.3 Multivariate statistical techniques

In the following sections, the background of the multivariate statistical techniques used in this Thesis is overviewed. In particular, an insight is given on both the theory and the algorithms behind the principal component analysis and the projection on latent structures. The principal component analysis will be exploited in the project as a data dimensionality reduction technique while the projection on latent structures will be used to build empirical correlations that model the behaviour of those processes that cannot be mapped via first principles.

1.3.1 Principal component analysis

Principal component analysis (PCA) is a widely used statistical technique for unsupervised data dimensionality reduction and feature extraction. In modern chemical plants huge volumes of data are continuously acquired; PCA synthesizes efficiently the information stored in these

heavy loads of noisy and correlated data identifying a few fictitious orthogonal variables which capture the variability and the correlations of the original dataset. Performing a PCA, indeed, allows to convert through an orthogonal transformation a set of observation of the (commonly correlated) process variables into a set of values of uncorrelated variables named principal components (PCs). The PCs are linear combination of the original variables that are able to describe the main data trends (Wise and Gallagher, 1996). Wisely selecting the number of PC to be taken into account, it is possible to reduce dramatically the dimensionality of the original dataset while retaining most of its variance.

Under the hypothesis that the process data are collected into a matrix $\mathbf{X}$ of dimensions $(I \times J)$ where I is the number of samples and J the number of process variables, mathematically PCA relies on an eigenvector decomposition of the covariance matrix of the process variables (Wise and Gallagher, 1996):

$$\mathrm{cov}(\mathbf{X}) = \frac{\mathbf{X}^{\mathrm{T}}\mathbf{X}}{I-1} \ . \tag{1.1}$$

PCA decomposes the rank R data matrix $\mathbf{X}$ into a sum of r rank 1 matrices:

$$\mathbf{X} = \mathbf{M}_1 + \mathbf{M}_2 + \mathbf{M}_3 + \ldots + \mathbf{M}_r + \ldots + \mathbf{M}_R \ . \tag{1.2}$$

The matrix $\mathbf{M}_r$ can be represented as the outer product of two vectors $\mathbf{t}_r$ e $\mathbf{p}_r$, therefore equation (1.2) can be reformulated as follows:

$$\mathbf{X} = \mathbf{t}_1\mathbf{p}_1^{\mathrm{T}} + \mathbf{t}_2\mathbf{p}_2^{\mathrm{T}} + \mathbf{t}_3\mathbf{p}_3^{\mathrm{T}} + \ldots + \mathbf{t}_r\mathbf{p}_r^{\mathrm{T}} + \ldots + \mathbf{t}_R\mathbf{p}_R^{\mathrm{T}} \ , \tag{1.3}$$

where the vectors $\mathbf{t}_i$ are and $\mathbf{p}_i$ are known as scores and loadings, respectively. The scores contain information on how the samples relate to each other while the loadings contain information on how the variables relate to each other.

The loadings $\mathbf{p}_i$ are the eigenvectors of the covariance matrix. For each $\mathbf{p}_i$, indeed:

$$\mathrm{cov}(\mathbf{X})\mathbf{p}_a = \lambda_a\mathbf{p}_a \ , \tag{1.4}$$

where λ_i is the eigenvalue associated with the eigenvector $\mathbf{p}_i$.

If the dataset $\mathbf{X}$ is not a full rank matrix (namely when the process variables are highly correlated), it is possible to capture the vast majority of its variance trough a small number of PCs.

Under the assumption that A PCs are taken into account, performing a PCA decomposes the data matrix $\mathbf{X}$ as follows:

$$\mathbf{X} = \sum_{a=1}^{A} \mathbf{t}_a\mathbf{p}_a^{\mathrm{T}} + \mathbf{E} = \mathbf{T}\mathbf{P}^{\mathrm{T}} + \mathbf{E} \ , \tag{1.5}$$

where $\mathbf{E}(I \times J)$ is the residual matrix, $\mathbf{T}(I \times A)$ is the score matrix and $\mathbf{P}(J \times A)$ is the loading matrix. PCA summarizes the valuable information of the original J-dimensional process variables space projecting the original observations onto an A-dimensional latent subspace of PCs. The number of PCs A must be less than or equal to the smaller dimension of $\mathbf{X}$ ($A \leq \min\{I, J\}$). In most of the cases, however, since it is wanted to summarize the information of the heavy load of input data with the lowest amount of latent variables, $A << J$.

Once performed the analysis, the $\mathbf{t}_i/\mathbf{p}_i$ pairs are arranged in descending order according to the associated eigenvalue λ_i. The magnitude of the eigenvalue λ_i, is an indicator of the amount of variance explained by the i^{th} PC.

In this context the variance is intended as the amount of information of the original dataset $\mathbf{X}$. Therefore, when the following approximation is carried out:

$$\widehat{\mathbf{X}} = \mathbf{T}\mathbf{P}^{\mathbf{T}} \ , \tag{1.6}$$

the residuals in matrix $\mathbf{E}$ represent the information (of the initial dataset $\mathbf{X}$) that the model $\widehat{\mathbf{X}}$ is not able to explain. If the number of PC to be taken into account is selected properly, however, most of the relevant information is retained by the model leaving only measurements noise in the residual matrix.

From a geometrical point of view, as it can be noticed in Figure 1.4, the loadings are the director cosines of the PCs, while the scores are the coordinates of the data in the new system defined by the latent variables.

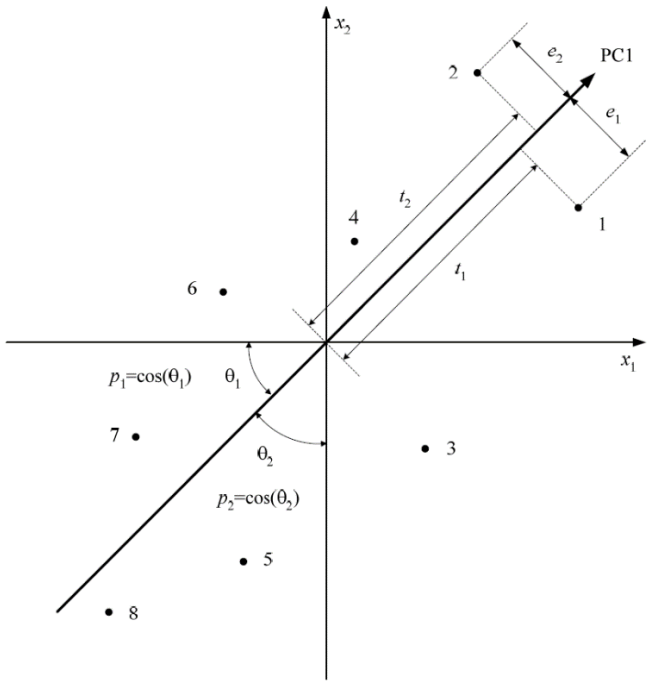


Figure 1.4. *Geometrical interpretation of the scores and the loadings of the PCA method for a dataset with I=8 observations of J=2 variables (x1 and x2). From: Facco (2009).*

The concept of PCA, moreover, can be further explained through Figure 1.5.

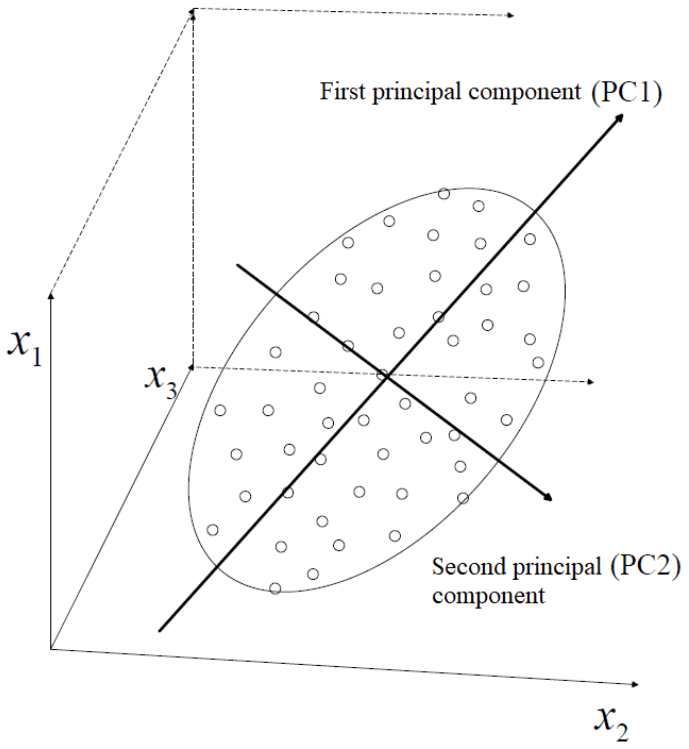


Figure 1.5. *PC model of a three-dimensional dataset lying primarily in a single plane.*

As it can be noticed in Figure 1.5, a three-dimensional dataset where the data lie primarily in a plane can be efficiently described by a two PCs model. The first eigenvector or PC aligns with the greatest variation in the data while the second PC represents the greatest amount of variation that is orthogonal to the first PC.

The directions with the greatest variation in the data are found through a least squares optimization of the residuals $e_{i,j}$.

1.3.2 Partial least squares regression

Partial least squares (PLS) regression is a statistical multivariate method that relates the information of two data matrices combining PCA and multiple linear regression. The main goal of the PLS is to predict a set of response variables **Y** through a set of regressors **X**. However, differently from a standard PCA (which decomposes a data matrix **X** with the aim to find the components that best explain its variance), performing a PLS allows to find the direction of greatest variation of the dataset **X** that best predicts **Y**.

Specifically, PLS regression searches for a set of latent vectors that perform a simultaneous decomposition of **X** and **Y** with the constraint that these components explain as much as possible of the covariance between **X** and **Y**.

PLS derives its usefulness from its ability to analyse data with many, noisy, collinear, and even incomplete variables in both **X** and **Y** (Wold *et al.*, 2001).
The method consists of two outer relations and one inner relation. The outer relations are the decompositions of the matrices **X** and **Y**:

$$\begin{cases} \mathbf{X} = \sum_{a=1}^{A} \mathbf{t}_a \mathbf{p}_a^{\mathrm{T}} + \mathbf{E} = \mathbf{T}\mathbf{P}^{\mathrm{T}} + \mathbf{E} \\ \mathbf{Y} = \sum_{a=1}^{A} \mathbf{u}_a \mathbf{q}_a^{\mathrm{T}} + \mathbf{F} = \mathbf{U}\mathbf{Q}^{\mathrm{T}} + \mathbf{F} \end{cases} , \tag{1.7}$$

where $\mathbf{t}_a$ and $\mathbf{u}_a$ are the score vectors (in the score matrices **T** and **U**), $\mathbf{p}_a$ and $\mathbf{q}_a$ are the loading vectors (in the loading matrices **P** and **Q**) and **E** and **F** are the residual matrices.
The procedure implies to minimize the norm of the residual matrices $\|\mathbf{E}\|$ and $\|\mathbf{F}\|$.
The inner relation among the scores of the matrices is:

$$\mathbf{u}_a = b_a \mathbf{t}_a \ , \tag{1.8}$$

where b_a are the regression coefficients:

$$b_a = \frac{\mathbf{u}_a^T \mathbf{t}_a}{\mathbf{t}_a^T \mathbf{t}_a} \ . \tag{1.9}$$

Since the PCs are calculated for the two blocks separately, however, this procedure exhibits poor computational efficiency.
Hence, commonly, the parameters for the PLS models are estimated through a slightly modified version of the NIPALS algorithm, which, as stated by Jackson (1991), was originally developed by Wold with the name NILES. A detailed description of the modified version of the NIPALS algorithm for the PLS model parameters estimation is given in the following section (§1.3.2.1).

1.3.2.1 Nonlinear iterative partial least squares algorithm (NIPALS)

The modified version of the NIPALS algorithm commonly employed for the PLS model parameters estimation has been comprehensively described by Geladi and Kowalski (1986).
The adapted procedure, besides the scores and the loading of the matrices **X** and **Y**, calculates also an additional set of vectors known as weights. The weights are employed as a mathematical artifice to maintain orthogonal scores.
The algorithm consists on the following steps (Geladi and Kowalski 1986):

1. Select a column of the matrix **Y** as the starting estimate of $\mathbf{u_1}$ (usually the column with the greatest variance is chosen):

$$\mathbf{u_1} = \mathbf{y} \ , \tag{1.10}$$

2. Calculate at iteration r:

$$\mathbf{w}_r = \frac{\mathbf{X}^\mathbf{T}\mathbf{u}_r}{\|\mathbf{X}^\mathbf{T}\mathbf{u}_r\|} \ , \tag{1.11}$$

$$\mathbf{t}_r = \mathbf{X}\mathbf{w}_r \ , \tag{1.12}$$

$$\mathbf{q_1} = \frac{\mathbf{u}_r^\mathbf{T}\mathbf{t}_r}{\|\mathbf{u}_r^\mathbf{T}\mathbf{t}_r\|} \ , \tag{1.13}$$

$$\mathbf{u}_r = \mathbf{Y}\mathbf{q}_r \ . \tag{1.14}$$

3. Check for convergence by comparing $\mathbf{t}_r$ in equation (1.12) with $\mathbf{t}_{r-1}$; if they are equal within a predefined tolerance, proceed.
4. Calculate the loadings:

$$\mathbf{p_r} = \frac{\mathbf{X}^\mathbf{T}\mathbf{t}_r}{\|\mathbf{t}_r^\mathbf{T}\mathbf{t}_r\|} \ , \tag{1.15}$$

5. Update the loadings, scores and weights:

$$\mathbf{p}_{r,new}^\mathbf{T} = \frac{\mathbf{p}_{r,new}^\mathbf{T}}{\|\mathbf{p}_{r,old}^\mathbf{T}\|} \ , \tag{1.16}$$

$$\mathbf{t}_{r,new} = \mathbf{t}_{r,old} \ \|\mathbf{p}_{r,old}\| \ , \tag{1.17}$$

$$\mathbf{w}_{r,new} = \mathbf{w}_{r,old} \ \|\mathbf{p}_{r,old}\| \ , \tag{1.18}$$

6. Compute the regression coefficients through equation (1.9).
7. Calculate the residuals:

$$\mathbf{E}_r = \mathbf{X}_r - \mathbf{t}_r\mathbf{p}_t^\mathbf{T} \ , \tag{1.19}$$

$$\mathbf{F}_r = \mathbf{Y}_r - \mathbf{b}_r\mathbf{u}_r\mathbf{q}_r^\mathbf{T} \ , \tag{1.20}$$

8. Repeat the procedure for every retained PC going back to step 1 after replacing $\mathbf{X}$ and $\mathbf{Y}$ by $\mathbf{E}_r$ and $\mathbf{F}_r$, respectively.

1.3.3 Data collection and pretreatment

Black-box (or data-driven) models rely only on the I/O data structure to map the behaviour of the process, therefore, they are only able to provide valid predictions within the range of the data they have been trained on. For this reason, the model must be calibrated with the most representative set of samples.

Once selected a dataset that properly characterize the process of interest, the input data have to be pretreated. In this Thesis, the auto-scaling is adopted as data pre-treatment technique. The auto-scaling mean-centers and scales to unit variance all the variables.

The mean centering consists on subtracting from every variable $x_{i,j}$ the respective mean $\bar{x}_j$.

The mean is calculated as follows:

$$\bar{x}_j = \frac{\sum_{i=1}^{I} x_{i,j}}{I}, \qquad j = 1, \dots, J \ , \tag{1.21}$$

where $x_{i,j}$ is the element of $\mathbf{X}$ in row i and column j.

Once mean-centered, the data are auto-scaled dividing each measurement by the standard deviation of the corresponding variable:

$$\sigma_j = \sqrt{\frac{\sum_{i=1}^{I}(x_{i,j} - \bar{x}_j)^2}{I}} \ . \tag{1.22}$$

The auto-scaling allows to deal with the differences in the magnitude of the variables that present different unit of measurement and assign to all the variables the same weight.

Once preformed the auto-scaling on a matrix $\mathbf{X}$, moreover, its covariance matrix becomes its correlation matrix.

1.3.4 Selection of the PC subspace dimension

The selection of the latent variables subspace dimension (namely the number of PCs to be taken into account) can be performed according to many different rationales.

One of the most straightforward techniques involves neglecting the PCs associated with an eigenvalue smaller than a pre-defined threshold value.

As stated by Muñoz (2019), indeed, the eigenvalue of a PC can be roughly interpreted as the number of variables (of the original dataset) that the component is representing. It is therefore reasonable to neglect the PCs whose eigenvalue is smaller than 0.5 (namely those PCs that represent less than half a variable of the original dataset).

Similarly to this first method, a second approach implies the selection of as many PCs as needed to retain at least a certain amount of variance of the original dataset.

Anyway, the most commonly employed technique to solve the problem of the choice of the latent variables subspace dimension is the *k*-fold cross validation. The *k*-fold cross-validation method consists on:

1. Randomly splitting the dataset **X** into *k* subgroups of observations;
2. Building a reduced dataset deleting one of the subgroups;
3. Calibrating the model on the reduced dataset;
4. Evaluating the goodness-of-fit of the model on the deleted subgroup;
5. Iterating the procedure for all the subgroups;
6. Repeating the procedure changing the number of PCs.

The method is schematically represented in Figure 1.6.

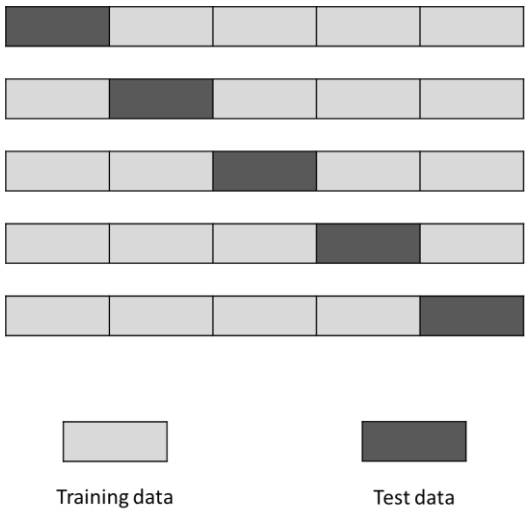


Figure 1.6. *Schematic representation of the k-fold cross-validation with k=5.*

The criterion used to assess the model goodness-of-fit in the fourth step relies on the analysis of the $RMSECV$ (acronym for root-mean-square error of cross validation):

$$RMSECV = \sqrt{\frac{PRESS}{I}} \ , \tag{1.23}$$

As it can be seen in equation (1.23), the $RMSECV$ is a function of the predicted residual error sum of squares ($PRESS$), which is defined as follows:

$$PRESS = \sum_{j=1}^{J} \sum_{i=1}^{I} e_{i,j}^2 \ . \tag{1.24}$$

where $e_{i,j}$ is the element of the residual matrix **E** in row i and column j.

Increasing the number of PCs that are taken into account in the model, a decrease of the $RMSECV$ is detected at first. Then, when the newly added PCs are only able to explain the noise

of the measurements of the original dataset, the value of the $RMSECV$ increases. Therefore, the optimal number of PCs can be found in correspondence to the minimum of the $RMSECV$.
In this thesis, the straightforward approach based on the eigenvalues is used to determine the latent variables subspace dimension when, inside the steady state detection algorithm, PCA is exploited as a data dimensionality reduction technique (§2).
When instead empirical correlations are built through partial least squares regression (§4), the cross-validation method is employed to ensure an effective minimization of the residuals, to achieve the highest possible predictive capability and to avoid the overfitting.

1.3.5 Diagnostic metrics for the partial least square regression

Although the root-mean-square error of cross validation ($RMSECV$) remains the decisive criterion for the choice of the number of PCs, the influence of the latent variables subspace dimension on the PLS regression model performances has been further investigated through two additional metrics: the determination coefficient R^2 and the predictive relevance Q^2. The determination coefficient R^2 allows to assess the capability of the model to represent the original data and is defined as follows:

$$R^2 = 1 - \frac{\sum_{j=1}^{J}\sum_{i=1}^{I}\left(x_{i,j} - \hat{x}_{i,j}\right)^2}{\sum_{j=1}^{J}\sum_{i=1}^{I}\left(x_{i,j} - \bar{x}_j\right)^2} = 1 - \frac{RSS}{TSS} \ , \tag{1.25}$$

where $\hat{x}_{i,j}$ is the element of the matrix that has been reconstructed through the model while RSS and TSS are the residual sum of squares and the total sum of squares, respectively.
The metric Q^2 provides an evaluation of the model predictive capability and is defined as follows:

$$Q^2 = 1 - \frac{PRESS}{TSS} \ . \tag{1.26}$$

The determination coefficient is bound between 0 and 1 while the predictive capability can assume negative values. The Q^2 is negative if, when used for predictions, the model performs worst than the no-model estimate (namely the mean response of the training dataset).
When both metrics show a low value, the model is underfitting the training data. Underfitting occurs when the model is too simple to capture the underlying data trend.
When the determination coefficient shows a high value (close to unity) but the predictive relevance remains small the model not only captures the data trend, but begins to describe the noise of the training dataset as well. This phenomenon is commonly referred to as ‘overfitting’. The models that overfit the training data have a low bias (due to the increased degrees of freedom of the model) and a high variance; therefore, their predictions can change dramatically with minor changes on the training dataset (Wilson and Sahinidis, 2017).

Lastly, if the metrics are both close to unity, the model performs well on both the training and test data.

1.4 ALAMO

In the following section a detailed description of the theory and the algorithms behind the ALAMO model building methodology is provided. Similarly to the partial least squares regression, ALAMO will be exploited in this Thesis to develop empirical models of those processes whose behaviour cannot be mapped through the first principles.

1.4.1 Background

Chemical process simulation software are widely employed both industrially and academically for the design and test of single equipment and/or entire processes. However, as stated by Cozad *et al.* (2014), despite these numerical models provide remarkable level of accuracy in their predictive capabilities, the structure of the simulations can impose challenges when used in an optimization setting.
The general optimization problem can be addressed as follows:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0 \\ & x \in A \subset \mathbb{R}^n \end{aligned} \tag{1.27}$$

where $f(x)$ is the cost function to be minimized with respect to the degrees of freedom x while $g(x) \leq 0$ is the set of constraints the degrees of freedom are required to satisfy. When the functions $f(x)$ and $g(x)$ are not available in algebraic form but are obtained through an input output black-box instead, the above optimization problem implies some challenges such as the need of derivative information and costly function evaluations. To overcome these challenges a significant effort has been made to generate highly accurate surrogate models of the functions $f(x)$ and/or $g(x)$. Cozad *et al.* (2014), in particular, developed ALAMO (acronym for Automatic Learning of Algebraic Models for Optimization), a model-building methodology that identifies algebraic correlation from a set of measured or simulated data. Through the learning software ALAMO, however, Cozad *et al.* (2014) aimed not only to achieve remarkable models accuracy but focused on the reduction of models complexity as well. The models built with ALAMO, hence, aim to reduce the difficulty and improve the tractability of the optimization procedure.
In the ALAMO approach firstly the surrogate models are developed through an integer programming-based best subset technique that considers a large number of explicit transformations of the original input variables. Then, if the number of points of the training

dataset is not fixed (namely when new points can be added to the training dataset carrying out further simulations) the model is improved through an iterative approach. The iterative approach implies that the current surrogate models are tested against the simulation through an adaptive sampling methodology. The adaptive sampling technique named by Cozad *et al. 'error maximization sampling (EMS)'* finds areas in the problem space that maximize the model error. Once the error has been calculated in the newly sampled points, if no areas of sufficient model mismatch are found the algorithm terminates and the methodology has converged to the final surrogate model. If instead the model is proved to be inconsistent above a specified tolerance, the points identified by the EMS are added to the training set and the model is retrained.
Anyway, since in this work the algebraic correlations between input and output variables will be developed starting from a training dataset with a fixed number of points, no further information regarding the adaptive sampling technique will be given.
In the following section (§1.4.2) a detailed description of the model-building methodology will be provided instead.

1.4.2 ALAMO model-building methodology

Given a set of N training points where each data points contains a set of input data x_{id} and a set of responses z_{ik} (where $i = 1, \dots, N$, $d \in D$ and $k \in K$), under the hypothesis that the analytical form of the response surface is not known, it is wanted to generate a model for each response.
In order to develop models with sufficient complexity to achieve accurate predictions and enough simplicity to ensure the tractability of the optimization, firstly a simple set of basis functions and a constant term are defined (Cozad *et al.* 2015).
The basis functions $X_j(x)$ (with $j \in B$) are non-linear transformations of the input variables.
The most employed functions are summarized in Table 1.1, where the value of the exponent α is defined by the user.

Table 1.1. *Commonly used basis functions.*

Category	$X_j(x)$
Polynomial	$(x_d)^\alpha$
Multinomial	$\prod_{d \in D' \subseteq D} (x_d)^{\alpha_d}$
Exponential and logarithmic forms	$exp(x_d)^\alpha, log\ (x_d)^\alpha$

The resulting surrogate model is a linear combination of the non-linear basis functions:

$$\hat{z}(x) = \sum_{j \in B} \beta_j X_j(x) \tag{1.28}$$

where the values of the model parameters β_j (namely the coefficients that multipy the basis functions) can be estimated solving an ordinary least squares regression problem.

Unfortunately, although the model developed starting from the expanded set of regressors will typically experience a low error on the training set, it still may not contain an adequate representation of the underlying process. As the number of regressors increases, indeed, the model generally begins to overfit the training data.

In order to avoid the overfitting, the superfluous regressors must be removed from the model. Model reduction techniques as the backward elimination are common method that allows to reduce the number of terms in a model. However, although these methods are able to attenuate the overfitting using only a subset of the original set of basis functions, they can easily miss synergistic effects of groups of bases that may show poor fitting capabilities if taken into account individually. For this reason, a best subset method is implemented to take into account for all the possible combinations and to identify, through a measure of the model fitness that is sensitive to overfitting, the best subset of basis.

The general best subset problem can be addressed as follows:

$$\begin{aligned} &\min_{S,\beta} \quad \Phi(S,\beta) \\ &\text{s.t.} \quad S \subseteq B \end{aligned} \tag{1.29}$$

where $\Phi(S,\beta)$ is the model goodness of fit for the regression coefficients β and the subset of basis function S.

Solving the problem addressed in equation (1.29) allows to find a model that uses the best subset of functions S to achieve the most effective bias-variance trade-off (Cozad *et al.*, 2014):

$$\hat{z}(x) = \sum_{j \in S} \beta_j X_j(x) \tag{1.30}$$

Starting from equation (1.29) Cozad *et al.* (2014) performed a series of reformulations that simplified the solution of the problem.

The final version of the methodology and the main steps of the reformulation procedure will be presented hereinafter while a more comprehensive and detailed description will be given in appendix A.

The first simplification is achieved tracking which basis functions are active in the model through a binary vector y defined as follows: whenever a basis function $j \in B$ is active in the model ($j \in S$), then $y_j = 1$; otherwise $y_j = 0$.

The vector y allows to reformulate equation (1.29) into a mixed-integer non linear problem and to describe equation (1.30) over the full bases set B.

The next step implies decoupling the goodness-of-fit measure into two parts as follows:

$$\min_{\beta,T,y} \Phi(\beta,T,y) = \min_{\mathrm{T}} \left\{ \min_{y,\beta} \left[\Phi_{y,\beta}(y,\beta)|_T \right] + \Phi_T(T) \right\} \tag{1.31}$$

where $\Phi_T(T)$ is the model sizing part while $\Phi_{y,\beta}(y,\beta)|_T$ refers to the selection of the basis functions and the parameters (Cozad *et al.*, 2014).
Hence, the best subset selection problem is posed as a nested minimization as follows:

$$\begin{aligned} \min_{T\in\{1,\dots,T^u\}} \quad & \left[\Phi_{y,\beta}(y,\beta)|_T\right] + \Phi_T(T) \\ \text{s.t.} \quad & \min_{y,\beta} \Phi_{y,\beta}(y,\beta)|_T \\ & \text{s.t.} \quad \sum_{j\in B} y_i = T \\ & \beta^l y_j \le \beta_j \le \beta^u y_j \quad j \in B \\ & y_j \in \{0,1\} \quad\quad j \in B \end{aligned} \tag{1.32}$$

The inner minimization concerns with the selection of the basis functions and of the parameters while the outer minimization determines the complexity of the model.
The solution of the best subset selection problem is highly influenced by the choice of the model fitness measure. A proper measure must properly reflect the accuracy of the model while remaining sensitive to overfitting. Two common methods to assess the model fitness are cross-validation and information criteria. Both these methods are sensitive to empirical error and overfitting, but the cross-validation is not able to account directly for the model complexity.
For this reason and due to the large number of basis functions available, Cozad *et al.* (2014) decided to use as goodness-of fit measure the corrected Akaike information criteria (Hurvich and Tsai, 1993):

$$AICc(S,\beta) = Nlog\left(\frac{1}{N}\sum_{i=1}^{N}\left(z_i - \sum_{j\in S}\beta_j X_{ij}\right)^2\right) + 2|S| + \frac{2|S|(|S|+1)}{N-|S|-1} \tag{1.33}$$

Starting from equation (1.32), at last, two more series of reformulation were performed to improve the tractability and efficiency of the algorithm. Firstly, the finite solution space of the outer minimization is parametrized with respect to T, which is an indicator for the complexity of the model. Then the inner minimization is posed as the following mixed-integer linear problem (Cozad *et al.*, 2014):

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{N} w_i \\
\text{s.t.} \quad & w_i \geq z_i - \sum_{j \in B} \beta_j X_{ij}, && i = 1, \dots, N \\
& w_i \geq \sum_{j \in B} \beta_j X_{ij} - z_i, && i = 1, \dots, N \\
& \sum_{j \in B} y_i = T \\
& -U_j(1 - y_i) \leq \sum_{j \in B} X_{ij} \left(z^i - \sum_{j \in B} \beta_j X_{ij} \right) \leq U_j(1 - y_i) \ , && j \in B \\
& \beta^l y_j \leq \beta_j \leq \beta^u y_j, && j \in B \\
& y_{kj} \in \{0,1\} && j \in B \\
& \beta_j^l \leq \beta_j \leq \beta_j^u, && j \in B
\end{aligned}
\tag{1.34}
$$

The set of equations (1.34) is used to identify the best T-term subset of the original set of bases B. Solving the equations with a small value of T and then increasing that value until the information criterion worsens it is possible to efficiently solve the best subset problem finding the most accurate low-complexity model.

Chapter 2

Steady-state detection

This chapter deals with the issue of the steady-state identification in industrial processes. The reader is firstly provided with a brief review of the state-of-the art scientific literature regarding steady-state detection algorithms and the approaches they exploit to tackle the problem. Particular attention is devoted to the method proposed by Kelly and Hedengren (2013), which is the technique the novel algorithm developed in this work aims to improve.
Finally, the reliability of the novel approach is evaluated analysing the results obtained applying the steady-state detection algorithm to the cumene process case study.

2.1 Overview

With the increasing use of steady-state model based techniques in industrial continuous processes, rigorous steady-state identification has become critical in a wide range of application areas, such as process performance assessment, data reconciliation, soft sensor development, process optimization, fault detection and process control (Jiang *et al.*, 2003).
Process data employed for building black-box or hybrid steady-state models should only be collected when the plant is actually operating steadily, otherwise erroneous parameter or entire modeling failure can occur. Moreover, since applying a steady-state model to a non-stationary process would not provide any meaningful results, attaining the steady-state, besides triggering data collection, represents also the necessary condition that, once verified, allows for models implementation. The term steady-state refers to a process operating around a stable point or within a stationary region where the accumulation of material, energy and momentum is statistically negligible (Kelly and Hedengren, 2013).
With the development of distributed control system (DCS) technologies, a huge amount of process data (both steady and unsteady) can be collected and recorded for state identification. When a process variable is measured, in particular, the output displayed by the sensor stands for the true process value with additive noise and disturbance.
At steady-state, the true process values stay unchanged; therefore detecting windows in which a process is operating in a state of steadiness would be trivial if the process signals were noiseless. In this case, it would indeed be enough to state the constancy of the sensors output signals to ensure that the plant is operating steadily. Unfortunately, process measurements are inherently corrupted by several sources of error (such as instrument malfunction or

miscalibration and/or process noise); hence, the sensors output signals keep changing even when the process is operating at steady-state.
The steady-state detection methods must be able to recognize if the variation of the process variables is due to a non-stationary drift or if it is only driven by noise fluctuations. Misunderstanding the nature of the variations could lead either to Type I or Type II error. Type I errors (false positives) arise when the algorithms trigger a 'steady-state' when the process is not at steady-state while type II errors (false negatives) consist on triggering a 'not at steady-state' response when the process is actually at steady-state (Cao and Rhinehart, 1995).

2.1.1 Literature survey

In the past few decades, several steady-state detection methods have been proposed. According to Wang *et al.* (2018), these approaches can be classified into three main categories: model-based, statistical theory based and trend extraction based methods.
The model-based techniques detect process steady-state by analysing deeply the physical and chemical background of the process. An example is the approach proposed by Prabhakar and Kumar (2014) to assess the voltage stability margins, which is based on the P-Q-V curve and Thevenin's theorem. The quality of the state identification provided by the model-based approaches depends highly on the accuracy of the process model. Accurate process models, however, especially for complicated large-scale industrial plants, are very difficult and costly to develop. Therefore, since the process state is reflected in the real time collected process data, it is more reasonable to carry out SSD (acronym for steady-state detection) through data-driven methods. The main strength of the data-driven methods is their versatility. Indeed, since these approaches do not rely on process models, they are suitable to be applied to a broad spectrum of different processes. The statistic based and the trend extraction based methods (namely the second and third category of SSD approaches according to the classification of Wang *et al.*) are both data-driven techniques. Among the statistical methods, Cao and Rhinehart (1995) developed a computationally efficient approach based on the R-statistic. The R-statistic evaluates the ratio of two variances measured on the same set of data by means of two different techniques: the mean squared deviation and the mean of the squared difference of successive data. The computational burden of the method is minimized employing conventional exponentially weighted moving averages (namely a first order filter). More recently, Kelly and Hedengren (2013) computed the probability of a process to be steady performing a Student-t test. Since the novel steady-state detection method proposed in this work aims to improve Kelly and Hedengren's algorithm, a detailed description of their approach will be given in §2.1.2.
The statistical methods mostly provide steady-state evaluations on fixed time intervals, rather than assessing the steadiness of the process state in each time point. Cao and Rhinehart's technique is an exception, but, as a consequence of the filtering procedure, it is affected by a delay in the prediction. Furthermore, since the optimal detection parameters of each method

depend on the characteristics of the signal (such as the noise variance), it is needed to re-tune the algorithm parameters each time a new application is taken into account.
The last category of SSD methods according to Wang's classification are the trend extraction approaches. One of the most employed technique to carry out trend extraction is wavelet transform. Jiang *et al.* (2003), for instance, proposed to assess the steadiness of the process computing the value of the first and second order wavelet transform modulus. The computational burden is commonly the main issue of the trend extraction based steady-state detection methods.

2.1.2 Kelly and Hedengren's method

In 2013, Kelly and Hedengren proposed a statistical approach based on a student t-test to tackle the steady-state detection problem. The description of the algorithm assumptions and steps is given in §2.1.2.1 and §2.1.2.2 as if only a single process signal x_t was taken into account for process steadiness assessment.
As a consequence of the growing development of distributed control system technologies, however, the amount of measured process variables is commonly very large. Therefore, it is essential for a steady-state detection algorithm to be able to deal with multiple process signals. The approach Kelly and Hedengren followed to provide a unique prediction for the overall process steadiness when more than one process signal is available is described in §2.1.2.3.

2.1.2.1 Algorithm assumptions

The first step of the algorithm consists on splitting the dataset that stores all the measurements of the process signal x_t through the definition of time windows.
Each time window is defined to include n sampled values of the process variable x_t, which are equally spaced in time. Therefore, the first fundamental assumption is that the measurements collection is performed with a constant sampling frequency.
Once the dataset has been splitted, moreover, Kelly and Hedengren defined the value of the process signal x_t inside the window through the following equation:

$$x_t = mt + \mu + a_t \ , \tag{2.1}$$

where (Figure 2.1):

- m : slope of the regression line that fits the trajectory of the process signal x_t;
- t : relative time running within the window;
- μ : intercept of the regression line that fits the trajectory of the process signal x_t;
- a_t : white noise sequence.

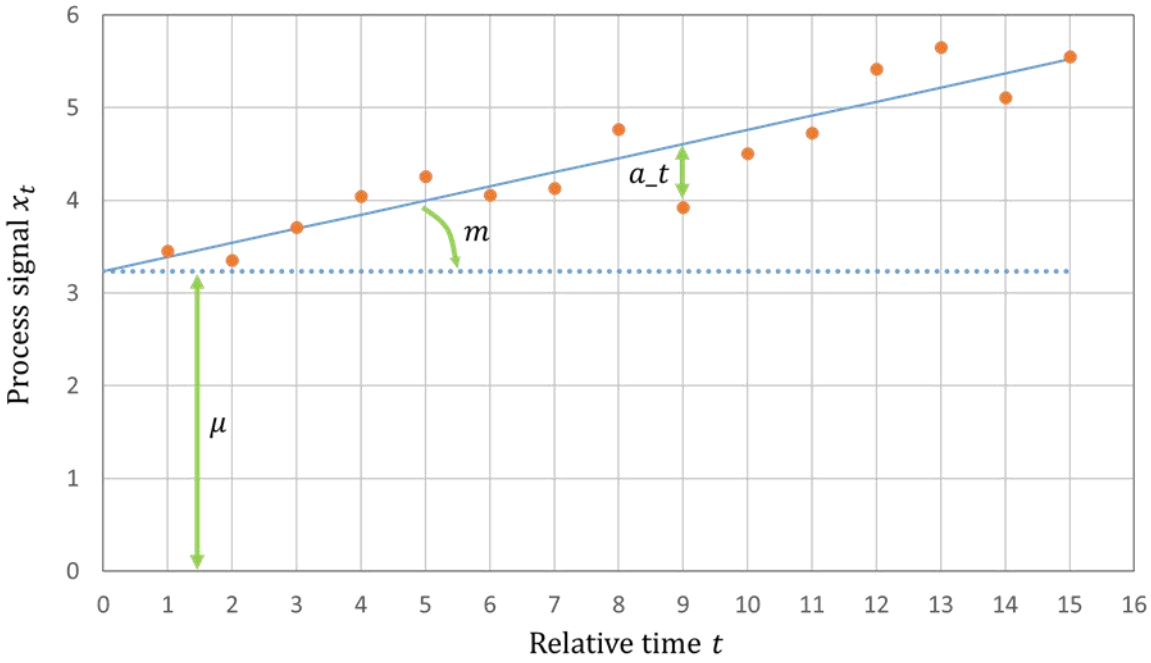


Figure 2.1. *Process signal value x_t as a function of the relative time t running within the window.*

The intercept of the regression line corresponds to the mean of the hypothetical stationary process (namely the arithmetic average of the values of x_t over the time window under the condition that the slope m is null). The white noise sequence is assumed to have zero mean and standard deviation σ_a.

2.1.2.2 Algorithm steps

Given a dataset with all the measurements of a process variable x_t and the value of the following parameters:

- α : significance level of the Student's t-test;
- Tr: threshold value for the assessment of the process steadiness in the time window;
- n : number of sampled values included in the data window.

the following step need implementing:

1. Split the dataset collecting all the measurements in time windows with n sampled values of x_t each.
2. For each time window:
   2.1 Fit with a linear model the trajectory of the process signal x_t in order to obtain the values of the slope and of the intercept of the regression line.
   2.2 Compute the standard deviation of the white noise shocks as follows:

$$\sigma_a = \sqrt{\frac{1}{n-2}\sum_{t=1}^{n}(x_t - mt - \mu)^2} \ . \tag{2.2}$$

2.3 Perform the following Student's two-tailed t-test:

$$\begin{aligned} &If\ |x_t - \mu| \le t_{crit}\sigma_a\text{:} \\ &\qquad Set\ y_t = 1\ . \\ &Else\text{:} \\ &\qquad Set\ y_t = 0\ . \end{aligned} \tag{2.3}$$

where t_{crit} is the critical t value of a two-tailed test with significance level α.

2.4 Compute the fraction of time τ within the window in which the process is deemed to be at steady-state through the following equation:

$$\tau = \frac{\sum_{t=1}^{n} y_t}{n} \ . \tag{2.4}$$

2.5 Assess the stability of the process signal in the time window through the comparison of τ with the threshold value Tr:

$$\begin{aligned} &If\ \tau \ge Tr:\\ &\text{The process signal is deemed to be at steady-state in the window.}\\ &Else:\\ &\text{The process signal is deemed not to be at steady-state in the window.} \end{aligned} \tag{2.5}$$

The practical meaning of the threshold value Tr can be clarified through the following brief example: setting Tr equal to 0.99 means that in order to deem the signal to be steady at least 99% of the points included in the time window must be at steady-state.

2.1.2.3 Multiple process signals and time window length

As stated previously, the development of distributed control system (DCS) technologies increased dramatically the amount of collected process data. Even if several process signals are available for process state identification, the SSD algorithm must be able to provide a unique assessment for the steadiness of the entire process. This comprehensive state prediction must take into account for the steadiness of all the measured key process variables.
In order to deal with multivariate system, Kelly and Hedengren proposed to:

1. Select among all the available process signals those referring to variables that have a key influence on the overall process behaviour.
2. Reduce the significance level α of the Student's t test through the Sidak inequality:

$$\alpha' = 1 - (1 - \alpha)^{\frac{1}{k}} \ . \tag{2.6}$$

   where k is the number of selected key variables.
3. Apply the algorithm (§2.1.2.2) to each key process signals employing α' as the significance level of the Student's t test.

As an outcome of the third step, a steadiness prediction for each process signal is obtained. However, no further indication on how to deal these predictions is provided. It is reasonable to assume that Kelly and Hedengren decided to deem the process to be at steady-state in a certain time window only if all the selected key signals are steady within that window.
Lastly, as regards the selection of the window size n, it is suggested to fulfill the following inequalities:

$$3 \leq \frac{\tau_p}{\Delta t} \leq 5 \ . \tag{2.7}$$

where Δt is the sampling period and τ_p is the process time constant. The selection of the key variables as well as the estimation of the process time constant require having prior process expertise.

2.2 A new method for steady-state detection

In this work, a novel steady-state detection method is developed combining a modification of Kelly and Hedengren's t-test with one of the most accredited data dimensionality reduction techniques: principal component analysis (PCA). The steady-state detection algorithm has been coded in Python™ 3.7.3. The packages '*numpy 1.16.2'* and '*pandas 0.24.2'* have been extensively used to deal with arrays and matrices while the package '*scikit-learn 0.20.3'* has been exploited to perform PCA. Lastly, the module *'stats'* from the package '*scipy 1.2.1'* has been used to compute the critical t value of a two-tailed test with significance level α.

2.2.1 Motivation

The design of the new algorithm is mainly styled after Kelly and Hedengren's approach. The reason why PCA has been exploited is the need of improving the management of multiple process signals. Kelly and Hedengren proposed to deal with multivariate systems through Sidak's inequality (2.6). One of the assumptions of Sidak's inequality, however, is that the k process signals employed for steady-state detection refer to variables that are independent from

each other. Unfortunately, in chemical plants dramatic correlations and redundancies arise among process variables. The validity of equation (2.6) is therefore strongly questionable. When taking into account complicated large-scale industrial applications, moreover, the identification of the key process variable could be very difficult to carry out. The possibility of exploiting PCA rather than Sidak's inequality to deal with multiple process signals has been suggested by Wang *et al.* (2018). Through PCA, it is possible to find low-dimensional representations for high-dimensional observed data maintaining the main variance of the original dataset. The low-dimensional representation is given by latent variables that are named principal components (PCs). Since the variance of the original dataset is mostly retained by the PCs, furthermore, the information that are needed to identify accurately the state of the process (steady or unsteady) are kept in the latent variables subspace, while process noise is left in the residuals. Carrying out steady-state detection on PCs rather than on process variables improve therefore the accuracy and the efficiency of the prediction. In addition, beside the capture of the main trends of the original dataset, PCA eliminates also the correlation between the process variables.

Since commonly more than one PC is taken into account, it can be argued that the multivariate system issue has not been tackled (because the problem of considering multiple process variables has just been replaced by the problem of considering multiple latent variables). Actually, differently from process variables (whose relative importance cannot be estimated quantitatively), after PCA has been carried out, the relevance of each PC is assessed by the percentage of variance of the original dataset that the latent variable is able to explain. It can be therefore easily computed a composite SSD index (namely a steadiness prediction for the entire process) weighting the prediction of each single PC through their relative importance.

The last motivation that required the development of a novel steady-state detection algorithm concerns with the definition of the time window. Kelly and Hedengren proposed the implementation of a static approach in which the algorithm detects the time window as a whole to be in a steady or unsteady-state. In order to provide a steady-state prediction for each sampling instant, however, a moving window would be needed. In Figure 2.2 the differences between the two techniques are graphically displayed.

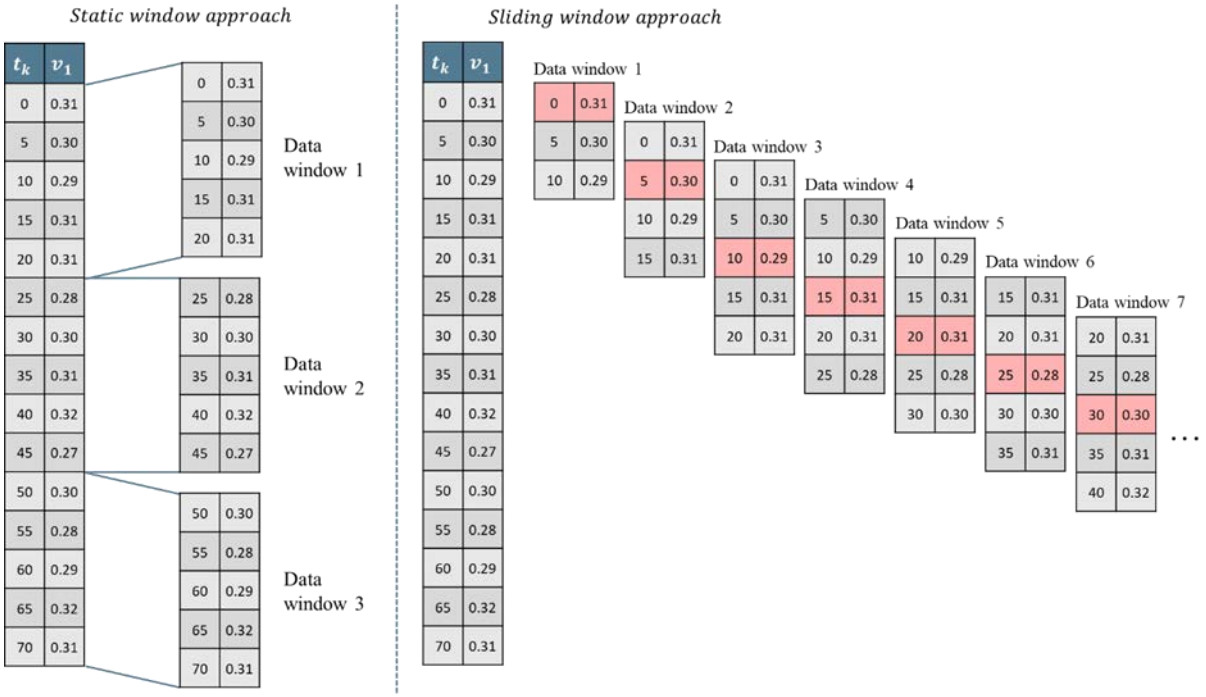


Figure 2.2. *Static window approach vs. sliding window approach.*

When a sliding window is used, the degree of steadiness of each point included in the window is exploited to assess if the variable (v_1 in the example) is at steady-state in the sampling instant t_k located in the middle of the sliding window. The time points located in the middle of the sliding windows are highlighted in red in Figure 2.2. Since assessing the stability of the process in each sampling instant is essential for online application, in the new SSD algorithm it has been decided to implement the sliding window approach.

2.2.2 Data preprocessing

The presence of missing data and invalid measurements in industrial datasets is common and unavoidable. The proposed steady-state detection algorithm is not able to handle incomplete datasets. Hence, a Python™ function was coded to remove the missing values, which commonly are designated as 'NaN' (acronym for Not a Number). The removal is carried out on a row basis. That means that when one sensor (or more than one) experiences a fault and displays a NaN in one sampling instant, the measurements of all the sensors taken at that time point t_k are discarded.

In Figure 2.3, the NaN removal procedure carried out by the developed function is visually summarized. As it can be seen, after the identification and the elimination of the corrupted rows (namely the rows with at least one NaN), three different datasets are obtained. Those three datasets are supplied individually to the steady-state detection algorithm. Otherwise, the constraint of keeping constant the sampling frequency would not be fulfilled (merging the three

matrices would generate a dataset in which for example the sampling instant 1505 is followed by the time point 1520, while the constant sampling period is equal to 5).

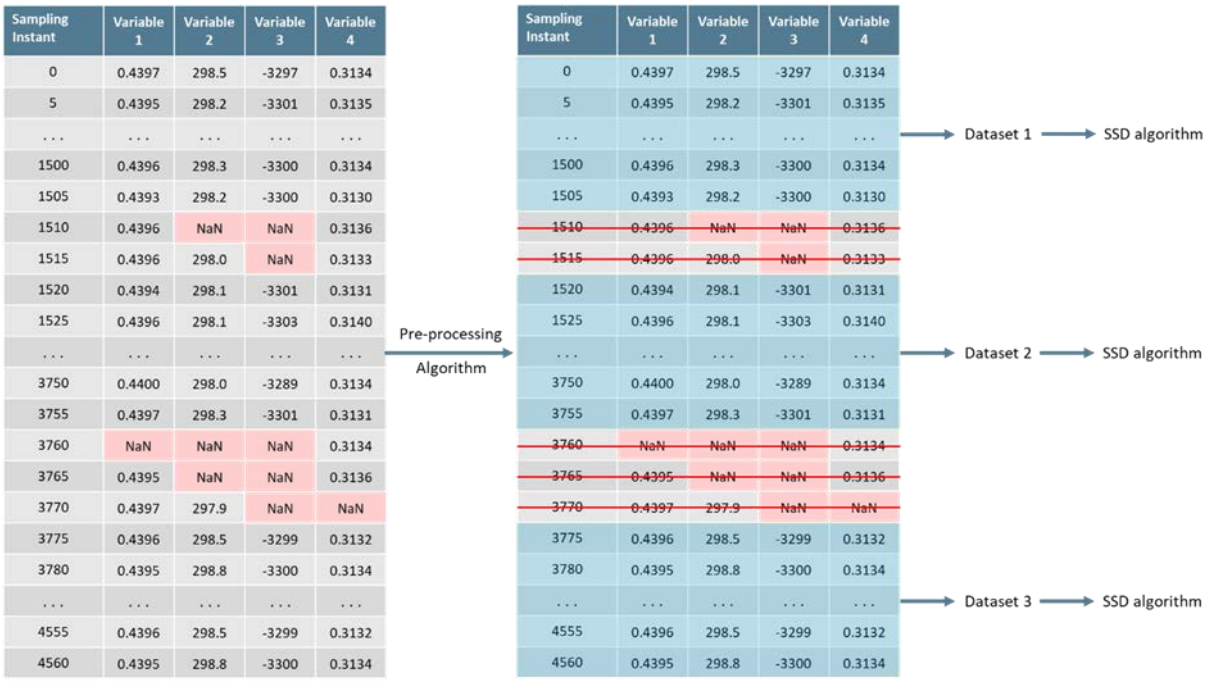


Figure 2.3. *Invalid measurements removal procedure.*

All the industrial datasets upon which the NaN removal procedure was carried out in this work had a low percentage of missing data (< 5%). When more corrupted datasets are taken into account, the removal approach could not be the best choice. Therefore, it is left as a future work the development of a proper and efficient imputation method for missing values replacement.

2.2.3 Algorithm steps

The algorithm inputs and output are summarized graphically in Figure 2.4.

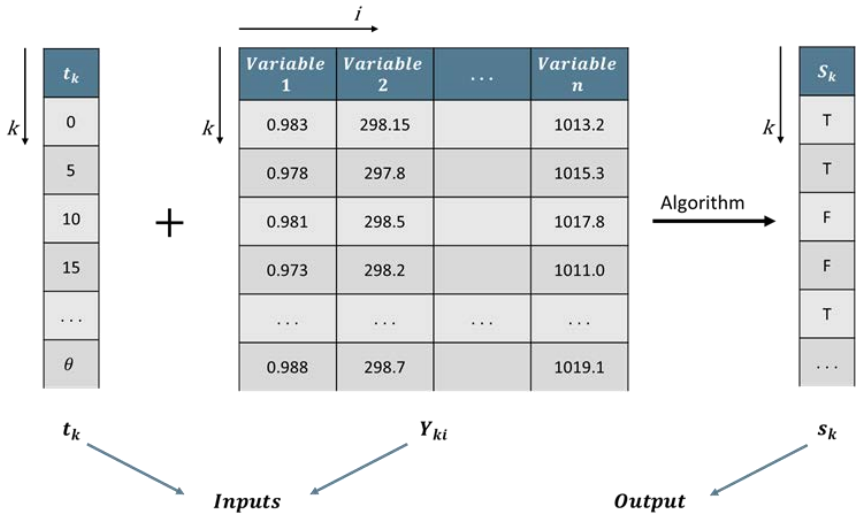


Figure 2.4. *Algorithm inputs and output.*

Given:

- A time series $\{\mathbf{t}_k, \mathbf{Y}_{ki}, k = 0,1, \dots, (\theta - 1);\ i = 0,1, \dots, (n-1)\}$ where $\mathbf{t}_k$ denotes a uniform measurement time grid and $\mathbf{Y}_{ki}$ is a matrix of n measured variables (θ is the total number of sampling instants that are taken into account);
- v : minimum percentage of variance of the original dataset explained by the retained PCs (0%-100%, default 95%);
- N : number of points included in the sliding window;
- α : significance level of the Student's t-test (0%-100%, default 1%);
- T_1: first threshold value (0-1, default 0.93);
- T_2: second threshold value (0-1, default 0.93).

Determine:

- Boolean time series $\{\mathbf{s}_k, k = 0,1, \dots, (\theta - 1)\}$ where if $s_k = 1$ the system is deemed to be at steady-state at the sampling instant t_k while when $s_k = 0$ the system is considered not at steady-state at the sampling instant t_k. The algorithm inputs and output are summarized graphically in Figure 2.4.

1. For every i, compute the mean $\overline{y_i}$ and the standard deviation σ_i over all data points θ.
2. For every k, i, set:

$$y_{ki} = \frac{y_{ki} - \overline{y_i}}{\sigma_i} \quad . \tag{2.8}$$

3. Apply PCA to the data matrix $\mathbf{Y}_{ki}$ to determine:
   - The lowest number of PCs m that ensures that at least v% of the variance of the original dataset is explained.
   - The time series $\{\mathbf{t}_k, \mathbf{X}_{kj}, k = 0,1, \dots, (\theta - 1);\ j = 0,1, \dots, (m-1)\}$ where $\mathbf{X}_{kj}$ is the matrix of the first m PCs.
   - The vector $\{\mathbf{ex\_var}_j, j = 0,1, \dots, (m-1)\}$ that stores the percentage of variance that each of the first m PC explains (when taken into account individually).
4. For every j, set:

$$ex\_var'_j = \frac{ex\_var_j}{\sum_{j=1}^{m} ex\_var_j} \quad . \tag{2.9}$$

5. For every $q = 0\ ,1\ ,2\ ,3\ , \dots\ , (\theta - 1)$:
   5.1. Set:

$$k_s = \max\left(0\,, q - \frac{N-1}{2}\right) . \tag{2.10}$$

$$k_f = \min\left(q + \frac{N-1}{2}\,, \theta\right) \tag{2.11}$$

5.2. Set:

$$q = q + 1 \ . \tag{2.12}$$

5.3. For every PC j:

5.3.1. Use the vectors $\{\mathbf{t}_k, k = k_s\,, \ldots, k_f\}$ and $\{\mathbf{x}_{kj}, k = k_s\,, \ldots, k_f\}$ to estimate a linear model of the form:

$$x_j(t_k) = \mu_j + \gamma_j t_k \ . \tag{2.13}$$

5.3.2. Compute the standard deviation of each de-trended PC over the sliding window:

$$\sigma_j = \sqrt{\frac{1}{(k_f - k_s + 1) - 2}\sum_{k=k_s}^{k_f}\left(x_{kj} - \gamma_j t_k - \mu_j\right)^2} \ . \tag{2.14}$$

$k_f - k_s + 1$ is the number of time points included in the sliding window.

5.3.3. Define the scalar $= 0$.

5.3.4. Calculate t_{crit} using the function *'$stats.t.ppf()$'* imported from the Python™ package *'$scipy$'*:

$$t_{crit} = stats.t.ppf\left(1 - {}^{1}/_{2}\,\alpha\,, k_f - k_s\right) \ . \tag{2.15}$$

t_{crit} is the critical value of the Student's t-test.

5.3.5. For $k = k_s\,, \ldots, k_f$:

5.3.5.1. Perform the following two-tailed Student's t-test:

$$If\ \left|x_{kj} - \left[\,\mu_j + \gamma_j \cdot \left(t_{k_s} - \Delta t_k\right)\right]\right| \le t_{crit}\sigma_j :$$
$$\text{Set } SS = SS + 1 \ . \tag{2.16}$$

where Δt_k is the constant sampling period.

5.3.6. Assess the steadiness of the PC in the sampling instant q:

$$If \frac{SS}{k_f - k_s + 1} > T_1 :$$
$$\text{Set } p = p + ex\_var_j' \ . \tag{2.17}$$

5.4. Assess the steadiness of the entire process in the sampling instant q:

$$If\ p > T_2 :$$
$$\text{Set } s_q = 1 \ .$$
$$Else :$$
$$\text{Set } s_q = 0 \ . \tag{2.18}$$

2.2.4 Steps description

The first and the second steps perform the standardization of the data matrix columns removing the mean and scaling to unit variance, while the third step carry out the PCA through the function *'PCA'* included in the Python™ package '*scikit-learn'*. The dimensionality reduction is computed defining the parameter v: the minimum percentage of variance of the original dataset that must be explained by the retained PCs. The parameter v has a dramatic influence on the performance of the feature extraction approach. A proper selection of its value, indeed, allows to hold the main data information in the PCs leaving only process noise in the residues. In this work, an optimum value for v has been estimated analysing the eigenvalues related to each PC.

As stated by Garcia-Muñoz (2019) the eigenvalue of a PC can be roughly interpreted as the number of variables (of the original dataset) that the component is representing. It is therefore reasonable to neglect the PCs whose eigenvalue is smaller than 0.5 (namely those PCs that represent less than half a variable of the original dataset). Figure 2.5 graphically summarizes through an example how the rule of thumb is applied to estimate the parameter v. As it can be seen in the example, from the fifth PC on, the eigenvalues are smaller than 0.5. Therefore, only the first four PCs should be kept. The values of v that allows keeping four PCs are those bound between 91.0209 and 95.4848 % (the inequality bounds can be easily identified from the column referring to the cumulated retained variance percentages).

Number of PC	Eigenvalue	Retained variance [%]	Cumulated retained variance [%]
1	12.9417	76.12763	76.12763
2	1.4199	8.35237	84.48
3	1.11195	6.5409	91.0209
4	0.75886	4.4639	95.4848
5	0.36685	2.15794	97.64274
6	0.20977	1.23391	98.87666
7	0.0967	0.56884	99.44549
8	0.04605	0.27088	99.71637
9	0.00911	0.05361	99.76998
10	0.00818	0.04811	99.81809
11	0.00707	0.04158	99.85967
12	0.0067	0.03941	99.89908
13	0.0054	0.03176	99.93085
14	0.00466	0.02743	99.95828
15	0.00393	0.02314	99.98142
16	0.00192	0.01129	99.9927
17	0.00124	0.0073	100

Figure 2.5. *Procedure followed to determine the parameter v.*

The fourth step calculates then the relative importance of each retained PC and concludes the section of the algorithm that deals with the PCA. The weights calculated in step 4 will be exploited in later steps to calculate a composite steady-state detection index and therefore to assess the steadiness of the entire process.

From the fifth step on, the section of the algorithm that deals with the steady-state assessment begins. Step 5, in particular, defines an iterator that runs across all the sampling instants, while step 5.1 and 5.2 deal with the implementation of the sliding window. Step 5.3, then, defines an iterator that runs across the PCs. After one PC is selected, its steadiness in the sliding window is assessed through a procedure styled after Kelly and Hedengren's approach (which has been described in detail in §2.1.2.2). In step 5.3.5, the equation implemented to carry out the Student's t test in the novel approach (2.16) is slightly different from the original one proposed by Kelly and Hedengren (2.3). The new expression reduces the computational burden of the algorithm avoiding to define a new relative time (running within the window) whenever a new time window is considered. Through equation (2.16), moreover, the code is enabled to deal directly with datasets where the constant sampling period is different from one.

Once the statistical test has been computed for each sampling instant included in the sliding window, in step 5.3.6 the steadiness of the PC in the time point q (located in the center of the sliding window) is assessed checking if the fraction of points at steady-state within the sliding window overcomes the threshold value T_1. Still in step 5.3.6 a composite steady-state index p is calculated to evaluate with a number bound between zero and one the steadiness of the entire process (in the time point q). Since the computationally inexpensive way through which the

index p is computed in the final version of the code (step 5.3.6) could be difficult to understand, an extended derivation is provided in §2.2.4.1.

Finally, in the last step of the method, the steady-state composite index is compared with a second threshold value T_2 obtaining the final algorithm output: the time series $\{\mathbf{s}_k, k = 0,1,\dots,(\theta-1)\}$.

2.2.4.1 Derivation of the composite steady-state index p

Originally, in the new method, the calculation of the composite steady-state index p and of the Boolean time series $\mathbf{s}_k$ were carried out outside the loop that iterates across each sampling instant. At the end of step 5, indeed, it was only computed a Boolean matrix $\{\mathbf{R}_{kj}, k = 0,1,\dots,(\theta-1);\ j = 0,1,\dots,(m-1)\}$ where the single element r_{kj} was set equal to one whenever the PC j was deemed to be at steady-state in the sampling instant k and zero otherwise. Differently from the latest algorithm version, moreover, the values of the composite steady-state index p (one value for each sampling instant) were all calculated through a single operation and stored in a dedicated vector $\mathbf{p}_k$. The single operation (which was present in the older algorithm versions) that allowed to compute the vector $\mathbf{p}_k$, as showed in Figure 2.6, is a weighted average of the element of the matrix $\mathbf{R}_{kj}$:

$$\mathbf{p}_k = \sum_{j=0}^{m-1} r_{kj} \cdot ex\_var'_j \quad . \tag{2.19}$$

The weights $ex\_var'_j$ are the normalized percentages of explained variance computed in step 4. Therefore, the composite steady-state detection index is merely an average of the predictions of each PC weighted by their relative importance.

Finally, once carried out equation (2.19), the older algorithm versions computed the Boolean time series $\mathbf{s}_k$ testing if the value of each of the element of the vector $\mathbf{p}_k$ exceeded the threshold value T_2. The whole procedure followed to obtain $\mathbf{s}_k$ and $\mathbf{p}_k$ starting from the matrix $\mathbf{R}_{kj}$ is summarized graphically in Figure 2.6.

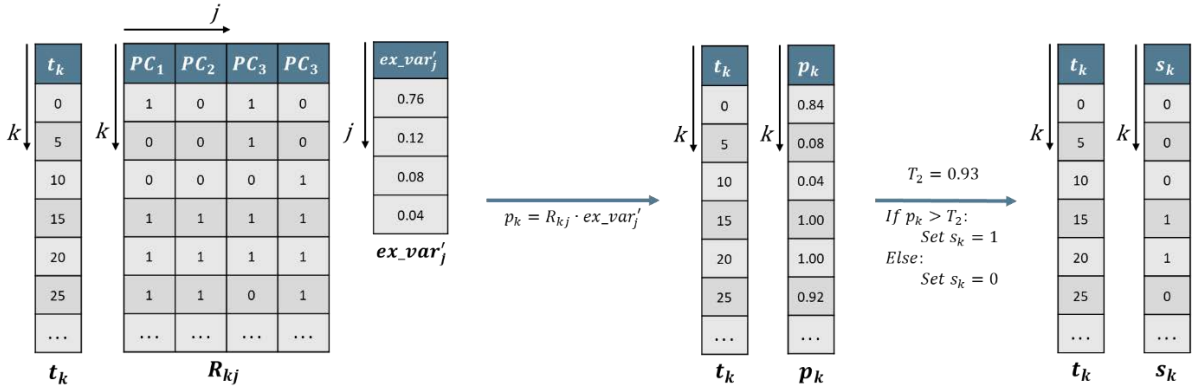


Figure 2.6. *Procedure followed to* $\mathbf{s}_k$ *and* $\mathbf{p}_k$ *starting from* $\mathbf{R}_{kj}$.

Although the definition of the composite steady-state index stays unchanged, in the final version of the code, through step 5.3.6 and 5.4, the calculation of the Boolean time series $\mathbf{s}_k$ is computed directly inside the loop that iterates across each sampling instant without explicitly defining the matrix $\mathbf{R}_{kj}$ and without collecting the values of the composite SS index in the vector $\mathbf{p}_k$. This allowed achieving a significant reduction of the algorithm running time.

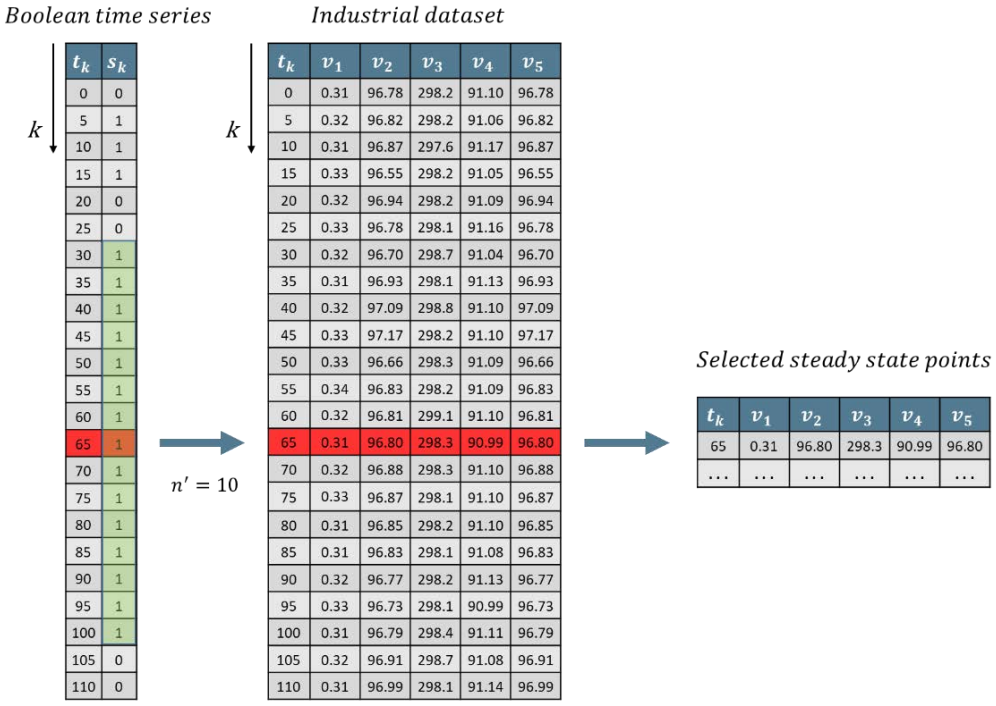
2.2.5 Representative steady-state points collection

Once the algorithm has provided the user with the boolean time series that stores the steadiness predictions for the whole system in each sampling instant, commonly a considerable amount of steady-state points is achieved. In most of the cases, however, such as for data reconciliation, not all the points are needed to carry out later analyses.

Therefore, in order to retrieve from the Boolean time series some representative steady-state points, a Python™ function was developed to:

1. Identify the intervals of time in which the process is at steady-state for more than n' consecutive time points;
2. Determine the sampling instant located in the middle of those steady-state windows;
3. Collect from the industrial dataset the measurements of all the process variables (both the ones employed for steady-state detection and those that have not been exploited) in that sampling instant.

The value of the parameter n' depends on the sampling period and on the amount of steady-state points that it is wanted to collect. The steps carried out by the function developed to collect representative steady-state points are graphically summarized in Figure 2.7.



t_k	s_k
0	0
5	1
10	1
15	1
20	0
25	0
30	1
35	1
40	1
45	1
50	1
55	1
60	1
65	1
70	1
75	1
80	1
85	1
90	1
95	1
100	1
105	0
110	0

t_k	v_1	v_2	v_3	v_4	v_5
0	0.31	96.78	298.2	91.10	96.78
5	0.32	96.82	298.2	91.06	96.82
10	0.31	96.87	297.6	91.17	96.87
15	0.33	96.55	298.2	91.05	96.55
20	0.32	96.94	298.2	91.09	96.94
25	0.33	96.78	298.1	91.16	96.78
30	0.32	96.70	298.7	91.04	96.70
35	0.31	96.93	298.1	91.13	96.93
40	0.32	97.09	298.8	91.10	97.09
45	0.33	97.17	298.2	91.10	97.17
50	0.33	96.66	298.3	91.09	96.66
55	0.34	96.83	298.2	91.09	96.83
60	0.32	96.81	299.1	91.10	96.81
65	0.31	96.80	298.3	90.99	96.80
70	0.32	96.88	298.3	91.10	96.88
75	0.33	96.87	298.1	91.10	96.87
80	0.31	96.85	298.2	91.10	96.85
85	0.31	96.83	298.1	91.08	96.83
90	0.32	96.77	298.2	91.13	96.77
95	0.33	96.73	298.1	90.99	96.73
100	0.31	96.79	298.4	91.11	96.79
105	0.32	96.91	298.7	91.08	96.91
110	0.31	96.99	298.1	91.14	96.99

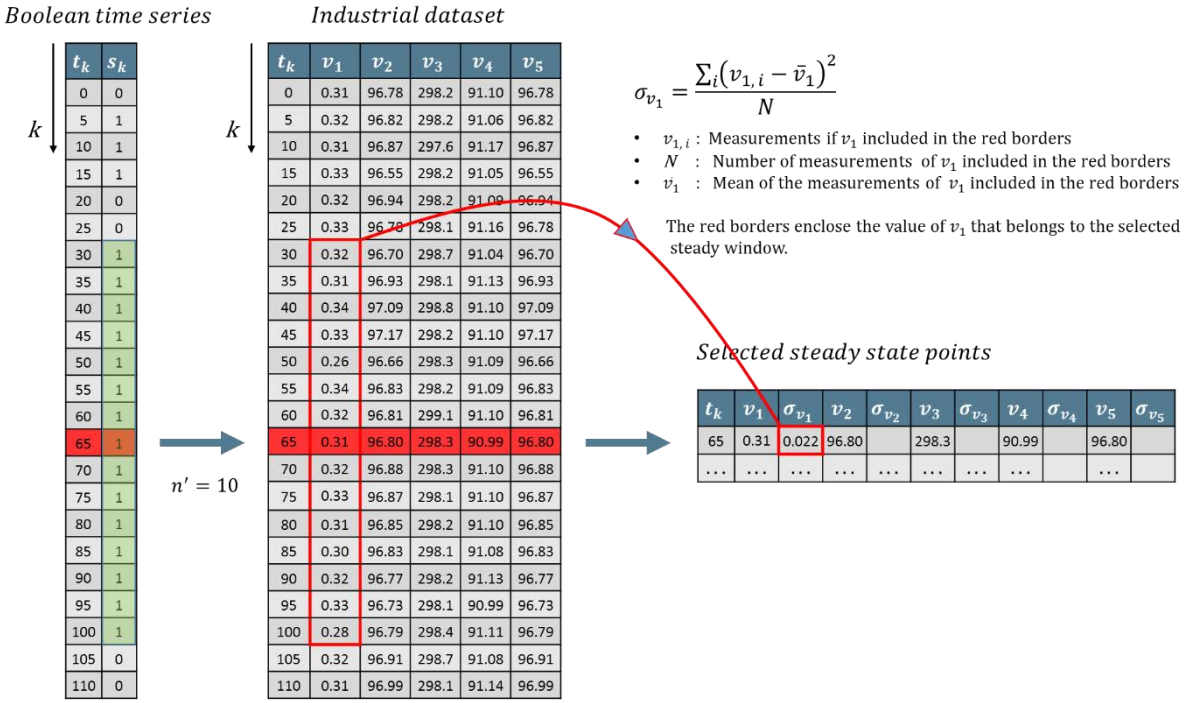
t_k	v_1	v_2	v_3	v_4	v_5
65	0.31	96.80	298.3	90.99	96.80
…	…	…	…	…	…

Figure 2.7. *Steady-state points collection procedure.*

Additional code lines were later added to the function in order to also:

1. For each steady-state point and for each variable;
   1.1. Identify the steady window from which the measurement was collected;
   1.2. Calculate the standard deviation of the measurements included in that steady window.

The so computed standard deviations are intended to be accurate estimations of the measurement noise affecting each collected steady-state point in the steady window from which it was retrieved. The additional feature of the function is demonstrated in Figure 2.8 carrying out graphically the calculation of the standard deviation of the first variable v_1 (for the first collected steady-state point).



t_k	s_k
0	0
5	1
10	1
15	1
20	0
25	0
30	1
35	1
40	1
45	1
50	1
55	1
60	1
65	1
70	1
75	1
80	1
85	1
90	1
95	1
100	1
105	0
110	0

t_k	v_1	v_2	v_3	v_4	v_5
0	0.31	96.78	298.2	91.10	96.78
5	0.32	96.82	298.2	91.06	96.82
10	0.31	96.87	297.6	91.17	96.87
15	0.33	96.55	298.2	91.05	96.55
20	0.32	96.94	298.2	91.09	96.94
25	0.33	96.78	298.1	91.16	96.78
30	0.32	96.70	298.7	91.04	96.70
35	0.31	96.93	298.1	91.13	96.93
40	0.34	97.09	298.8	91.10	97.09
45	0.33	97.17	298.2	91.10	97.17
50	0.26	96.66	298.3	91.09	96.66
55	0.34	96.83	298.2	91.09	96.83
60	0.32	96.81	299.1	91.10	96.81
65	0.31	96.80	298.3	90.99	96.80
70	0.32	96.88	298.3	91.10	96.88
75	0.33	96.87	298.1	91.10	96.87
80	0.31	96.85	298.2	91.10	96.85
85	0.30	96.83	298.1	91.08	96.83
90	0.32	96.77	298.2	91.13	96.77
95	0.33	96.73	298.1	90.99	96.73
100	0.28	96.79	298.4	91.11	96.79
105	0.32	96.91	298.7	91.08	96.91
110	0.31	96.99	298.1	91.14	96.99

t_k	v_1	σ_{v_1}	v_2	σ_{v_2}	v_3	σ_{v_3}	v_4	σ_{v_4}	v_5	σ_{v_5}
65	0.31	0.022	96.80		298.3		90.99		96.80	
…	…	…	…	…	…	…	…		…	

Figure 2.8. *Measurement's noise estimation.*

In Figure 2.8 the measurements of v_1 included in the steady-state window are enclosed in red borders.

2.2.6 Algorithm performances assessment

Once the steadiness of the process has been assessed through the proposed steady-state identification algorithm, the quality and precision of the detection can be evaluated comparing the predicted running state with the trajectories of the PCs or of the process variables.

In particular, three types of plots are provided to the user at the end of the detection routine:

1. The comparison of the trajectory of a single PC with the steadiness prediction;
2. A grouped comparison of the trajectories of all the retained PCs with the steadiness prediction;
3. A comparison of the trajectory of a single process variable with the steady-state prediction.

Hereinafter, an example of each type of graph is provided. The plots that will be presented, in particular, are referred to the cumene process case study. In the cumene process case study, no industrial datasets were employed for the development of the hybrid model, but digital process signals were created simulating a dynamic flowsheet in gPROMS through an accurate first principle model. The first principle model that was employed refers to the design proposed by Luyben (2010) and was available inside the Process Systems Enterprise's libraries.

Disturbances in the feed were employed to introduce transient states, while noises and invalid measurements were added at a later date through Excel and Python™. Further information regarding how the data have been generated, how transient states have been induced, how noise and invalid measurements have been added to the dataset and a detailed description of the process flowsheet are given in §3, the chapter which is dedicated to the industrial plant for the production of cumene.

For a proper understanding of the plots, additional details regarding the dataset structure must be provided. The digital plant historian collects the values of 83 different process variables (temperatures, pressures, flowrates, compositions and heat duties). Each variable was virtually measured 33050 different times with a constant sampling period of 120 seconds. The percentage of invalid measurements was around 0.01%. Among the 83 available process signals, 17 were exploited to carry out steady-state detection. The PCA was performed so as to keep at least 95% of the variance of the original dataset. That involved the collection of four PCs. The values of all the other algorithm parameters are reported in Table 2.1.

Table 2.1. *Algorithm parameters.*

$window\_length$	α	T_1	T_2
139	0.01	0.93	0.93

The plots reported hereinafter refers to a submatrix of the virtual plant historian collecting 3750 samples. The algorithm assessed the steadiness of the process in all the sampling instants with a running time of 60 seconds (laptop Lenovo T450 with the processor Intel Core i7-5600U vPro). In Figure 2.9, the trajectory of the first PC is displayed together with the predicted running state. The steadiness forecasts are visualized in the lower part of the graph through a histogram: whenever the process is deemed to be at steady-state between two consecutive sampling instants a blue bar with height equal to 1 is displayed between those two time points. Otherwise, no bars are plotted.

As it can be seen from Figure 2.9, through visual inspection of the first PC trajectory it can be stated that transient behavior arises roughly when the time over the sampling period is:

- between 400 and 600;
- between 800 and 1000;
- between 1200 and 1400;
- between 1600 and 1700;
- between 2100 and 2200;
- between 2500 and 2600;
- between 2800 and 3000;
- between 3300 and 3500.

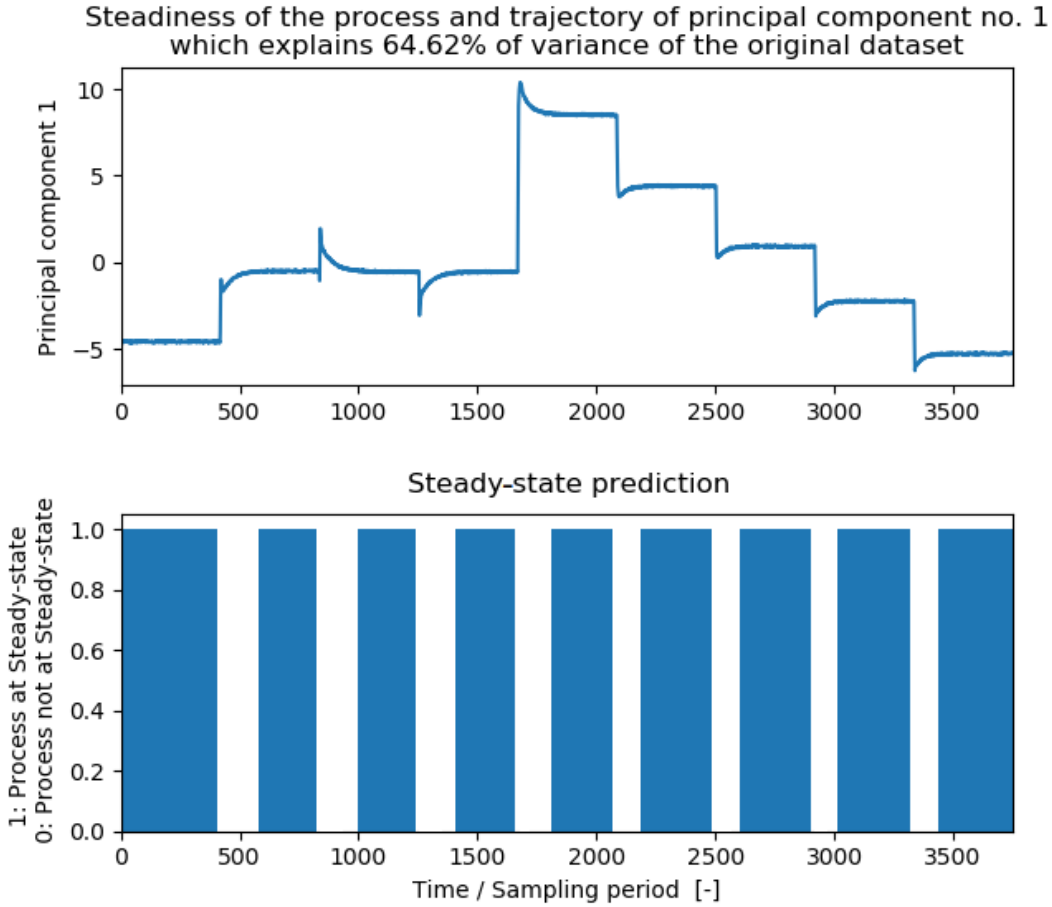


Figure 2.9. *Trajectory of the first PC and steady-state predictions.*

Since the algorithm prediction matches quite accurately the evaluation achieved through visual inspection, it can be concluded that the novel method is able to handle efficiently heavy data loads providing correct assessments of the steadiness of the overall process. Minor detection issues can be however noticed. Indeed, although the method is able to detect almost instantaneously when a disturbance enters the system (displaying promptly the 'not at steady-state' condition), in the regions in which the transient is expiring (and the system is starting to attain a new steady-state) it deems to be at steady-state intervals of times in which the effect of

the disturbance has not completely vanished. This misprediction, however, do not pose any issues because, as described in §2.2.5, the points selected to carry out further analyses are picked from the middle of the steady intervals, where the reliability of the prediction is ensured. In the second kind of plots the trajectories of all the retained PCs are simultaneously displayed together with the steadiness assessments. Those kinds of graphs are particularly helpful when the PCs show transient behaviours in different times. Analysing only the trajectory of a single PC, indeed, one could find intervals of time in which, despite the trajectory of the PC is steady, the whole system has been deemed to be not at steady-state. This commonly do not happen because of an algorithm forecasting failure, but rather because another PC is experiencing transient behaviours in that specific interval of time. Plotting all the trajectories of the PCs allow the user to easily identify which PCs are bringing the whole system to unsteadiness in a given sampling instant. In the reported example, however, the transient behaviours arise for all the latent variables almost at the same time, hence the usefulness of the grouped visualization is not highlighted.

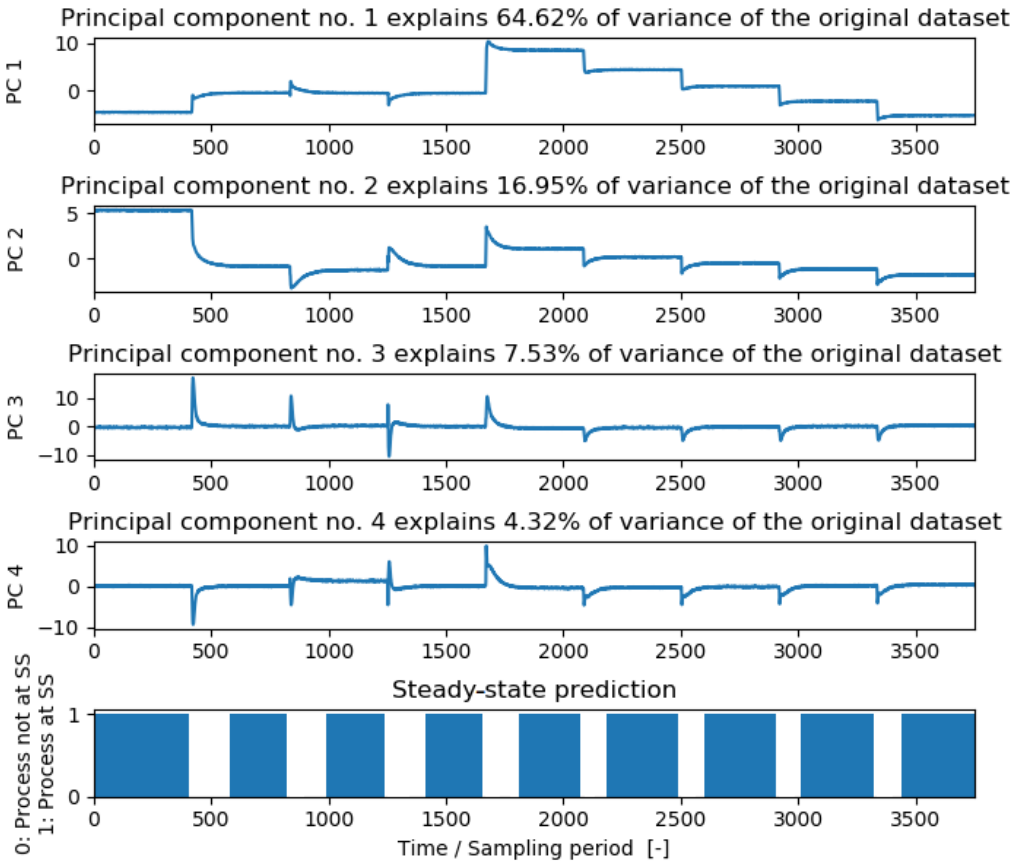


Figure 2.10. *Trajectories of the retained PCs and steady-state predictions.*

As for the first kind of plots, in any case, also through Figure 2.10 the reliability of the steadiness assessment can be tested comparing the predictions expressed by the histogram with what it can be stated by inspecting visually the trajectories.

Lastly, in Figure 2.11 (an example of the third kind of plots), the trajectory of a process variable, the molar flowrate of cumene (the desired product), is displayed together with the steady-state predictions.

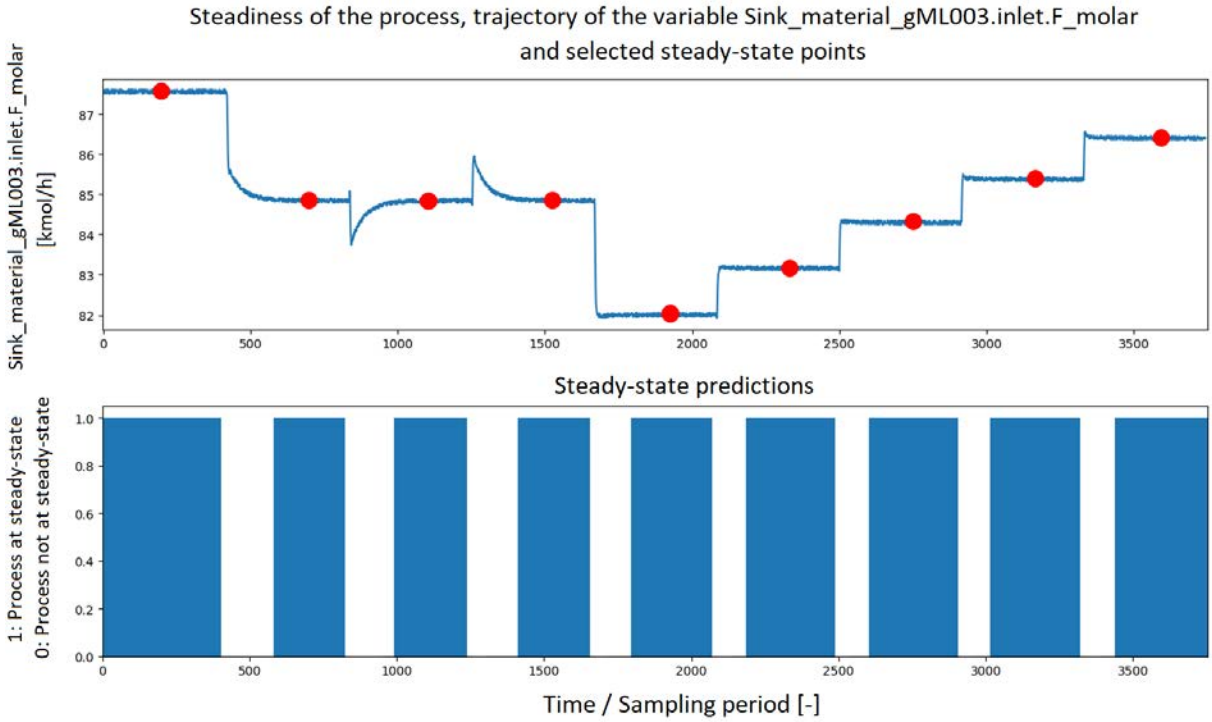


Figure 2.11. *Process variable trajectory, collected steady-state points and steadiness predictions.*

The most important feature of this type of graph consists on the presence of the red dots. The red dots indicate the representative steady-state points collected (with the function explained in §2.2.5) to carry out further analyses. The third type of plots are therefore essential for the user in order to quickly verify if the points retrieved for later applications are affected by major prediction issues or, as it is desired, truly belong to a steady window.

Chapter 3

Industrial process for the production of cumene

This Chapter overviews the industrial process for the production of cumene and the methodology through which the process data employed for the development of the hybrid model of the plant have been collected. First, details regarding the kinetics of the reactions involved in the process and the phase equilibrium are given. Then the flowsheet and the plantwide control strategy are presented. Finally, the procedure through which the process data are collected is described.

3.1 Reaction kinetics and phase equilibrium

Cumene (isopropylbenzene) is a colourless water-insoluble aromatic hydrocarbon used in the manufacture of several chemicals including phenol and acetone (https://pubchem.ncbi.nlm.nih.gov/compound/Cumene). The cumene manufacturing process involves the following two reactions:

$$C_6H_6 + C_3H_6 \rightarrow C_9H_{12} \ , \tag{3.1}$$

$$C_9H_{12} + C_3H_6 \rightarrow C_{12}H_{18} \ . \tag{3.2}$$

The main reaction is the Friedel-Crafts alkylation of benzene with propylene to produce cumene (3.1) while the alkylation of cumene with propylene (3.2) is an undesired reaction that forms *p*-diisopropylbenzene. The reactions are carried out in a high temperature, high pressure gas-phase reactor in the presence of a solid catalyst.
The rates of reaction are expressed as follows:

$$r_1 = K_1 e^{-\frac{E_1}{RT}} C_{C_6H_6} C_{C_3H_6} \ , \tag{3.3}$$

$$r_2 = K_2 e^{-\frac{E_2}{RT}} C_{C_9H_{12}} C_{C_3H_6} \ , \tag{3.4}$$

Both reaction rates have unit of [kmol/(m^3·s)]. The values of the activation energy E_i and of the pre-exponential factor K_i are reported in Table 3.1.

Table 3.1. *Kinetic parameters. From: Luyben (2010).*

	Main reaction R1	**Side reaction R2**
Pre-exponential factor K_i $[m^3/kmol \cdot s]$	2.8×10^7	2.32×10^9
Activation energy E_i $[kJ/kmol]$	104174	146742

As can be noticed in Table 3.1, the activation energy of the undesired reaction is larger than the one of the main reaction; therefore, low temperatures improves selectivity (Luyben, 2010).
The selectivity can be also enhanced keeping the amount of cumene and propylene low in the reactive section. This can be achieved operating the plant with a large excess of benzene, but in order to keep the process economically feasible, after the reactor, the excess must be properly recovered and recycled.
As regards the thermodynamics, the normal boiling points of benzene, cumene and *p*-diisopropylbenzene are 80.1, 152.4 and 209 °C, respectively. Therefore, under the assumption that the separation of propane and propylene is uneconomical (Luyben, 2010), all the other separations are fairly easily achievable through standard distillation.

3.2 Flowsheet

Figure 3.1 shows the optimized flowsheet of the industrial plant for the production of cumene with the process operating conditions and the equipment sizes. As it can be noticed in the flowsheet, the fresh streams of pure benzene and mixed C3 (propylene and propane) enter the plant as liquids. The feed 'Fresh C3' has a flowrate equal to 101.93 kmol/h while its composition is 95% propylene and 5% propane on a molar basis. The fresh feed of pure benzene, instead, is equal to 98.78 kmol/h (Luyben, 2010).
The liquid fresh feeds are mixed with a liquid recycle stream (the distillate of the first distillation column C1) and together are sent to a vaporizer. The saturated gas exits the vaporizer at 210 °C and 25 bar and then is pre-heated through two heat exchangers. The first heat exchanger, named 'FEHE', recovers heat from the stream exiting the reactor while the second heat exchanger (HX1) raises the temperature of the gaseous stream entering the reactor up to 358°C.
The reactor is a tubular cooled reactor that generates high-pressure steam exploiting the heat released by the exothermic reactions. The reactor is made of 1500 tubes with a length of 6 meters and filled with a solid catalyst. The coolant steam is assumed to enter the reactor with the same temperature of the reactants mixture (358°C).
Once left the tubular reactor at 358.5°C, then, the mixture is cooled down to 279°C in the feed-effluent heat exchanger (FEHE) and then it is sent to a condenser. In the condenser the temperature is further decreased down to 90°C using cooling water. In the meanwhile, a valve decreased the pressure of the stream from 25 to 1.75 bar (Luyben, 2010).

After the condenser the two-phase stream is fed to a flash tank. The task of the flash operation is to remove the propane which enters continuously the plant in the fresh C3 stream. Since propane does not react, indeed, if not vented, it would accumulate in the plant.

The liquid exiting the flash drum is then fed to the first distillation column C1. The first distillation column has 15 stages, an operating pressure of 1.75 bar and a reflux ratio equal to 0.44. Its distillate is mostly benzene (95.2% on a molar basis) and is recycled back to the reactor. Since the target cumene purity is equal to 99.9% mol, the first distillation column must prevent benzene from dropping out of the bottom. It is therefore required that the composition of benzene in the residue is equal to 0.05% mol.

The residue of the first column is then fed into a second distillation column. The task of the second distillation column is to attain a high purity cumene in the distillate (99.9% mol) and minimize the loss of cumene in the bottoms.

Moreover, the second column has 20 stages and operates at a pressure equal to 1 bar and with a reflux ratio of 0.63 (Luyben, 2010).

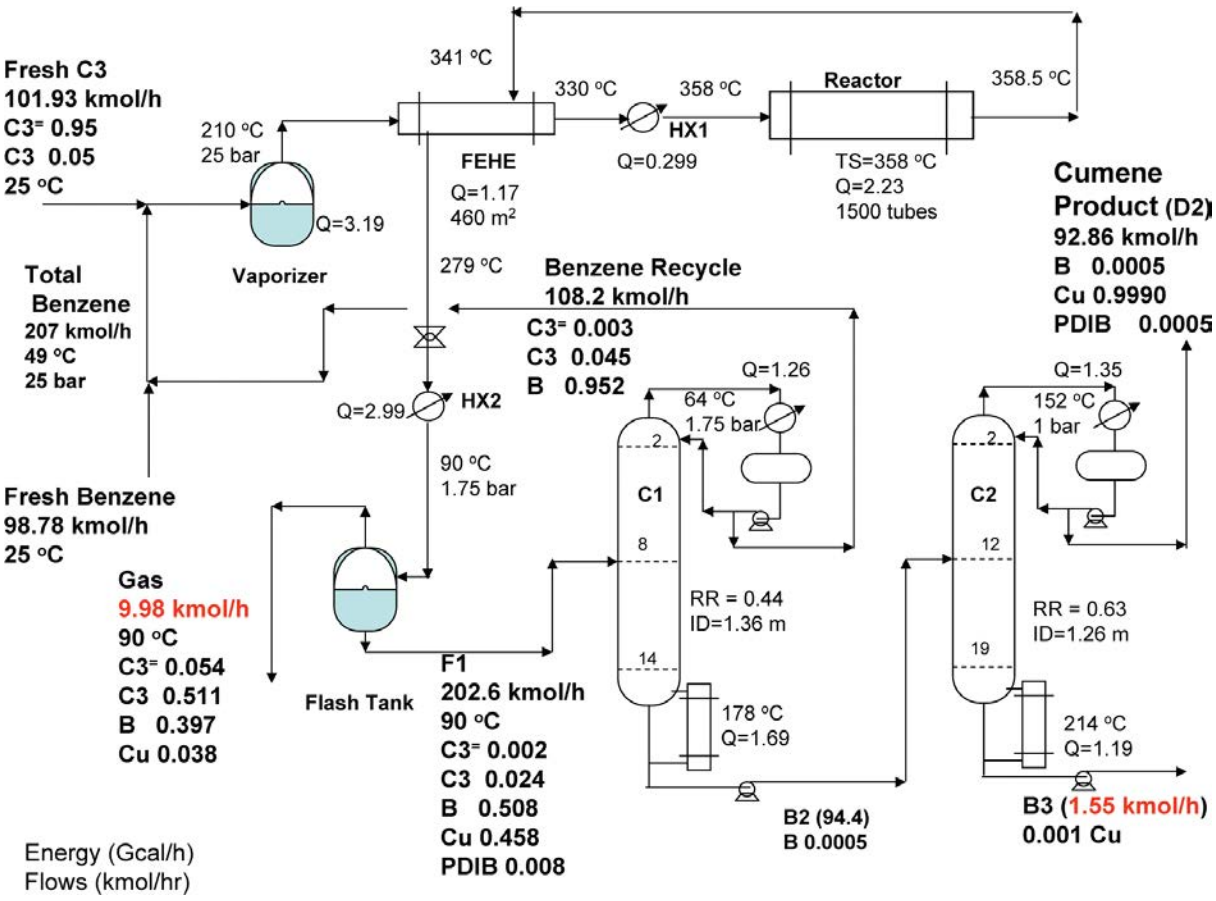


Figure 3.1. *Optimized flowsheet for the cumene process production proposed by Luyben (2010). From: Luyben (2010).*

3.2.1 Plantwide control strategy

The plantwide control structure is summarized in Figure 3.2. indicating in the flowsheet all the controllers that are implemented in the plant.

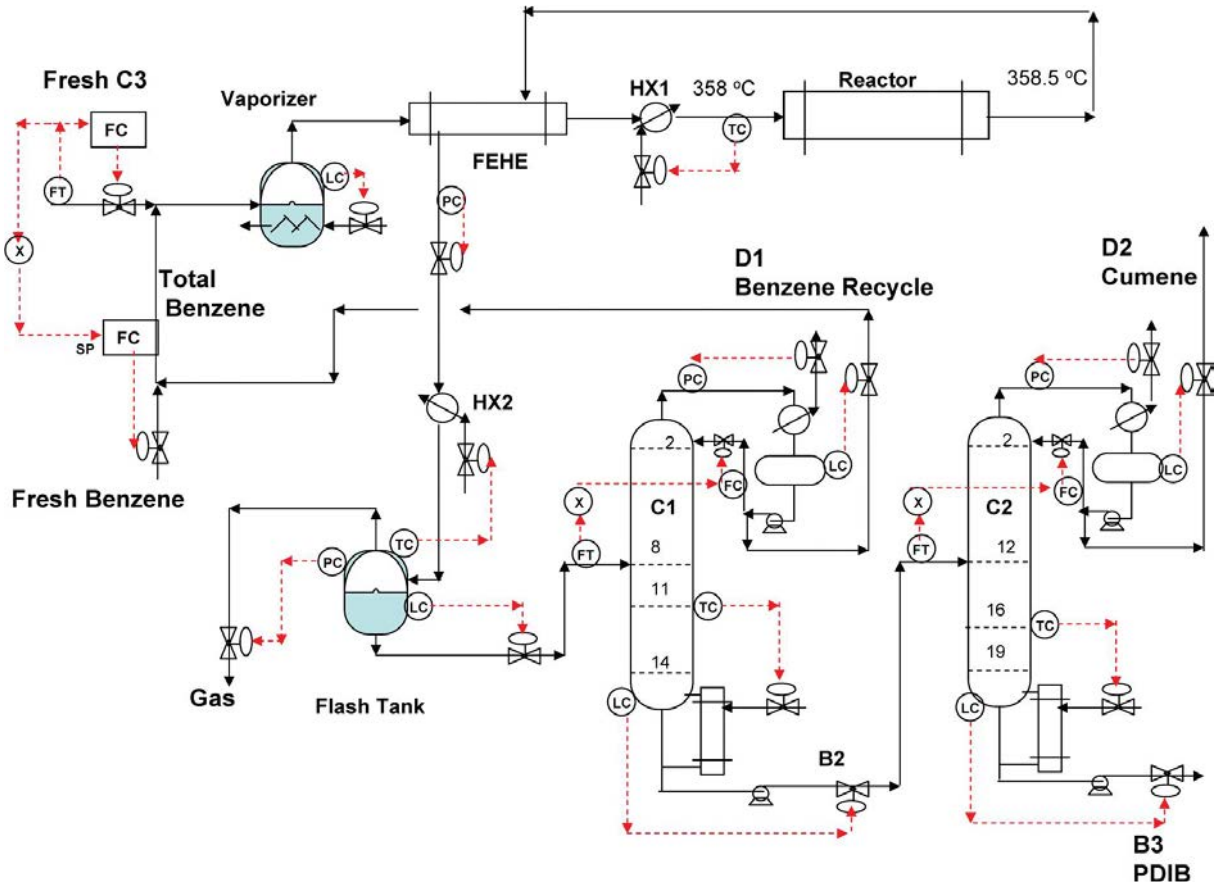


Figure 3.2. *Plantwide control structure proposed by Luyben (2010). From: Luyben (2010).*

Hereinafter the control loops are listed pointing out the manipulated and the controlled variables (Luyben, 2010):

1. The flowrate of fresh C3 is controlled acting on the valve stem position;
2. The total benzene flowrate (the fresh benzene plus the recycled) is ratioed to the flowrate of fresh C3;
3. The level in the vaporizer is controlled through its heat duty;
4. The reactor inlet temperature is controlled manipulating the heat duty of the heat exchanger HX1;
5. The reactor outlet temperature is controlled manipulating the temperature of the coolant steam;
6. The pressure of the stream with the reacted mixture is controlled acting on the stem of the valve located after the FEHE heat exchanger;

7. The temperature in the flash drum is controlled acting on the heat duty of the condenser HX2;
8. The pressure in the flash drum is controlled acting on the flowrate of the vented gases;
9. The liquid level in the flash drum is controlled acting on the flowrate of the liquid effluent;
10. The temperature in the 11^{th} stage of the first distillation column is controlled manipulating the reboiler heat duty;
11. The temperature in the 16^{th} stage of the second distillation column is controlled manipulating the reboiler heat duty;
12. The reboiler liquid level of both columns is controlled acting on the residues flowrates;
13. The condenser liquid level of both columns is controlled manipulating the distillates flowrates;
14. Reflux flowrates of both distillation columns are ratioed to the column feed.

3.3 Data acquisition

Digital process signals were generated performing in gPROMS dynamic simulations of a detailed first principles model of the cumene production plant. The first principles model that was employed reproduces accurately the design and the control strategy proposed by Luyben (2010) and was already available inside the Process Systems Enterprise's libraries.
Once performed the simulations, in order to obtain a dataset, which is as similar as possible to a real industrial dataset, it was decided to corrupt the virtual process signals provided by gPROMS adding noise and invalid measurements. In the following sections it is firstly described the how the dynamic simulations have been carried out and then the rationale through which the noise and the invalid measurements have been added to the virtual process signals is overviewed.

3.3.1 Dynamic simulation

The virtual set of process data have been generated in gPROMS simulating a first principles model that, as it has been previously pointed out (§3.3), accurately reproduces the flowsheet and the control strategy proposed by Luyben (2010).
Figure 3.3 shows the model of the industrial plant for the production of cumene implemented in gPROMS simulation interface.

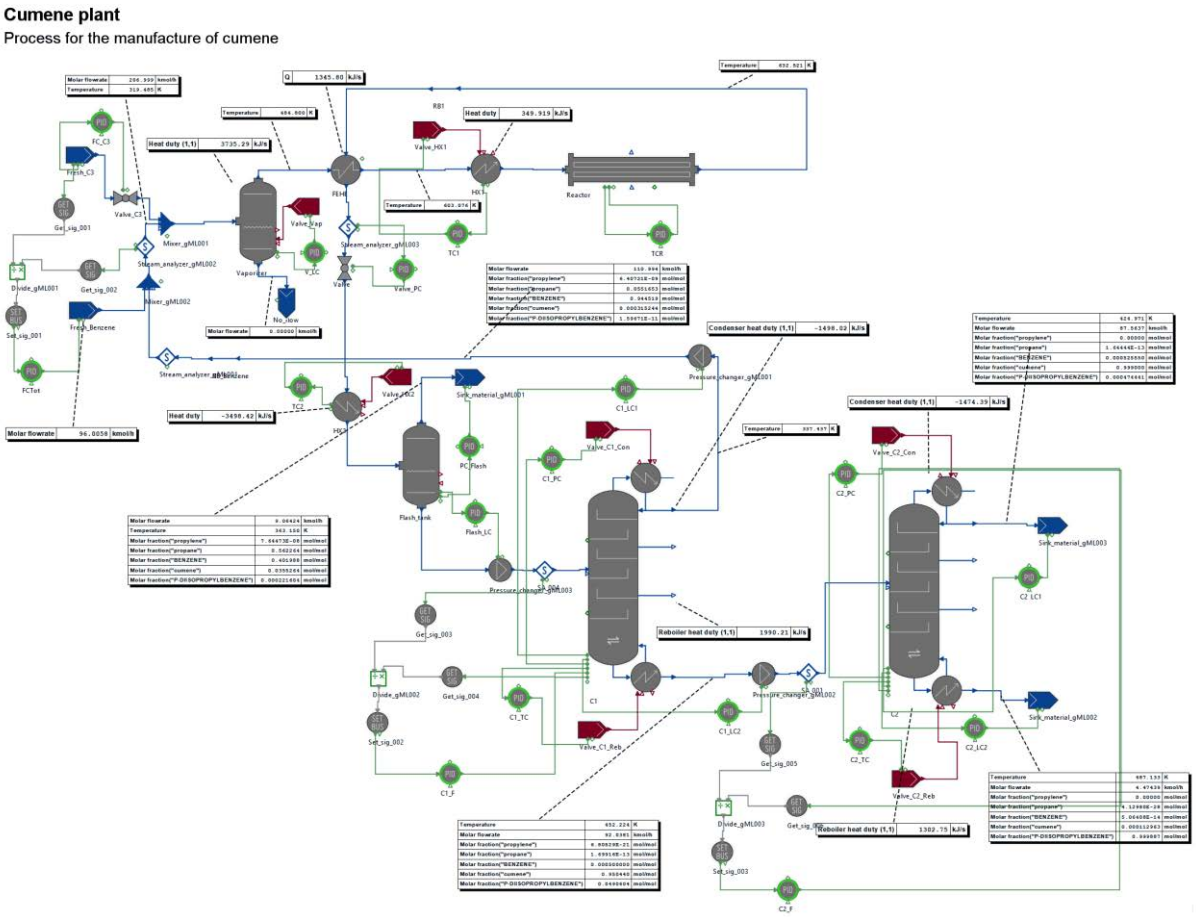


Figure 3.3. *First principles model of the industrial plant for the production of cumene implemented in gPROMS.*

In order to obtain information at different operating points, the dynamic simulations were carried out introducing disturbances in the flowrate, temperature and composition of the fresh C3 feed. After each disturbance enters the plant, the flowsheet is run without introducing changes until a new steady-state operating point is attained.

The schedule of the dynamic simulations is listed hereinafter pointing out for each disturbance the shape of the perturbation and the affected input variable:

1. The flowrate of the fresh C3 feed is increased or decreased by x% with respect to its nominal value through a rampchange;
2. The temperature of the fresh C3 feed is raised to 45°C through a ramp change;
3. The temperature of the fresh C3 feed is decreased back to its nominal value (25°C) through a ramp change;
4. The molar fraction of propylene in the fresh C3 feed is decreased to 0.90 through a stepchange;
5. The molar fraction of propylene in the fresh C3 feed is raised to 0.92 through a stepchange;

6. The molar fraction of propylene in the fresh C3 feed is raised to 0.94 through a stepchange;
7. The molar fraction of propylene in the fresh C3 feed is raised to 0.96 through a stepchange;
8. The molar fraction of propylene in the fresh C3 feed is raised to 0.98 through a stepchange.

where with concern to the x value in point 1, 9 different dynamic simulations were carried varying the magnitude of the first disturbance. In the first simulation the flowrate was left unchanged ($x = 0$). In the second set of six simulations the flowrate was increased by 4.5, 8, 10, 12, 15 and 17% with respect of its nominal value, respectively. Finally, in the last two simulations, the flowrate was decreased respectively by 3 and 5% with respect to its nominal value. The magnitude of all the perturbations (including the composition stepchanges and the temperature ramp increases/decreases) have been selected arbitrarily with the purpose to stick to what it could really happen in an industrial plant. The gPROMS interface through which the schedule listed above has been implemented is reported in Figure 3.4.

Figure 3.4. *Schedule of the gPROMS dynamic simulation in which the fresh C3 flowrate is decreased by 3% ($x = -3\%$).*

Furthermore, Figure 3.5 shows the effect of the disturbances on the purity of the cumene that leaves as distillate the second distillation column.

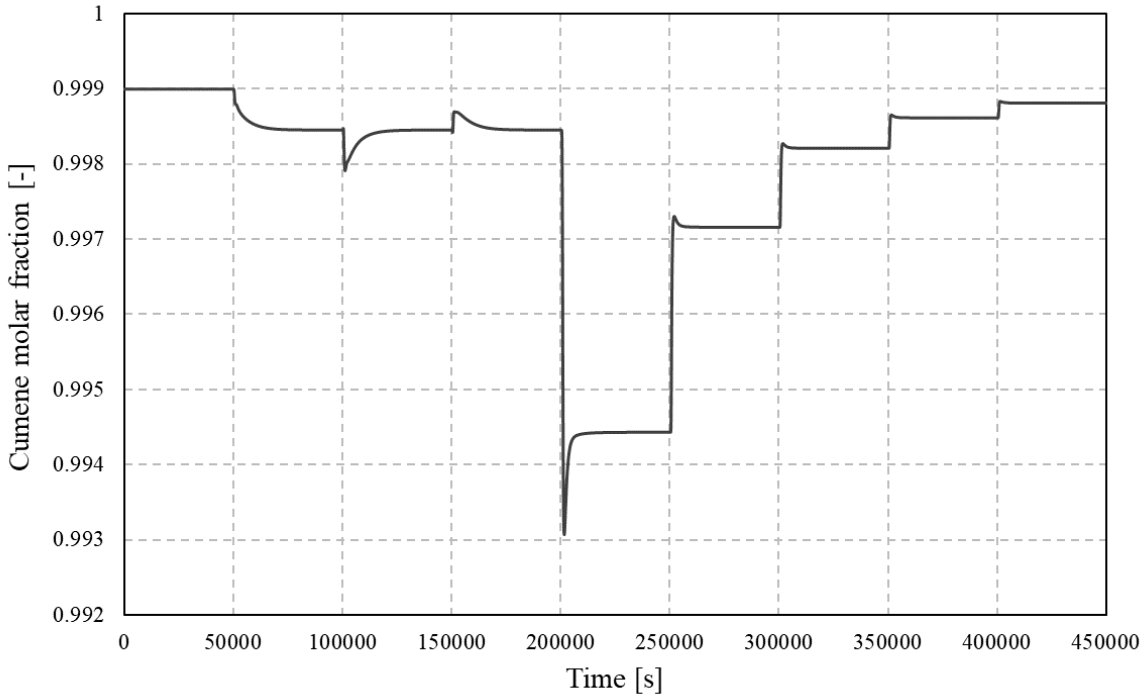


Figure 3.5. *Cumene molar fraction as a function of time ($x = -3\%$).*

The dynamic simulations have been performed setting 120 seconds as reporting interval, therefore each process variable is sampled every two minutes. Whenever a simulation is finished, an execution output is displayed in the gPROMS interface. From the execution output the user has access to the trajectories of all the variables that are taken into account by the process simulator in the model.

Once all the dynamic simulations ended, from each execution output, a table collecting the values of 83 process variables was exported and saved as a CSV file. The set of 83 process variables includes: the reflux ratios of the distillation columns, the heat duties of the condensers and the of the reboilers and the temperature, pressure, flowrate and composition of some relevant streams. Examples of relevant streams are the fresh C3 feed, the distillates, the residues and the feeds of the distillation columns and the reactor inlet and outlet.

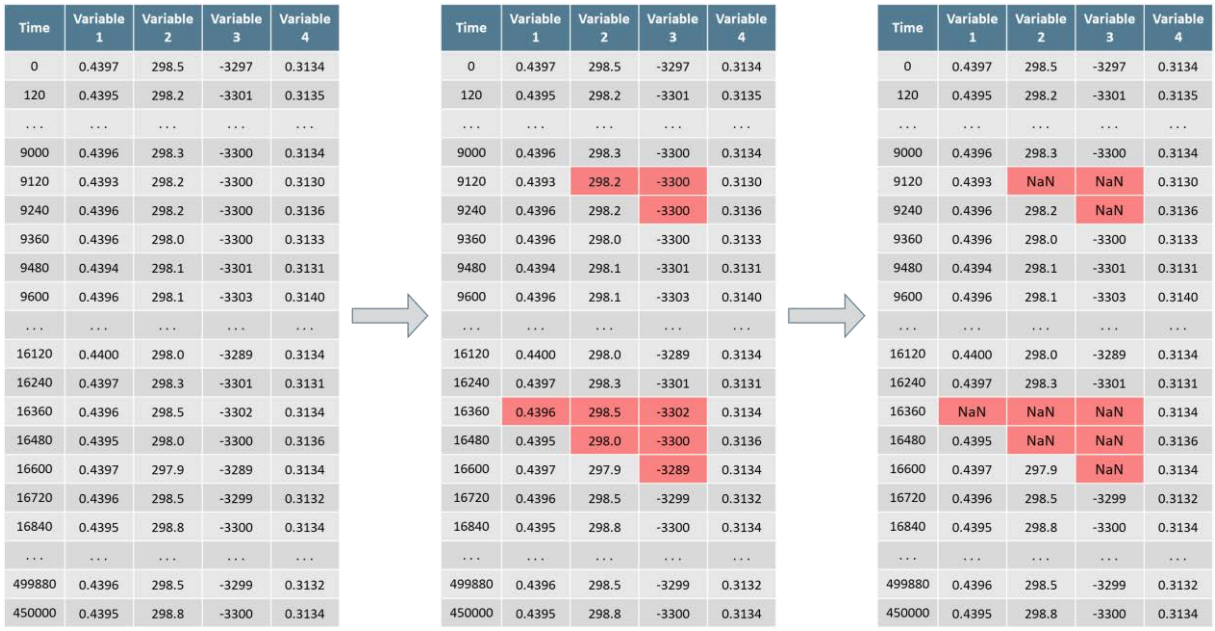
3.3.2 Noise and invalid measurement addition

The data collected from the sensors installed in a real chemical plant are inherently corrupted by measurement noise and missing/invalid readings (the latter usually caused by sensor malfunctions or miscalibration). Therefore, once exported the virtual data from gPROMS, the dataset was corrupted on purpose with these sources of error in order to accurately reproduce a real industrial dataset.

3.3.2.1 Invalid measurements

The addition of invalid measurements has been performed manually in excel, meaning that the content of some (randomly selected) cells of the dataset have been deleted and substituted by the string NaN (acronym for Not a Number).

The procedure is summarized in Figure 3.6 taking into account a generic dataset with 4 variables.



Time	Variable 1	Variable 2	Variable 3	Variable 4
0	0.4397	298.5	-3297	0.3134
120	0.4395	298.2	-3301	0.3135
…	…	…	…	…
9000	0.4396	298.3	-3300	0.3134
9120	0.4393	298.2	-3300	0.3130
9240	0.4396	298.2	-3300	0.3136
9360	0.4396	298.0	-3300	0.3133
9480	0.4394	298.1	-3301	0.3131
9600	0.4396	298.1	-3303	0.3140
…	…	…	…	…
16120	0.4400	298.0	-3289	0.3134
16240	0.4397	298.3	-3301	0.3131
16360	0.4396	298.5	-3302	0.3134
16480	0.4395	298.0	-3300	0.3136
16600	0.4397	297.9	-3289	0.3134
16720	0.4396	298.5	-3299	0.3132
16840	0.4395	298.8	-3300	0.3134
…	…	…	…	…
499880	0.4396	298.5	-3299	0.3132
450000	0.4395	298.8	-3300	0.3134

Time	Variable 1	Variable 2	Variable 3	Variable 4
0	0.4397	298.5	-3297	0.3134
120	0.4395	298.2	-3301	0.3135
…	…	…	…	…
9000	0.4396	298.3	-3300	0.3134
9120	0.4393	298.2	-3300	0.3130
9240	0.4396	298.2	-3300	0.3136
9360	0.4396	298.0	-3300	0.3133
9480	0.4394	298.1	-3301	0.3131
9600	0.4396	298.1	-3303	0.3140
…	…	…	…	…
16120	0.4400	298.0	-3289	0.3134
16240	0.4397	298.3	-3301	0.3131
16360	0.4396	298.5	-3302	0.3134
16480	0.4395	298.0	-3300	0.3136
16600	0.4397	297.9	-3289	0.3134
16720	0.4396	298.5	-3299	0.3132
16840	0.4395	298.8	-3300	0.3134
…	…	…	…	…
499880	0.4396	298.5	-3299	0.3132
450000	0.4395	298.8	-3300	0.3134

Time	Variable 1	Variable 2	Variable 3	Variable 4
0	0.4397	298.5	-3297	0.3134
120	0.4395	298.2	-3301	0.3135
…	…	…	…	…
9000	0.4396	298.3	-3300	0.3134
9120	0.4393	NaN	NaN	0.3130
9240	0.4396	298.2	NaN	0.3136
9360	0.4396	298.0	-3300	0.3133
9480	0.4394	298.1	-3301	0.3131
9600	0.4396	298.1	-3303	0.3140
…	…	…	…	…
16120	0.4400	298.0	-3289	0.3134
16240	0.4397	298.3	-3301	0.3131
16360	NaN	NaN	NaN	0.3134
16480	0.4395	NaN	NaN	0.3136
16600	0.4397	297.9	NaN	0.3134
16720	0.4396	298.5	-3299	0.3132
16840	0.4395	298.8	-3300	0.3134
…	…	…	…	…
499880	0.4396	298.5	-3299	0.3132
450000	0.4395	298.8	-3300	0.3134

Figure 3.6. *Invalid measurements addition procedure.*

3.3.2.2 Measurement noise

In order to add automatically the noise to the whole virtual dataset, a Python™ function was developed. Once given a dataset, the function corrupts each process signal with noise according to the following rationale:

$$x_{i,\,noised} = x_i + k[\max(x_i) - \min(x_i)] \cdot \frac{\text{random}(-1E6, +1E6)}{1E6}\,, \tag{3.5}$$

where x_i is the noiseless virtual measurement, k is a parameter that allows to modulate the amplitude of the fluctuations, $\max(x_i)$ and $\min(x_i)$ are respectively the highest and the lowest values assumed by the process variable x_i throughout the simulation and finally the term $\text{random}(-1E6, +1E6)/1E6$ is a generator of random numbers with six decimal places and bound between -1 and +1.

In Figure 3.7 the noise addition procedure is overviewed through a graphical example.

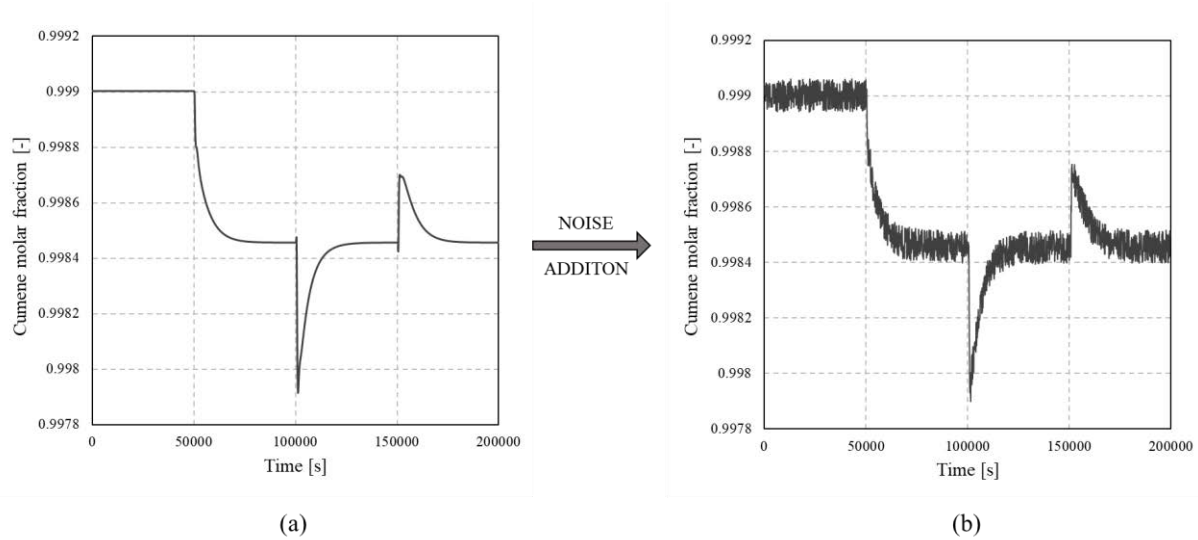


Figure 3.7. *Comparison between the (a) noiseless virtual process signal and the (b) noised signal generated by the Python™ function ($k = 0.1$).*

Different values of the parameter k have been exploited to generate datasets with different level of noise. In particular, the values of k which have been used are 0.025, 0.1 and 0.25. Eventually, the empirical correlations on which the hybrid model of the plant is based on will be developed starting from the dataset noised with $k = 0.1$.

The other datasets will instead be used to analyse the effect of the noise of the training and validation data on the empirical relations obtained through partial least squares regression.

Since the dataset corrupted with noise and invalid measurements is in every way similar to an actual industrial datasets, the hybrid model generation procedure developed in this Thesis and described in the next Chapter (§4) is indeed meant to be exploited when the hybrid models have to be developed directly from actual plant data.

Chapter 4

Hybrid model

This Chapter overviews the procedure followed to develop a steady-state hybrid model of the industrial plant for the production of cumene, discusses its implementation in the process simulator gPROMS and evaluates its performances. Firstly, it is described how steady-state operating points are retrieved from a plant historian through the SSD algorithm developed in this Thesis. Then it is explained how data reconciliation has been carried out exploiting gPROMS optimization tool and how the empirical correlations that constitutes the black-box portion of the hybrid model have been obtained through PLS and ALAMO. Finally, the implementation of the hybrid model in the process simulator gPROMS is discussed and the tests carried out to assess the model predictive capabilities are reviewed.

4.1 Hybrid model development

In order to demonstrate the potential of hybrid modelling, as explained in §3, it was decided to take into account the cumene manufacturing process, generate data in gPROMS with an accurate first principles model and corrupt the virtual process signals with noise and invalid measurements (to obtain a dataset as similar as possible to an actual plant historian). The availability of a fast, robust and accurate first principle model, however, is in conflict with the concept of hybrid modelling. Indeed, one of the most common scenario in which the grey-box models are employed involves dealing with chemical plants (or single equipment) in which it is not economically convenient, computationally fast enough or even not possible at all to fully map the behaviour of the process through a first principles model. Therefore, once simulated the historian in gPROMS, it was assumed that it was not possible to model with the first principles the train of distillation columns. Hence, the task became firstly to develop empirical correlations to model the behaviour of the separation section and then to integrate the data-driven elements within the first principles model of the rest of the plant. The assumption is summarized graphically in Figure 4.1 pointing out with green and red borders the section of the plant that will be modelled with first principles and the equipment whose behaviour will be mapped through data-driven elements, respectively.

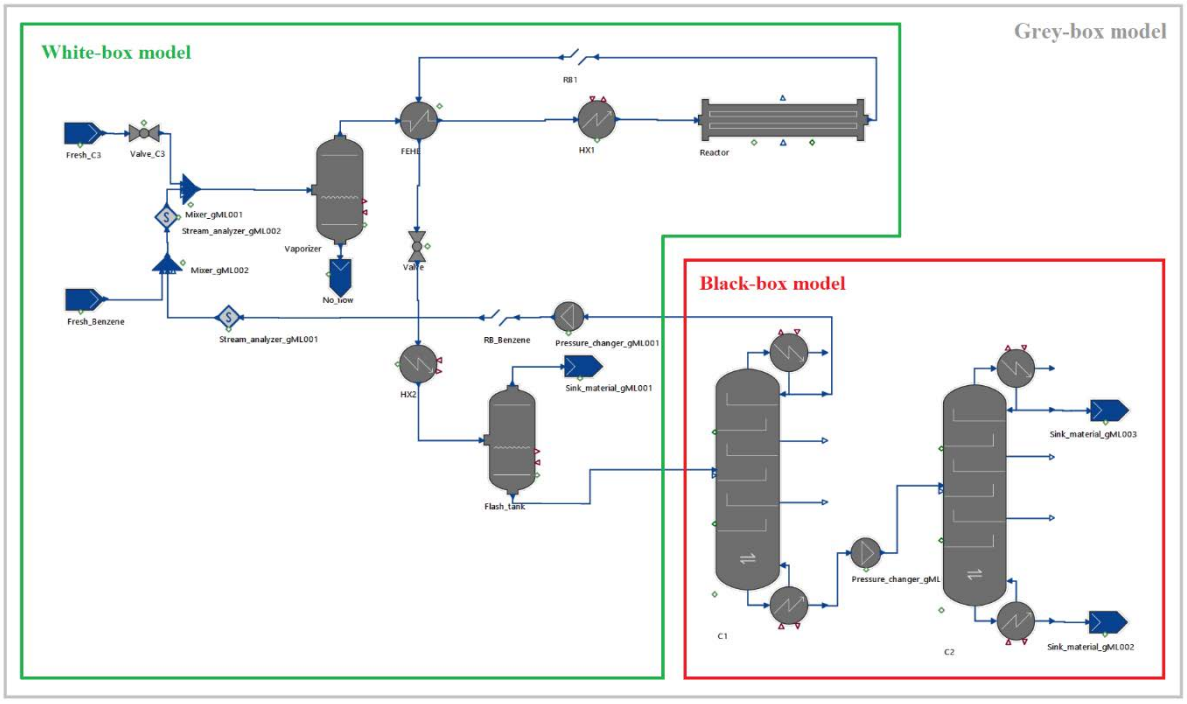


Figure 4.1. *Insight on the structure of the hybrid model of the industrial plant for the production of cumene.*

As can be noticed in Figure 4.1, from the combination of a white and a black-box model, a hybrid model is generated.

Once available the historian of the chemical plant under investigation, the procedure followed to develop the grey-box model consisted on the following steps:

1. Steady-state operating points identification;
2. Data reconciliation;
3. Data-driven element development.

Each point listed above will be discussed in detail in the following sections.

4.1.1 Steady-state operating points identification

Once concluded the data generation described in Chapter §3, a simulated plant historian collecting 33050 observations of 82 key process variables was obtained. However, since the target was to develop a steady-state model of the process, not all the data were useful and could be used to generate the data-driven elements. Hence, firstly the windows of time in which the plant was operating at steady-state were identified so that the data referring to transient states could be easily discarded. Then, among all the samples collected when the plant was operating steadily, only a single observation per each steady-state operating point was retained. Dealing with big data, indeed, not necessarily imply dealing with a lot of meaningful information. In particular, when more than one observation referring to the same steady-state operating point

are included into the dataset used to calibrate a model, the ability of the model to capture the behaviour of the process is not improved.
As described in chapter §2, the steady-state detection algorithm developed in this Thesis carry out automatically both the identification of the steady-state windows and the selection of the representative observations (one for each steady-state operating point). The selection of the representative points was performed identifying and picking the observations located in the middle of the steady-state intervals. This straightforward rationale ensures that all the samples collected for further analysis truly refer to steady operations.
The algorithm processed entirely the historian with a running time of 595 seconds (laptop Lenovo T450 with the processor Intel Core i7-5600U vPro) providing as outcome a dataset that collects the values of the 82 key process variables in 72 different steady-state operating points.
Before proceeding with the next step of the hybrid model development procedure, a last check was carried out analysing the set of diagnostic graphs described in §2. Again, no detection issues were discovered.

4.1.2 Data reconciliation

Data reconciliation is a state-of-the art technique that targets to make raw plant data match energy and mass balances minimizing the weighted squared error sum of the deviations between measured and estimated values under equality or inequality constraints (Dempf and List, 1998). Relying on the concept of redundancy (duplicated sensors or algebraic constraints) and on a process model, the technique corrects the noisy and faulty measurements forcing the observations to fulfil the first principles physical laws. Carrying out data reconciliation, therefore, *i*) avoid the inclusion of corrupted data (outliers), *ii*) allows to detect sensors systematic errors and, if needed, *iii*) serves as an estimator for variables that in the real plant are not actually measured (Pitarch *et al.*, 2017).
Within the framework of the hybrid model development, data reconciliation was carried out on the section of the plant that had to be modelled with the data-driven elements. In the cumene case study, therefore, it was performed on the separation section.
Since no white-box models were assumed to be available for the distillation columns, however, a proper reconciliation in which the observations are forced to fulfil both the mass and energy balances could not be performed. Instead, as can be noticed in Figure 4.2, a straightforward mass balance model was developed (in gPROMS) substituting each distillation column with a mass balance component splitter. Then, a simplified data reconciliation was performed forcing the compositions and the molar flowrates to fulfil the mass conservation law.

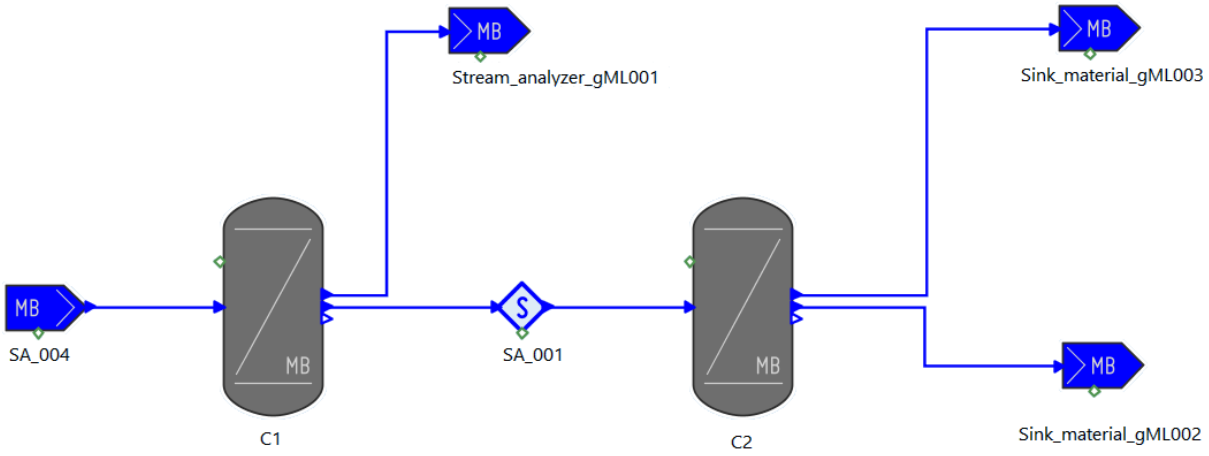


Figure 4.2. *Mass balance model of the separation section employed to carry out data reconciliation.*

Differently from the molar flowrates and the compositions, the measurements of pressure and temperature were not included in the analysis (namely were not reconciled) because in the mass balance model shown in Figure 4.2, the energy balance is not taken into account.

The component splitters through which the separation section is described are predefined gPROMS models that represent theoretical separation stages. The splitters, in particular, redistribute each component that is present in their feed among the distillate and the residue according to the value of the component split fraction. The component split fraction is the fraction of the inlet flowrate of the component that leaves the splitter in the overhead stream. Commonly the values of the split fractions of all the components involved in the separation are entered by the user before running the simulation. In the cumene case study, however (besides for the mitigation of the samples gross and random errors), data reconciliation was also exploited as a tool to estimate the split fractions of benzene and cumene in the first distillation column and the split fractions of cumene and *p*-diisopropylbenzene in the second distillation column (in each of the 72 steady-state operating point). The other split fractions, instead, were set, as summarized in Table 4.1, equal to constant values.

Table 4.1. *Assumptions on the values of the split fractions in the mass balance model of the separation section.*

Component	First distillation column	Second distillation column
Propylene	1	1
Propane	1	1
Benzene	Estimated	1
Cumene	Estimated	Estimated
P-diisopropylbenzene	0	Estimated

From a mathematical point of view, data reconciliation is a constrained optimization of a NLP (Nonlinear Programming Problem). In this Thesis, the technique was performed exploiting the gPROMS optimization tool, which allows to minimize a user defined objective function varying the values of specific decision variables.
The objective function that has been minimized is defined as follows:

$$\text{obj}_{\text{fun}} = \sum_{i=1}^{24} \left(\frac{x_i^{calc} - x_i^{meas}}{\sigma_i} \right), \tag{4.1}$$

where:

- x_i^{calc} : value of the variable calculated by the model in gPROMS;
- x_i^{meas}: value of the variable measured by the sensor;
- σ_i : standard deviation of the measured value estimated by the SSD algorithm (§2).

The 24 measurements that have been taken into account in the objective function are: the overall molar flowrate and the component molar fractions of the feed and the distillate of the first distillation column and the overall molar flowrate and the component molar fractions of the distillate and the residue of the second distillation column.

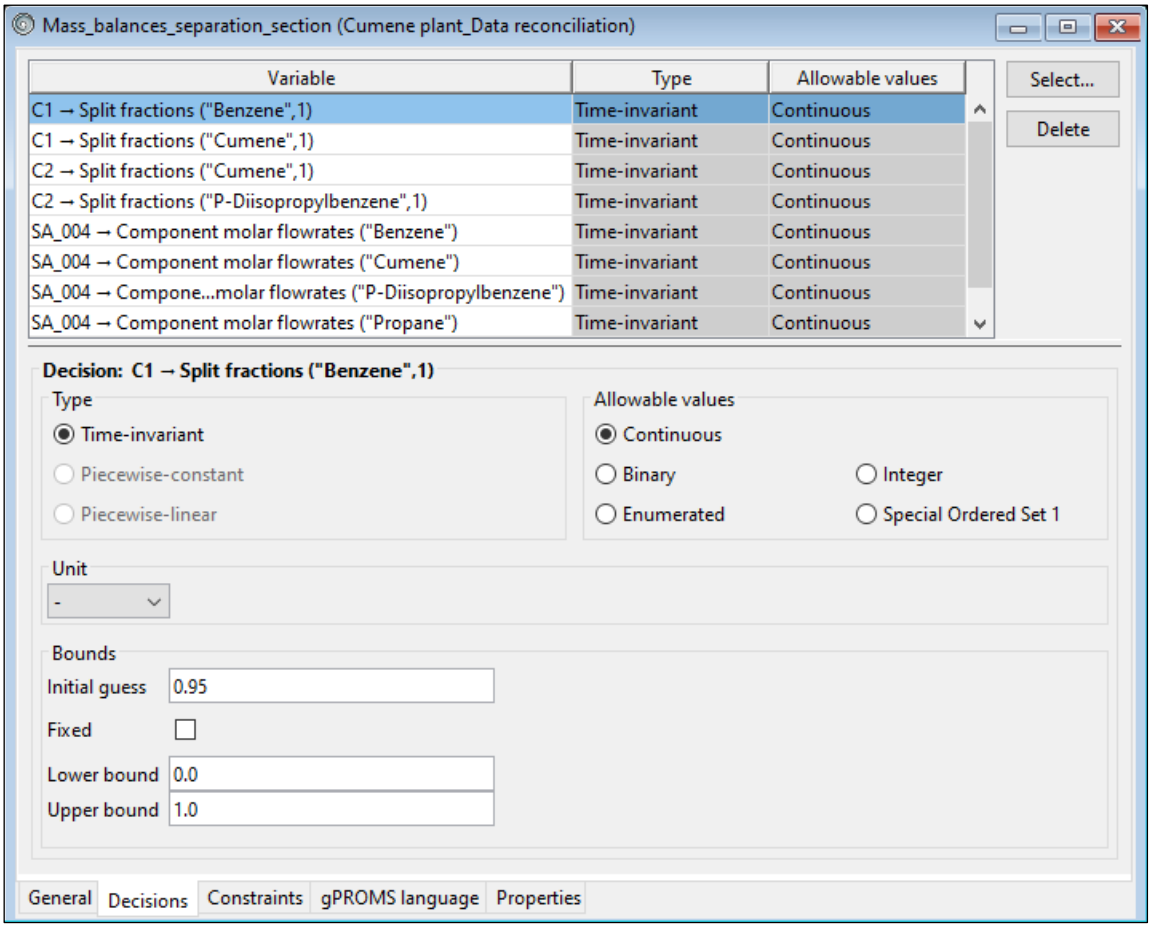


Figure 4.3. *gPROMS optimization interface for the selection of the decision variables.*

Furthermore, as can be seen in Figure 4.3, the minimization was carried out by varying the values of the split fractions of benzene and cumene in the first splitter, the split fraction of cumene and *p*-diisopropylbenzene in the second splitter and the molar flowrates of propane, cumene, benzene and *p*-diisopropylbenzene in the feed of the first splitter. The molar flowrate of propylene in the feed of the first splitter is not listed as a decision variable (namely those variables whose value is allowed to change to minimize the objective function) because it was a priori assumed constant and equal to zero. The choice to employ as decision variables the component flowrates rather than the overall flowrate and the molar fractions is meant to ensure that the sum of the component molar fractions is equal to one. The constraint, indeed, when the optimization is carried out varying the molar fractions, is not always matched due to numerical uncertainties.

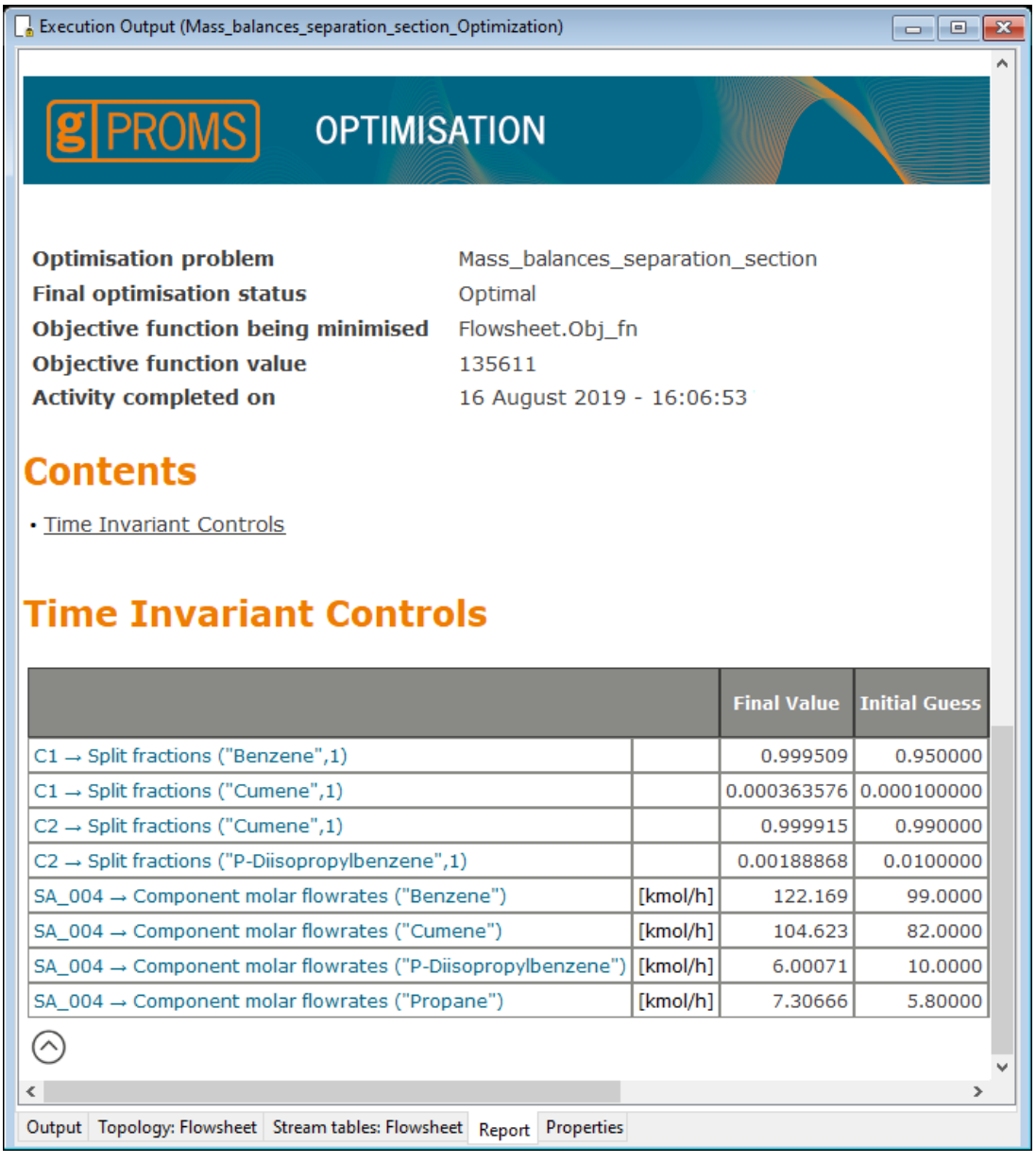


Figure 4.4. *Optimization execution output.*

In Figure 4.4 it is shown the execution output displayed by the process simulator once the flowrates and the molar fractions of a single steady-state operating point were reconciled. In the execution output the initial and final value of the decision variables and the final value of the objective function are summarized. The final values of the decision variables allow to minimize the deviations between the measured and the calculated process variables (while ensuring the fulfilment of the mass balance).
The optimization was carried out for all the 72 different steady-state operating point obtaining for each of them an estimate of the split fractions and a set of reconciled flowrates and molar fractions that fulfil the mass balance.

4.1.3 Data-driven element development

The last step required to generate the hybrid model of the plant for the production of cumene consisted on the development of a data-driven element to model the behaviour of the separation section. Each distillation column, once again, was approximated with a mass balance component splitter. This time, though, exploiting the outcomes of the steady-state detection and of the data reconciliation, empirical correlations were generated with both the PLS regression and the ALAMO model building methodology to predict the component split fractions. In the following sections, firstly the general structure of the black-box models of the distillation columns is described. Then the development of the empirical correlations through PLS and ALAMO is overviewed.

4.1.3.1 Black-box model of the first distillation column

The first distillation column targets to separate the unreacted benzene and the traces of propane and propylene that have not been vented off in the flash drum from the main product cumene and the undesired *p*-diisopropylbenzene.
The variables that need predicting are:

- the overall molar flowrate of the distillate F_{D1};
- the component molar fractions in the distillate $x_{i,D1}$;
- the overall molar flowrate of the residue F_{R1};
- the component molar fractions in the residue $x_{i,R1}$.

The process variables that, instead, are supposed to be known in advance (and therefore those variables that can be fed as inputs to the data-driven element) are:

- the overall molar flowrate of the feed F_{F1};
- the component molar fractions in the feed $x_{i,F1}$;
- the temperatures of the feed, the residue and the distillate: T_{F1}, T_{R1} and T_{D1};
- the pressures of the feed, the residue and the distillate: P_{F1}, P_{R1} and P_{D1};
- the reflux ratio of the distillation column r_1;
- the heat duties of the condenser and of the reboiler: $HC1$ and $HR1$.

The known and unknown variables are graphically summarized in the flowsheet reported in Figure 4.5. The variables marked in red are those that have to be predicted.

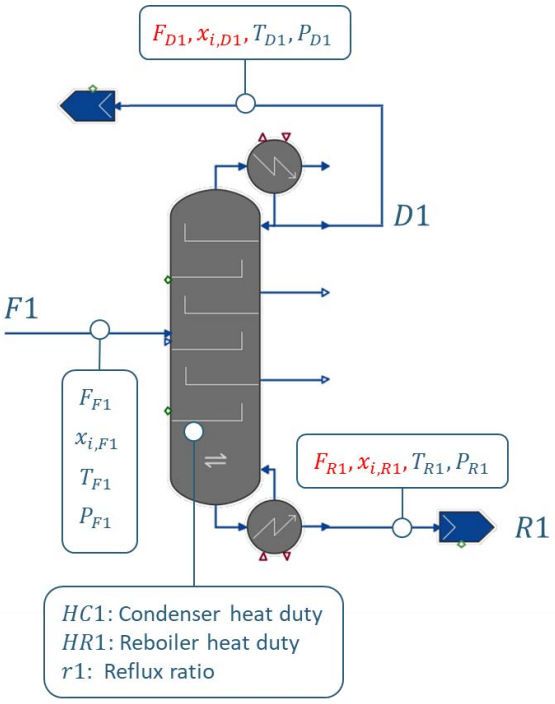


Figure 4.5. *Insight on the input and output variables of the black-box model of the first distillation column.*

As previously stated, the target of the column black-box model is to estimate the flowrate and the composition of both the distillate $D1$ and the residue $R1$. Since the column was approximated with a mass balance component splitter, however, under the assumption that the feed flowrate and composition are known, the task could be accomplished by estimating the component split fractions.

Therefore, data-driven correlations for the prediction of the split fractions of cumene and benzene were developed through ALAMO and PLS regression exploiting the steady-state operating points dataset generated by the SSD algorithm and partially treated (only the flowrates and compositions) via data reconciliation.

The general formulation of the empirical correlations for the cumene and benzene split fraction prediction are expressed as follows:

$$SF_{C_9H_{12},C1} = f\left(T_{F1}, P_{F1}, F_{F1}, x_{i,F1}, T_{D1}, P_{D1}, T_{R1}, P_{R1}, HC1, HR1, r1\right) , \tag{4.2}$$

$$SF_{C_6H_6,C1} = f\left(T_{F1}, P_{F1}, F_{F1}, x_{i,F1}, T_{D1}, P_{D1}, T_{R1}, P_{R1}, HC1, HR1, r1\right) . \tag{4.3}$$

The explicit formulas depend on the methodology used to develop the correlations (ALAMO or PLS) and therefore are provided afterwards in the dedicated sections.

The values of the split fractions of propane, propylene and *p*-diisopropylbenzene, differently from those of cumene and benzene, were assumed a priori equal to constant values. The assumed split fractions are summarized in Table 4.2.

Table 4.2. *Assumptions on the values of the split fractions in the first distillation column.*

Component	Split fraction
Propylene	1
Propane	1
P-diisopropylbenzene	0

4.1.3.2 Black-box model of the second distillation column

The second distillation column targets to separate the main product cumene from the undesired product. Similarly to the first distillation column the variables that it need predicting are:

- the overall molar flowrate of the distillate F_{D2};
- the component molar fractions in the distillate $x_{i,D2}$;
- the overall molar flowrate of the residue F_{R2};
- the component molar fractions in the residue $x_{i,R2}$.

Instead, the process variables whose value is assumed to be available are:

- the overall molar flowrate of the feed F_{F2};
- the component molar fractions in the feed $x_{i,F2}$;
- the temperatures of the feed, the residue and the distillate: T_{F2}, T_{R2} and T_{D2};
- the pressures of the feed, the residue and the distillate: P_{F2}, P_{R2} and P_{D2};
- the reflux ratio of the distillation column r_2;
- the heat duties of the condenser and of the reboiler: $HC2$ and $HR2$.

The process variables involved in the analysis are graphically visualized in Figure 4.6.

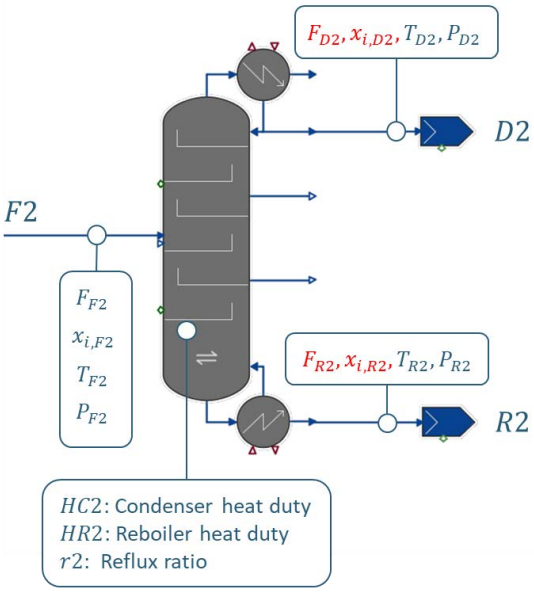


Figure 4.6. *Insight on the input and output variables of the black-box model of the second distillation column.*

In the second distillation column case, the data-driven correlations were developed to predict the split fractions of cumene and *p*-diisopropyllbenzene. The general formulation of the empirical correlations are:

$$SF_{C_9H_{12},C2} = f\left(T_{F2}, P_{F2}, F_{F2}, x_{i,F2}, T_{D2}, P_{D2}, T_{R2}, P_{R2}, HC2, HR2, r2\right) , \quad (4.4)$$

$$SF_{C_{12}H_{18},C2} = f\left(T_{F2}, P_{F2}, F_{F2}, x_{i,F2}, T_{D2}, P_{D2}, T_{R2}, P_{R2}, HC2, HR2, r2\right) . \quad (4.5)$$

The split fractions of propylene, propane and benzene, instead, as can be noticed in Table 4.3, were assumed to be constant and equal to one.

Table 4.3. *Assumptions on the values of the split fractions in the second distillation column.*

Component	Split fraction
Propylene	1
Propane	1
Benzene	1

4.1.3.3 Regressor matrices and response variables vectors

Since the regressors matrices and the response variables vectors through which the empirical correlation were developed are independent from the model building methodology that is being employed (either ALAMO or PLS regression), before discussing the outcomes of the model building techniques, it is convenient to firstly present the data on which the data-driven element were calibrated.

Table 4.4 reports an extract of the matrix of the regressors that have been exploited to predict the split fraction of benzene and cumene in the first distillation column.

Table 4.4. *Extract of the matrix of the regressors used to predict the split fractions in the first distillation column. The unit of measure of temperature is [°C], the unit of measure of pressure is [bar] and the unit of measure of the flowrates is [kmol/h]. The molar fractions, instead, are dimensionless.*

Index	T_{F1}^{m}	P_{F1}^{m}	F_{F1}^{m}	$x_{C_3H_8,F1}^{R}$	$x_{C_6H_6,F1}^{R}$	$x_{C_9H_{12},F1}^{R}$	$x_{C_{12}H_{18},F1}^{R}$	T_{D1}^{m}	P_{D1}^{m}	T_{R1}^{m}	P_{R1}^{m}	$r1^{m}$	$HC1^{m}$	$HR1^{m}$
1	363.15	1.781	203.1	0.03019	0.51669	0.43094	0.02218	337.50	1.829	452.23	1.925	0.440	-1498	1990
2	363.14	1.781	203.0	0.03017	0.51659	0.43107	0.02217	337.40	1.829	452.22	1.925	0.440	-1499	1990
3	363.15	1.781	203.0	0.03021	0.51652	0.43104	0.02224	337.44	1.829	452.23	1.926	0.440	-1498	1989
4	363.15	1.780	198.8	0.02969	0.52963	0.42301	0.01767	338.56	1.829	451.84	1.922	0.430	-1485	1947
5	363.15	1.780	200.5	0.02987	0.52402	0.42700	0.01912	338.04	1.829	451.96	1.923	0.435	-1489	1965
6	363.15	1.781	203.8	0.03023	0.51431	0.43202	0.02344	337.22	1.829	452.32	1.927	0.442	-1501	1999
7	363.15	1.781	205.4	0.03041	0.50993	0.43369	0.02596	336.85	1.829	452.53	1.928	0.445	-1506	2016
8	363.15	1.779	196.9	0.03016	0.51706	0.43033	0.02246	337.44	1.829	452.10	1.919	0.440	-1453	1930
9	363.15	1.779	197.0	0.03018	0.51696	0.43031	0.02255	337.41	1.829	452.10	1.919	0.440	-1453	1929
10	363.15	1.779	196.9	0.03012	0.51695	0.43026	0.02267	337.47	1.828	452.11	1.919	0.440	-1453	1931
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
72	363.12	1.792	236.9	0.03002	0.51922	0.42947	0.02129	337.67	1.831	453.06	1.964	0.438	-1752	2333

Table 4.5, instead, reports an extract of the matrix of the regressors that have been exploited to predict the split fraction of cumene and *p*-diisopropylbenzene in the second distillation column.

Table 4.5. *Extract of the matrix of the regressors used to predict the split fractions in the second distillation column. The unit of measure of temperature is [°C], the unit of measure of pressure is [bar] and the unit of measure of the flowrates is [kmol/h]. The molar fractions, instead, are dimensionless.*

Index	T_{F2}^{m}	P_{F2}^{m}	F_{F2}^{m}	$x_{C_6H_6,F2}^{R}$	$x_{C_9H_{12},F2}^{R}$	$x_{C_{12}H_{18},F2}^{R}$	T_{D2}^{m}	P_{D2}^{m}	T_{R2}^{m}	P_{R2}^{m}	$r2^{m}$	$HC2^{m}$	$HR2^{m}$
0	429.54	1.072	92.0	0.00050	0.95055	0.04895	424.97	1.073	1769	1.127	0.630	-1473	2743
1	429.55	1.072	92.0	0.00050	0.95059	0.04891	424.97	1.073	1769	1.127	0.630	-1475	2744
2	429.56	1.072	92.0	0.00050	0.95045	0.04905	424.97	1.073	1769	1.127	0.630	-1474	2750
3	428.96	1.066	87.6	0.00048	0.95942	0.04010	425.00	1.072	1770	1.118	0.624	-1413	2344
4	429.18	1.069	89.5	0.00049	0.95666	0.04285	424.98	1.073	1770	1.122	0.626	-1440	2498
5	429.43	1.071	91.2	0.00050	0.95278	0.04672	424.97	1.073	1769	1.126	0.629	-1463	2665
6	429.67	1.074	92.9	0.00050	0.94805	0.05145	424.97	1.073	1769	1.129	0.632	-1486	2843
7	429.96	1.075	94.4	0.00051	0.94301	0.05648	424.96	1.073	1768	1.132	0.635	-1504	3053
8	429.39	1.067	89.2	0.00049	0.94992	0.04959	424.99	1.072	1770	1.121	0.630	-1430	2686
9	429.41	1.067	89.2	0.00049	0.94971	0.04980	424.99	1.072	1770	1.120	0.630	-1430	2684
…	…	…	…	…	…	…	…	…	…	…	…	…	…
72	430.88	1.104	110.8	0.00054	0.94534	0.05412	424.95	1.075	1763	1.172	0.634	-1772	3492

The measurements with the apex ‘R’ are those which have been reconciled while the measurements with the apex ‘m’ are picked directly from the outcome of the steady-state detection procedure (namely the dataset collecting the information of the 82 key process variables in 72 different steady-state operating points).

Furthermore, Table 4.4 does not display the column referring to the molar fraction of propylene, while Table 4.5 omits both the values of the molar fractions of propane and propylene. These columns were removed from the matrices of the regressors because, as a consequence of the assumptions taken while performing data reconciliation (the assumptions on the split fraction and on the composition of the feed of the first distillation column), the molar fraction of propylene in the feed of the first distillation column and the molar fractions of propane and propylene in the feed of the second distillation column are constant and equal to zero in all the 72 steady-state operating points.

Table 4.6 reports an extract of the vectors collecting the values of the response variables: the split fractions of cumene and benzene in the first distillation column and the split fractions of cumene and *p*-diisopropylbenzene in the second distillation column.

Table 4.6. *Extract of the response variables vectors: the split fractions of (a) benzene and (b) cumene in the first distillation column and the split fractions of (c) cumene and (d) p-diisopropylbenzene in the second distillation column. The split fractions are dimensionless [-].*

Index	$SF^{R}_{C_6H_6,C1}$
1	0.9995607
2	0.9995611
3	0.9995604
4	0.9995983
5	0.9995830
6	0.9995693
7	0.9995555
8	0.9995407
9	0.9995662
10	0.9995658
…	…
72	0.9995143

(a)

Index	$SF^{R}_{C_9H_{12},C1}$
1	0.0003988
2	0.0003999
3	0.0003989
4	0.0004566
5	0.0004312
6	0.0004095
7	0.0003905
8	0.0003701
9	0.0004013
10	0.0004007
…	…
72	0.0003627

(b)

Index	$SF^{R}_{C_9H_{12},C2}$
1	0.9999623
2	0.9999625
3	0.9999625
4	0.9999747
5	0.9999701
6	0.9999650
7	0.9999596
8	0.9999536
9	0.9999683
10	0.9999682
…	…
72	0.9999158

(c)

Index	$SF^{R}_{C_{12}H_{18},C2}$
1	0.0090037
2	0.0092133
3	0.0092347
4	0.0296009
5	0.0161727
6	0.0106362
7	0.0081569
8	0.0064097
9	0.0197420
10	0.0189934
…	…
72	0.0018817

(d)

The values of the split fractions which are displayed in Table 4.6 are those that have been estimated performing data reconciliation.

4.1.3.4 Data-driven element for the estimation of the benzene split fraction in the first distillation column obtained via PLS regression

Once the data of Table 4.4 and the vector (a) of Table 4.6 were loaded in Python™, a PLS regression along with a 5-fold cross validation were performed through the scikit-learn package. The outcome of the analysis is summarized in Table 4.7.

Table 4.7. *Performances of the data-driven model developed for the estimation of the benzene split fraction.*

Number of PC	RMSECV $\times 10^8$	Q^2	R^2
1	4.0753	0.9799	0.9839
2	3.1615	0.9882	0.9903
3	2.8895	0.9903	0.9931
4	2.6408	0.9922	0.9940
5	2.4798	0.9932	0.9947
6	2.4406	0.9934	0.9953
7	2.4627	0.9933	0.9955
8	2.5411	0.9928	0.9957
9	2.5460	0.9928	0.9957
10	2.5747	0.9926	0.9957
11	2.6625	0.9921	0.9958
12	2.6719	0.9920	0.9958
13	2.6823	0.9920	0.9958
14	2.7334	0.9916	0.9958

As can be noticed in Table 4.7, the model that achieve the lowest $RMSECV$ and the highest predictive relevance Q^2 is the one which employs six PCs. The influence of the number of PCs on the model predictive capabilities is graphically visualized in the validation curve reported in Figure 4.7.

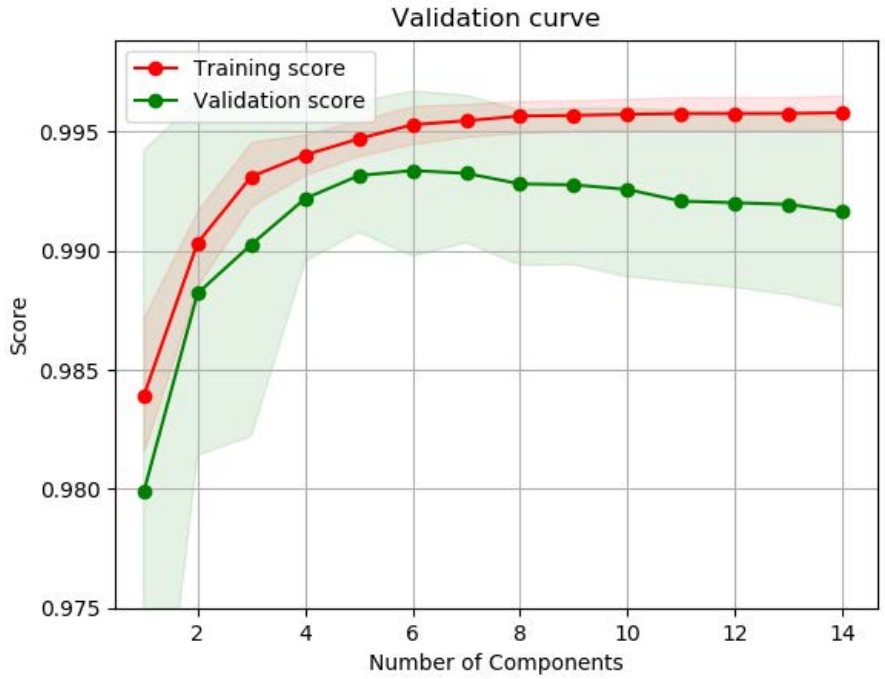


Figure 4.7. *Validation curve of the PLS regression model developed for the estimation of the benzene split fraction in the first distillation column.*

In Figure 4.7. the validation and the training score (Q^2 and R^2, respectively) are plotted as a function of the number of PCs taken into account in the model. As stated in §2.2.5, the most effective model is achieved in correspondence of the maximum of the validation score. Indeed, as can be noticed in Figure 4.7, despite the training score remains very close to unity, when more than six PCs are taken into account in the model, the predictive relevance Q^2 start decreasing and the model start overfitting the calibration data. Since performing cross validation implies estimating the scores five different times (one for each fold), the calculated metrics present a certain degree of uncertainty. Hence, a standard deviation band is shown alongside the curves, quantifying the degree of uncertainty of each score. The standard deviation (std) has been calculated as follows:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N}\left(y_{predicted} - y_{observed}\right)^2}{N}}, \tag{4.5}$$

where N is the total number of observation (namely 72, the number of steady-state operating points).

Once the optimal latent variables subspace dimension has been determined, a learning curve was plotted to investigate the relationship between the training set size and the metrics Q^2 and R^2 (namely the validation and training score, respectively).

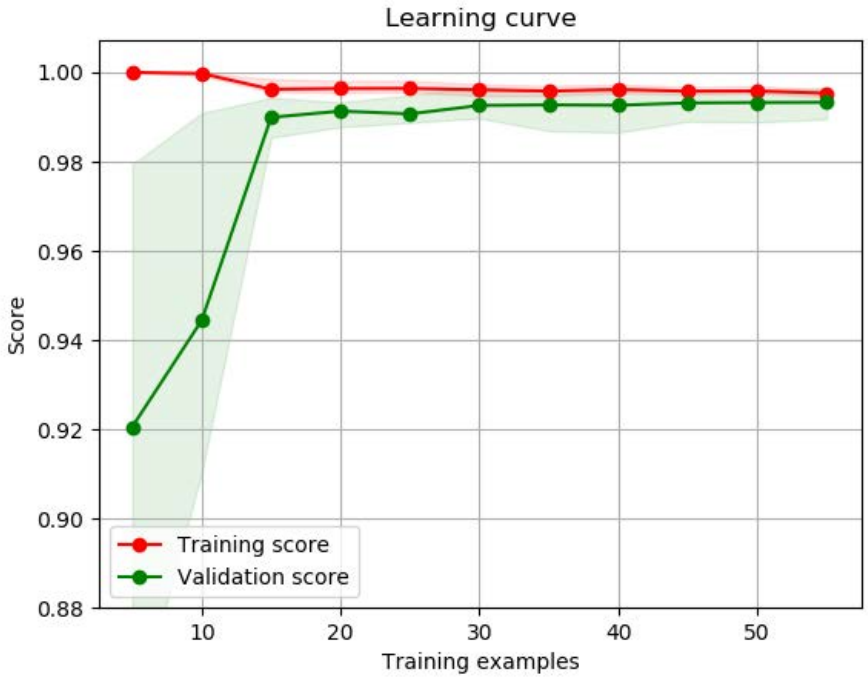


Figure 4.8. *Learning curve of the PLS regression model for the estimation of the benzene split fraction in the first distillation column.*

The learning curve allows to assess the capability of the model to achieve an effective bias-variance trade-off. Indeed, when the validation score is much smaller than the training score for a given number of training examples, the model suffers from high variance and therefore is prone to overfit the data. When instead both metrics converge to a score, which is much smaller than one and does not improve with the addition of more training examples, the model suffers from high bias and therefore tends to underfit the calibration dataset. Since the number of PCs that lead to the most effective bias-variance trade-off has been already chosen according to the $RMSECV$, the learning curve should instead be employed to assess how the model could benefit from an increase of the size of the training set. A model would benefit from the addition of more training examples when the gap between the validation and the training curve, despite showing the tendency to decrease, is still noticeable in correspondence of the maximum number of training instances.

In the case of the PLS regression model for the estimation of the benzene split fraction, as can be noticed in Figure 4.8, the validation and the training curves converge rapidly to a point of stability with a minimal gap between the two scores (both very close to one), therefore the model would not benefit from the addition of more training examples.

The empirical correlation for the split fraction prediction obtained through PLS regression is defined as follows:

$$
\begin{aligned}
SF_{C_6H_6,C1} = {} & 0.999555953 - 2.88 \times 10^{-7}\left(\frac{T_{F1} - \overline{T_{F1}^m}}{\sigma_{T_{F1}^m}}\right) + 3.05 \times 10^{-7}\left(\frac{P_{F1} - \overline{P_{F1}^m}}{\sigma_{P_{F1}^m}}\right) \\
& - 2.96 \times 10^{-6}\left(\frac{F_{F1} - \overline{F_{F1}^m}}{\sigma_{F_{F1}^m}}\right) + 1.23 \times 10^{-6}\left(\frac{x_{C_3H_8,F1} - \overline{x_{C_3H_8,F1}^R}}{\sigma_{x_{C_3H_8,F1}^R}}\right) \\
& + 4.95 \times 10^{-6}\left(\frac{x_{C_6H_6,F1} - \overline{x_{C_6H_6,F1}^R}}{\sigma_{x_{C_6H_6,F1}^R}}\right) - 7.56 \times 10^{-6}\left(\frac{x_{C_9H_{12},F1} - \overline{x_{C_9H_{12},F1}^R}}{\sigma_{x_{C_9H_{12},F1}^R}}\right) \\
& - 9.67 \times 10^{-7}\left(\frac{x_{C_{12}H_{18},F1} - \overline{x_{C_{12}H_{18},F1}^R}}{\sigma_{x_{C_{12}H_{18},F1}^R}}\right) - 1.04 \times 10^{-6}\left(\frac{T_{D1} - \overline{T_{D1}^m}}{\sigma_{T_{D1}^m}}\right) \\
& + 8.38 \times 10^{-7}\left(\frac{P_{D1} - \overline{P_{D1}^m}}{\sigma_{P_{D1}^m}}\right) - 3.05 \times 10^{-6}\left(\frac{T_{R1} - \overline{T_{R1}^m}}{\sigma_{T_{R1}^m}}\right) \\
& - 1.30 \times 10^{-6}\left(\frac{P_{R1} - \overline{P_{R1}^m}}{\sigma_{P_{R1}^m}}\right) - 4.64 \times 10^{-6}\left(\frac{r1 - \overline{r1^m}}{\sigma_{r1^m}}\right) \\
& + 1.87 \times 10^{-6}\left(\frac{HC1 - \overline{HC1^m}}{\sigma_{HC1^m}}\right) - 3.04 \times 10^{-6}\left(\frac{HR1 - \overline{HR1^m}}{\sigma_{HR1^m}}\right),
\end{aligned}
\tag{4.6}
$$

where $\overline{T_j^m}$, $\overline{P_j^m}$, $\overline{F_j^R}$, $\overline{x_{i,j}^R}$, $\overline{r1^m}$, $\overline{HC1^m}$, $\overline{HR1^m}$, $\sigma_{T_j^m}$, $\sigma_{P_j^m}$, $\sigma_{F_j^m}$, $\sigma_{x_{i,j}^R}$, σ_{r1^m}, σ_{HC1^m} and σ_{HR1^m} are the averages and the standard deviations of the columns of the regressors matrix reported in Table 4.4.

Finally, a graphical evaluation of the model predictive capabilities is provided in both Figure 4.9 and Figure 4.10.

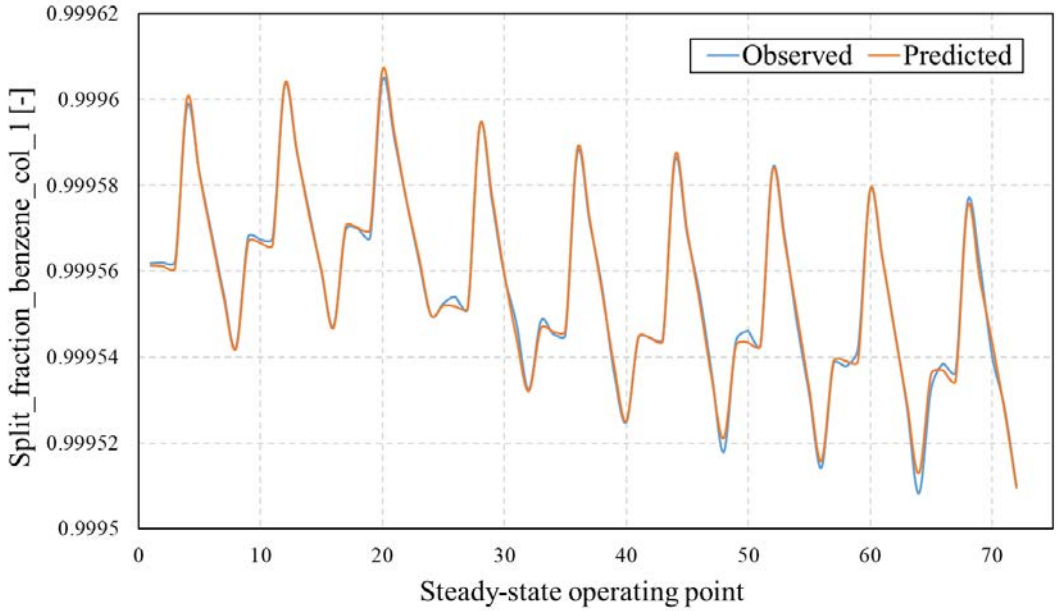


Figure 4.9. *Observed and predicted values of the benzene split fraction in the first distillation column as a function of the steady-state operating point.*

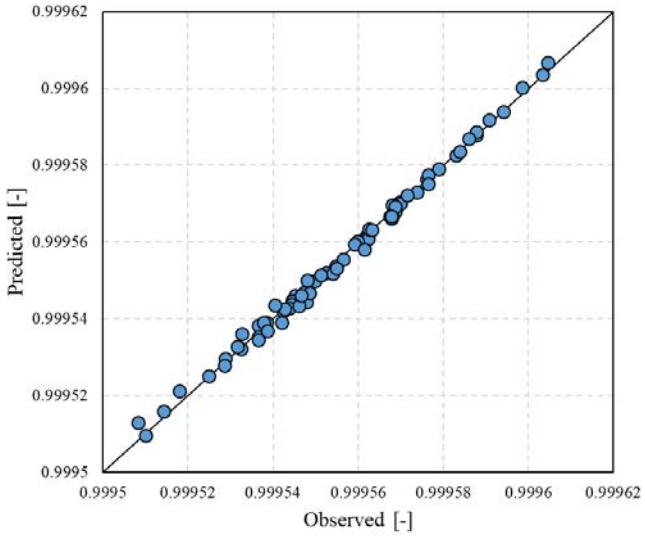


Figure 4.10. *Predicted vs. observed values of the benzene split fraction in the first distillation column.*

In Figure 4.9 the predicted and observed split fractions are compared pointing out the steady-state operating point they refer to, while in Figure 4.9 they are displayed ones against each other together with the $y = x$ line. Since all the points in Figure 4.10 are located near the $y = x$ line, the quality of the model predictions is ensured.

4.1.3.5 Data-driven element for the estimation of the cumene split fraction in the first distillation column obtained via PLS regression

Similarly to §4.1.3.4, a PLS regression (along with a 5-fold cross validation) was carried out in Python™ (through the scikit-learn package) taking into account the data of Table 4.4 and of vector (b) of Table 4.6. The results of the analysis are reported in Table 4.8.

Table 4.8. *Performance of the data-driven model developed for the estimation of the cumene split fraction.*

Number of PC	RMSECV ×10^8	Q^2	R^2
1	6.0028	0.9493	0.9732
2	2.2496	0.9941	0.9961
3	2.1498	0.9947	0.9966
4	2.0977	0.9947	0.9969
5	2.1076	0.9945	0.9971
6	1.9681	0.9951	0.9976
7	1.9820	0.9950	0.9978
8	2.0037	0.9950	0.9979
9	2.0329	0.9949	0.9980
10	2.0506	0.9950	0.9980
11	2.0153	0.9950	0.9980
12	2.0208	0.9950	0.9980
13	2.0294	0.9950	0.9980
14	2.0443	0.9949	0.9980

As can be noticed from Table 4.8, the model with the lowest $RMSECV$ and the highest Q^2 is the one that takes into account six PCs. The influence of the number of PCs on the model predictive capabilities is graphically visualized in the validation curve reported in Figure 4.11.

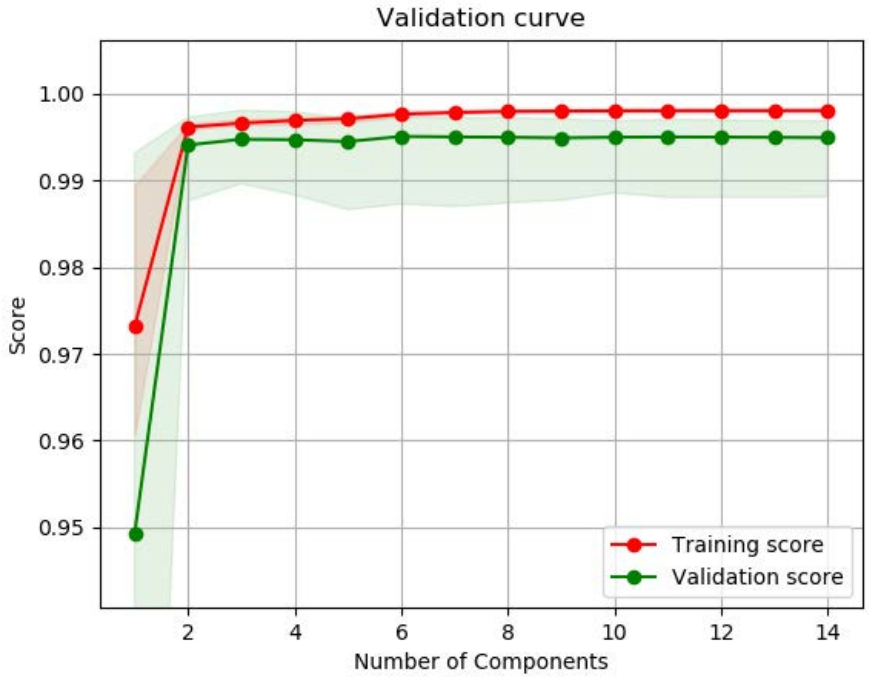


Figure 4.11. *Validation curve of the PLS regression model for the estimation of the cumene split fraction in the first distillation column.*

Figure 4.11. displays the values of the Q^2 and of the R^2 as a function of the number of PCs employed in the model. Differently from the benzene split fraction case, the decrease of the Q^2 experienced when more than six PCs are accounted is so small that the phenomenon of the overfitting cannot be appreciated in the validation curve. This behaviour is likely due to the fact that noise has been added artificially to the virtual dataset generated performing dynamic simulations in gPROMS. As will be clarified in Appendix C, if the value of the parameter k that modulates the amplitude of the fluctuations increases, the resulting models suffer more from overfitting when the number of PCs is larger than the optimal one.

Once the optimal latent variables subspace dimension has been determined, a learning curve was plotted to investigate the relationship between the training set size and the metrics Q^2 and R^2. As can be noticed in Figure 4.12, similarly to the benzene split fraction case, the validation and the training curves plateau rapidly to a point of stability with a tiny gap between the two scores (both very close to one), therefore the model does not require the addition of more training instances.

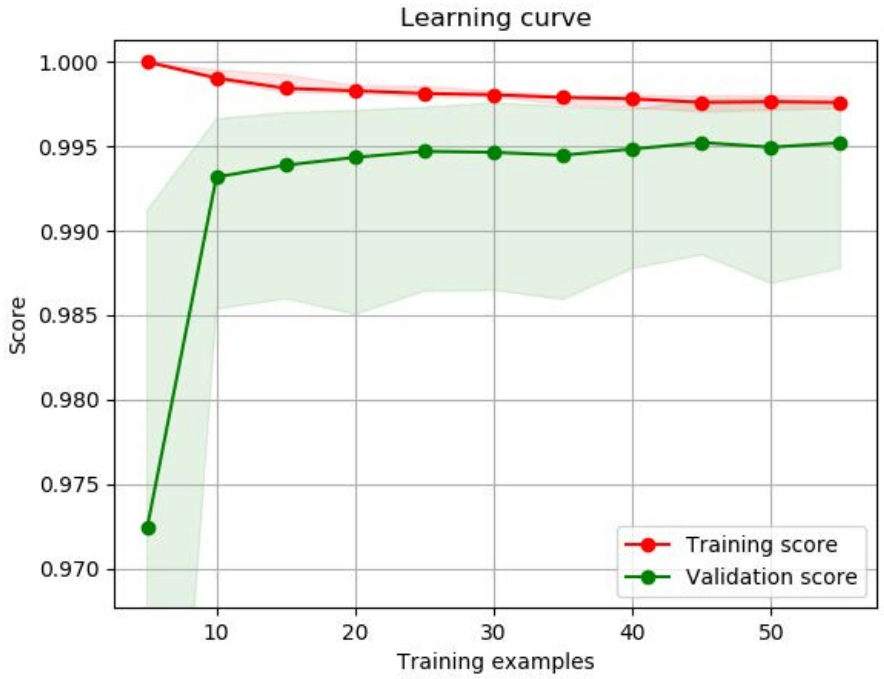


Figure 4.12. *Learning curve of the PLS regression model developed for the estimation of the cumene split fraction in the first distillation column.*

The empirical correlation for the split fraction prediction obtained through PLS regression is defined as follows:

$$
\begin{aligned}
SF_{C_9H_{12},C1} = {} & 0.000403996 + 1.52\times10^{-7}\left(\frac{T_{F1}-\overline{T_{F1}^{m}}}{\sigma_{T_{F1}^{m}}}\right) - 4.72\times10^{-7}\left(\frac{P_{F1}-\overline{P_{F1}^{m}}}{\sigma_{P_{F1}^{m}}}\right) \\
& + 2.31\times10^{-7}\left(\frac{F_{F1}-\overline{F_{F1}^{m}}}{\sigma_{F_{F1}^{m}}}\right) + 1.13\times10^{-6}\left(\frac{x_{C_3H_8,F1}-\overline{x_{C_3H_8,F1}^{R}}}{\sigma_{x_{C_3H_8,F1}^{R}}}\right) \\
& + 6.44\times10^{-6}\left(\frac{x_{C_6H_6,F1}-\overline{x_{C_6H_6,F1}^{R}}}{\sigma_{x_{C_6H_6,F1}^{R}}}\right) - 6.06\times10^{-6}\left(\frac{x_{C_9H_{12},F1}-\overline{x_{C_9H_{12},F1}^{R}}}{\sigma_{x_{C_9H_{12},F1}^{R}}}\right) \\
& - 7.49\times10^{-6}\left(\frac{x_{C_{12}H_{18},F1}-\overline{x_{C_{12}H_{18},F1}^{R}}}{\sigma_{x_{C_{12}H_{18},F1}^{R}}}\right) + 8.90\times10^{-7}\left(\frac{T_{D1}-\overline{T_{D1}^{m}}}{\sigma_{T_{D1}^{m}}}\right) \\
& - 7.32\times10^{-7}\left(\frac{P_{D1}-\overline{P_{D1}^{m}}}{\sigma_{P_{D1}^{m}}}\right) - 3.08\times10^{-6}\left(\frac{T_{R1}-\overline{T_{R1}^{m}}}{\sigma_{T_{R1}^{m}}}\right) \\
& - 8.11\times10^{-7}\left(\frac{P_{R1}-\overline{P_{R1}^{m}}}{\sigma_{P_{R1}^{m}}}\right) - 3.99\times10^{-6}\left(\frac{r1-\overline{r1^{m}}}{\sigma_{r1^{m}}}\right) \\
& - 5.26\times10^{-7}\left(\frac{HC1-\overline{HC1^{m}}}{\sigma_{HC1^{m}}}\right) + 6.12\times10^{-7}\left(\frac{HR1-\overline{HR1^{m}}}{\sigma_{HR1^{m}}}\right),
\end{aligned}
\tag{4.7}
$$

where $\overline{T_j^m}$, $\overline{P_j^m}$, $\overline{F_j^R}$, $\overline{x_{i,j}^R}$, $\overline{r1^m}$, $\overline{HC1^m}$, $\overline{HR1^m}$, $\sigma_{T_j^m}$, $\sigma_{P_j^m}$, $\sigma_{F_j^m}$, $\sigma_{x_{i,j}^R}$, σ_{r1^m}, σ_{HC1^m} and σ_{HR1^m} are the averages and the standard deviations of the columns of the regressors matrix reported in Table 4.4.

At last, a visual assessment of the model predictive capabilities is provided in Figure 4.13 and Figure 4.14.

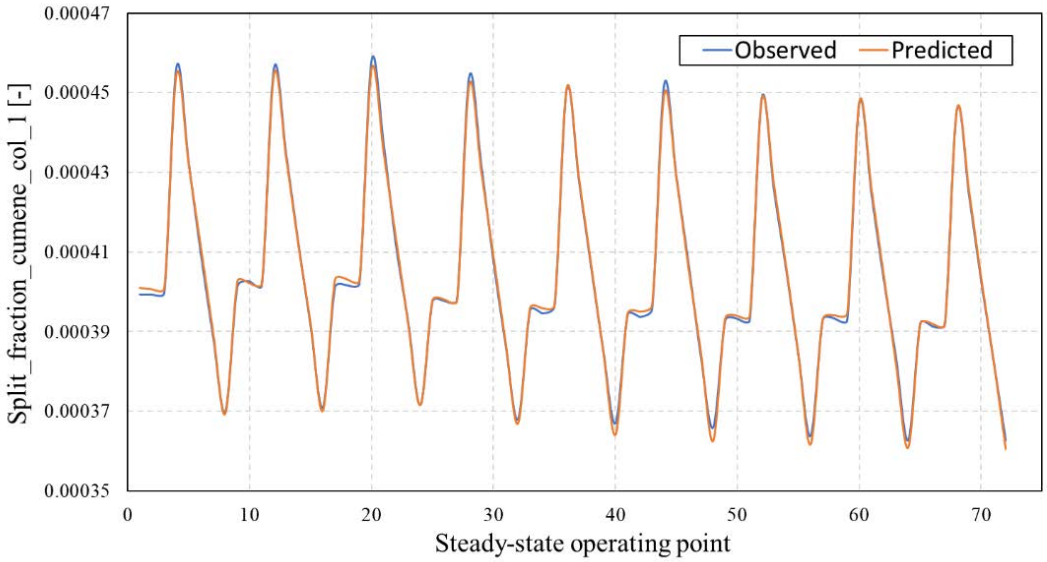


Figure 4.13. *Observed and predicted values of the cumene split fraction in the first distillation column as a function of the steady-state operating point.*

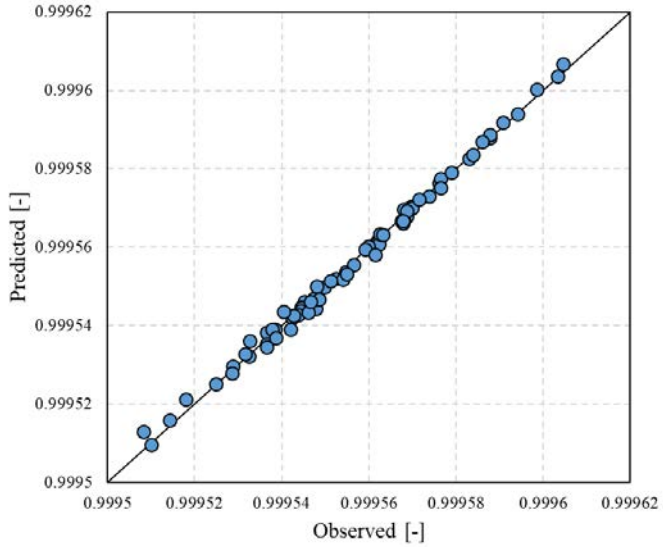


Figure 4.14. *Predicted vs. observed values of the cumene split fraction in the first distillation column.*

Since all the predicted/observed pairs lie close the $y = x$ line, the model is able to provide accurate estimations.

4.1.3.6 Data-driven element for the estimation of the cumene split fraction in the second distillation column obtained via PLS regression

Once loaded in Python™ the data stored in Table 4.5 and in vector (c) of Table 4.6, with the help of the scikit-learn package, a PLS regression and a 5-fold cross validation were performed. The results of the analysis are collected in Table 4.9.

Table 4.9. *Performance of the data-driven model developed for the estimation of the cumene split fraction.*

Number of PC	RMSECV $\times 10^8$	Q^2	R^2
1	3.1046	0.9765	0.9847
2	2.2297	0.9884	0.9920
3	1.4495	0.9952	0.9965
4	1.1809	0.9968	0.9977
5	1.1279	0.9971	0.9980
6	0.9320	0.9980	0.9986
7	0.9346	0.9981	0.9987
8	0.9488	0.9980	0.9988
9	0.9718	0.9979	0.9988
10	0.9723	0.9979	0.9988
11	0.9768	0.9979	0.9988
12	0.9765	0.9979	0.9988
13	0.9765	0.9979	0.9988

As can be noticed in Table 4.9, the model with the lowest $RMSECV$ employs six PCs. The model validation curve is reported in Figure 4.15.

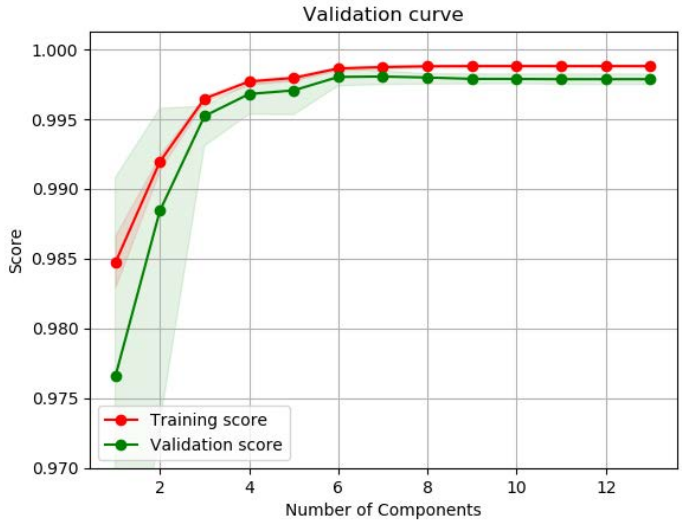


Figure 4.15. *Validation curve of the PLS regression model developed for the estimation of the cumene split fraction in the second distillation column.*

The overfitting arises when more than six PCs are taken into account, but, once again, as in the previous case (§4.1.3.5), it can be barely noticed in Figure 4.15.

The learning curve of the model is reported in figure 4.16.

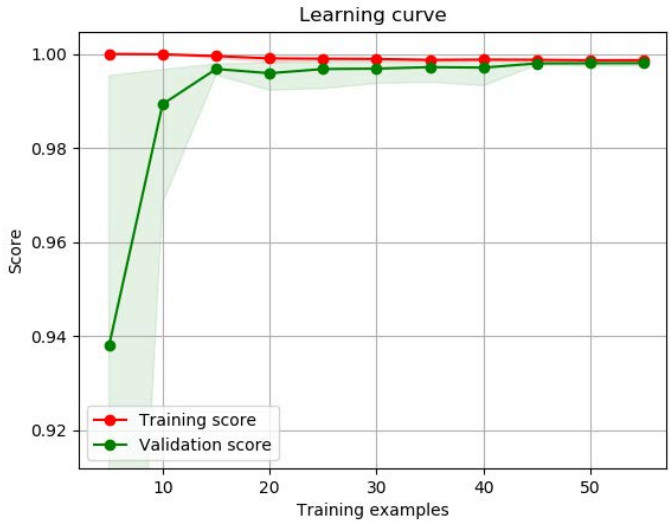


Figure 4.16. *Learning curve of the PLS regression model developed for the estimation of the cumene split fraction in the second distillation column.*

As can be noticed in Figure 4.16, the validation and a training curves converge steeply to a point of stability with a tiny gap between the two scores (both very close to one), therefore no more training instances are needed to calibrate the model.

The empirical correlation for the split fraction prediction obtained through PLS regression is defined as follows:

$$
\begin{aligned}
SF_{C_9H_{12},C2} = {} & 0.999951513 - 2.56 \times 10^{-6}\left(\frac{T_{F2} - \overline{T_{F2}^{m}}}{\sigma_{T_{F2}^{m}}}\right) + 6.70 \times 10^{-7}\left(\frac{P_{F2} - \overline{P_{F2}^{m}}}{\sigma_{P_{F2}^{m}}}\right) \\
& + 2.51 \times 10^{-6}\left(\frac{F_{F2} - \overline{F_{F2}^{m}}}{\sigma_{F_{F2}^{m}}}\right) - 1.90 \times 10^{-7}\left(\frac{x_{C_6H_6,F2} - \overline{x_{C_6H_6,F2}^{R}}}{\sigma_{x_{C_6H_6,F2}^{R}}}\right) \\
& - 1.13 \times 10^{-6}\left(\frac{x_{C_9H_{12},F2} - \overline{x_{C_9H_{12},F2}^{R}}}{\sigma_{x_{C_9H_{12},F2}^{R}}}\right) \\
& - 2.50 \times 10^{-6}\left(\frac{x_{C_{12}H_{18},F2} - \overline{x_{C_{12}H_{18},F2}^{R}}}{\sigma_{x_{C_{12}H_{18},F2}^{R}}}\right) - 2.91 \times 10^{-8}\left(\frac{T_{D2} - \overline{T_{D2}^{m}}}{\sigma_{T_{D2}^{m}}}\right) \\
& - 3.87 \times 10^{-6}\left(\frac{P_{D2} - \overline{P_{D2}^{m}}}{\sigma_{P_{D2}^{m}}}\right) - 1.82 \times 10^{-7}\left(\frac{T_{R2} - \overline{T_{R2}^{m}}}{\sigma_{T_{R2}^{m}}}\right) \\
& - 2.24 \times 10^{-6}\left(\frac{P_{R2} - \overline{P_{R2}^{m}}}{\sigma_{P_{R2}^{m}}}\right) - 1.25 \times 10^{-6}\left(\frac{r2 - \overline{r2^{m}}}{\sigma_{r2^{m}}}\right) \\
& - 1.12 \times 10^{-6}\left(\frac{HC2 - \overline{HC2^{m}}}{\sigma_{HC2^{m}}}\right) - 1.40 \times 10^{-5}\left(\frac{HR2 - \overline{HR2^{m}}}{\sigma_{HR2^{m}}}\right),
\end{aligned}
\tag{4.8}
$$

where $\overline{T_J^m}$, $\overline{P_J^m}$, $\overline{F_J^R}$, $\overline{x_{i,J}^R}$, $\overline{r2^m}$, $\overline{HC2^m}$, $\overline{HR2^m}$, $\sigma_{T_j^m}$, $\sigma_{P_j^m}$, $\sigma_{F_j^m}$, $\sigma_{x_{i,j}^R}$, σ_{r2^m}, σ_{HC2^m} and σ_{HR2^m} are the averages and the standard deviations of the columns of the regressors matrix reported in Table 4.5.

Finally, Figure 4.17 and Figure 4.18 provide a graphical evaluation of the model predictive performances.

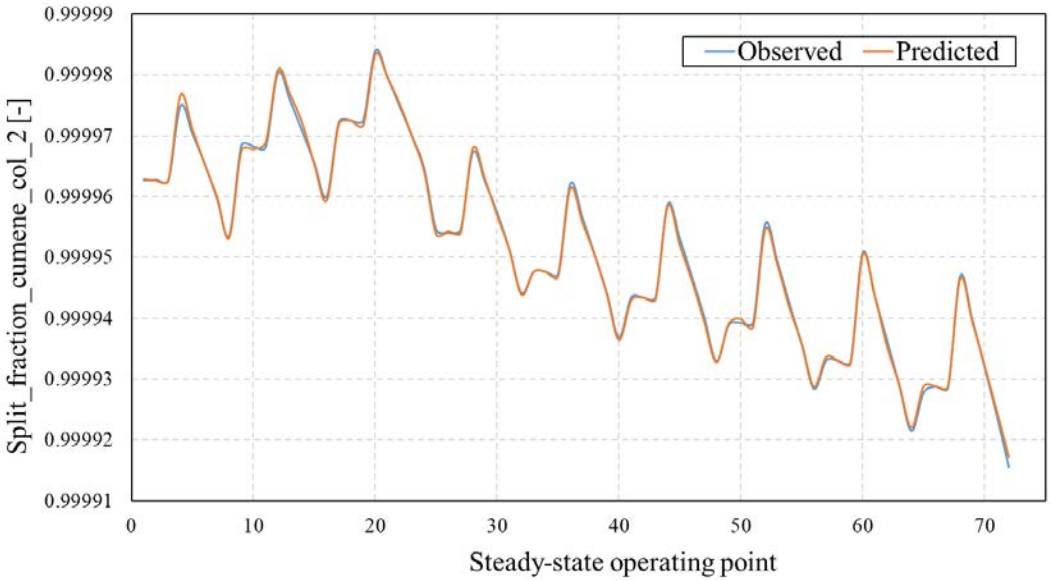


Figure 4.17. *Observed and predicted values of the cumene split fraction in the second distillation column as a function of the steady-state operating point.*

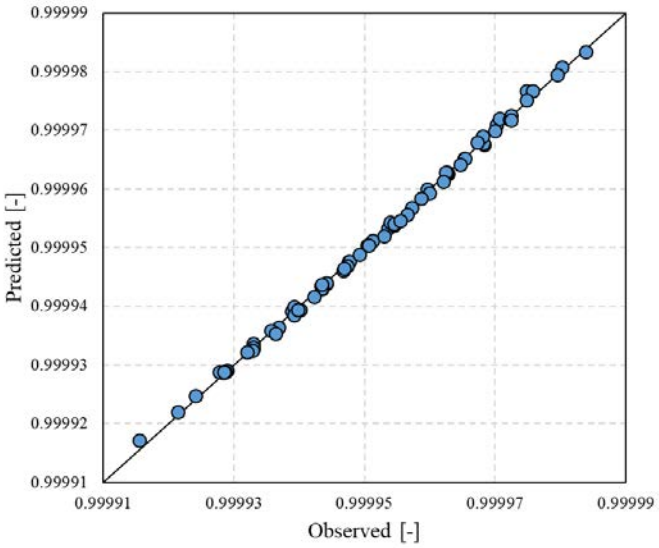


Figure 4.18. *Predicted vs. observed values of the cumene split fraction in the second distillation column.*

As in previous cases, since the trajectories in Figure 4.17 are very close to each other and all the points in Figure 4.18 are located in the proximity of the $y = x$ line, the quality of the model predictions is guaranteed.

4.1.3.7 Data-driven element for the estimation of the *p*-diisopropylbenzene split fraction in the second distillation column obtained via PLS regression

After loading the data of Table 4.5 and of the vector (d) of Table 4.6 in Python™, a PLS regression along with a 5-fold cross validation were carried out exploiting the scikit-learn package.

The outcome of the study is summarized in Table 4.10.

Table 4.10. *Performance of the data-driven model developed for the estimation of the p-diisopropylbenzene split fraction.*

Number of PC	RMSECV $\times 10^4$	Q^2	R^2
1	3.7181	0.3824	0.5207
2	1.3493	0.8111	0.9350
3	0.5837	0.9771	0.9915
4	0.5818	0.9768	0.9917
5	0.5817	0.9764	0.9921
6	0.5655	0.9769	0.9926
7	0.5661	0.9771	0.9927
8	0.5672	0.9766	0.9928
9	0.5668	0.9767	0.9928
10	0.5695	0.9763	0.9928
11	0.5697	0.9763	0.9928
12	0.5696	0.9763	0.9928
13	0.5696	0.9763	0.9928

As can be noticed in Table 4.10, the model with the lowest $RMSECV$ employs six PCs. Since the order of magnitude of the $RMSECV$ is remarkably greater than in previous cases, the split fraction of *p*-diisopropylbenzene is by far the most difficult to model. The model validation curve is reported in Figure 4.19.

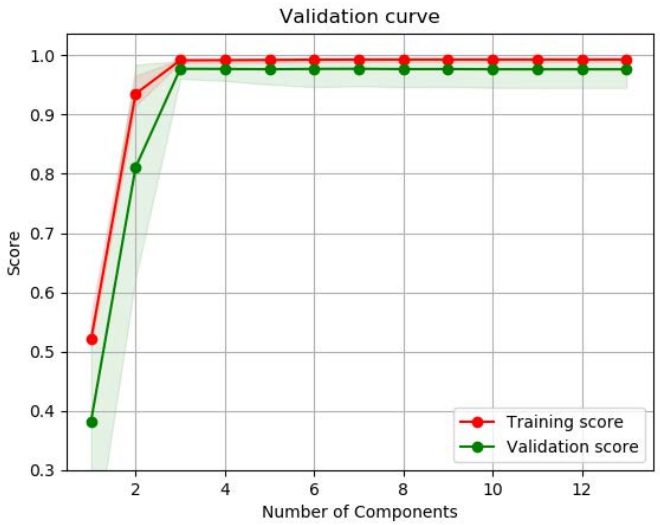


Figure 4.19. *Validation curve of the PLS regression model developed for the estimation of the p-diisopropylbenzene split fraction in the second distillation column.*

The learning curve of the model, instead, is reported in figure 4.20.

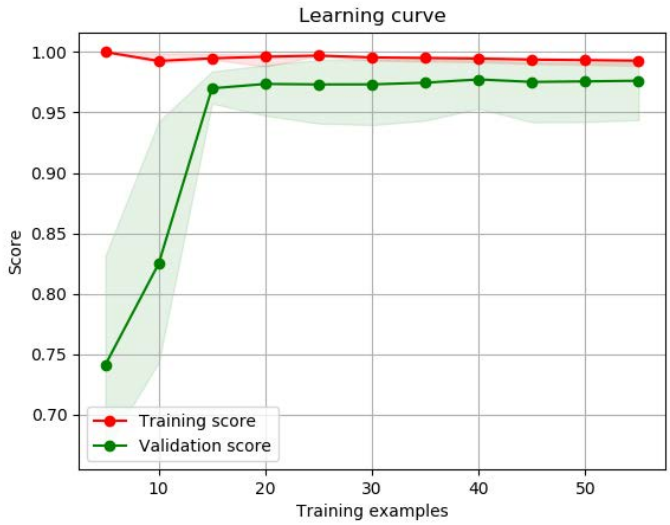


Figure 4.20. *Learning curve of the PLS regression model developed for the estimation of the p-diisopropylbenzene split fraction in the second distillation column.*

Despite the higher values of the $RMSECV$, similarly to the previous cases, the model would not benefit from the addition of more training examples because, as can be noticed in Figure 4.20, the validation and the training curves converge rapidly to a point of stability with a small gap between the two scores. In this case, however, the scores are not as close to one, indicating once again that the models calibrated with the information stored in Table 4.5 struggle the most when predicting the *p*-diisopropylenbenzene split fraction.

The empirical correlation for the split fraction prediction obtained through PLS regression is defined as follows:

$$
\begin{aligned}
SF_{C_{12}H_{18},C2} = {} & 0.016882056 + 3.46 \times 10^{-2}\left(\frac{T_{F2} - \overline{T_{F2}^{m}}}{\sigma_{T_{F2}^{m}}}\right) + 1.10 \times 10^{-3}\left(\frac{P_{F2} - \overline{P_{F2}^{m}}}{\sigma_{P_{F2}^{m}}}\right) \\
& + 1.13 \times 10^{-3}\left(\frac{F_{F2} - \overline{F_{F2}^{m}}}{\sigma_{F_{F2}^{m}}}\right) - 6.03 \times 10^{-3}\left(\frac{x_{C_6H_6,F2} - \overline{x_{C_6H_6,F2}^{R}}}{\sigma_{x_{C_6H_6,F2}^{R}}}\right) \\
& + 2.05 \times 10^{-5}\left(\frac{x_{C_9H_{12},F2} - \overline{x_{C_9H_{12},F2}^{R}}}{\sigma_{x_{C_9H_{12},F2}^{R}}}\right) \\
& - 1.13 \times 10^{-3}\left(\frac{x_{C_{12}H_{18},F2} - \overline{x_{C_{12}H_{18},F2}^{R}}}{\sigma_{x_{C_{12}H_{18},F2}^{R}}}\right) + 1.00 \times 10^{-2}\left(\frac{T_{D2} - \overline{T_{D2}^{m}}}{\sigma_{T_{D2}^{m}}}\right) \\
& - 1.45 \times 10^{-3}\left(\frac{P_{D2} - \overline{P_{D2}^{m}}}{\sigma_{P_{D2}^{m}}}\right) + 5.38 \times 10^{-3}\left(\frac{T_{R2} - \overline{T_{R2}^{m}}}{\sigma_{T_{R2}^{m}}}\right) \\
& - 1.66 \times 10^{-3}\left(\frac{P_{R2} - \overline{P_{R2}^{m}}}{\sigma_{P_{R2}^{m}}}\right) - 1.98 \times 10^{-2}\left(\frac{r2 - \overline{r2^{m}}}{\sigma_{r2^{m}}}\right) \\
& + 5.48 \times 10^{-3}\left(\frac{HC2 - \overline{HC2^{m}}}{\sigma_{HC2^{m}}}\right) + 2.73 \times 10^{-2}\left(\frac{HR2 - \overline{HR2^{m}}}{\sigma_{HR2^{m}}}\right),
\end{aligned}
\tag{4.9}
$$

where $\overline{T_J^m}$, $\overline{P_J^m}$, $\overline{F_J^R}$, $\overline{x_{i,J}^R}$, $\overline{r2^m}$, $\overline{HC2^m}$, $\overline{HR2^m}$, $\sigma_{T_j^m}$, $\sigma_{P_j^m}$, $\sigma_{F_j^m}$, $\sigma_{x_{i,j}^R}$, σ_{r2^m}, σ_{HC2^m} and σ_{HR2^m} are the averages and the standard deviations of the columns of the regressors matrix reported in Table 4.5.

Finally, the model performances are visualized graphically in Figure 4.21 and Figure 4.22.

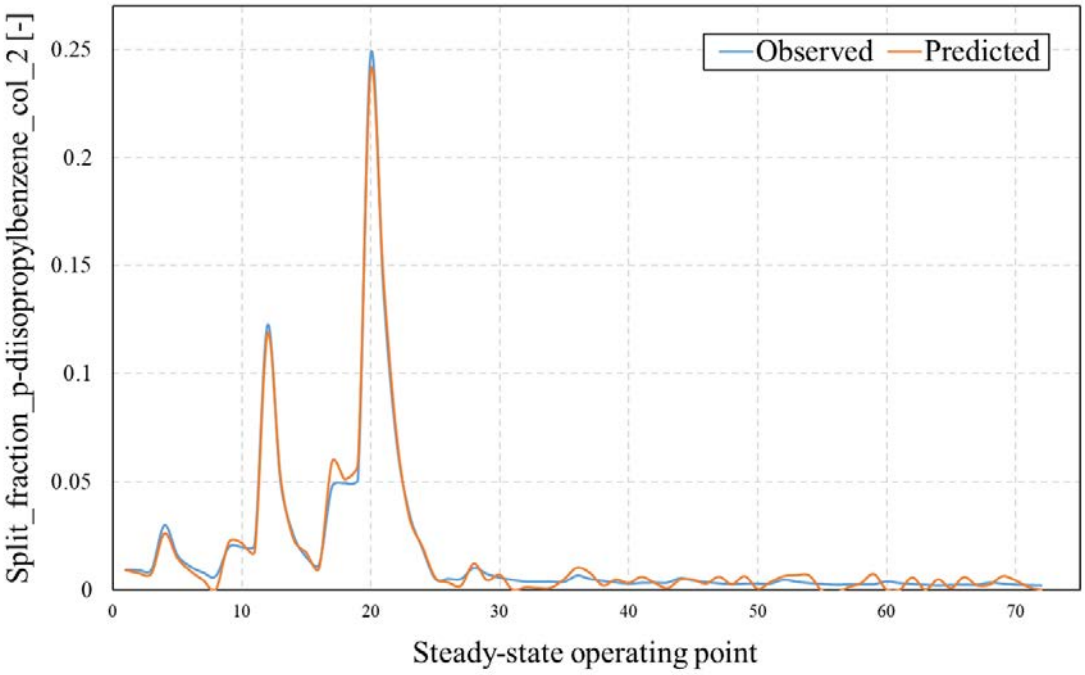


Figure 4.21. *Observed and predicted values of the p-diisopropylbenzene split fraction in the second distillation column as a function of the steady-state operating point.*

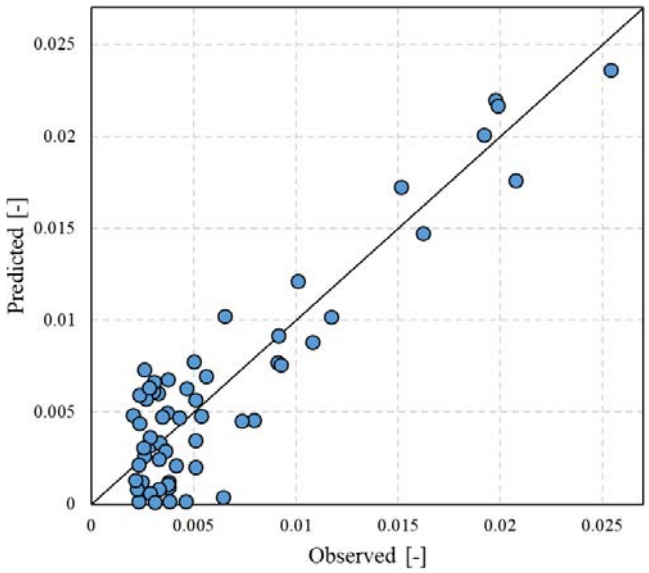


Figure 4.22. *Predicted vs. observed values of the p-diisopropylbenzene split fraction in the second distillation column.*

As can be seen in Figure 4.21 and 4.22, the trajectories are not perfectly close to each other (in particular in the last ten steady-state operating points) and the predicted/observed pairs do not lie exactly on the $y = x$ line as in benzene and cumene cases. Nonetheless, the empirical correlation is able to capture the overall trend of the observed values with a good level of accuracy and the predicted/observed pairs of Figure 4.22 do not depart excessively from the $y = x$ line. Therefore, the overall performance of the model is still satisfactory.

4.1.3.8 Data-driven elements built with the ALAMO model building methodology

Once the data stored in Table 4.4, Table 4.5 and Table 4.6 were loaded in Python™, the function implementing the ALAMO model building methodology was recursively ran with the aim to find the empirical correlations for the split fractions prediction. The ALAMO approach involves at first the definition of a set of non-linear transformations of the input variables (§2.2.2). In this Thesis, in particular, the set of basis functions consisted of polynomial and multinomial terms (no exponential or logarithmic terms were employed instead). The values of exponents α and α_d, that have been taken into account are summarized in Table 4.11.

4.11. *Values of the exponents of the polynomial and of the multinomial terms that have been taken into account throughout the development of the empirical correlations.*

	α	α_d
$SF_{C_6H_6,C1}$	± 1	± 1
$SF_{C_9H_{12},C1}$	$\pm 1, +2$	$\pm 1, +2$
$SF_{C_9H_{12},C2}$	± 1	± 1
$SF_{C_9H_{12},C2}$	$\pm 1, +2, +3$	$\pm 1, +2, +3$

Thus, for instance, when developing the data-driven element for the benzene split fraction prediction ($SF_{C_6H_6,C1}$) the ALAMO algorithm was forced to identify the best subset of bases among a set of multinomial and polynomials with either exponent +1 or -1 (e.g. x_i, x_i^{-1}, $x_i x_j$, $x_j^{-1} x_j^{-1}$). When the model building methodology was exploited to identify the empirical correlation for the estimation of the *p*-diisopropylbenzene ($SF_{C_9H_{12},C2}$), an extended starting basis functions set including also polynomials and multinomial with exponent +2 and +3 (namely $x_i^2, x_i^3, \; x_i^2 x_j^2, x_j^3 x_j^3$) was employed.

The empirical correlations for the split fractions prediction built with the ALAMO approach are defined as follows

$$SF_{C_6H_6,C1} = 0.999556056 - 1.7419 \times 10^{-5}\left(\frac{r1 - \overline{r1^m}}{\sigma_{r1^m}}\right) - 1.0805 \times 10^{-5}\left(\frac{P_{F1} - \overline{P_{F1}^m}}{\sigma_{P_{F1}^m}}\right), \qquad (4.10)$$

$$SF_{C_9H_{12},C1} = 0.000404396 - 2.478 \times 10^{-5}\left(\frac{x_{C_3H_8,F1} - \overline{x^R_{C_3H_8,F1}}}{\sigma_{x^R_{C_3H_8,F1}}}\right) - 2.740 \times 10^{-7}\left(\frac{r1 - \overline{r1^m}}{\sigma_{r1^m}}\right)^2\left(\frac{T_{F1} - \overline{T^m_{F1}}}{\sigma_{T^m_{F1}}}\right)^2, \quad (4.11)$$

$$SF_{C_{12}H_{18},C2} = 0.0042251425 - 0.01052\left(\frac{P_{R2} - \overline{P^m_{R2}}}{\sigma_{P^m_{R2}}}\right) - 0.01931\left(\frac{F_{F2} - \overline{F^m_{F2}}}{\sigma_{F^m_{F2}}}\right)\left(\frac{T_{D2} - \overline{T^m_{D2}}}{\sigma_{T^m_{D2}}}\right), \quad (4.12)$$

$$\begin{aligned} SF_{C_9H_{12},C2} = {} & 0.999952 - 1.5096 \times 10^{-5}\left(\frac{P_{R2} - \overline{P^m_{R2}}}{\sigma_{P^m_{R2}}}\right) + 3.394 \times 10^{-8}\left(\frac{x_{C_6H_6,F2} - \overline{x^R_{C_6H_6,F2}}}{\sigma_{x^R_{C_6H_6,F2}}}\right) \\ & + 3.241 \times 10^{-6}\left(\frac{x_{C_9H_{12},F2} - \overline{x^R_{C_9H_{12},F2}}}{\sigma_{x^R_{C_9H_{12},F2}}}\right) + 6.3728 \times 10^{-10}\left(\frac{P_{D2} - \overline{P^m_{D2}}}{\sigma_{P^m_{D2}}}\right)^{-1} \\ & - 1.888 \times 10^{-7}\left(\frac{T_{D2} - \overline{T^m_{D2}}}{\sigma_{T^m_{D2}}}\right)\left(\frac{x_{C_6H_6,F2} - \overline{x^R_{C_6H_6,F2}}}{\sigma_{x^R_{C_6H_6,F2}}}\right) \\ & + 1.072 \times 10^{-6}\left(\frac{x_{C_9H_{12},F2} - \overline{x^R_{C_9H_{12},F2}}}{\sigma_{x^R_{C_9H_{12},F2}}}\right)\left(\frac{P_{F2} - \overline{P^m_{F2}}}{\sigma_{P^m_{F2}}}\right). \end{aligned} \quad (4.13)$$

As can be noticed in the equations (4.10), (4.11), (4.12) and (4.13), the empirical correlations developed with ALAMO usually employ a much smaller number of terms with respect to the data-driven elements built with PLS (which consist on a linear regression exploiting all the process variables stored in the regressors matrices).

The performances of the correlations for the prediction of the split fractions of benzene, cumene and *p*-diisopropylbenzene are firstly assessed in Figure 4.23, 4.24, 4.25 and 4.26 comparing the predicted and observed values along with the corresponding steady-state operating point. Then, in Figure 4.27, 4.28, 4.29 and 4.30 the predicted and observed values of the benzene, cumene and *p*-diisopropylbenzene split fractions are directly displayed ones against each other together with the $y = x$ line.

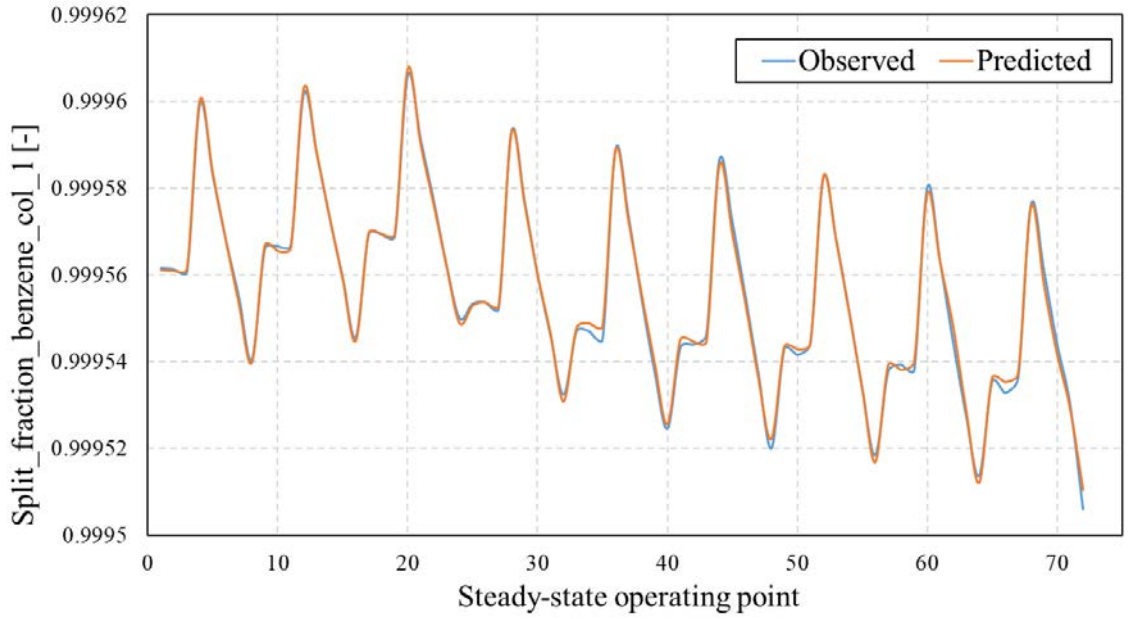


Figure 4.23. *Observed and predicted values of the benzene split fraction in the first distillation column as a function of the steady-state operating point.*

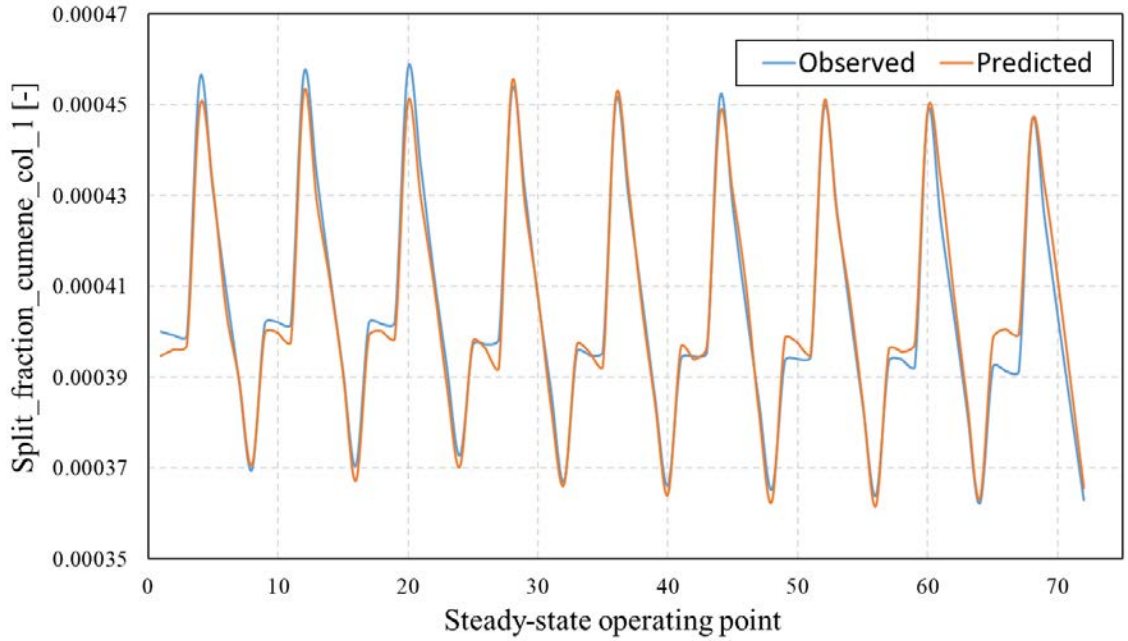


Figure 4.24. *Observed and predicted values of the cumene split fraction in the first distillation column as a function of the steady-state operating point.*

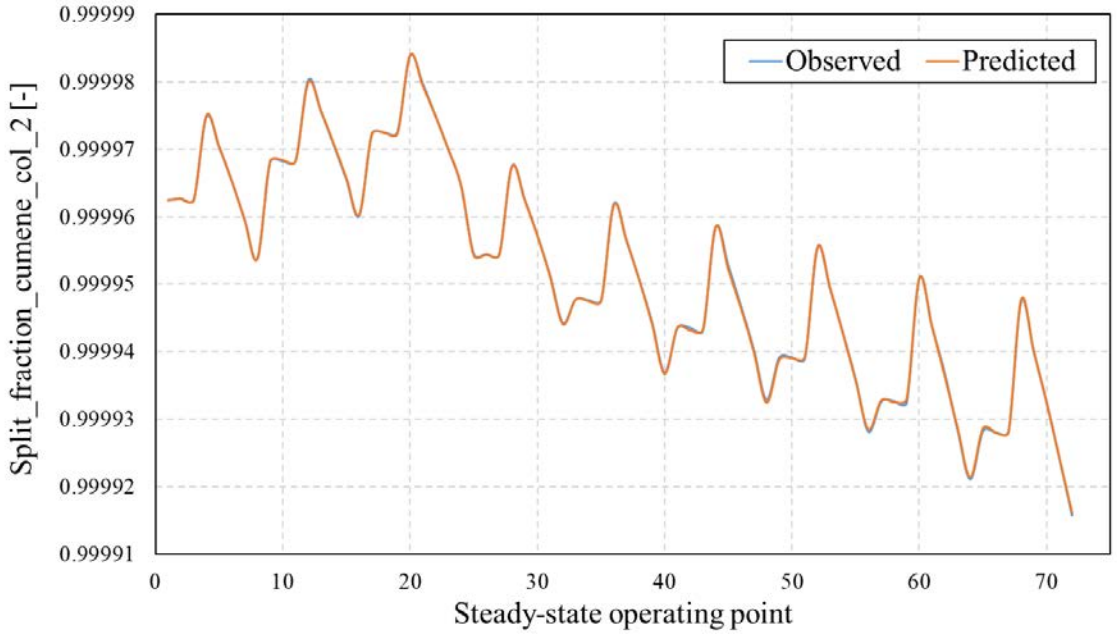


Figure 4.25. *Observed and predicted values of the cumene split fraction in the second distillation column as a function of the steady-state operating point.*

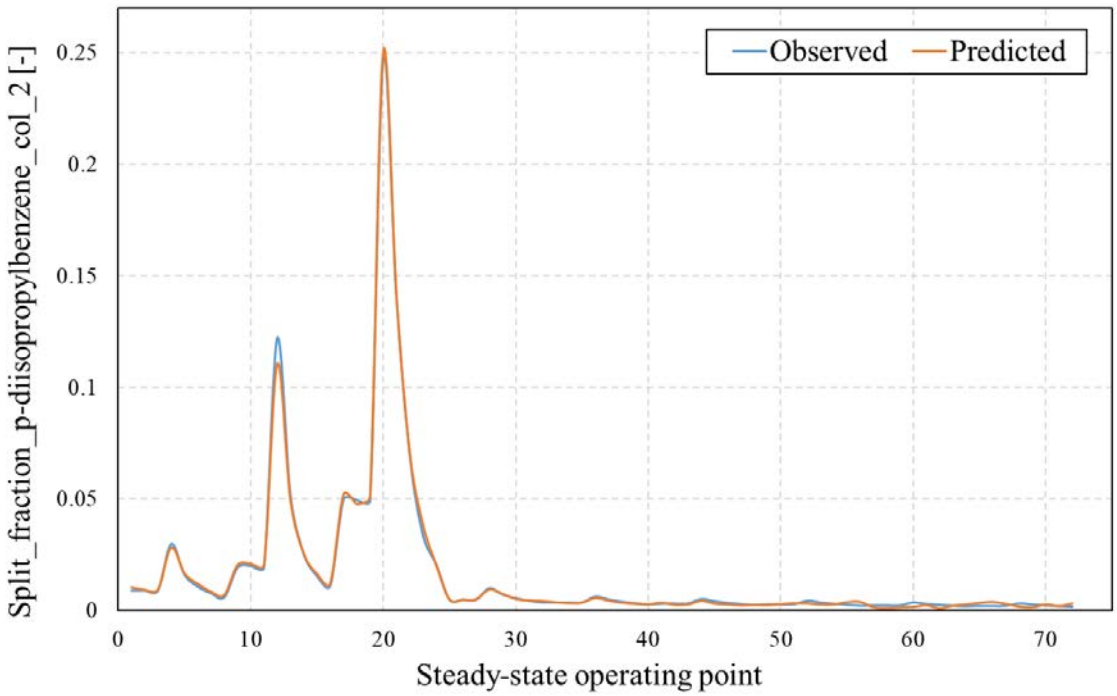


Figure 4.26. *Observed and predicted values of the p-diisopropylbenzene split fraction in the second distillation column as a function of the steady-state operating point.*

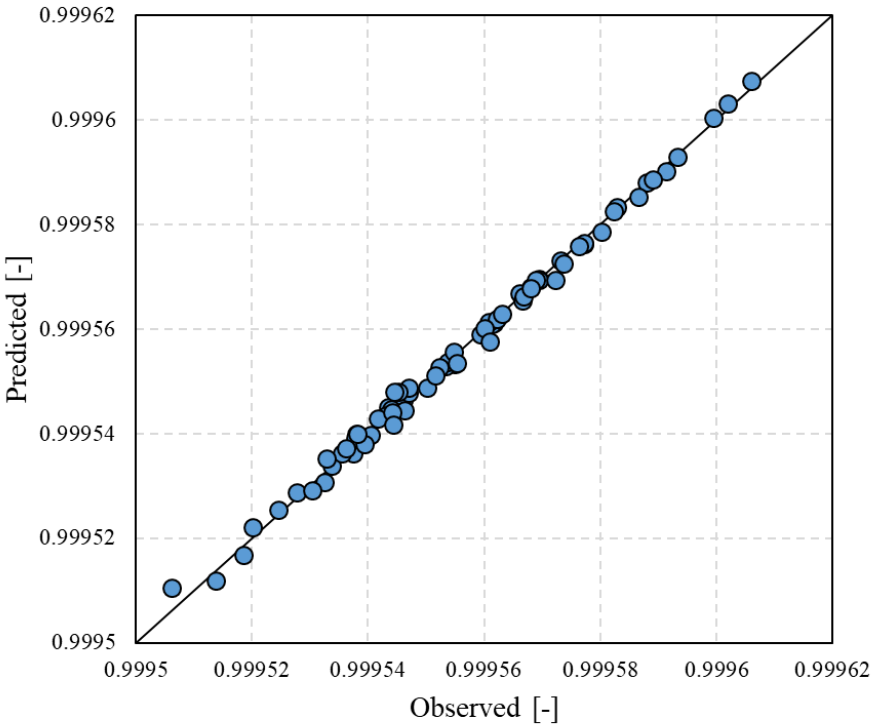


Figure 4.27. *Predicted vs. observed values of the benzene split fraction in the first distillation column.*

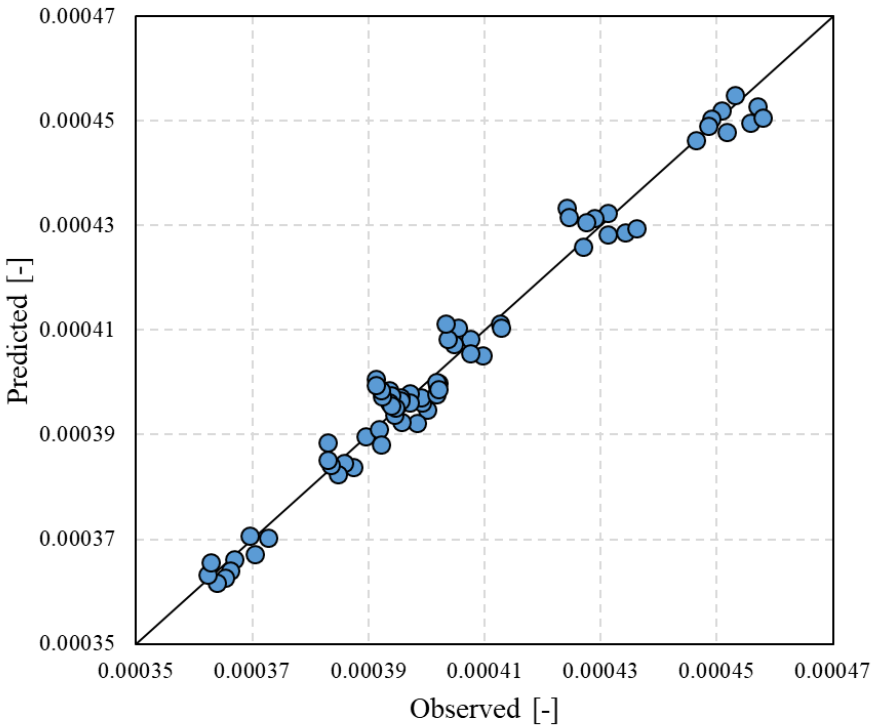


Figure 4.28. *Predicted vs. observed values of the cumene split fraction in the first distillation column.*

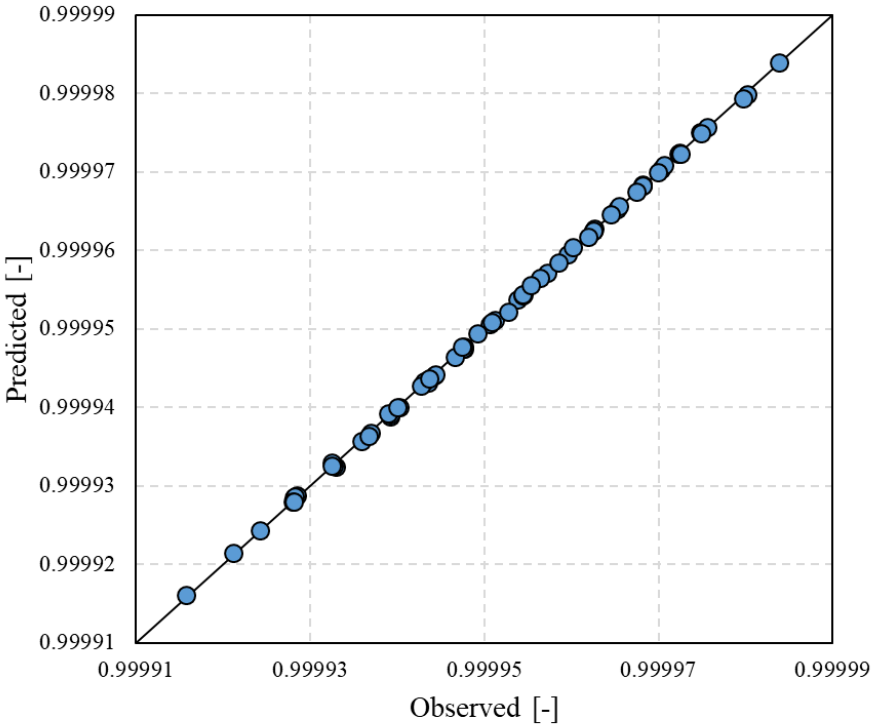


Figure 4.28. *Predicted vs. observed values of the cumene split fraction in the second distillation column.*

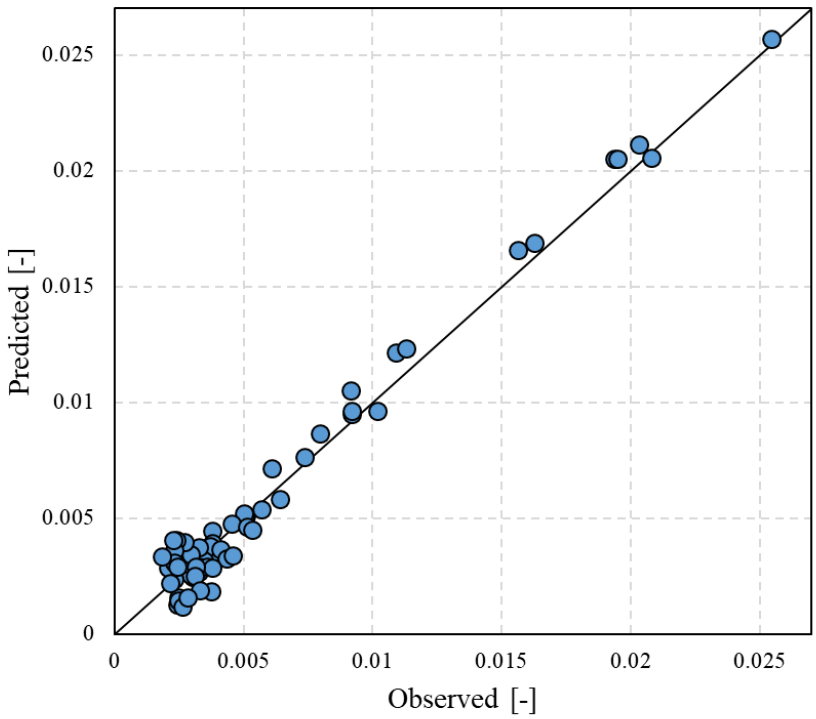


Figure 4.30. *Predicted vs. observed values of the p-diisopropylbenzene split fraction in the second distillation column.*

As can be seen in Figure 4.23, 4.24 and 4.25, the trajectories of the predicted and observed values are very close to each other in all the 72 steady-state operating points, therefore, with what concerns the prediction of the cumene and benzene split fractions, the performances of the empirical correlations developed with ALAMO are as satisfactory as those of the correlations built with PLS regression.
However, note that since with respect the pairs obtained with the PLS model, the predicted/observed pairs of Figure 4.30 lie consistently closer to the $y = x$ line, the ALAMO approach provides a more accurate data-driven element for the prediction of the *p*-diisopropylbenzene split fraction.

4.1.3.9 Final remarks on the data-driven elements development

In the last step of the hybrid model generation procedure a set of data-driven elements were developed to model the behaviour of the separation section. With both the ALAMO approach and the PLS regression, in particular, empirical correlations were built to predict the values of the split fractions of benzene and cumene in the first distillation column and the values of the split fractions of cumene and *p*-diisopropylbenzene in the second distillation column. Despite employing a lower number of inputs, the data-driven models built with the ALAMO methodology proved to be able to provide predictions which are as accurate as (or in the case of the split fraction of *p*-diisopropylbenzene even more accurate than) the predictions of the models developed through PLS regression. Therefore, in the gPROMS implementation of the hybrid model of the plant for the production of cumene, it was decided to exploit the empirical correlations built with the ALAMO methodology.

4.2 Hybrid model implementation in gPROMS

Once have been developed the empirical correlations needed to model the behaviour of the separation section, a steady-state hybrid model of the industrial plant for the production of cumene was implemented in gPROMS combining the black-box model of the train of distillation columns with the first principles models of the rest of the equipment. In the following sections, firstly it is discussed how the empirical correlations were integrated in the gPROMS model. Then, the tests which were carried out to assess the hybrid model predictive capabilities are reviewed.

4.2.1 Data based mass balance component splitter

In Figure 4.31 it is reported the graphical representation of the hybrid model as displayed in the gPROMS interface.

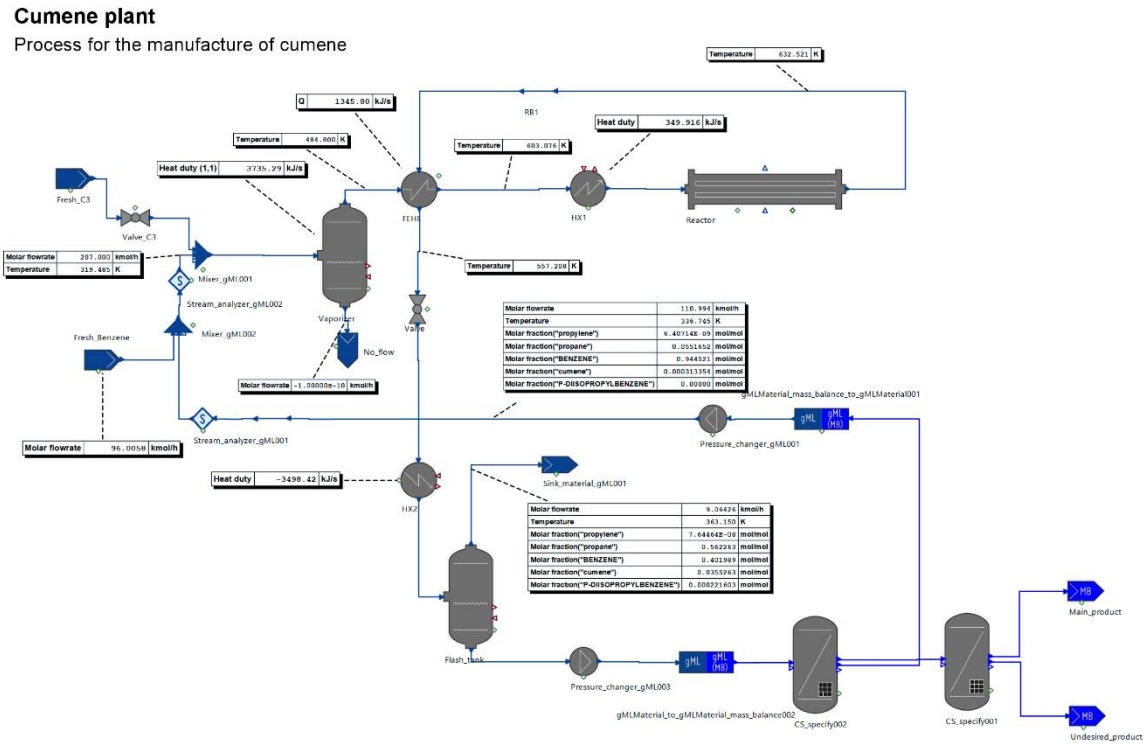


Figure 4.31. *Hybrid model of the plant for the production of cumene (gPROMS interface).*

As can be noticed in Figure 4.31 and Figure 4.32, the separation section is modelled with two data based mass balance component splitter. The gPROMS model 'data based mass balance component splitter' is a particular type of component splitter that allows to define the split fractions as a function of some input variables and parameters.

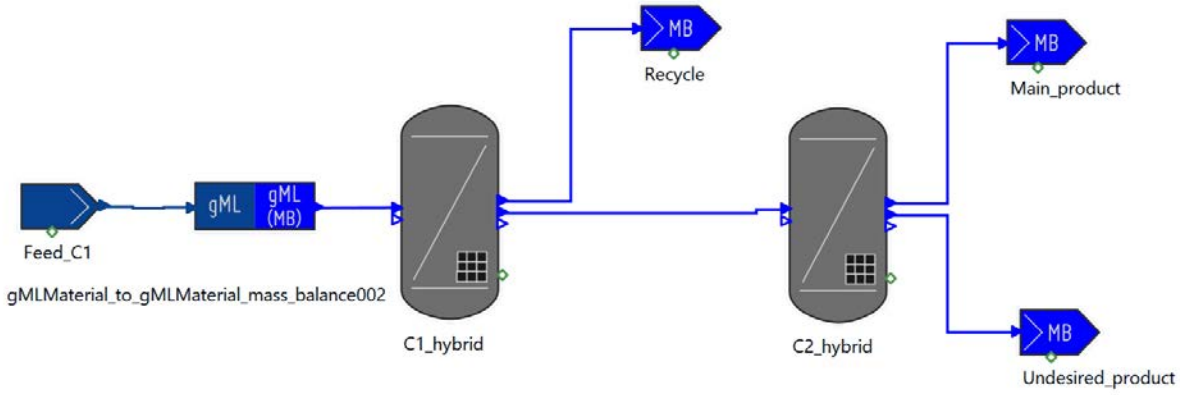


Figure 4.32. *Insight on the separation section of the hybrid model of the plant for the production of cumene.*

In order to overview the functionalities of the data based model, the first component splitter (named C1_hybrid in Figure 4.31) is taken as an example.

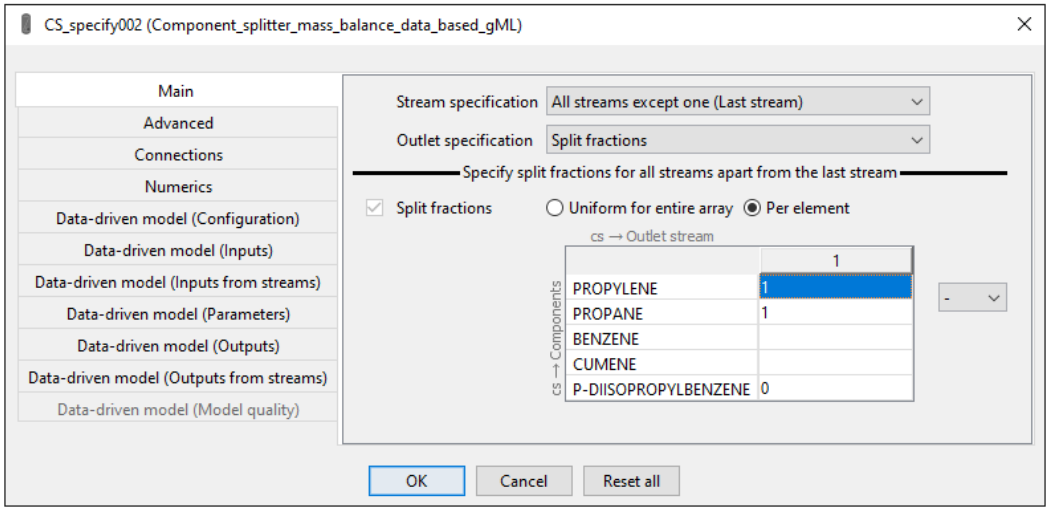


Figure 4.33. *Main tab of the first data based component splitter.*

In the 'main' tab of the model, as can be noticed in Figure 4.33, the user is required to specify a priori the values of those split fraction which, instead of being estimated through an empirical correlation, will remain constant throughout the simulation. Indeed, consistently with the assumptions summarized in table 4.2, the values of the split fractions of propylene and propane were set equal to 1 while the split fraction of *p*-diisopropylbenzene was set equal to zero.

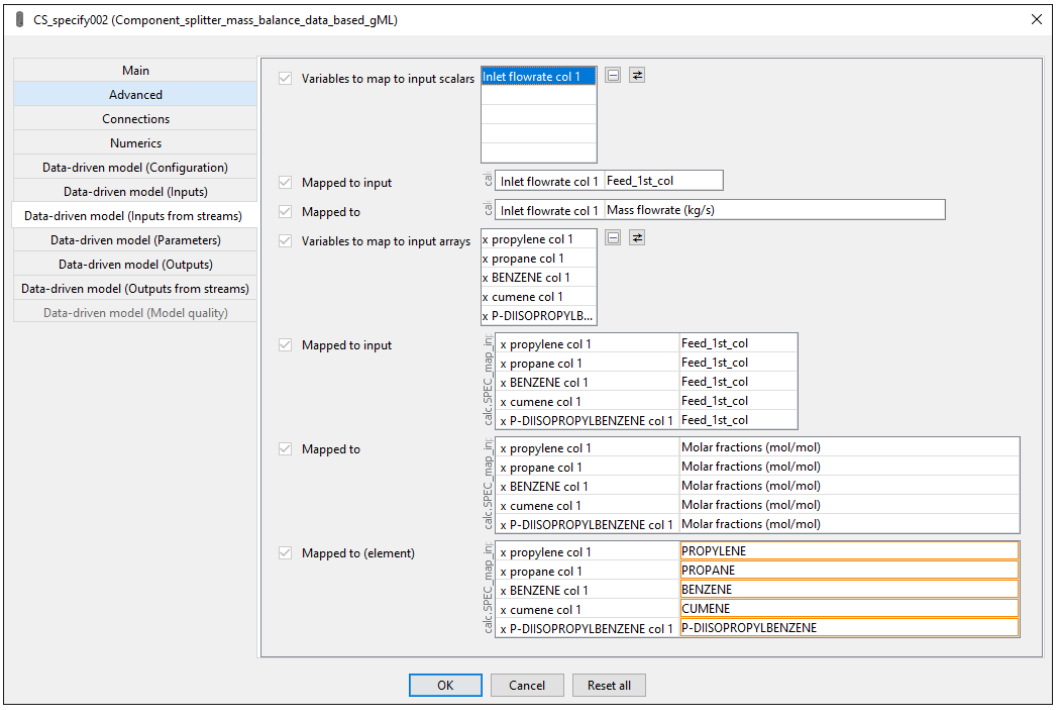


Figure 4.34. *Input from stream tab of the first data based component splitter.*

The ‘input from streams’ tab points out those inputs that are calculated in the previous section of the model, while the ‘parameters’ tab summarizes the process variables the user is required to assign a value to. In the case of the first component splitter of the separation section, as can be noticed in Figure 4.34 and 4.35, the model updates autonomously the values of the overall molar flowrate and of the composition of the feed, while temperatures and pressures of feed, distillate and residue as well as the reflux ratio must be specified by the user.

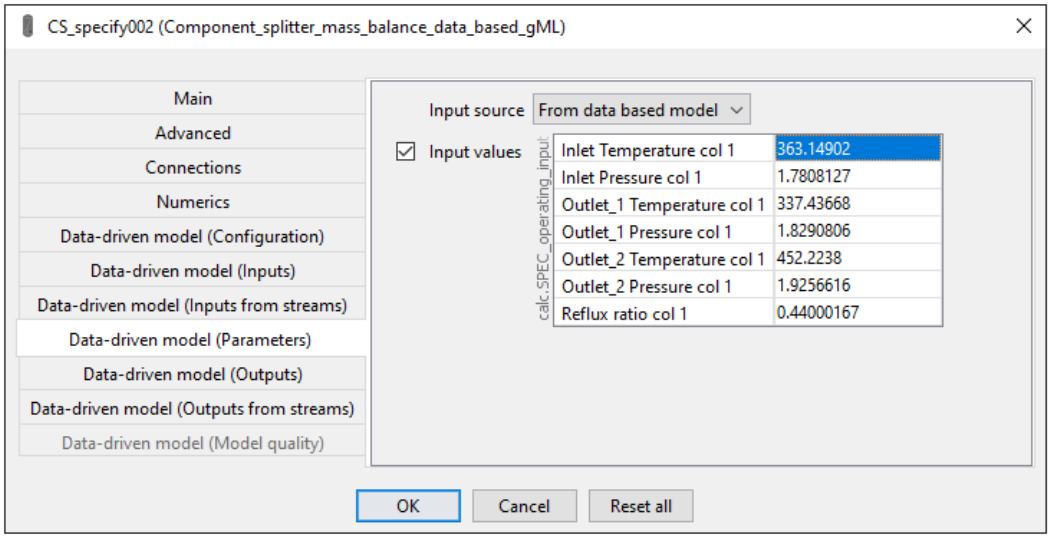


Figure 4.35. *Parameters tab of the first data based component splitter.*

Finally, in the ‘output from stream’ tab, the variables estimated by the data-driven element are indicated. The first component splitter, in particular, as can be seen in the bottom right of Figure 4.36, provides a prediction for the benzene and cumene split fractions.

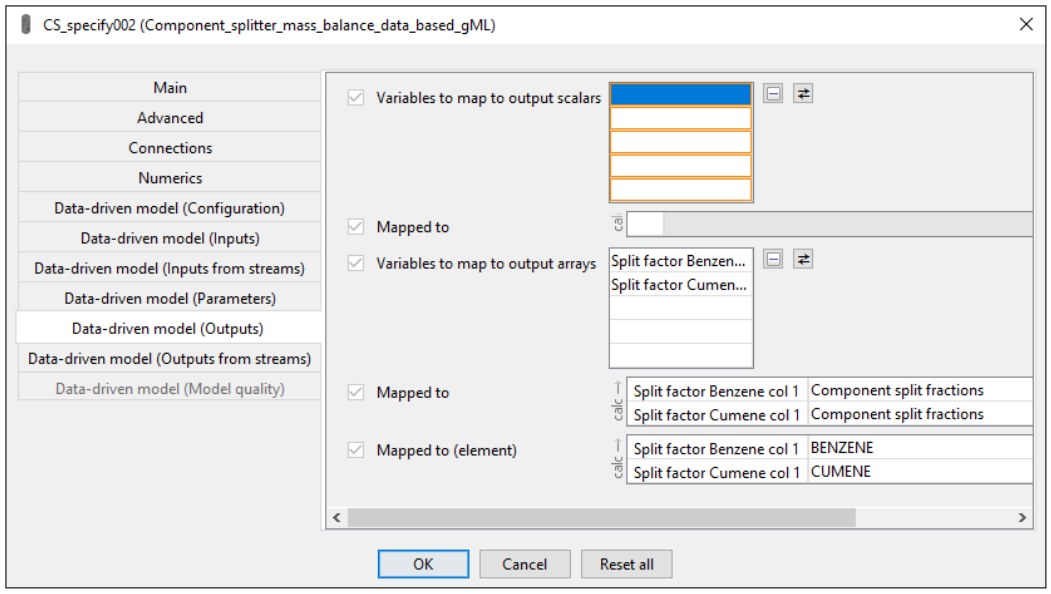


Figure 4.36. *Outputs tab of the first data based component splitter.*

The equation of the empirical correlations employed for the split fraction prediction (equation 4.10 and 4.11) are not specified directly in the data based splitter but, as can be seen in Figure 4.37, are defined in an external custom model.

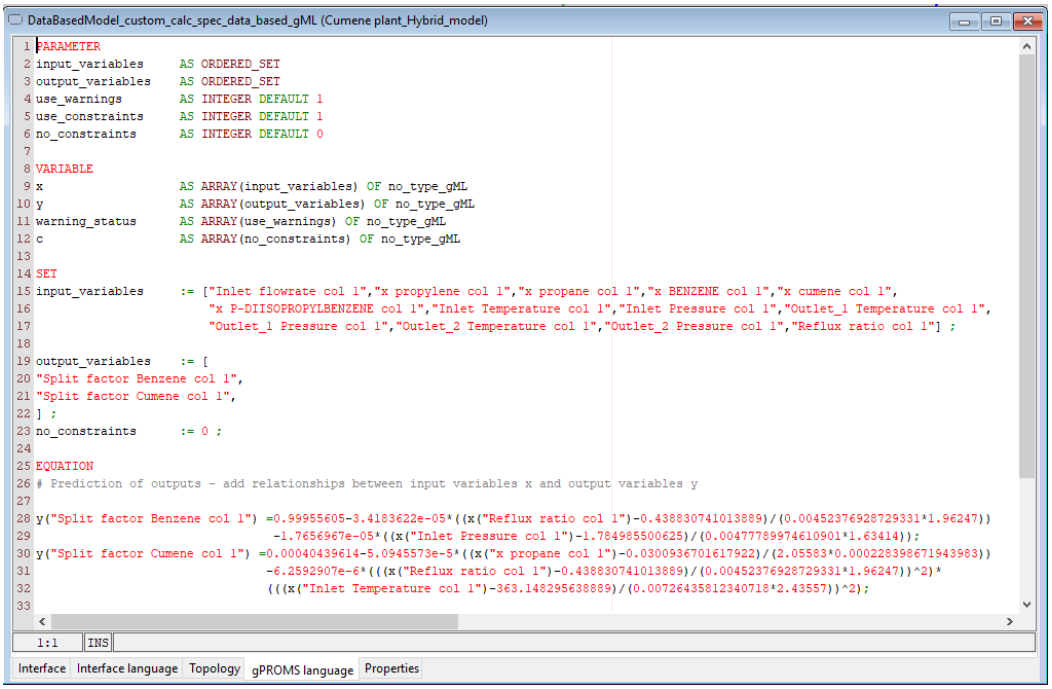


Figure 4.37. *Parameters tab of the first data based component splitter.*

Once the correlations are defined, the external custom model is connected with the data based mass balance component splitter in the 'configuration' tab.

4.2.2 Hybrid model performance assessment

Since an accurate and robust first principles model of the industrial plant for the production of cumene was available (and was already implemented in gPROMS), it was decided to assess the predictive capabilities of the hybrid model testing its outcomes against the predictions of the first principles model.

Hence, in order to collect information about some steady-state operating points that were not employed to calibrate the data-driven element of the hybrid model, four new dynamic simulations (of the first principles model) were carried out in gPROMS introducing disturbances which are different from the ones listed in §4.1.3.3 In particular, in the first simulation the flowrate of fresh C3 feed was decreased by 4% with respect to its nominal value through a rampchange; in the second simulation the flowrate of fresh C3 feed was increased by 6% with respect to its nominal value through a rampchange; in the third simulation the molar fraction of propylene in the fresh C3 feed was decreased to 0.93 through a stepchange; and finally, in the fourth simulation, the flowrate of fresh C3 feed was increased by 9% with respect to its nominal value through a rampchange and the molar fraction of propylene in fresh C3 feed

was decreased to 0.93 through a stepchange. In all four dynamic simulations, after the last disturbance, the flowsheet was allowed to run until a new steady-state was attained. Then, the values of the process variables in the last sampling instant were saved for a posterior comparison. In the meanwhile, the same steady-state operating points were simulated with the hybrid model.

The reference variables that were taken into account for the prediction comparison are the five component molar flowrates in the main product stream (namely the distillate of the second distillation column). The values estimated by the first principles and by the hybrid models are summarized in Table 4.11 and Table 4.12, respectively.

Table 4.11. *First principles model predictions.*

Steady-state operating point	$F_{C_3H_6,D2}$ [kmol/h]	$F_{C_3H_8,D2}$ [kmol/h]	$F_{C_6H_6,D2}$ [kmol/h]	$F_{C_9H_{12},D2}$ [kmol/h]	$F_{C_{12}H_{18},D2}$ [kmol/h]
1	6.57×10^{-24}	1.47×10^{-11}	0.04352	83.7987	0.12936
2	1.16×10^{-25}	1.71×10^{-11}	0.04990	92.9856	0.02047
3	2.43×10^{-24}	1.48×10^{-11}	0.04457	86.2514	0.05249
4	0.0	1.81×10^{-11}	0.05226	95.9787	0.01645

Table 4.12. *Hybrid model predictions.*

Steady-state operating point	$F_{C_3H_6,D2}$ [kmol/h]	$F_{C_3H_8,D2}$ [kmol/h]	$F_{C_6H_6,D2}$ [kmol/h]	$F_{C_9H_{12},D2}$ [kmol/h]	$F_{C_{12}H_{18},D2}$ [kmol/h]
1	0.0	0.0	0.04356	83.7988	0.13389
2	0.0	0.0	0.04989	92.9858	0.01987
3	0.0	0.0	0.04455	86.2511	0.05474
4	0.0	0.0	0.05224	95.9787	0.01566

As can be noticed in Table 4.12 and 4.13, the hybrid model estimations of the benzene and cumene molar flowrates match almost perfectly with the values computed by the first principles model. In the case of the *p*-diisopropylbenzene, instead, the molar flowrate calculated by the hybrid model provides a satisfactory approximation of the value calculated by the first principles model, but the prediction is not accurate as in the benzene and cumene cases.

The performances of the hybrid model were then further evaluated graphically in Figure 4.38, 4.39 and 4.40 generating some predicted vs. observed plots. Since the values calculated by the first principles model are assumed to be the correct ones, in the plots they are considered as the observed values, while the hybrid model estimations are considered as the predicted values.

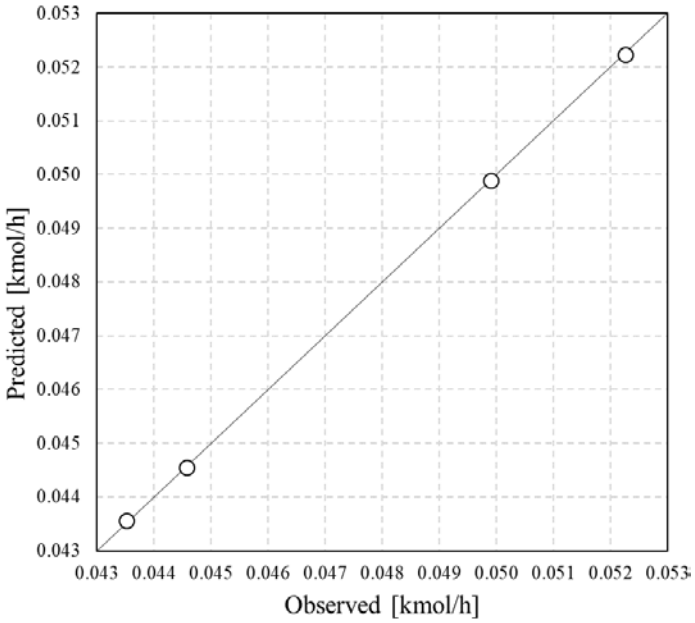


Figure 4.38. *Predicted vs. observed benzene molar flowrate.*

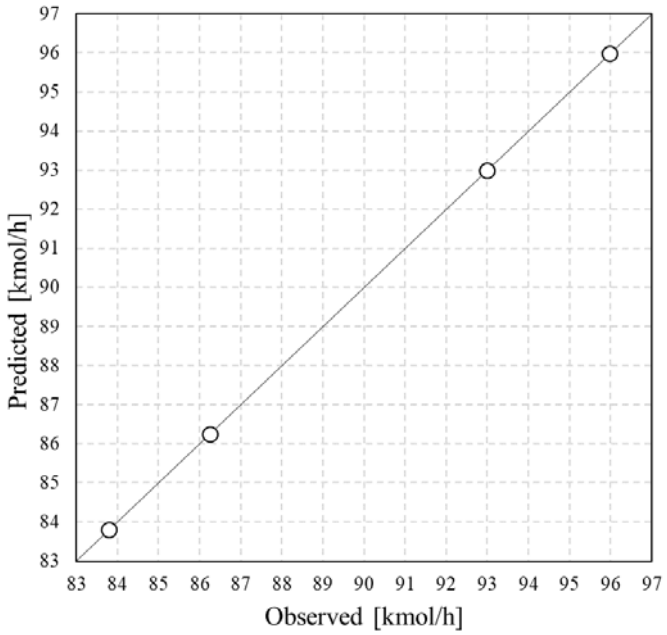


Figure 4.39. *Predicted vs. observed cumene molar flowrate.*

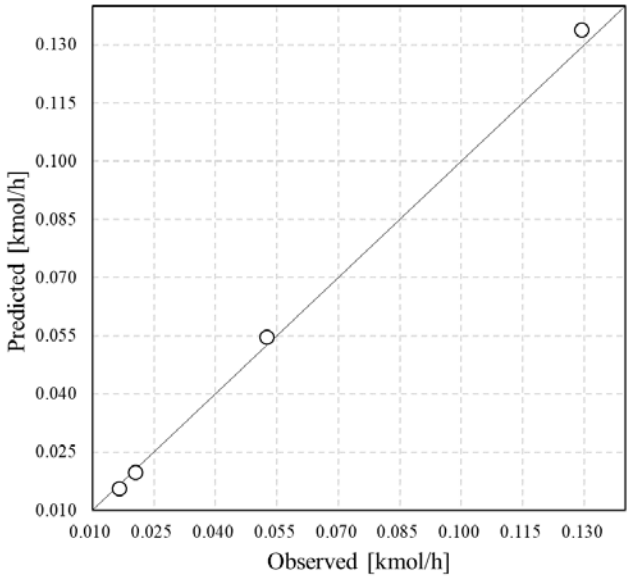


Figure 4.40. *Predicted vs. observed p-diisopropylbenzene molar flowrate.*

As can be noticed in Figure 4.38 and 4.39, since the predicted/observed pairs lie almost perfectly on the $y = x$ axis, the hybrid model is able to provide extremely accurate estimations of the benzene and cumene molar flowrates. In the case of the *p*-diisopropylbenzene, the predicted/observed pairs are still located close to the $y = x$ line, indicating a satisfactory predictive capability, but the accuracy of the estimation is not as remarkable as in the benzene and cumene case. The difference in the accuracy of the predictions falls back on the quality of the empirical correlations developed for the split fractions prediction. As shown in §4.1.3.7 and §4.1.3.8, the *p*-diisopropylbenzene split fraction proved to be the split fraction that the data-driven model struggled the most to predict.

Conclusions

The aim of this Thesis was to demonstrate the potential of hybrid modelling. For this purpose, the industrial process for the production of cumene was taken into account as a case study and, under the assumption that the separation section could not be modelled through first principles, a hybrid model of the plant was developed and implemented in gPROMS.

In order to reach these goals, the project was organized as follows.

First, a virtual plant historian was generated in gPROMS simulating in dynamic mode a detailed first principles model of the process which was already available within the Process Systems Enterprise libraries. Then, the historian was corrupted on purpose with noise and invalid measurements in Excel and Python™ to make it as similar as possible to an actual industrial process signals dataset.

Second, a steady-state detection algorithm was developed to identify, within the historian, the steady-state operating points. The performances of the novel steady-state algorithm were assessed through a series of diagnostic graphs. The outcome of the analysis proved that the steadiness predictions of the algorithm match accurately the evaluations that can be achieved inspecting visually the trajectory of the process signals.

Third, data reconciliation was performed on the separation section forcing the measurements of flowrate and composition to fulfill the mass conservation laws. Data reconciliation was carried out exploiting gPROMS optimization tool, which allows to minimize a user defined objective function varying the values of specific decision variables.

Fourth, a black-box model of the separation section was developed exploiting both the partial least squares regression and the ALAMO model building methodology. Empirical correlations were generated to predict the values of the split fractions of benzene and cumene in the first distillation column and the values of the split fractions of cumene and *p*-diisopropylbenzene in the second distillation column. Despite employing a lower number of inputs, the data-driven elements built with the ALAMO methodology proved to be able to provide predictions which are at least as accurate as those of the models developed through partial least squares regression. Therefore, it was decided to proceed with the hybrid model implementation in gPROMS employing the data-driven elements built with the ALAMO methodology.

In the last step of the project the hybrid model was implemented in gPROMS. The data-driven elements that map the behaviour of the separation section were integrated with the white-box models of the other process units.

Once implemented in gPROMS, the predictive capabilities of the hybrid model were tested against those of the first principles model considering four steady-state operating points which were not included in the calibration set of the data-driven elements. The predictions of the hybrid model matched accurately the values computed by the first principle model. Therefore,

it was demonstrated that, when the behaviour of a system or an equipment within a process cannot be easily described by the fundamental conservation laws, if process data are available, data-driven elements can be developed specifically to model those poorly understood systems. Then, those data-driven elements can be integrated within the white-box model of the rest of the plant without compromising the robustness and the reliability of the overall process model. The hybrid model development methodology described in this Thesis was successfully carried out with a real industrial case study too, starting the analysis from an actual plant historian and developing the black-box correlations for the process equipment whose behaviour could not be described via first principles modelling. Although results cannot be disclosed due to confidentiality reasons, the hybrid model proved to perform satisfactorily, thus confirming the potential of the methodology.

Finally, future work will aim first to make the steady-state detection algorithm suitable for online process steadiness evaluation. Second, it is wanted to allow the user to carry out hybrid modelling entirely within the gPROMS platform. For this purpose, the ALAMO model building methodology is currently being implemented inside the process simulator environment.

Appendix A

Insight on ALAMO's model-building methodology

In this Appendix the steps of the procedure through which Cozad *et al.* (2014) reformulated the best subset selection problem are described in detail.

A.1 Reformulation of the best subset selection problem

Cozad *et al.* (2014) reformulated the general best subset problem described by equation (1.29) in order to obtain a form that can be efficiently solved.

The first simplification is performed tracking which basis functions are active in the model through a binary vector y defined as follows: whenever a basis function $j \in B$ is active in the model ($j \in S$), then $y_j = 1$; otherwise $y_j = 0$.

The vector y allows to reformulate equation (1.29) into a mixed-integer nonlinear problem as follows:

$$\begin{aligned} &\min_{y,\beta} \quad \Phi(y,\beta) \\ &\text{s.t.} \quad y_j \in \{0,1\} \ . \end{aligned} \tag{A.1}$$

Through the binary vector, moreover, equation (1.30) can be described over the full set of bases B as follows:

$$\hat{z}(x) = \sum_{j \in B} y_j \beta_j X_j(x) \ . \tag{A.2}$$

The second step of the reformulation procedure implies replacing $y_j \beta_j$ with the following big-M constraints:

$$\beta^l y_j \leq \beta_j \leq \beta^u y_j \quad j \in B \ , \tag{A.3}$$

where β^l and β^u are the lower and the upper bounds, respectively. Through this step the complication linked with the integer bilinear terms is removed.

The third step of the reformulation consists on decoupling the goodness-of-fit measure into two parts as follows:

$$\min_{\beta,T,y} \Phi(\beta,T,y) = \min_{\mathrm{T}} \left\{ \min_{y,\beta} \left[\Phi_{y,\beta}(y,\beta)|_T \right] + \Phi_T(T) \right\} \ , \tag{A.4}$$

where $\Phi_T(T)$ is the model sizing part while $\Phi_{y,\beta}(y,\beta)|_T$ refers to the selection of the basis and the parameters.

Once decoupled the goodness-of fit measure, the best subset selection problem is posed as a nested minimization as follows:

$$\begin{aligned} \min_{\mathrm{T}\in\{1,\dots,\mathrm{T^u}\}} \quad & \left[\Phi_{y,\beta}(y,\beta)|_T \right] + \Phi_T(T) \\ \text{s.t.} \quad & \min_{y,\beta} \Phi_{y,\beta}(y,\beta)|_T \\ & \text{s.t.} \quad \sum_{j\in B} y_i = T \\ & \qquad \beta^l y_j \le \beta_j \le \beta^u y_j \qquad j \in B \\ & \qquad y_j \in \{0,1\} \qquad\qquad j \in B \ , \end{aligned} \tag{A.5}$$

where, as stated in §1.4.2, the inner minimization concerns with the selection of the basis functions and of the parameters while the outer minimization determines the complexity of the model.

The next step consists on the selection of the goodness-of-fit measure, Cozad *et al.* (2014) decided to use the corrected Akaike information criteria (Hurvich and Tsai, 1993):

$$AICc(S,\beta) = Nlog\left(\frac{1}{N} \sum_{i=1}^{N} \left(z_i - \sum_{j\in S} \beta_j X_{ij} \right)^2 \right) + 2|S| + \frac{2|S|(|S|+1)}{N-|S|-1} \ . \tag{A.6}$$

Equation (A.6) is then manipulated in order to be posed as a nested minimization and to be given in the form of (A.4).

Once the goodness-of-fit measure has been defined and implemented, then the solution space of the outer minimization is parametrized with respect to T and the constraint:

$$\sum_{j\in B} y_i = T \qquad j \in B \tag{A.7}$$

is included to highlight the fact that the solution of the inner minimization is carried out increasing the value of T until a minimum is reached. Variable T is an indicator for the complexity of the model.

The last steps of the reformulation procedure, finally, concerns with the inner minimization problem. In order to pose the inner minimization as a mixed-integer linear problem (MILP) firstly the nonlinear objective is replaced by the following L_1-norm error:

$$\min SE = \sum_{i=1}^{N} \left| z_i - \sum_{j \in B} \beta_j X_{ij} \right| \tag{A.8}$$

Then each instance of $|w|$ in (A.8) is replaced by w' and the following constraints are added:

$$\begin{aligned} w' &\geq w \\ w' &\geq -w \end{aligned} \tag{A.9}$$

In order to retain the least square nature of the coefficients, moreover, the stationary condition with respect to the parameter β is used as follows:

$$\frac{d}{d\beta_j} \sum_{i=1}^{N} \left(z_i - \sum_{j \in B} \beta_j X_{ij} \right)^2 \propto \sum_{i=1}^{N} X_{ij} \left(z^i - \sum_{j \in B} \beta_j X_{ij} \right) = 0, \quad j \in S \tag{A.10}$$

Finally, the equation (A.10) is used as big-M constraints to define the basis coefficient:

$$-U_j(1 - y_i) \leq \sum_{j \in B} X_{ij} \left(z^i - \sum_{j \in B} \beta_j X_{ij} \right) \leq U_j(1 - y_i) \tag{A.11}$$

After all the reformulations, the inner minimization can be expressed as the following mixed-integer linear problem (MILP):

$$\begin{aligned} \min \quad & \sum_{i=1}^{N} w_i \\ \text{s.t.} \quad & w_i \geq z_i - \sum_{j \in B} \beta_j X_{ij}, && i = 1, \dots, N \\ & w_i \geq \sum_{j \in B} \beta_j X_{ij} - z_i, && i = 1, \dots, N \\ & \sum_{j \in B} y_i = T \\ & -U_j(1 - y_i) \leq \sum_{j \in B} X_{ij} \left(z^i - \sum_{j \in B} \beta_j X_{ij} \right) \leq U_j(1 - y_i) , && j \in B \\ & \beta^l y_j \leq \beta_j \leq \beta^u y_j, && j \in B \\ & y_{kj} \in \{0,1\} && j \in B \\ & \beta_j^l \leq \beta_j \leq \beta_j^u, && j \in B \end{aligned} \tag{A.12}$$

As stated in (§1.4.2), solving the set of equations (A.12) with increasing values of T until the Akaike information criteria worsens, it can be identified the most accurate low-complexity model.

Appendix B

Software tools

This appendix briefly overviews the software that have been used to generate the hybrid model. At first an introduction on the Python™ programming language is provided and details on the packages and on the tools that allowed to implement the multivariate statistical techniques and the ALAMO methodology are given. Then, the programming language in which the steady state detection algorithm (§2) has been coded is specified. Finally, an insight on the process simulator gPROMS is given pointing out the tools of the software that have been exploited the most when carrying out the project.

B.1 Python™

Python™ is an object-oriented, interactive, interpreted programming language that combines remarkable power with an effective and clear syntax. It incorporates modules, high level dynamic data types and classes. Furthermore, Python™ is extensible in C or C++ and has interfaces to many system calls and libraries, as well as to various window systems (https://www.python.org).

B.1.1 Scikit-learn package

Scikit-learn is a Python™ module that provides a wide range of state-of-the-art machine learning algorithms for medium-scale supervised and unsupervised problems (Pedregosa *et. al.*, 2011). The Scikit-learn package presents an easy-to-use interface which is tightly integrated with the Python™ language. The aim of the module, indeed, is to bring machine learning to non-specialists using a general-purpose high-level language. Emphasis is therefore put on ease of use, performance, documentation and API consistency. The PC analysis and the projection on latent structures performed in this Thesis have been computed in Python™ using the algorithms implemented in the Scikit-learn library.

B.1.2 ALAMO

The ALAMO approach to model building described in §1.4 has been implemented by Process System Enterprise (PSE) in Python™. All the empirical correlations developed in this Thesis

following the steps of the ALAMO algorithm, therefore, have been obtained through the function provided by the company.

B.1.3 Steady-state detection algorithm

In this project, a steady-state detection algorithm has been developed from scratch in order to identify, from the trajectories of the process variables, the windows of time in which the plant is operating at steady-state. The detection algorithm has been coded in Python™.

B.2 gPROMS software®

The hybrid model developed in this Thesis has been implemented in the process simulator gPROMS, which is a modelling software by Process Systems Enterprise (PSE).
The gPROMS platform provides equation-oriented modelling and optimisation framework upon which the several PSE's gPROMS products operate. In particular, it allows for flowsheeting, powerful high-fidelity custom models construction, advanced parameter estimation, model validation and both steady-state and dynamic model simulation (https://www.psenterprise.com).

B.2.1 gPROMS ProcessBuilder

gPROMS ProcessBuilder is an advanced process simulation tool for model-based support of key process design and operating decisions. Built on the gPROMS Platform, gPROMS ProcessBuilder allows users to construct process flowsheets by dragging and dropping models, for basic and advanced unit operations, from a palette of model libraries. The process flowsheet can then be simulated or used in optimisation or model validation studies.
gPROMS can be used to optimise the steady-state and/or the dynamic behaviour of a continuous or batch process. Both plant design and operational optimisation can be carried out. The form of the objective function and the constraints can be quite general. Moreover, the optimisation decision variables can be either functions of time ("controls") or time-invariant quantities.
By default, gPROMS treats optimisation problems as dynamic ones, optimising the behaviour of a system over a finite non-negative time horizon. However, in some cases, it is desired to optimise a system at a single time point performing a so-called "point" optimisation. From the mathematical point of view, this is equivalent to solving a purely algebraic problem in which a generally nonlinear objective function is maximised or minimised subject to (generally) nonlinear constraints by manipulating a set of optimisation decision variables that may be either continuous or discrete. All the optimizations performed in this Thesis have been solved exploiting the DAEBDF solver which is the standard mathematical solver used by default by gPROMS to solve mixed sets of differential and algebraic equations (DAEs).

The DAEBDF solver is based on variable time step, variable order Backward Differentiation Formulae (BDF). The solver, in particular, adjust automatically each time step taken so that the following criterion is satisfied:

$$\sqrt{\frac{1}{n_d}\sum_{i=1}^{n_d}\left(\frac{\epsilon_i}{a + r|z_i|}\right)} < 1 \ , \tag{B.1}$$

where:

- n_d : number of differential variables in the problem;
- ϵ_i : solver estimate for the local error in the i^{th} differential variable;
- a : absolute error tolerance;
- r : relative error tolerance;
- z_i : current value the i^{th} differential variable.

This means that the error ϵ_i incurred in a particular variable z_i over a single time step is not allowed to exceed an estimate of $a + r|z_i|$.

The DAEBDF solver is designed to deal with large, sparse systems of equations in which variable values are restricted to specified lower and upper bounds. Moreover, it can handle situations in which some of the partial derivatives of the equations with respect to the variables are available analytically while the rest must be approximated.

Appendix C

The impact of measurement noise amplitude on the data-driven element predictive capabilities

In this Appendix it is discussed how different measurement noise amplitudes affect the performances of the black-box model of the train of distillation columns developed through PLS regression.

C.1 Outcome of the analysis

Once the virtual plant historian was generated performing in gPROMS dynamic simulations of the detailed first principles process model, as previously stated in §3, noise and invalid measurements were added on purpose in order to obtain a dataset which is as similar as possible to a real industrial process variables dataset. The noise addition, in particular, was carried out through a Python^{TM} function that allowed to modulate the amplitude of the fluctuations by changing the parameter k. Different values of the parameter k have been exploited to generate datasets with different levels of noise ($k = 0.025$, 0.1 and 0.25). Then, after carrying out data reconciliation, a PLS regression and 5-fold cross validation were performed with each of these datasets in order to assess the effect of noise in the training and validation data on the predictive capabilities of the separation section black-box model. The empirical correlation for the prediction of the benzene split fraction in the first distillation column was taken into account as a reference for the comparison. The outcome of the analysis is reported in Table C.1, C.2 and C.3 and in Figure C.1, C.2 and C.3.

Table C.1. *Performances of the data-driven model developed for the estimation of the benzene split fraction (with k = 0.025).*

Number of PC	RMSECV $\times 10^8$	Q^2	R^2
1	3.4028	0.9877	0.9891
2	1.5783	0.9971	0.9974
3	1.4541	0.9979	0.9982
4	0.9111	0.9990	0.9992
5	0.8900	0.9991	0.9992
6	0.7526	0.9993	0.9995
7	0.6703	0.9995	0.9996
8	0.6402	0.9995	0.9997
9	0.6213	0.9995	0.9997
10	0.6476	0.9994	0.9997
11	0.6370	0.9995	0.9997
12	0.6390	0.9995	0.9997
13	0.6600	0.9995	0.9997
14	0.6652	0.9994	0.9997

Table C.2. *Performances of the data-driven model developed for the estimation of the benzene split fraction (with k = 0.1).*

Number of PC	RMSECV $\times 10^8$	Q^2	R^2
1	4.0753	0.9799	0.9839
2	3.1615	0.9882	0.9903
3	2.8895	0.9903	0.9931
4	2.6408	0.9922	0.9940
5	2.4798	0.9932	0.9947
6	2.4406	0.9934	0.9953
7	2.4627	0.9933	0.9955
8	2.5411	0.9928	0.9957
9	2.5460	0.9928	0.9957
10	2.5747	0.9926	0.9957
11	2.6625	0.9921	0.9958
12	2.6719	0.9920	0.9958
13	2.6823	0.9920	0.9958
14	2.7334	0.9916	0.9958

Table C.3. *Performances of the data-driven model developed for the estimation of the benzene split fraction (with k = 0.25).*

Number of PC	RMSECV ×10^8	Q^2	R^2
1	4.7453	0.9666	0.9768
2	4.1472	0.9744	0.9828
3	4.2569	0.9742	0.9835
4	4.2788	0.9733	0.9842
5	4.3300	0.9707	0.9848
6	4.5321	0.9667	0.9855
7	4.4729	0.9672	0.9858
8	4.5408	0.9659	0.9862
9	4.5258	0.9639	0.9865
10	4.6242	0.9652	0.9868
11	4.6611	0.9655	0.9869
12	4.6912	0.9654	0.9869
13	4.7380	0.9645	0.9869
14	5.0122	0.9642	0.9873

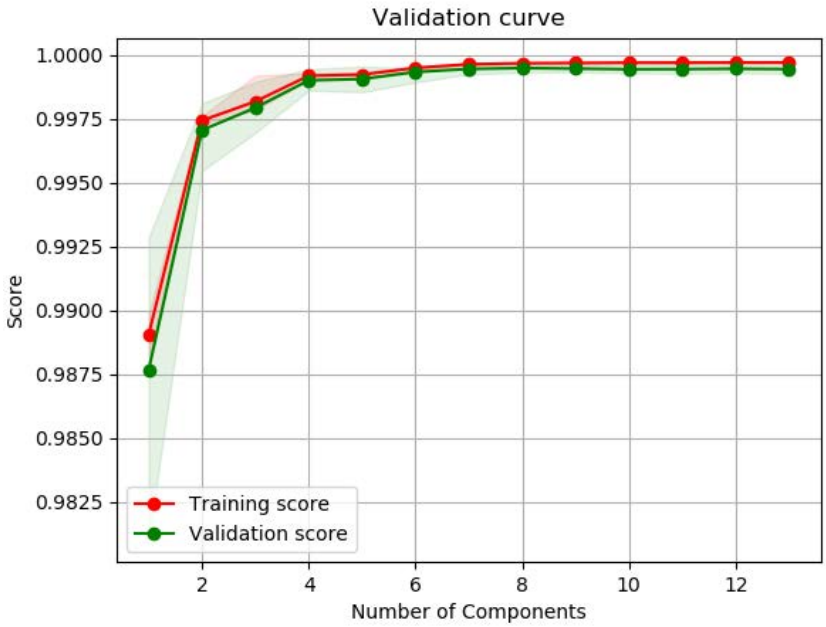


Figure C.1. *Validation curve of the PLS regression model developed for the estimation of the benzene split fraction in the first distillation column (with k = 0.025).*

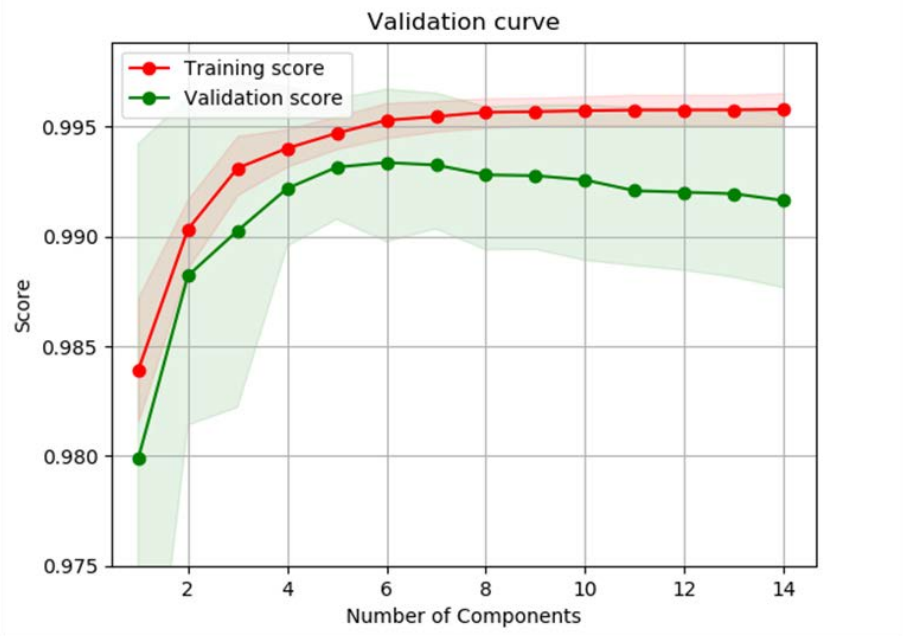


Figure C.2. *Validation curve of the PLS regression model developed for the estimation of the benzene split fraction in the first distillation column (with k = 0.1).*

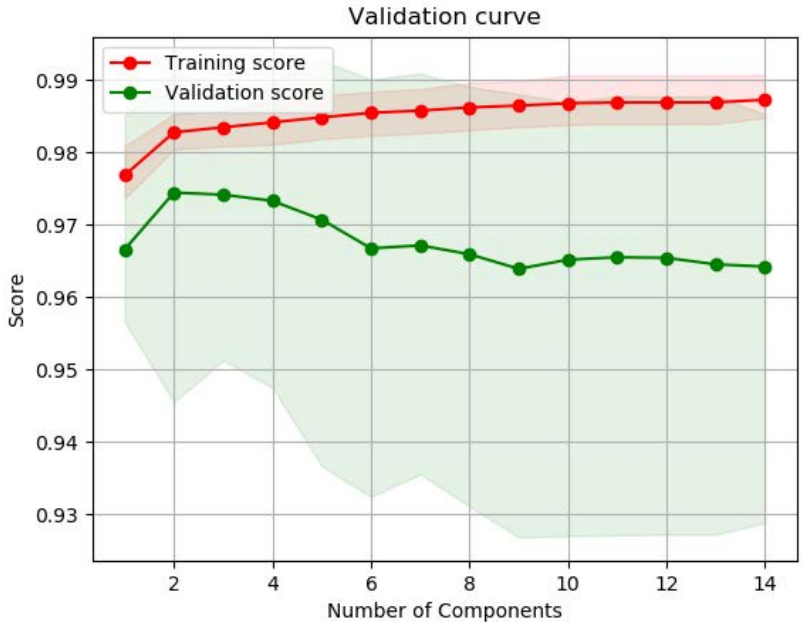


Figure C.3. *Validation curve of the PLS regression model developed for the estimation of the benzene split fraction in the first distillation column (with k = 0.25).*

As can be seen in Table C.1, C.2 and C.3, as the amplitude of the measurement noise of the training dataset increases, the $RMSECV$ increases and the predictive relevance Q^2 decreases. Therefore, as expected, the data-driven elements performances degrade with the increase of the noise in the training data. As can be noticed in Figure C.1, C.2 and C.3, moreover, the increase of the amplitude of the measurement noise leads to validation curves that decrease with a greater slope when the number of PCs exceed the optimal one. Hence, the noisier the training dataset is, the more the black-box model will be prone to overfit it.

References

Chen L., Y. Hontoir, D. Huang, J. Zhang and A. J. Morris (2004). Combining first principles with black-box techniques for reaction systems. *Control Engineering Practice*, **12**, 819-826.

Cao S. and R. Rhinehart (1995). An efficient method for on-line identification of steady-state. *Journal of Process Control*, **5**, 363-374.

Cozad A., N.V. Sahinidis and D.C. Miller (2014). Learning surrogate models for simulation-based optimization. *AIChE Journal*, **60**, 2211–2227.

Cozad A., N.V. Sahinidis and D.C. Miller (2015). A combined first-principles and data-driven approach to model building. *Computers and Chemical Engineering*, **73**, 116-127.

Dempf D. and T. List (1998). ON-Line Data Reconciliation in Chemical Plants. *Computers & Chemical Engineering*, **22**, 1023-1025.

De Prada C., D. Hose, G. Gutierrez and J.L. Pitarch (2018). Developing grey-box dynamic process models. *IFAC PapersOnLine*, **51**, 523-528.

Engell S. and K. Dadhe (2001). Neural Networks for Modelling and Control of Reactive Distillation. *IFAC Proceedings Volumes*, **34**, 354-359.

Facco P. (2009). Development of multivariate statistical techniques for quality monitoring in the batch manufacturing of high value added products. *Ph.D. Thesis*, Università degli Studi di Padova.

Hurvich C. M. and C. L. Tsai (1993). A corrected Akaike information criterion for vector autoregressive model selection. *Journal of Time Series Analysis*, **14**, 271-279.

Jackson J. E. (1991). *A user's guide to principal components*. John Wiley & Sons Inc., New York (USA).

Jiang T., B. Chen, X. He and P. Stuart (2003). Application of steady-state detection method based on wavelet transform. *Computers & Chemical Engineering*, **27**, 569-578.

Kelly J. D. and J. D. Hedengren (2013). A steady-state detection (SSD) algorithm to detect non-stationary drifts in processes. *Journal of Process Control*, **23**, 326-331.

Liu J., S. Chen, X. Nie and M. O. Robbins (2007). A continuum–atomistic simulation of heat transfer in micro- and nano-flows. *Journal of Computational Physics*, **227**, 279-291.

Luyben L. W. (2010). Design and Control of the Cumene Process. *Industrial & Engineering Chemistry Research*, **49**, 719-734.

Muñoz S. G. (2019). Introduction to Process Analytics using Multivariate Methods. *Course material*, Imperial College London.

Mjalli F. S. and A. Al-Mfargi (2009). Neural network-based heat and mass transfer coefficients for the hybrid modeling of fluidized reactors. *Chemical Engineering Communications*, **197**, 318-342.

Pedregosa F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, **12**, 2825-2830.

Pitarch J. L., C. G. Palacín, C. De Prada, B. Voglauer and G. Seyfriedsberger (2017). Optimisation of the resource efficiency in an industrial evaporation system. *Journal of Process Control*, **56**, 1-12.

Prabhakar P. and A. Kumar (2014). Performance evaluation of voltage stability index to assess steady state voltage collapse. *6th IEEE Power India International Conference*, 1-6.

Safavi A. A., A. Nooraii and J. A. Romagnoli (1999). A hybrid model formulation for a distillation column and the on-line optimisation study. *Journal of Process Control*, **9**, 125-134.

Wang Y., K. Sun, X. Yuan, Y. Cao, L. Li and H. N. Koivo (2018). A Novel Sliding Window PCA-IPF Based Steady-State Detection Framework and Its Industrial Application. *IEEE Access*, **6**, 20995-21004.

Wilson Z. T. and N.V. Sahinidis (2017). The ALAMO approach to machine learning. *Computers & Chemical Engineering*, **106**, 785-795.

Wise B. M. and N. B. Gallagher (1996). The process chemometrics approach to process monitoring and fault detection. *Journal of Process Control*, **6**, 329-348.

Wold S., M. Sjöström and L. Eriksson (2001). PLS-regression: a basic tool of chemometrics. *Chemometrics and Intelligent Laboratory Systems*, **58**, 109-130.

Xiong Q. and A. Jutan (2002). Grey-box modelling and control of chemical processes. *Chemical Engineering Science*, **57**, 1027-1039.

Zendehboudi S., N. Rezaei and A. Lohi (2018). Applications of hybrid models in chemical, petroleum, and energy systems: A systematic review. *Applied Energy*, **228**, 2539-2566.

`https://pubchem.ncbi.nlm.nih.gov/compound/Cumene` (last access: 10/10/2019).

`https://www.python.org` (last access: 18/10/2019).

`https://www.psenterprise.com` (last access: 20/10/2019).

Acknowledgments

First of all, I would like to thank Professor Costas Pantelides and Professor Fabrizio Bezzo for giving me the opportunity to work for six months at Process Systems Enterprise (PSE). Working in such a collaborative and professional environment was a truly enriching experience. Thanks to my PSE Supervisor, Dr. Maarten Nauta, for his careful guidance. It was a pleasure working with you during my internship.

Thanks to all the new friends I met during my stay in London: Mariana, Isabel, Gonçalo, Lylia, Mauro, João, Cristian and Luca.
Thanks to Filippo for his warm welcome in London.

Finally, I am grateful to Professor Fabrizio Bezzo for his priceless advices and his invaluable support not only during the master thesis project, but throughout my entire academic career.

Mamma Nicoletta e papà Franco, grazie per essere sempre al mio fianco. Sono diventato la persona che sono grazie a voi. Mamma, grazie perché con il tuo esempio mi insegni ad essere forte tutti i giorni.
Grazie a tutta la mia famiglia: nonni, zii e cugini. In particolare, grazie al nonno Giuseppe, sei la mia più grande fonte di ispirazione.

Grazie al Prof. Fabrizio Bezzo per la disponibilità, la professionalità e il supporto durante l'intero arco della mia carriera universitaria, dalla stesura della tesi triennale alla realizzazione del progetto di tesi magistrale, passando per il periodo di studio all'estero.

Grazie all'Ing. Sergio Peron per avermi seguito con dedizione e interesse nella mia prima esperienza lavorativa, durante la stesura della tesi triennale.

Grazie a Serena ed Alberto per essermi vicino sempre, nonostante tutto.

Grazie agli amici dell'Erasmus: Mia, Linda e Davide. Mi avete fatto sentire a casa anche in una città sperduta della Germania dell'est.

Grazie ai miei compagni di corso. In particolare, grazie a Federico, Luca e Riccardo, avete reso spensierati questi anni di studio matto e disperatissimo.