TÉCNICO LISBOA

Hybrid Modelling/ Machine learning for soft-sensing and process modelling

Alexandre José Gomes da Silva

Thesis to obtain the Master of Science Degree in

Chemical Engineering

Supervisors: Professor Doctor Carla Isabel Costa Pinheiro
Doctor Maarten Nauta

Examination Committee

Chairperson: Professor Doctor Carlos Manuel Faria de Barros Henriques
Supervisor: Professor Doctor Carla Isabel Costa Pinheiro
Members of the Committee: Professor Doctor João Almeida Lopes

November 2018

Acknowledgments

Firstly, I would like to thank Prof. Dr. Carla Pinheiro and Prof. Dr. Costas Pantelides for granting me the opportunity to partake in a 6 months internship at Process Systems Enterprise (PSE). It was a truly enriching experience, both professionally and personally.

To Dr. Maarten Nauta and Prof. Dr. Carla Pinheiro, my supervisors, I express my gratitude for all the guidance and help provided, whenever possible. A special thanks goes to Eng. Renato and Eng. João for sharing with me their gPROMS knowledge and for all the great advice regarding my thesis.

To my housemates: Diogo, Mauro and Patrícia I express my eternal gratitude, they were my family in London and without them I'm sure this experience wouldn't have been as great! A special thanks goes to André, Artur, Renato and Tom for housing three complete strangers and for sharing with us their London expertise. To all the new and old friends in London: Mariana, Francisco, Renato, Artur, João, Lylia, Christian, Beatriz and Barras goes my greatest appreciation. Without them this experience wouldn't have been the same. To all the guests we had along the way a big thank you for making us feel a little less homesick.

Finally, I would like to show my deepest gratitude to my mother Manuela, my father José and my sister Catarina. Without them I wouldn't be the person I am today, all I have accomplished I owe to them.

Abstract

It is common to have systems within processes that are poorly understood and cannot be easily represented by first-principle models. A prototype tool was developed, in Python, aiming to soft-sense parameters belonging to these systems. This tool allows for the creation of predictive models using input and output data. These models were subsequently integrated in dynamic simulations in gPROMS®.

For this purpose, a partial least squares regression (PLSR) algorithm was implemented. This algorithm has the particularity of needing the models dimensions to be chosen prior to creating the model, therefore a 10-fold cross-validation (CV) method was implemented along with other tools to help judge the model's quality. The Distance to the model and Hotelling's T^2 statistical tests were also implemented for outlier detection.

In order to study the models behaviour when integrated in a dynamic simulation, data from two different cases studies, a granulation and a compression case, was used. Furthermore, data from a solid oxide fuel cell case was used to study the creation of a model capable of predicting several response variables. These cases led to the implementation of a 2^{nd} order polynomial transformation and a reciprocal transformation.

The simulations showed the high importance of choosing training data that covers the operative conditions of the simulation. Otherwise the results obtained will be unreliable since the model is being used to predict data outside the range it was meant to be used in.

Keywords: Hybrid Modelling, Partial Least Squares Regression, Soft-sensing, Distance to the model, Hotelling's T^2

Resumo

É comum a existência de sistemas cujas relações entre variáveis não são bem conhecidas tornando-se difícil a sua representação com modelos de primeiros princípios. Com o objetivo de fazer *soft-sensing* a parâmetros pertencentes a estes sistemas, foi desenvolvido o protótipo de uma ferramenta, em *Python*, que permite a criação de modelos usando dados de entrada e saída. Os modelos criados foram integrados em simulações dinâmicas usando o *software gPROMS*®.

Com esse propósito, foi implementada uma regressão de mínimos quadrados parciais. Visto este algoritmo necessitar que as dimensões do modelo sejam escolhidas, foi implementada uma *10 fold cross validation* juntamente com outras ferramentas que ajudam a averiguar a qualidade do modelo. Os testes estatísticos *Distance to the model* e *Hotelling's* T^2 foram implementados para detecção de *outliers*.

De modo a estudar como os modelos se comportam quando integrados em simulações dinâmicas, foram usados dados de um caso de granulação e de um de compressibilidade. Adicionalmente, dados de um caso de *solid oxide fuel cell* foram usados para estudar a criação de um modelo para previsão várias variáveis de resposta. Estes casos levaram à implementação de uma tranformação polinomial de 2^o grau e de uma transformação recíproca.

As simulações acentuaram a importância de escolher dados para a criação do modelo que abranjam as condições operatórias da simulação. Caso contrário os resultados não serão de confiança, uma vez que o modelo está a ser usado para prever valores fora do alcance para o qual foi concebido.

Keywords: Modelação Híbrida, Regressão de mínimos quadrados parciais, Soft-sensing, *Distance to the model*, Hotelling's T^2

Contents

List of Tables

List of Figures

Nomenclature

Acronyms

ADDoPT	Advanced Digital Design of Pharmaceutical Therapeutics
AZ	AstraZeneca
CV	Cross-validation
D_{32}	Sauter mean diameter
D_{43}	Volume weighted mean
DmodX	Distance to the model of an observation
dof	Degrees of freedom
FO	Foreign object
GSA	Global System Analysis
LOOCV	Leave one out cross-validation
LV	Latent variables
MLR	Multiple linear regression
MSE CV	Mean squared error of cross-validation
NIPALS	Non-linear iterative partial least squares
PLS	Partial least squares
PLSR	Partial least squares regression
PRESS	Prediction error sum of squares
PSE	Process Systems Enterprise
Q^2	Goodness of prediction
R^2	Goodness of fit
SOFC	Solid oxide fuel cell
std	Standard deviation
TSS	Total sum of squares
VIP	Variable influence on projection

Symbols

ν	Distance to model correction factor
A	Number of components of the PLS model
f_i	Volume fraction
K	Number of X variables
N	Number of samples in a dataset
n_i	Number of particles

Chapter 1

Introduction

In the process industry, a large number of sensors are commonly implemented with the purpose of delivering data for process monitoring and control. Approximately two decades ago, researchers started to use the large amounts of data being measured and stored by them to build predictive models. These models are referred to as soft-sensors.[1]

Two main classes of soft-sensors exist: model-driven and data-driven. Model-driven soft-sensors, also called white-box models, are based on first-principle models and therefore have full phenomenological knowledge regarding the process background. Data-driven soft-sensors, also called black-box models, are based on the data measured from a process and therefore the model itself has no knowledge of the process. Given that model-driven sensors are primarily used for planning and development of process plants and inferential control, the focus of soft-sensing is usually concentrated in data-driven models.[1]

As a consequence of being based on data measured directly from the process, data-driven sensors describe the real process conditions. Their usefulness is derived from the fact that they can build multivariate features related to different process variables that have a direct influence in the process's output quality. Several modelling approaches are used for these models, being the PLSR one of the most popular in chemical engineering and chemometrics, among others.[1]

The inclusion of a data-driven soft sensor in dynamic simulations implies its use alongside first-principle models. This combination of white and black box models constitutes another type of modelling strategy: grey-box modelling, also referred to as hybrid modelling (Figure 1.1).[2]

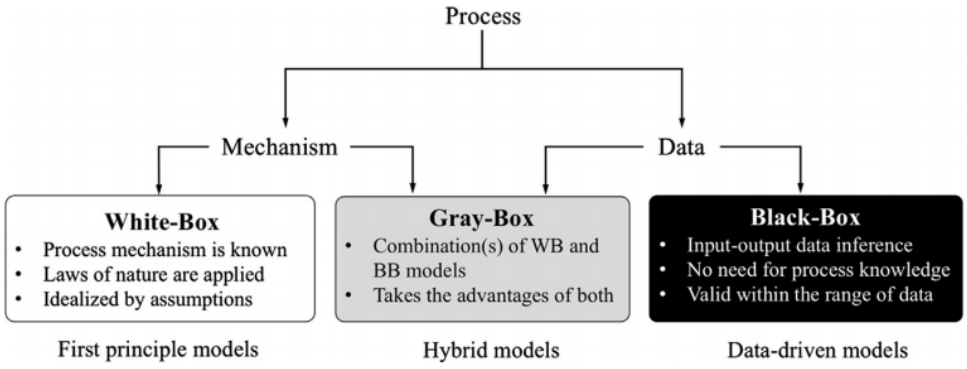


Figure 1.1: Different types of modelling strategies.[2]

These types of models are specially appealing when a process is non-linear. When that is the case, hybrid modelling combines white and black box modelling, compensating for the shortcomings of the standalone implementation of these strategies. As a consequence, hybrid modes offers a higher degree

of flexibility, enabling them to accomplish the modelling task based on precision, reliability and relevancy of the process knowledge and process data.[2]

1.1 Motivation

Despite many technological advancements in the field, there are still many chemical processes that include poorly understood systems, making it hard, or even impossible, to model them tough first-principle models. This leads to approximations being made during simulations of the process, culminating in final results that stray from reality.

In this regard, there is the need to include said systems in simulations, in order to get the best prediction for the outcome of the final product of a process. This is specially true in sectors where small variations in some parameters might have a significant impact on the overall quality of the process.

1.2 Scope

The present work aimed to develop a tool, in Python, able to create predictive models from process data. These models were then included in dynamic simulations for soft-sensing purposes. For the models' creation the PLSR was used.

There was a focus on model quality, accomplished by implementing several tools and methods that evaluate the predictive capabilities of the models, allowing the achievement of the best model possible. These methods were validated trough the comparison of the tool's results with the ones obtained from SIMCA® with the same data.

Model validity was another focus of this work. Its implementation consisted on assessing if the model is used in the data range it was meant to be used in. This is done trough statistical tests, namely the distance to the model of an observation (DmodX) and Hotelling's T^2, which were also validated trough comparison with SIMCA's results.

1.3 Outline

This thesis is structured as follows: Chapter 2 presents a literature review regarding the PLSR, explaining the different intricacies of the method and also the statistical tests applied to training data. Chapter 3 is composed by the descriptions of the several software's and packages used throughout this work.

Chapter 4 consists on the validation of both the model's creation, more specifically the creation of a model for the prediction of the FFC, and the statistical tests trough the comparison with the results obtained from SIMCA. Additionally, the model created will be implemented in a dynamic simulation and its validity will be assessed. Chapter 5 describes the results obtained for both the creation of a model that predicts the compressibility of a blend in a roller compactor and its implementation in a dynamic simulation. In Chapter 6 the capability of creating a model that predicts several outputs is tested.

Finally, Chapter 7 presents the all the conclusion obtained from the work developed in the present thesis, along with some considerations regarding future work.

Chapter 2

Background

Machine learning has been introduced steadily into the process industry in recent years, composing the main type of approach used for modelling soft-sensors. Even tough many different machine algorithms are available, for this work, the partial least squares regression (PLSR) machine learning algorithm was deemed the most suitable.[1]

2.1 Machine Learning in industrial processes

Machine learning refers to a set of tools for understanding data, more specifically, it is an algorithm based method that allows the estimation of an unknown dependency between a systems inputs and outputs from the available data. Four types of machine learning algorithm exist: supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning. However, in the process industry, most problems call for either supervised or unsupervised learning. Furthermore, most problems in this industry are regression problems i.e. problems with a quantitative response.[3][4][5]

Supervised learning consists on fitting a model that relates the output variables (response variables) to the input variables (predictor variables), using datasets with responses associated to each observation of the predictors. It either aims to accurately predict the response for future observations (prediction) or to better understand the relationship between the inputs and outputs (inference). On the other hand, it is not possible to fit a regression model in unsupervised learning because there is no response variable to predict. In this type of machine learning, for every observation in the dataset there is vector of predictors but no response variables. This work will focus on supervised learning for developing soft-sensors that aim to predict future responses.[3]

Machine learning is widely used in different areas of the process industry. It has been successfully applied in process optimization, monitoring and control in manufacturing. It has helped with the increasing complexity and high dimensionality of the manufacturing systems. This tendency leads to the prediction of an expanded use of machine learning in this field, specially trough hybrid approaches[6]. Toxicity prediction is another field were machine learning has been introduced, consequence of the significant challenge this prediction poses in environmental and drug development areas. This application consists on in vitro bioassay data being used to predict in vivo chemical toxicology[4]. For sensor fault detection, machine learning is also applied. It is used for the identification of the sensor's or process's faults effectively, also allowing to find which particular sensor or set of sensor is responsible for the fault[1]. Machine learning has also been used for the estimation of quality measurements, trough soft-sensors, that would usually be determined trough infrequent sampling or off-line analysis. Product composition estimation in distillation columns and particle size distributions in a grinding circuit are examples of such[7].

2.2 Partial Least Squares Regression

In multivariate problems it is common to use the widely understood multiple linear regression (MLR). This type of regression is applicable when there are few input variables, they are not significantly collinear and there is some understanding of how they relate with the output variable. However, in most of the problems tackled through machine learning, in the field of engineering, it is uncommon to comply with all these conditions, in fact, it is highly common to come across a problem with a high number of input variables, which present a higher degree of collinearity. This poses a problem when trying to create a model with this type of regression, therefore, in problems such as these, MLR becomes inefficient and thus the need to use another type of regression arises. The PLSR presents a useful alternative, especially when there is the need to obtain a predictive model for the output variable(s). This type of regression puts emphasis on getting a model that focuses on predicting responses and not necessarily on trying to understand the relation between variables.[8] [9]

Like any other type of regression, PLSR will relate the input variables - x - with the output variables - y - and it will create a multivariate linear model with predictive capabilities. Unlike others, this regression algorithm has the particularity of being able to tolerate moderate amounts of missing data in both the X and Y matrices, since the algorithm has the particularity of being able to iteratively substitute these values with predictions made by the model. The larger the matrices, the higher the proportion of missing data tolerated, provided that the data is not missing according to a systematic pattern, although it should be noted that, in cases with a high number of missing data, the models parameters and predictions will present a high degree of uncertainty.[9]

When modelling through PLSR, it is assumed that the process in question is influenced by just a few underlying variables called latent variables (LV). Usually, it is not known how many of them there are, being one of the aims of PLSR analysis to estimate their number. The X and Y variables are hypothesized to be realizations of these latent variables and, therefore, are not assumed to be independent. The LV assumptions approximately correspond to the application of microscopic concepts, e.g. molecules and reactions in chemistry and biology, making PLSR suitable for the modelling of chemical data. Whenever the number of LV's equals the number of input variables, the latter is independent and thus PLSR yields the same results as MLR. Consequently, we can interpret PLSR as a generalization of MLR, being that the first contains the other as a special case whenever the number of predictor and response variables is fairly small compared to the number of samples i.e when MLR has a solution.[9]

2.2.1 Partial Least Squares Regression Algorithm - PLS1 and PLS2

Even though there are several types of PLSR algorithms , only the PLS1 and PLS2 algorithms are implemented in the scikit-learn package. They differ from each other in the sense that PLS1 is particular case of PLS2, applied when there is only one response variable. Algorithm wise, the only differences between the two are the mathematical consequences on the several matrices that relate to Y derived from having only one response variable.

The algorithm starts with the treatment of the input data matrices, – X of dimensions (N,K) and Y of dimensions (N,M) – in order to make their distributions be fairly symmetrical. They are centred, through the subtraction of the mean of each variable to their respective column in the input matrix and, optionally, they can also be scaled by diving each variable by its standard deviation. When there are variables with a range of more than one magnitude of 10, a logarithmic transformation may be applied. Once the treatment has been applied, an iterative loop begins, the outer loop, whose first step begins with another loop, the inner loop.

In the inner loop the weight matrices W of dimensions (K, A) and C of dimensions (M, A) for X and Y

respectively are calculated. Said calculation is accomplished trough equations (2.1) and (2.2). In both equations, it is possible to observe that $X^{(r)}$ and $Y^{(r)}$ are used. This happens because they represent the residual matrix of the iteration number r of the outer loop, except for the first iteration, where they represent the treated input data matrices. The subscript i represents the number of the iteration of the inner loop.[9][10]

$$W_i = \frac{X^{(r)T} U_{i-1}}{U_{i-1}^T U_{i-1}} \qquad (U = Y^{(r)} \quad for \quad i = 1) \tag{2.1}$$

$$C_i = \frac{Y^{(r)T} T_{i-1}}{T_{i-1}^T T_{i-1}} \qquad (T = X^{(r)} \quad for \quad i = 1) \tag{2.2}$$

Upon being calculated, the X weight matrix (W) is normalized trough equation (2.3).

$$W_i = \frac{W_i}{\sqrt{W_i^T W_i}} \tag{2.3}$$

After normalization, the X-score – T (N, A) – and Y-score – U (M, A) – matrices will be calculated trough equations (2.4) and (2.5). These matrices will be used to calculate the weight matrices of the next iteration within the inner loop.

$$T_i = X^{(r)} W_i \tag{2.4}$$

$$U_i = Y^{(r)} C_i \tag{2.5}$$

At the end of each iteration the constrain in (2.6) is tested. The inner loop will run as long as this condition is not met.[10]

$$(W_i - W_{i-1})^T (W_i - W_{i-1}) < 10^{-6} \tag{2.6}$$

Once the inner loop is finished, the scores will once again be calculated trough equations (2.4) and (2.5), using the weights obtained at the end of the inner loop. The X-scores span in the same space as the LV, albeit they are usually not direct estimates of the LV. These matrices are of particular importance since they act as estimators of X and Y. With them, it is also possible to calculate the loading matrices: P (K, A) for X and Q (N, A) for Y.[11][9]

$$P = \frac{X^{(r)T} T}{T^T T} \tag{2.7}$$

$$Q = \frac{T^{(r)T} U}{U^{(r)} U} \tag{2.8}$$

At this point, it is possible to make estimations of the X and Y data through the calculated matrices,

as shown in equations (2.9) and (2.10).

$$\hat{X} = TP^T \tag{2.9}$$

$$\hat{Y} = UQ^T \tag{2.10}$$

This estimations will be subtracted to the X and Y matrices, in order to get the residual matrix for the next iteration, after which the algorithm will go back to the inner loop step. There will be A iterations, being A the number of components of the model i.e. the number of latent variables (model dimensions). Once the algorithm is finished, it is possible to get predictions of Y through X, since the X-scores have the property of being predictors of Y. This will be done through equation number (2.12), where W* represents a transformed weight matrix and a centred and scaled (possibly) response matrix is obtained - Y*.[9]

$$W^* = \frac{W}{P^T W} \tag{2.11}$$

$$Y^* = XW^*C^T \Leftrightarrow Y^* = XB^* \tag{2.12}$$

Through the equation shown above, it is possible to get the coefficient matrix B*, however, it cannot be applied directly to an X matrix without previous treatment of the latter. Therefore, this matrix needs to be treated, using the standard deviation (std) of both the input and output matrices, in order to be used with an untreated X matrix.

$$B = B^* \frac{STD(Y)}{STD(X)} \tag{2.13}$$

Once treated, prediction of new values will be accomplished though equation (2.14). It is worth noting that, in the equation displayed bellow, both means refer to the mean of the training matrices.

$$Y = XB + (Y_{mean} - X_{mean}B) \tag{2.14}$$

PLS2 has the disadvantage that, by modelling all the response variables at the same time, the number of components of the model will be the best compromise between the optimal value for each variable as obtained by cross-validation (CV). As such, one can apply PLS1 for each of the response variables in order to get an optimal model for said variable. Moreover, PLS1 presents the advantage of being able to eliminate some of the effects of interactions between response variables, which, in some cases, is a cause of nonlinearity.[12]

Regarding the other partial least squares (PLS) algorithms, PLS1 and PLS2 differ from them in the step referring to the update of the residual matrix after each iteration e.g the PLS-SVD algorithm updates the cross-product matrix X^TY instead. The advantage of this method of updating the residual matrix is more evident in cases with just one response variable since, when using other algorithms, it is easier to get a null residual matrix, terminating the algorithm before the loop has gone through A iterations. [11]

2.2.2 Choosing the Model's Dimensions

As mentioned above, the model's dimensions will determine the number of iterations, therefore it is important to determine this parameter correctly, avoiding overfitting i.e. getting a well fitted model with very little to none predictive power. This will be done by applying CV to models with different numbers of components and then choosing the one with the higher score.[9] [3]

Cross-validation consists on splitting the data into validation and training data. Then, the model is fitted to the training data and predictions are made with the validation data. The predictions are then compared with the actual values and the overall prediction error sum of squares (PRESS) (equation (2.15)) is calculated, estimating the predictive capacity of the model. The higher the prediction capability of the model, the lower the PRESS will be.

$$PRESS = \sum_{i}^{N} (Y_{predicted} - Y_{measured})^2 \tag{2.15}$$

Although this parameter is a good indicator of the overall prediction capability of the model, in this work, at the end of the cross-validation process, the mean squared error of cross-validation (MSE CV) - the PRESS score divided by the number of samples - will be the score displayed, making it easier to compare models with different number of samples in the input data matrix. Additionally, the goodness of prediction (Q^2) - equation (2.16) - will also be computed and displayed. This scoring method yields a score between 0 and 1 (it is possible for the score to be negative, however said case will correspond to a model which cannot predict any data correctly), that always presents itself as being equal or lower than thegoodness of fit (R^2), being that a model is good at predicting whenever it scores above 0.5. Even though the PRESS remains the decisive criteria for choosing the number of components, this scoring method allows for a better and easier interpretation of the model's quality, especially in multiple output cases, since the order of magnitude of the outputs will vary, turning the MSE CV into a score that only conveys if a model is better at predicting than others and not necessarily if it is a good predictive model.[13]

$$Q^2 = 1 - \frac{PRESS}{\sum_i^N (y_i - \bar{y})^2} = 1 - \frac{PRESS}{TSS} \tag{2.16}$$

As a result of using centred and scaled matrices when applying cross-validation, the calculation of total sum of squares (TSS) will be simplified, since this parameter will correspond to the number of samples in the matrix.

The two most widely used types of cross validation are the k-fold cross validation and leave one out cross-validation (LOOCV). In k-fold cross validation, the data is randomly divided into k groups (folds) of approximately equal size. The model is then fitted on k-1 groups and predictions are made using the remaining group, which will act as a validation set (Figure 2.1). A score is calculated and the procedure will be repeated k times, so that each group will act as a validation set once.[3]

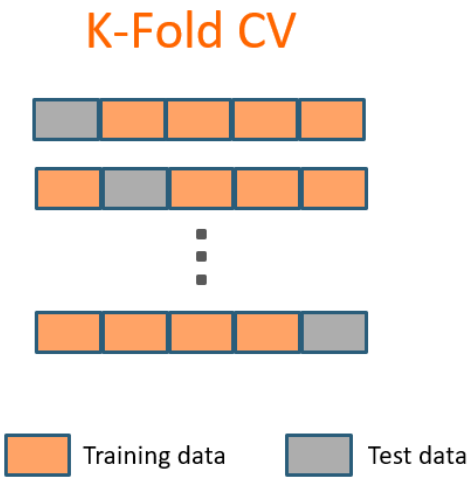


Figure 2.1: Schematic representation of a K-fold CV.

In LOOCV, the procedure is similar to the one in k-fold CV, however, in this case, k will be equal to the number of samples, so that the validation set consists of a single sample and each sample acts as a validation group once. At first glance, this type of CV poses itself as the most thorough and therefore, the one to yield the best results. However, according to Wold *et all*[9] this method is not recommendable when using PLSR. This happens because this type of cross-validation is asymptotically incorrect, which means that this type of cross-validation tends to choose the model of excessive size instead of the optimal model, unless, of course, the size of the optimal model is equal to the number of predictor variables.[3] [14]

In order to choose the number of components that yields the model with the most predictive power, a 10-fold CV will be applied to several models, each of them fitted using a different number of components. This parameter must be an integer number greater or equal to one and lower or equal to the overall number of x variables. Once CV has been applied to the different sized models, the MSE CV scores will be compared and the one that has the lowest one will be the chosen model.

2.2.3 Interpretation of the PLSR model

By making estimates for the latent variables , the PLSR creates ”new x variables” as linear combinations of the inputs, using them as predictors for Y. These new variables, i.e the scores, contain information regarding the x and y variables and how they compare with the given model. The interpretation of these scores is provided by the weigh matrices, that give information about the way the variables combine to form a quantitative relation between X and Y. Therefore, the weights present themselves essential to understand which x variables are of importance, quality indicated by large values in the W matrix (w_{Ka}, and which provide the same information, fact translated in similar profiles in the values of a certain column of W (w_a). The weights also express the direct correlations between X and Y and the ”compensation relations” necessary to predict Y from X without any secondary variation in X i.e. everything varying in X that is not directly related to Y. [9]

Geometric interpretation

PLSR, often referred to as projection on latent structure, is a projection method and, therefore has a simple geometric interpretation. In order to grasp the way the statistical tests work, there is a need to understand what happens geometrically in a PLSR. This explanation will be accompanied by Figures relative to an illustrative example, where there are 3 input variables, 3 outputs and the model has 2 components, in order to make the visualisation of the procedure easier. Its observations, after centring and scaling, are presented in Figure 2.2. [9]

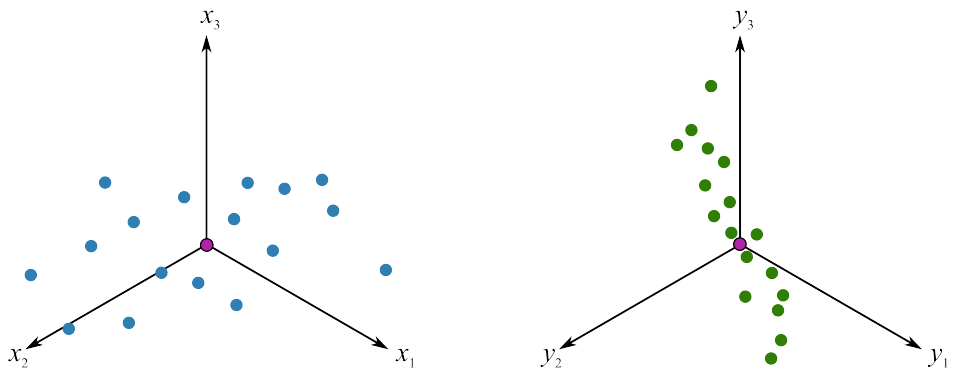


Figure 2.2: Example's input and output data after centring and scaling.[15]

Once the weight matrix is calculated, its column vectors act as direction vectors and each observation is perpendicularly projected onto said direction (Figure 2.3). The point were the projections lands is the X-space (or Y-space) score, t_a (or u_a), which is found so there is a maximization of the covariance between t_a and u_a. What this means is that the directions of the latent variables are oriented so that they best explain X and Y, having the greatest possible relation between the two.[15]

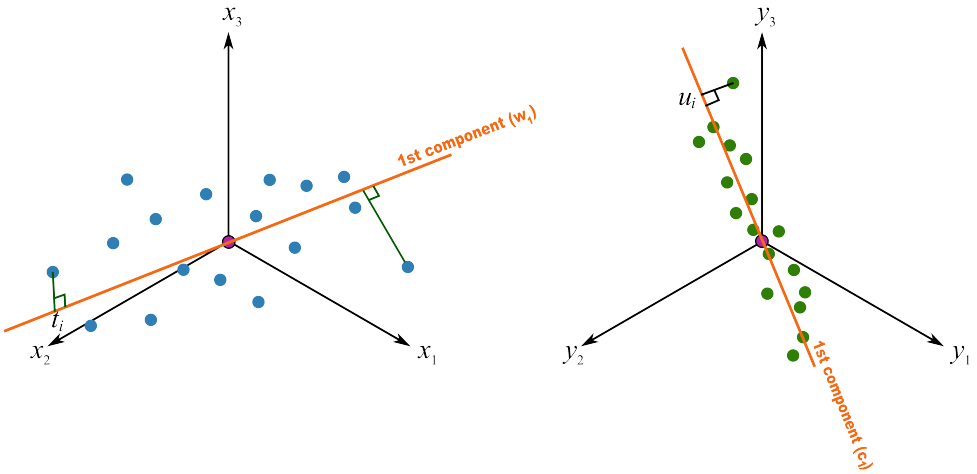


Figure 2.3: Perpendicular projections of the observations according to the weight matrix column vectors.[15]

The second component of the model will be orthogonal to the first component in the X space, not being necessarily orthogonal in the Y space, even though it is often close to orthogonal. As it can be seen in Figure 2.4, once all the model's components have been discovered, the projections of the observations in the directions of the weight vectors will create an A-dimensional hyper-plane. [9][15]

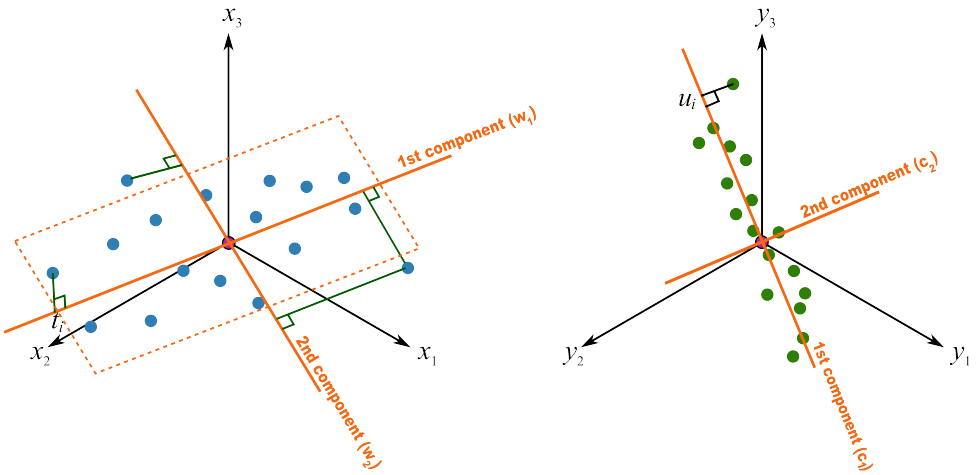


Figure 2.4: Creation of a plane by the model's components[15]

2.2.4 Statistics

The point of the model, as previously discussed, is to predict new values of Y. Therefore there is a need to apply statistical tests on the input data, so that some information regarding the applicability of the model, relative to the input data, can be obtained. In this work, two different tests will be applied: the distance to the model of an observation (DmodX) and the Hotelling's T^2.

The distance to the model test is a moderate outlier detection tool. It calculates the distance of a sample point to the model plane through equation (2.17). If the distance to the model is being calculated for a sample point of the training data, equation (2.17) should be multiplied by a distance to model correction factor (ν), obtained trough equation (2.18). This factor takes into account the fact that the DmodX is expected to be smaller when calculated for an observation that is part of the training data.

$$DmodX = \sqrt{\frac{\sum_k e_{ik}}{K-A}} \div \sqrt{\frac{\sum_i \sum_k e_{ik}}{(N-A-A_0)(K-A)}} \tag{2.17}$$

$$\nu = \frac{N}{N-A_0-A} \tag{2.18}$$

As it will be discussed further in this work, the equation for the correction factor (equation (2.18)) presented in the SIMCA® 15 User guide [16] does not yield satisfactory results and therefore was considered to be wrong. As such, for reasons explained in chapter 4.3, equation (2.19) was used for this parameter.

$$\nu = \frac{N}{N-0.542 \cdot A+0.487} \tag{2.19}$$

As shown above, the equation for DmodX has 2 parameters, being the first one the absolute distance to model and the second one a normalization parameter. They are dependent on the number of X variables (K), number of samples in a dataset (N) and number of components of the PLS model (A). The A_0 parameter takes a value of 1 if the matrix has been centred, and 0 otherwise. The e_{ik} parameter represents the residual for the variable k in the i^{th} sample. It is calculated through the subtraction of the data matrix being evaluated (after centring and/or scaling) with the estimation made with de X-score and

loading matrices.

$$\begin{bmatrix} e_{11} & \cdots & e_{1k} \\ \vdots & \ddots & \vdots \\ e_{i1} & \cdots & e_{ik} \end{bmatrix} = X - TP^T \tag{2.20}$$

When calculating the distance to the model of a prediction set there is the need to calculate a new X-score matrix, since this is the only non-model specific matrix (equation (2.21)). This can be easily achieved through the rotation matrix, W* (equation (2.11)) .[9]

$$T_{prediction} = X_{prediction}\frac{W}{P^T W} \Leftrightarrow T_{prediction} = X_{prediction}W^* \tag{2.21}$$

For each sample, the distance to the model will be calculated and compared with the critical distance, being that any sample with a distance higher than the critical one, will represent a possible outlier and a sample with a distance higher than twice the critical distance is considered an outlier. This parameter is represented by the squared root of an inverse cumulative F-distribution function for a significance level of 5%. The number of degrees of freedom of the denominator of said F-distribution amount to the number of degrees of freedom of the model.[16] [17]

$$DOF_{model} = \sqrt{(N - A_o - A)(K - A)} \tag{2.22}$$

The degrees of freedom (dof) of the numerator of the F-distribution in question match with the number of degrees of freedom of the observations in the training set. The way to calculate them will be dependent on the value of the dof of the model, as such:

$$DOF_{observations} = \frac{M + \sqrt{K - DOF_{model}} - A}{\nu} \qquad K > DOF_{model} \tag{2.23}$$

$$DOF_{observation} = \frac{M - A}{\nu} \qquad K < DOF_{model} \tag{2.24}$$

When calculating degrees of freedom, the M parameter represents the minimum value between K, the DOF_{model} and 100.[16]

Although a data point may be close to the model plane, it can still be far away from the centre of the model, where it has more significance. Therefore, the distance to the centre of the model of the projection of an observation in the model plane will be calculated through Hotelling's T^2 range. For each sample, the squared distance of each variable score to its average will be calculated and divided by the respective variance. The distance to the centre of the model of a sample will be the sum of the aforementioned calculation for all its x variables.[16]

$$T_i^2 = \sum_K \frac{(t_{ik} - \bar{t_k})^2}{s_{tk}^2} \tag{2.25}$$

This test is dependent on the X-score value for observation i and number of components k (t_{ik}), as well as the mean ($\bar{t_k}$) and standard deviation (s_{tk}) for the X-score vector with k number of components i.e. the mean and standard deviation of the X-score matrix's column number k.The critical distance for Hotelling's T^2, like the critical DmodX, is calculated through a cumulative F-distribution with a 5% significance level. The dof of the numerator of this distribution match the number of components, while the dof of the denominator correspond to the subtraction between the number of observations and the

model's dimensions. This test allows for the exposure of outliers through the following constrain:

$$T_i^2 > \frac{A(N-1)}{N-A} F_{critical}(p) \tag{2.26}$$

The samples with a T^2 value higher than the critical distance at a significance level of 5% (p = 0.05) represent suspected outliers, while the samples with a T^2 higher than the critical distance at a significance level of 1% represent outliers.[16]

Chapter 3

Software Tools

3.1 Python™language

Python is an interpreted, interactive, object-oriented programming language used for general-purpose programming. It incorporates modules, exceptions, high level dynamic data types and classes, putting emphasis on code readability by using significant whitespace. It is extensible in C or C++, having interfaces to many system calls and libraries.[18]

3.1.1 Scikit-learn package

Scikit-learn is a Python module which integrates a wide range of state-of-the-art machine learning algorithms for supervised and unsupervised problems. It does so while maintaining an user friendly interface integrated with the Python language. It aims to answer the ever increasing need for statistical data analysis by non-specialists in the software and web industries and other fields outside of computer-science. [10]

3.2 gPROMS software®

For the implementation of the models created in dynamic simulations, the gPROMS® platform was used. This modelling software, created by Process Systems Enterprise (PSE), provides equation-oriented modelling and optimisation framework on which the several PSE's gPROMS products operate. It allows for flowsheeting, paired with custom modelling of high-fidelity unit models with both steady-state and dynamic process modelling. [19]

3.2.1 gPROMS FormulatedProducts®

The dynamic simulations executed throughout this work were performed in gPROMS Formulated-Products. This is PSE's platform used for integrated digital design of robust formulated products and their respective manufacturing processes. It warrants the screening of formulations with complex phase structures for quality attributes, the ability to determine whether they can be manufactured or not and the exploring of the design space for the whole formulation and manufacturing chain.[19]

3.2.2 Foreign Objects

gPROMS ® allows the usage of external software components - foreign object (FO) - which provide a way of importing data. These may include physical property packages, external unit operation modules or complete computational fluid dynamics (CFD) software packages. In the present work, a single FO was used, the pyFO, of internal PSE development, responsible for trading of information between the platform and Python. [20]

3.2.3 Global system Analysis

gPROMS has the capability of performing uncertainty analysis with Monte Carlo methods on certain models trough the global system analysis (GSA) tool. The applied method stimulates the model in deterministically or probabilistically ways to asses the uncertainty of the model itself. When using this tool it is important for the user to understand the model properly due to the large impact the applied variations might have on the output variables chosen for consideration. GSA derives its usefulness from the fact that it allows for the clarification of some aspects of the problem. It does so by flagging models used out of context or with a degree of complexity not sustained by the available information. Two types of analysis can be performed with this tool: sensitivity analysis - the study of how the variance of the outputs of the model is dependent on the inputs factors affected by uncertainty - and uncertainty analysis - a Monte Carlo analysis consisting on performing multiple model evaluations with randomly selected model inputs, whose results are then used to determine the uncertainty in the models predictions.[20]

3.3 SIMCA® software

SIMCA is a statistical software used for multivariate data analysis. It allows for the interpretation and visualisation of large amounts of data, batch data and time-series data, among others. By being able to analyse process variations, identify critical parameters and predict final product quality, this software becomes suitable for data mining and process modelling. [21]

Chapter 4

Granulation Case Study

Like with any other model, there is a need for the validation of the partial least squares (PLS) model and, in this case, for the validation of the implemented statistical tests. With this purpose, a PLS model was fitted to data from a granulation case study. This granulation case was previously ran trough the umetric's SIMCA® software, which created a PLS model and tested it with the distance to model and Hotelling's T^2 tests.

4.1 Process description

In the case study in question, data from a granulation process was provided by AstraZeneca (AZ) as a part of the Advanced Digital Design of Pharmaceutical Therapeutics (ADDoPT) project. This project consists of a four year collaboration between pharmaceutical companies, solution providers and academia aiming to make existing and new digital design approaches widely usable within the pharmaceutical industry, increasing efficiency and effectiveness of drug development and manufacture.[22]

The flowsheet referring to the process where the training data for the model was collected (Figure 4.1) was also provided and assembled by AZ. In this dry granulation process, powder particles are fed to a roller compactor where they are compacted into a ribbon and subsequently milled. The process ends with the compacting of the granules from the mill in the tablet press. Linked to the output stream of the mill there is a particle size distribution sensor (PSD_sensor) that will obtain the results of a particle size distribution for said stream, namely the cumulative diameters (D_{10}, D_{50}, D_{90}) and the volume weighted mean (D_{43}) , which will then act as inputs for the model. This is possible since the data based sensor block (Sensor_data_based), a block that contains the developed model, was added to the flowsheet. This block receives the cumulative and volume mean diameters from the particle size distribution and calculates prediction for the flow-function coefficient (FFC).

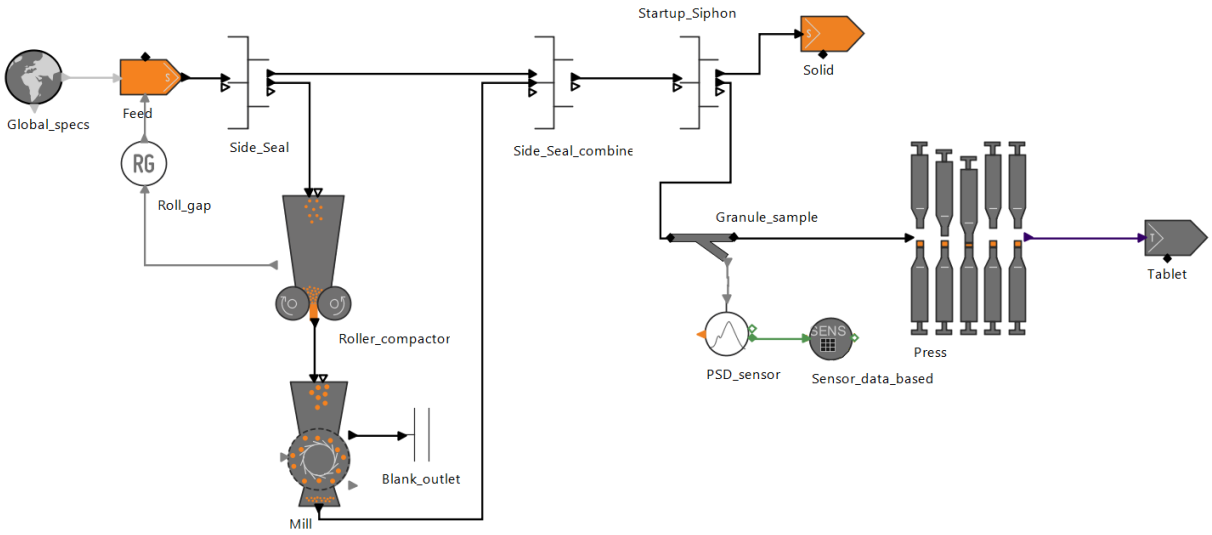


Figure 4.1: AZ's process flowsheet.

4.2 Model Development and Validation

The above mentioned data refers to several particle size distributions and the parameters obtained from it, i.e the cumulative diameters and the D_{43}. The cumulative diameters amount to the diameter at which a certain percentage of the particle sample has a diameter lower or equal than said value, e.g D_{10} is the diameter at which 10% of the sample's mass is comprised of particles of the same or lower diameters. D_{43} is calculated through equation (4.1), which is rearranged in order to get an equation dependent on the volume fraction (f_i) in a sieve of diameter D_i, and not the number of particles (n_i) captured in a sieve.[23]

$$D_{4,3} = \frac{\sum_i n_i D_i^4}{\sum_i n_i D_i^3} \Leftrightarrow D_{4,3} = \sum_i f_i \cdot D_i \tag{4.1}$$

Like stated previously, both the cumulative diameters and D_{43} are used as inputs for the model, in order to calculate the FFC. This coefficient refers to a function which is of importance in granulation, since it allows for the calculation of the flow performance of the powder. All these parameters were calculated for 19 different samples and their values can be consulted in Appendix A.1.

These parameters were loaded into SIMCA, by AZ, which created a PLS model with 3 components. This software, much like the model discussed in this work, applies cross-validation (CV) to different sized models and chooses the one with the best CV score. However, the scoring method used by the software differs from the one used. Instead of calculating the Q^2 score trough equation (2.16), it calculates a score trough the equation shown bellow.[16]

$$Q^2 = 1 - \frac{PRESS}{K(N-1)} \qquad A = 1 \tag{4.2}$$

$$Q^2 = 1 - \frac{PRESS_A}{\sum_i^N (y_i - \hat{y}_i)^2_{A-1}} \qquad A > 1 \tag{4.3}$$

Even though both the methods presented by equation (2.16) and equations (4.2) and (4.3) are widely used throughout the available literature, the first one was chosen because, instead of merely giving an

insight to prediction capability relative to the one of a model with one less component, it also gives an insight to the quality of the model. Since the total sum of squares (TSS) is a summed squared difference with the mean, the score ends up comparing the model with one whose output is solely the mean of the output data and, therefore, this method yields negative scores to models with little to none predictive capability, that is, models whose predictions are further away from the real value than the mean of the output data.

The model obtained by SIMCA was one of 3 components, with a goodness of fit (R^2) score of 0.8 and a goodness of prediction (Q^2) score of 0.75, whose coefficients are presented in table 4.1. It should be noted that a logarithmic transformation was applied to the output data in order to normalize it.[24]

Table 4.1: SIMCA's PLS model coefficients.

	Constant	D10	D50	D90	D43
coefficients	2.17E-01	1.05E-02	-5.51E-04	3.75E-04	-2.97E-04

The input data were inserted into Python™ and, with the help of the scikit-learn package, a partial least squares regression (PLSR) was applied, along with a 10-fold CV in order to choose the optimal number of components. The information regarding the model quality can be consulted in table 4.2.

Table 4.2: Model quality results obtained from a 10-fold CV.

Nº of components	MSE CV	Q2	R2
1	0.826	0.174	0.353
2	0.360	0.640	0.735
3	0.351	0.649	0.801
4	0.390	0.610	0.802

As it can be seen from the table shown, a model with 3 components was chosen due to its superior prediction capability (it has the lowest mean squared error of cross-validation (MSE CV) and higher Q^2), which, despite the differences in scoring method, is in line with what was obtained in SIMCA. To help visualise the behaviour of the different models during CV, the plotting of a validation curve was implemented (Figure 4.2). This curve represents both the training and validation scores obtained during CV for the different numbers of components possible, that is, the scores for the training and test sets. Since in CV 10 different scores are calculated, one for each fold, the errors calculated will have a certain degree of uncertainty. Because of this, a standard deviation (std) band (calculated trough Equation (4.4)) is shown alongside the curves, quantifying the degree of uncertainty for each score.

$$std = \sqrt{\frac{\sum_{i=1}^{N} (y_{predicted} - y_{measured})^2}{N}} \tag{4.4}$$

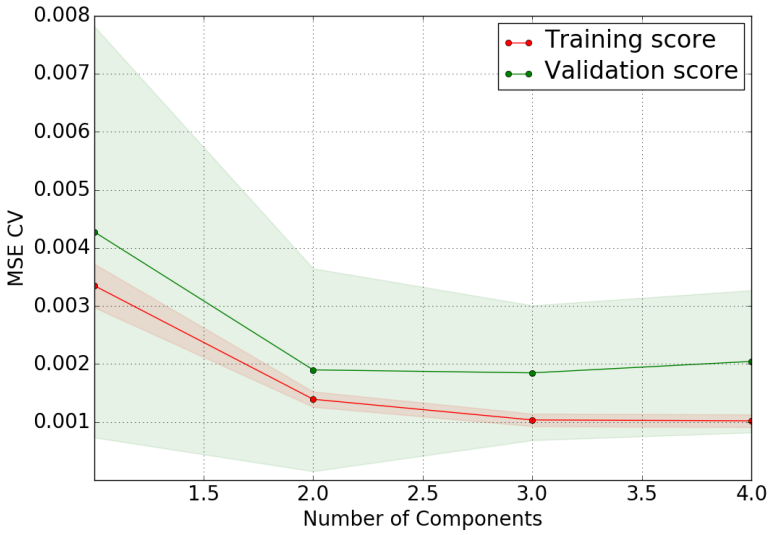


Figure 4.2: Validation curve for the granulation case.

Having obtained the same coefficients as SIMCA, the tool's capability of generating a PLS model with the optimal number of components was validated. However there is still the need to follow the usual steps in the development of a predictive multivariate model, in order to judge its results. The first one is to look at the plot of the predicted values against the the output training data (Figure 4.3). These values are analysed using a y=x plot as reference: the closer the values are to this reference line, the better the predictions. This reference plot also displays a band, which indicates the expected area where the predictions will land, according to the std of the predictions in CV. Additionally, the plots not only display the value of the aforementioned std, but also the values for the confidence intervals calculated for confidence levels of 95% and 99%. These intervals correspond to the range were the prediction are expected to land with 95% or 99% or certainty.

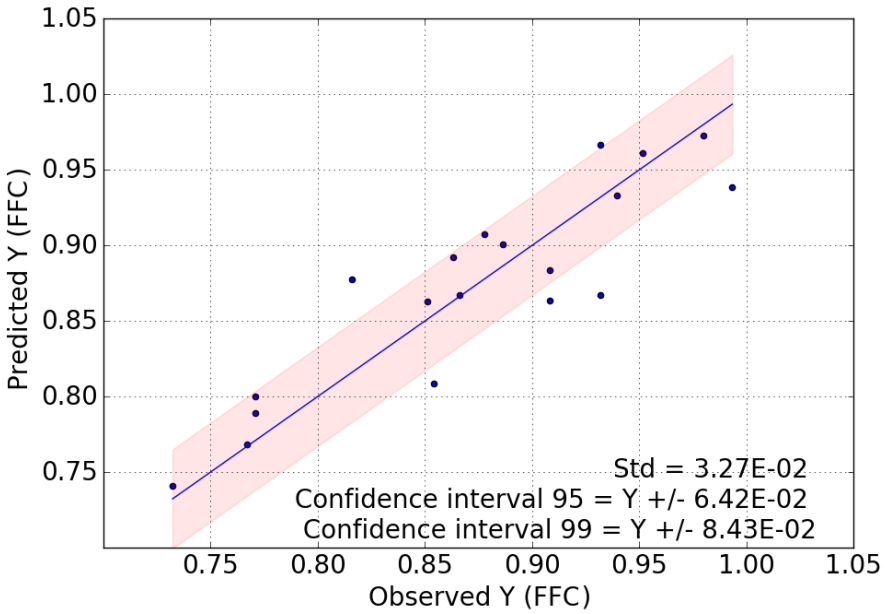


Figure 4.3: Predicted Y vs. Observed Y plot with reference.

Through the plot displayed, it can be seen that most of the predictions land close to the reference plot within the std band. This allow us to classify the quality of the predictions as being good. Some deviations are shown, but they are not severe. In order to get a better sense of the severity of these deviations, an option to add deviation plots was implemented and, as it can be inferred from Figure 4.4, it was concluded that the biggest deviations were not much greater than 5%.

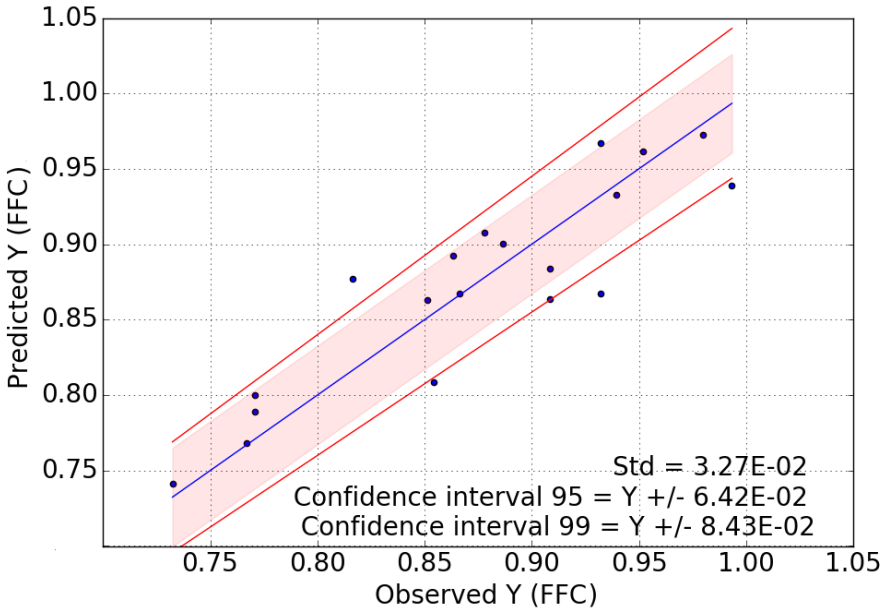


Figure 4.4: Predicted Y vs. Observed Y plot with 5% deviation reference.

Even though the deviations greater than the std, shown by some points, are not severe, there was still the suspicion that they might have been caused by overfitting. This suspicion comes from the fact that there is only a small number of observations, 19, in the training set. With the goal of analysing if said suspicion is true and to have a way to check for overfitting in future cases, a learning curve plot (Figure 4.5) was implemented. For this plot, a model is fitted for different numbers of observations (training examples) and both the training score of the model and the CV score are plotted alongside uncertainty bands, similar to the ones in the validation curve.

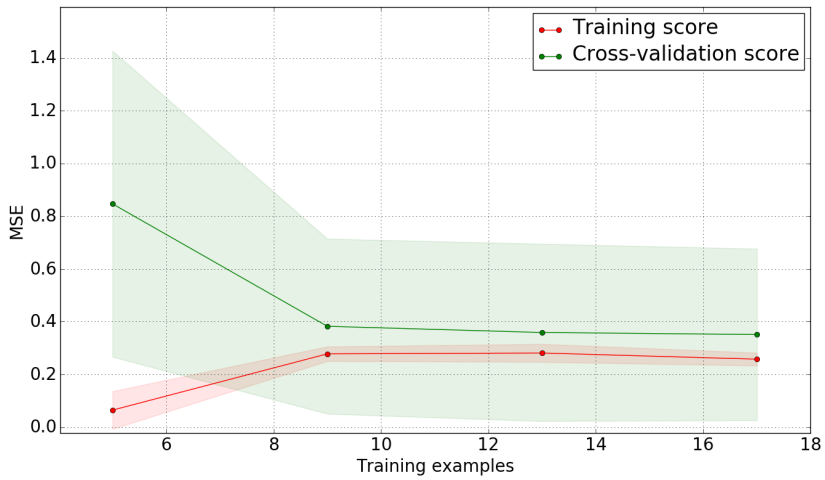


Figure 4.5: Learning curve for the granulation case.

For a small number of training examples, the learning curve shows a large separation between both the scores. This points out that, for up to 8 examples in the training set, the training error is low, indicating a good fit to the training data, and the error for the CV tests is high, indicating that the predictions made were not good. This facts when paired together, by definition, constitute overfitting. For higher numbers of samples the errors converge and there is a relatively small difference between both scores. This indicates that the number of samples used was appropriate and that there was no overffiting. The high degree of uncertainty in the CV score, displayed by the green band, indicates that, even though the score values are close together, it is not possible to claim with total certainty that there is no overfitting. Having obtained the same results as the model used for validation and having no more data regarding this case study, it was considered that there is no overfitting.

4.3 Statistical tests validation

An important step in judging the quality of the data set and, therefore, the model's, is outlier detection through statistical tests, namely the distance to the model of an observation (DmodX) and Hotelling's T^2 range. As such there is a need to validate the implemented functions and cross reference their results with the ones obtained from the SIMCA report.

4.3.1 Distance to the model

Naturally, the first statistical test applied is the DmodX that, as stated previously, measures the distance of a sample point to the plane created by the model. Both equations (2.17) and (2.18) were implemented, as the tests are being applied to the training data.

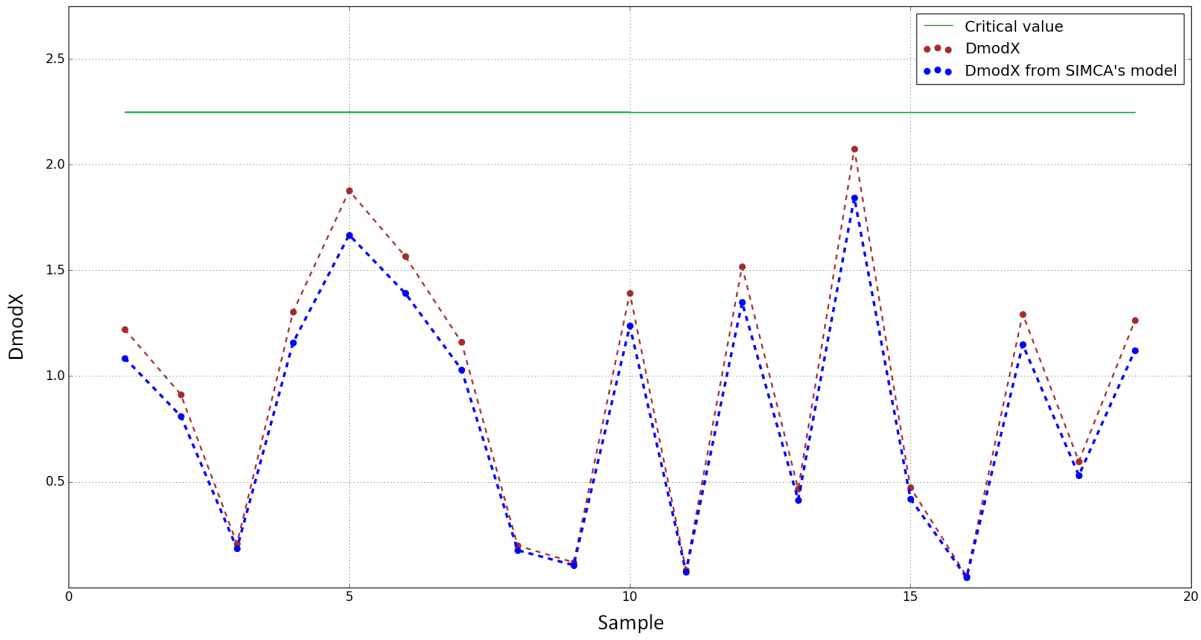


Figure 4.6: Comparison plot between the DmodX results from the implemented functions and the ones from SIMCA's model.

From the results displayed in Figure 4.6 a clear difference between the results and SIMCA's report is noted. Upon using some of the training data as a prediction set in SIMCA, the value for DmodX without a correction factor for some samples was obtained, presenting the same value as the implemented

function based on equation (2.17), what leads to the conclusion that the deviation in results is caused by an error in the implementation of the correction factor. Consequently, an hypothesis was created (equation (4.5)) that the correction factor was also dependent on the number of input variables similarly to how the DmodX is.

$$\nu = \frac{N}{N - A_0 - (K - A)} \tag{4.5}$$

In the interest of testing this hypothesis, several changes were made to the input data which was inserted in SIMCA's software both as training set and as a prediction set, so the correction factor could be obtained by the quotient between the DmodX for the training data set and the DmodX for the prediction data set. This correction factor was compared to both the hypothesised one and the one in equation (2.18), so as to withdraw conclusions, even if the hypothesis is wrong. In these tests the number of samples in a dataset (N), number of X variables (K) and number of components of the PLS model (A) were varied individually, and the conclusions withdrawn were as follows:

- Even though the correction factor was shown to be independent of the number of X variables, the hypothesised factor yielded better results than the one presented in the literature;
- The percentage of deviation of both correction factors did not show significant fluctuations when changing the number of samples in the data set, leading to believe that the dependency of the factor in question with this parameter is correct;
- The percentage of deviation of both correction factors showed high fluctuations when changing the number of components in the model and ergo the dependency with the number of components was considered incorrect.

Taking these conclusions into account, a new hypothesis was created, and the dependency with a new parameter, α, was included:

$$\nu = \frac{N}{N - \alpha} \tag{4.6}$$

This parameter was then calculated for the correction factors obtained with the unchanged training data set, for a number of components ranging from 1 to 3. The results obtained are displayed in Table 4.3.

Table 4.3: Values of α obtained for the models with different number of components.

A	α
1	1.033
2	1.565
3	2.118

The obtained results were fitted using a linear regression, having shown a high linear relation as such:

$$\alpha = 0.542 \cdot A + 0.487, \qquad R^2 = 0.999 \tag{4.7}$$

Equation (4.7) was then promptly implemented, having shown the desired results, displayed in Figure 4.7

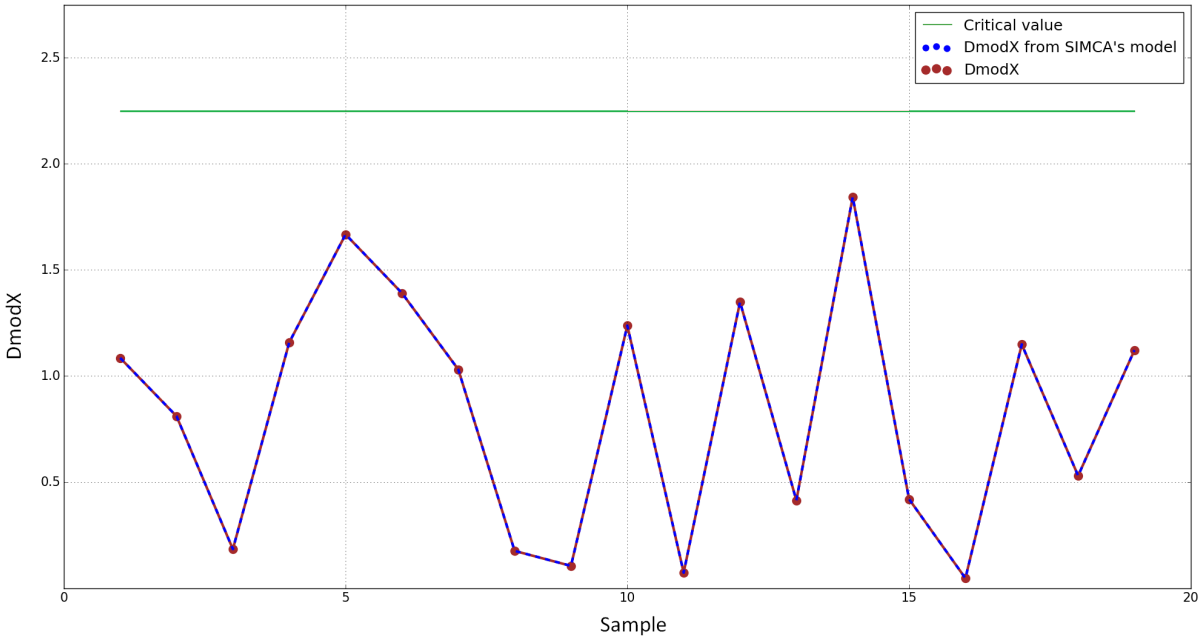


Figure 4.7: Comparison plot between the DmodX results from the implementation of the new ν and the ones from SIMCA's model.

Even though the results obtained were pleasing, one should be wary when using this correlation. Since it was obtained for a limited range of the number of components parameter, one is at risk of having a high number of components where the correlation fails at calculating the correction factor, specially considering the fact that it is quite common to create these types of models with datasets containing a high number of input variables. With the validation of equation (4.7) for high values of A in mind, the first 118 non binary parameters of the dataset presented in reference [25] were inserted into SIMCA and a a model with 118 components was created. The same data was then loaded into Python and an analogous model was created. The value obtained for the correction factor, as evidenced by table 4.4, was really close to the one from SIMCA, being that the deviation is negligible. In light of these results, the hypothesis was considered correct and was applied in the subsequent case studies.

Table 4.4: Values for ν obtained in validation for a high number of components.

A	SIMCA's ν	Equation's (4.7) ν	Relative error (%)
118	1.0011	1.0012	0.0097

4.3.2 Hotelling's T^2

Regarding the Hotelling's T^2, the validation process was more straightforward. In fact, the implemented function yielded results that match the ones from SIMCA's report (Figure 4.8) and, therefore, this test was considered validated. From the results it was also concluded that the dataset used to fit the model showed no outliers.

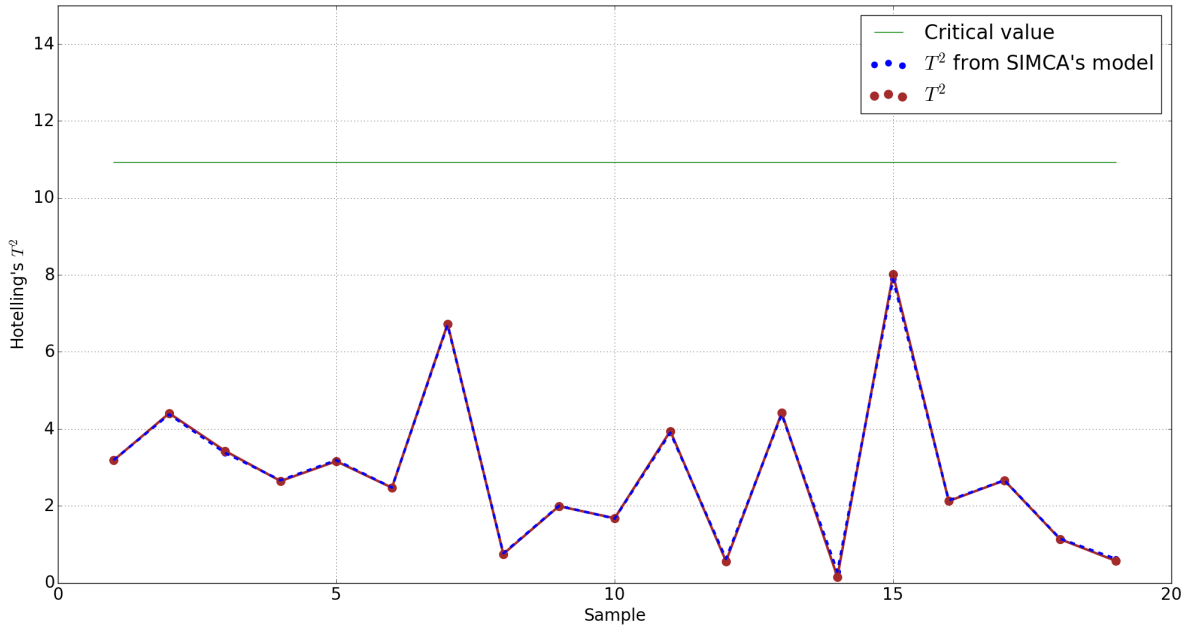


Figure 4.8: Comparison plot between results from the Hotelling's T^2 implemented and the ones from SIMCA.

4.4 gPROMS Simulation

Being the main purpose of this work the soft-sensing of parameters whose relations with the remaining operation variables is unknown, it is only natural that the next step in this case study is the implementation of the model in a dynamic simulation.

4.4.1 Implementation

The model's implementation is possible because the Python tool allows for the creation of an XML file containing all the relevant information of the model, such as the coefficients, the maximum and minimum values, amongst others. This XML file will be loaded into the gPROMS software which will use the information it contains and, paired with the Python tool, it'll predict values for the FFC over time. This pairing is only possible trough a foreign object (FO) that allows for the trading of information between Python and gPROMS.

During simulations, the predictions are calculated thanks to the data based sensor block. The sensor is able to make predictions because it is linked to the XML file and, therefore, contains all the model's information. This can be seen in its performance tab (Figure 4.9) where the path of the XML file is specified. In the path, the name of the FO (pyFO) that makes the bridge between Python and gProms and the path of Python file used are also specified.

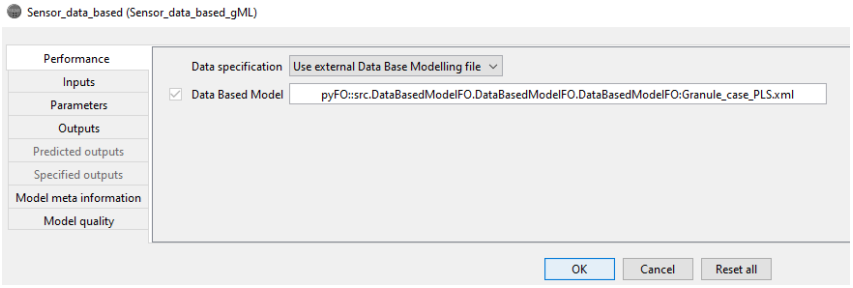


Figure 4.9: Data based sensor Performance tab.

In the Inputs tab of the block (Figure 4.10), the variables used as inputs for the model are selected. This variables are then mapped to the results coming from the PSD_sensor.

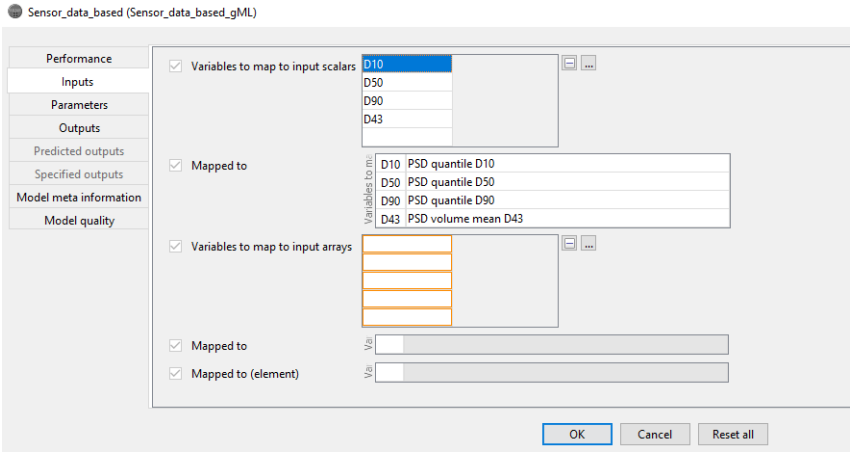


Figure 4.10: Data based sensor Inputs tab.

Both the Parameters and Outputs tabs were not used in this works. The Parameters tab is used whenever one wants to make predictions by inputting the data manually which, naturally, does not occur in a dynamic simulation. The outputs tab, on the other hand, is used to map the predictions as inputs of other blocks, which was not necessary since the only goal of the data based sensor is, in this case, soft-sensing. In the Model meta information tab (Figure 4.11), it is possible to check informations regarding the model and the tool used to create it. This is particularly useful since it informs the user of the type of model they are using and when and by whom it was created.

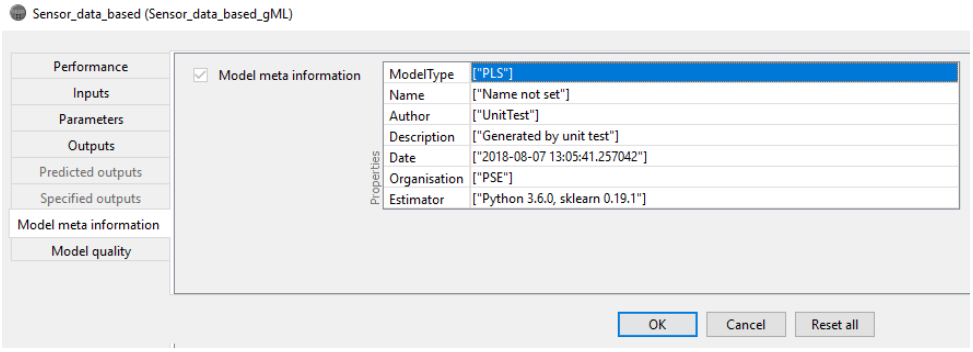


Figure 4.11: Data based sensor Model meta information tab.

In the model quality tab (Figure 4.12), as its name states, information regarding the quality of the model is displayed. Both the cross validation score and training scores are displayed, alongside the std

expected for the predictions, according to CV. Additionally, the minimum and maximum values of each parameter in the training data are displayed.

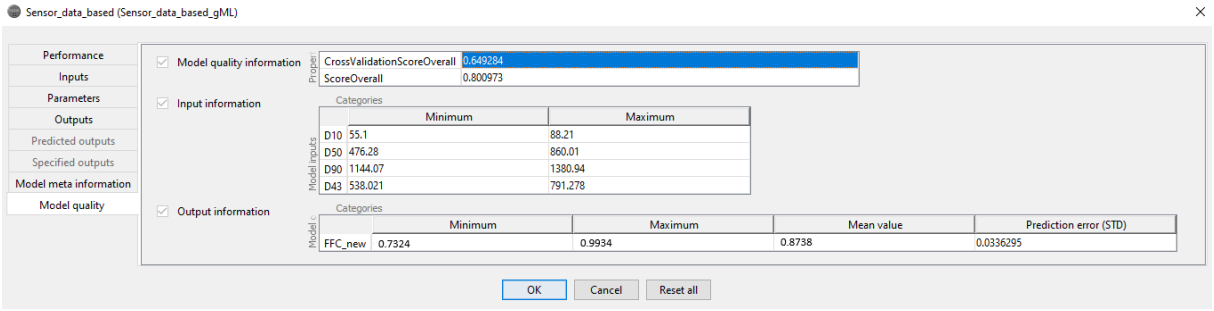


Figure 4.12: Data based sensor Model quality tab.

Once the simulation is done, if no warnings have been triggered, the data based sensor block will display a green ring around itself (Figure 4.13). In the case where one of the statistical tests has a value higher than its critical one or the input data used in the model has parameters with values higher than the maximum or lower than the minimum value of the parameter, in the training data, a warning will be triggered. When that happens, a red ring will be displayed around the block (Figure 4.13).

Figure 4.13: Data based sensor when no warnings are triggered (left) and when a warning is triggered (right).

4.4.2 Results

The simulation ran for 1800 s, with the PSD_sensor block registering different values for the particle size distribution. The values predicted for the FFC during the simulation are presented in Figure 4.14.

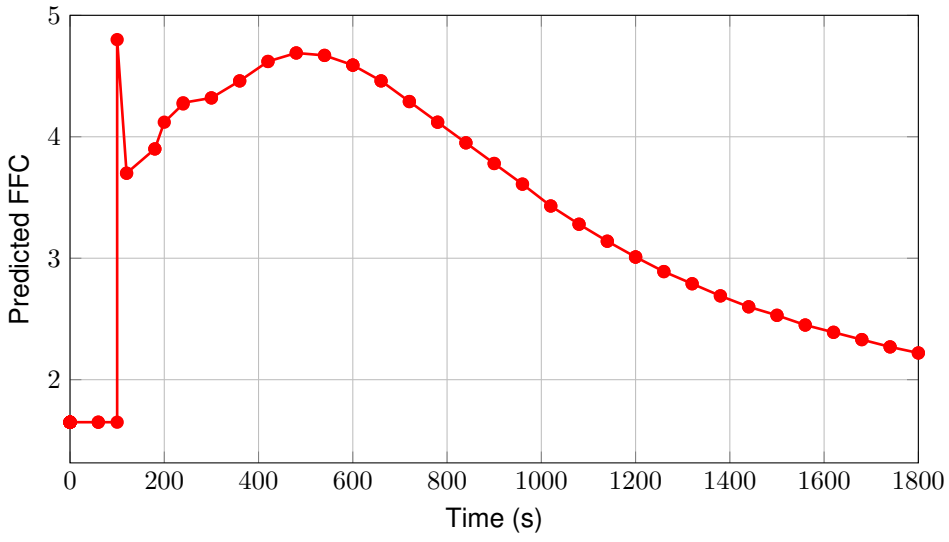
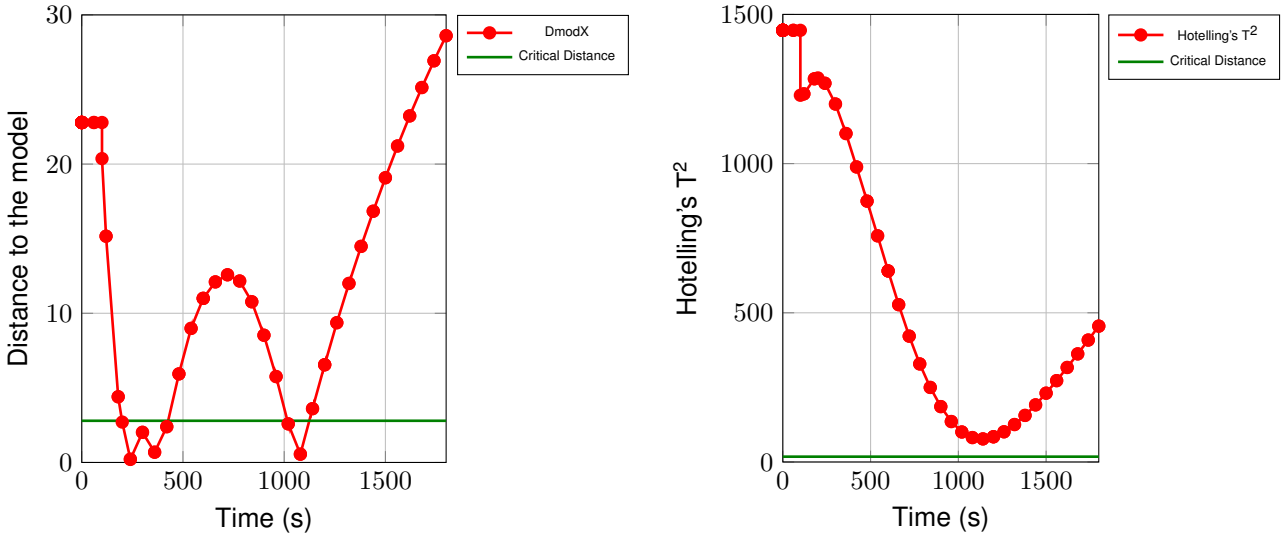


Figure 4.14: Predicted values for the FFC over time

As it can be seen, the model was able to make predictions when included in a dynamic simulation. However, at the end of the simulation, the data based sensor block present a red ring. Because of this, both the results of the statistical tests and the values taken by the model's inputs must be analysed in order to asses what triggered the warning. Figure 4.15 shows that, even though DmodX not always took a value over the critical distance, the Hotelling's T^2 consistently had values over the critical bound throughout the entire simulation.



(a) DmodX

(b) Hotelling's T^2

Figure 4.15: Statistical tests for the predictions of the FFC.

By looking at the comparison plots, presented in Figure 4.16 and Appendix A.2, it becomes apparent why the statistical tests failed. By having the input parameters assuming values higher than their maximum/ lower than their minimum values in the training data, it is assured that the observation points will present higher distances to the model plane/ its centre.

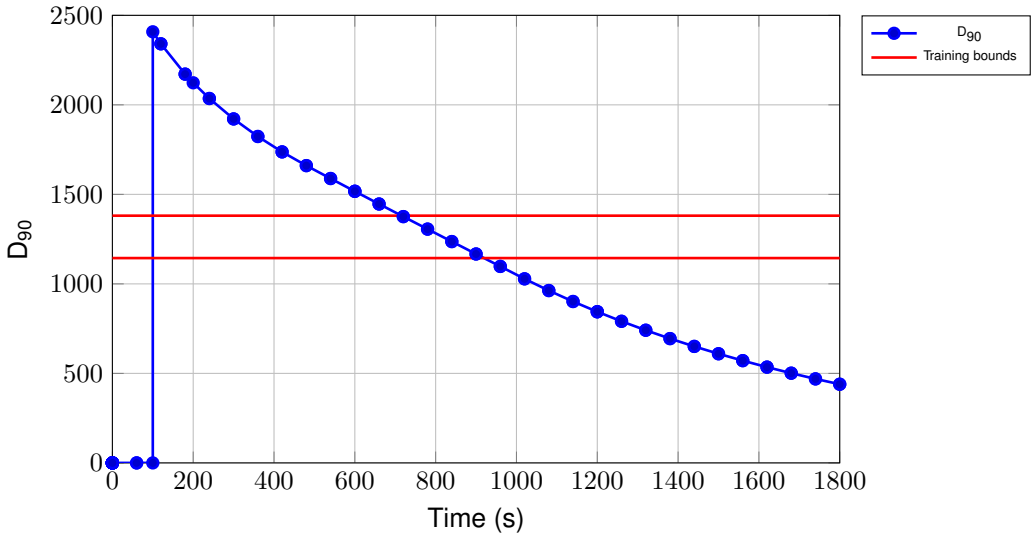


Figure 4.16: Values of D_{90}, μm, during the simulation and respective training bounds.

The model was fitted with data within certain bounds, however, by exceeding those bounds, one does not have a guarantee that the quality of the predictions, assessed previously, will uphold for the

new input data. This fact constitutes one of the main limitations of these type of models: no matter how broad the range of the training data, the quality of the predictions made with inputs outside that range can never be assured. Therefore, even though the main goal of soft-sensing the FFC was accomplished, the predictions cannot be considered reliable.

Chapter 5

Compression Case Study

Having the model creation/analysis tools validated, it was now possible to fit a model to new data. Therefore a partial least squares (PLS) model was developed for a compression case study. In this case study, with data provided by Pfizer, as a part of the Advanced Digital Design of Pharmaceutical Therapeutics (ADDoPT) project, a model was created for the prediction of the compressibility of a blend, defined as the fraction of volume reduction under pressure[26], from particle size distribution data. Analogously to what happened in the previous case study, the output was logarithmically transformed and predictions were made for this transformation.

5.1 Process description

The flowsheet in Figure 5.1, provided and assembled by Pfizer, represents the process where the training data for the model was collected. In this process, a fenofibrate is crystallized, milled and then mixed with an excipient. This blend will go trough a particle size distribution (blend_PSD block), whose results will be inserted into the data based sensor (Sensor_data_based), namely the cumulative diameters (D_5, D_{10}, D_{16}, D_{25}, D_{50}, D_{75}, D_{84}, D_{90} and D_{95}), the Sauter mean diameter (D_{32}) and the volume weighted mean (D_{43}) , in order to predict the compressibility of the blend in the roller compactor - parameter that influences the efficiency of this step of the process. The blend will then be milled and tablets will be created due to the usage of a tablet press.

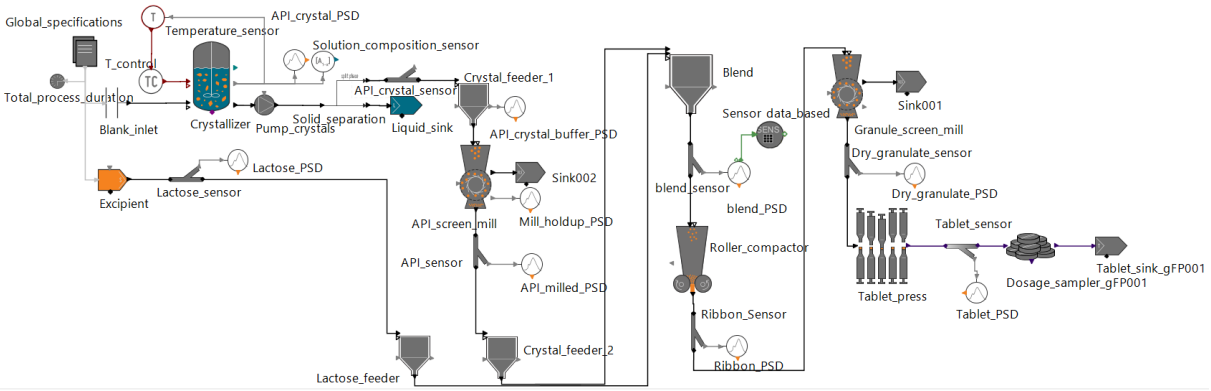


Figure 5.1: Pfizer's process flowsheet.

5.2 Model Development

The data provided consisted of 100 samples of untreated particle size distribution results, namely the diameters of each sieve and their respective blend volume fraction. With these parameters, the cumulative diameters D_5, D_{10}, D_{16}, D_{25}, D_{50}, D_{75}, D_{84}, D_{90} and D_{95}, the D_{32} and the D_{43} were calculated. Much like D_{43}, D_{32} is dependent on the diameters of the sieves (D_i) and their respective volume fraction (f_i) (Equation (5.1)).

$$D_{3,2} = \frac{\sum_i n_i D_i^3}{\sum_i n_i D_i^2} \Leftrightarrow D_{3,2} = \frac{1}{\sum_i \frac{f_i}{D_i}} \tag{5.1}$$

Once calculated, the data was loaded to Python and a model with 4 components and a goodness of prediction (Q^2) and goodness of fit (R^2) scores of 0.737 and 0.822, respectively, was obtained (Table 5.1). No coefficients will be shown for this case study because the number of inputs for the final model will be too great for it to be possible.

Table 5.1: Model quality for compression case study

Nº of components	MSE CV	Q^2	R^2
1	0.409	0.591	0.683
2	0.297	0.703	0.791
3	0.286	0.714	0.807
4	0.263	0.737	0.822
5	0.276	0.724	0.830
6	0.274	0.726	0.832
7	0.302	0.698	0.836
8	0.317	0.683	0.839
9	0.320	0.680	0.840
10	0.316	0.684	0.840

By interpreting the validation curve (Figure 5.2) it becomes apparent that there is a high degree of uncertainty prediction wise. For the optimal number of components case, this indicates that albeit the score was the lowest obtained, there is still the possibility to obtain high prediction errors with this model. Even though this is concerning, further analysis of the model is still required.

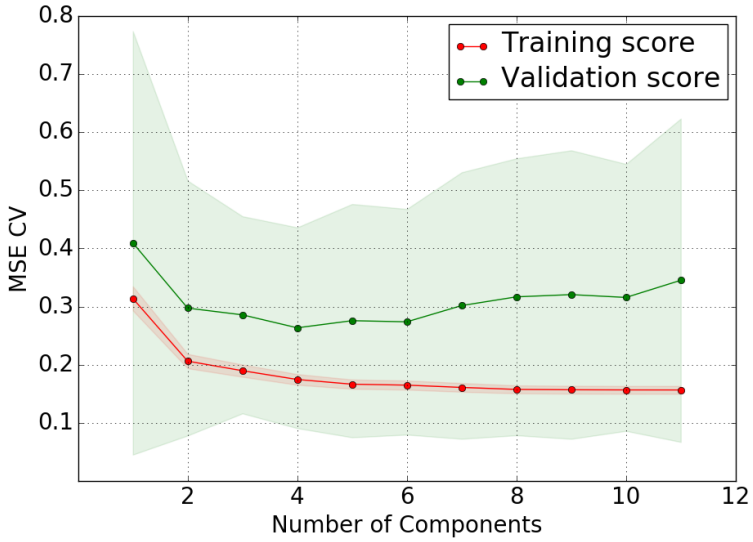


Figure 5.2: Validation curve for the compression case study.

The reason for this uncertainty becomes more apparent by plotting the observed values against the predicted ones (Figure 5.3). For high compressibility values, the predictions are close to the reference plot and within the uncertainty band, showing good quality. However, as the values of compressibility decrease, the deviations from the reference plot increase, with most points going outside the uncertainty band. This leads to the conclusion that during cross-validation (CV), folds with high values of compressibility will have a lower mean squared error of cross-validation (MSE CV) while folds with lower values of compressibility will have a higher MSE CV leading to the high degree of uncertainty showed in the validation curve. Furthermore, for lower values of compressibility a point is reached when the plotted points have non-negligible deviations from the reference plot and the uncertainty band. Therefore the calculated prediction for lower values cannot be considered reliable.

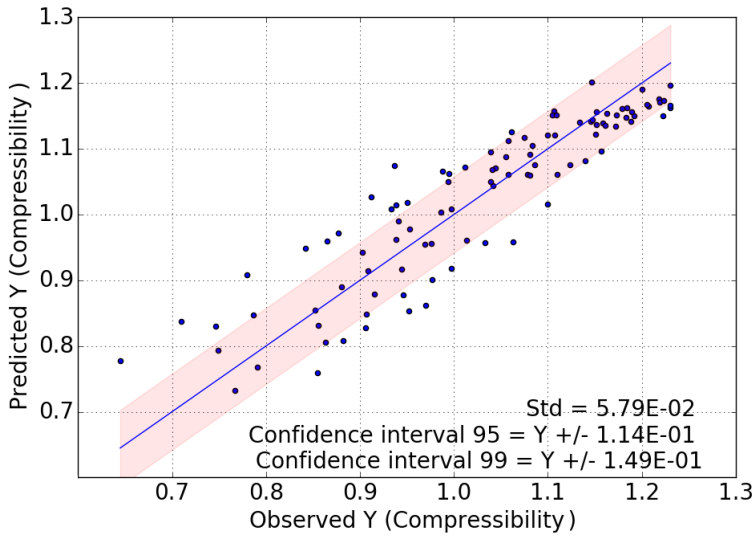


Figure 5.3: Predicted Y vs. Observed Y plot for the compression case study.

5.2.1 Polynomial Transformation

One possible cause for the bad quality of the predictions for lower values of compressibility is the simplicity of the model. In order to try and correct this issue, a 2nd order polynomial transformation was applied to add complexity to the model. This non-linear transformation - illustrated in Figure 5.4 - adds as input the value 1 (which, in this work, will always have a null coefficient) and the multiplication of each parameter by itself an by the remainder parameters. By training the data on non-linear functions of the input data, it is assured that the partial least squares regression (PLSR) is able to fit to a wider range of data while still generating a linear model. [10]

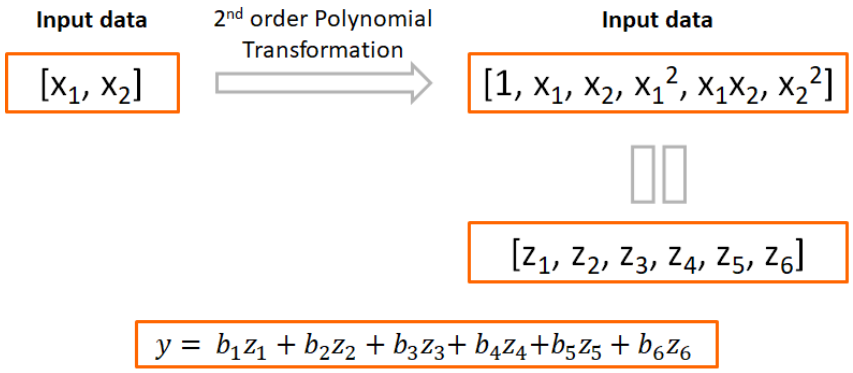


Figure 5.4: Schematic representation of a 2nd order Polynomial Transformation.

The transformed input data, now with 78 inputs, was fed to the tool and the results were as presented in Table 5.2.

Table 5.2: Model quality for compression case study with polynomial transformation.

Nº of components	MSE CV	Q^2	R^2
1	0.510	0.490	0.612
2	0.258	0.742	0.811
3	0.261	0.739	0.819
4	0.290	0.710	0.828
5	0.287	0.713	0.831
6	0.278	0.722	0.844
7	0.287	0.713	0.847
8	0.321	0.679	0.850
9	0.263	0.737	0.855
10	0.277	0.723	0.864
11	0.328	0.672	0.878
12	0.245	0.755	0.882
13	0.205	0.795	0.883
...	...	...	...
78	340.192	-339.192	0.987

The model obtained was one of 13 components, with higher scoring than the one obtained previously. In fact, as it can be seen by the presented slice of its validation curve (Figure 5.6) the new model not only shows a lower MSE CV, but also presents a much lower uncertainty regarding said score.

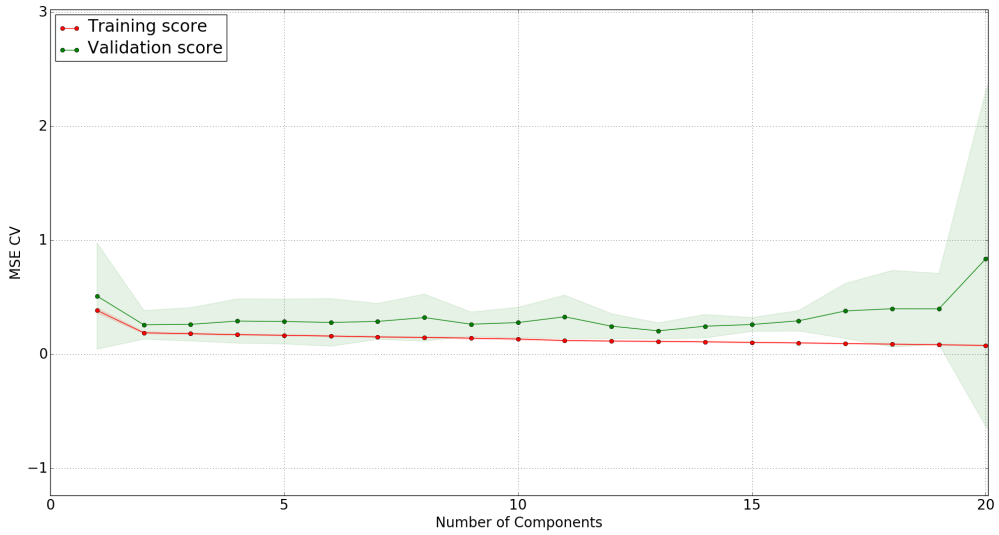


Figure 5.5: Validation curve for compression case study with polynomial transformation.

Figure 5.6 shows the new model was able to maintain a good quality of prediction for higher values of compressibility (although some loss in quality is registered), while being able to get better predictions for lower values. As it can be inferred from the plots in Figure B.1, presented in Appendix B.1, most points outside the uncertainty band present a deviation close to 10%, being that some values have slightly bigger deviations.

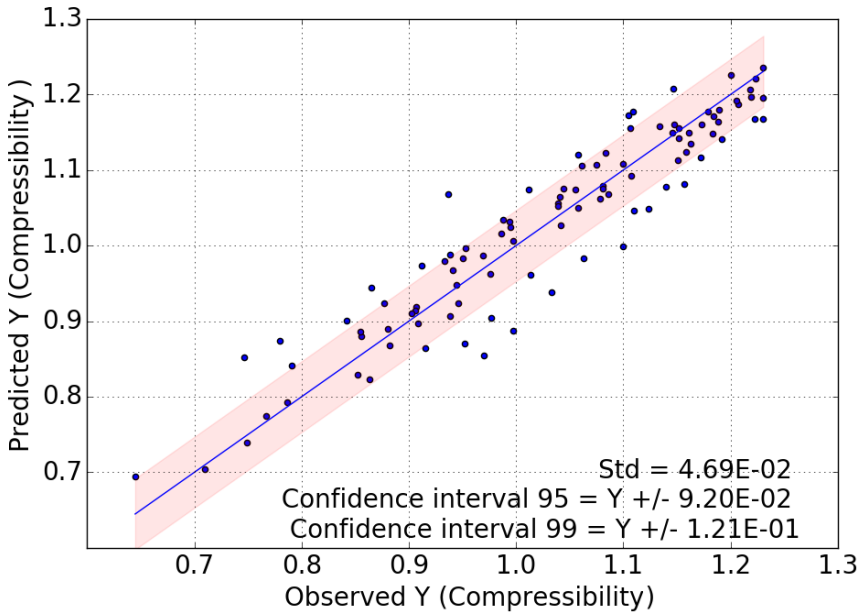


Figure 5.6: Predicted Y vs. Observed Y plot for the compression case study with polynomial transformation.

The results shown in the aforementioned plots serve to show that, by adding complexity to the training data, the polynomial transformation added robustness to the model, since it is now suitable for predicting values within the range of the compressibility in the training data.

Granting the training data has a sizeable number of samples, it is still necessary to take a look at

the learning curve to figure out if there is any possibility of overfitting. As demonstrated by the learning curve in Figure 5.7, there is only a small gap separating the scores indicating the absence of overfitting.

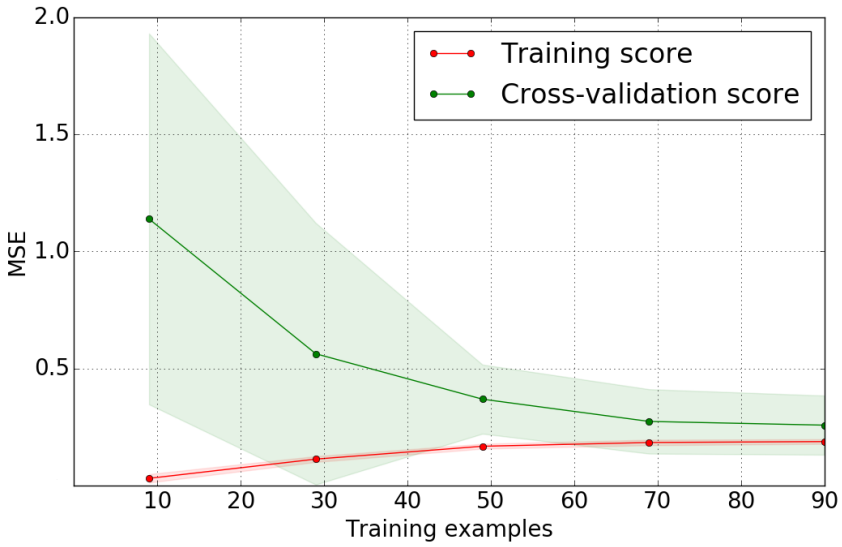
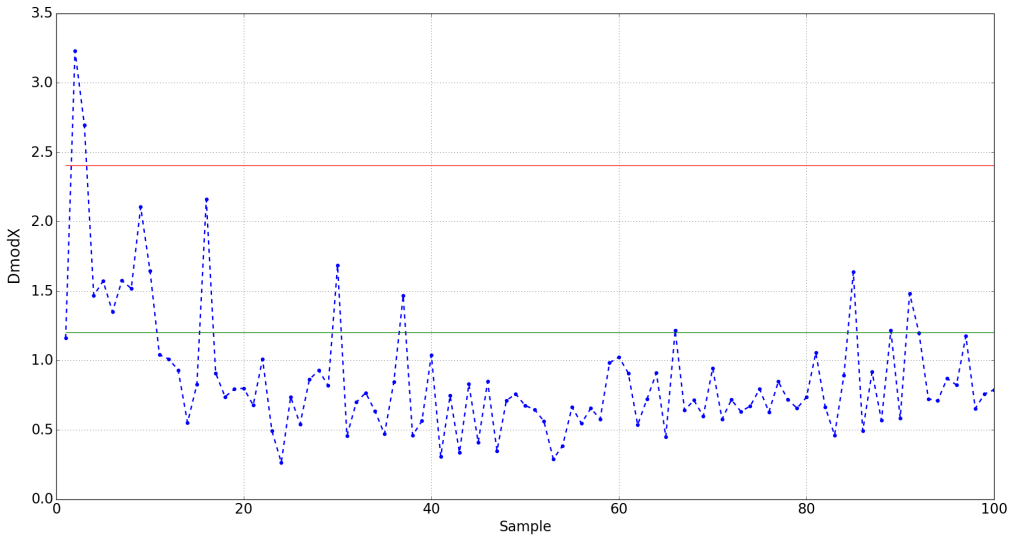


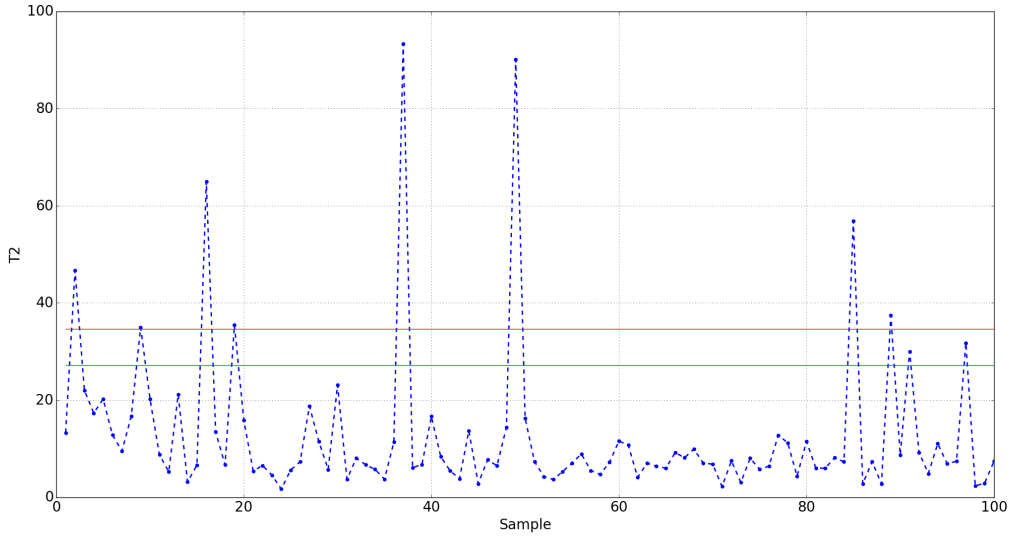
Figure 5.7: Learning curve for the compression case study with polynomial transformation.

5.3 Statistical tests

Once the predictive capability of the model had been established, the statistical tests were applied to the training data (Figure 5.8).



(a) DmodX



(b) Hotelling's T^2

Figure 5.8: Statistical tests for the training data of the compression model with polynomial transformation.

The test results display the existence of both moderate and strong outliers. Usually, these facts would lead to a critical analysis of the data to try to determine if these observations are explained by the process or not i.e. if the deviations were caused by certain operational conditions. If that is the case, then the observations are not removed from the dataset since they are a result of the process's behaviour and, because of that, should be accounted for. In case they are not explainable, they are removed from the dataset. Since the data was provided by a third party, the existing knowledge of the process is not enough to do the required analysis and so it was assumed that the outlier points were explained by the process.

5.4 gPROMS Simulation Results

The relevant information regarding the chosen model - a PLSR model with 13 components whose training data underwent a polynomial transformation - was stored into an XML file, which was then loaded into gPROMS with the aim of soft-sensing the compressibility of a blend.

The simulation ran for 18000 s, having the results of the particle size distribution of the blend registered (trough the blend_PSD block) and fed to the data based sensor. The sensor subsequently calculated the predictions for the compression of the blend in the Roller Compactor for each point in time, which are reported in Figure 5.9.

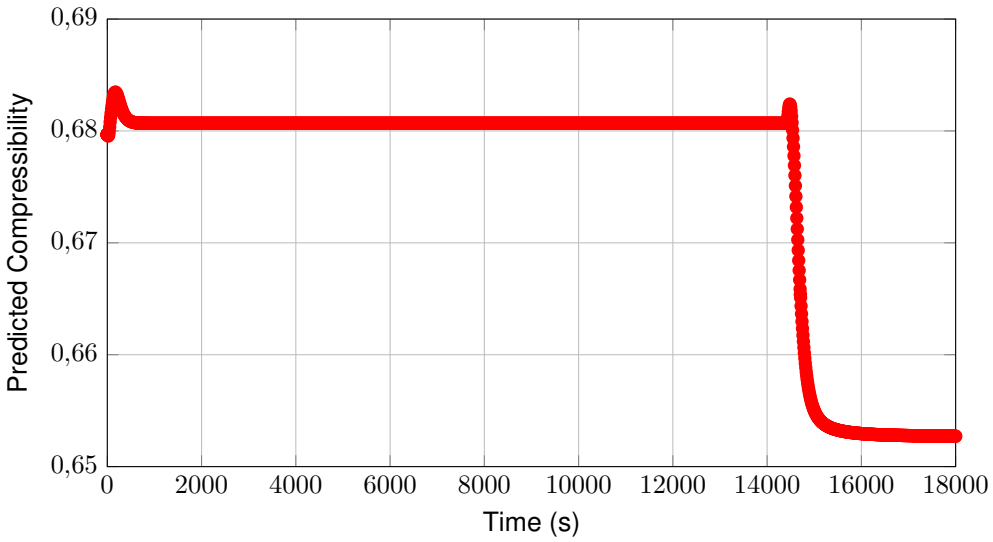
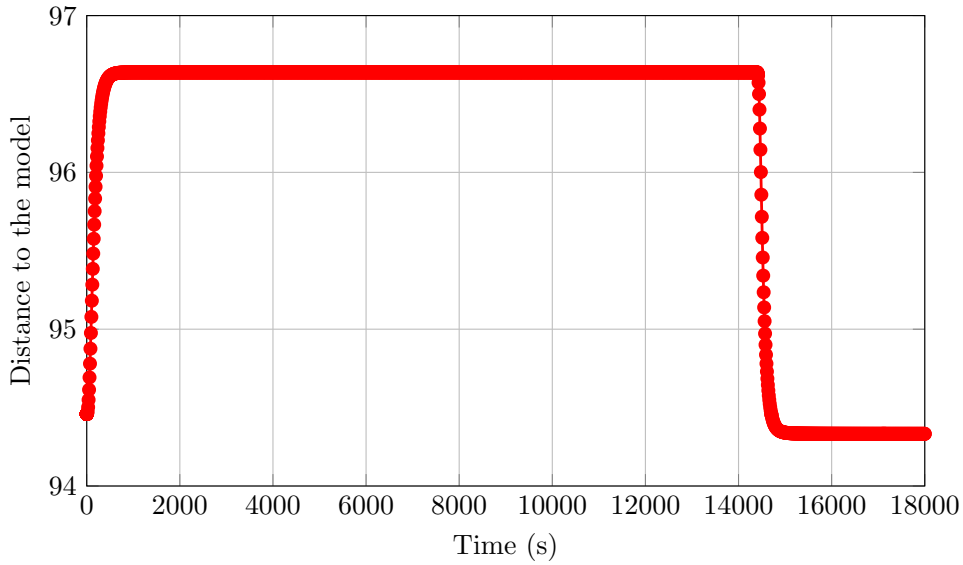
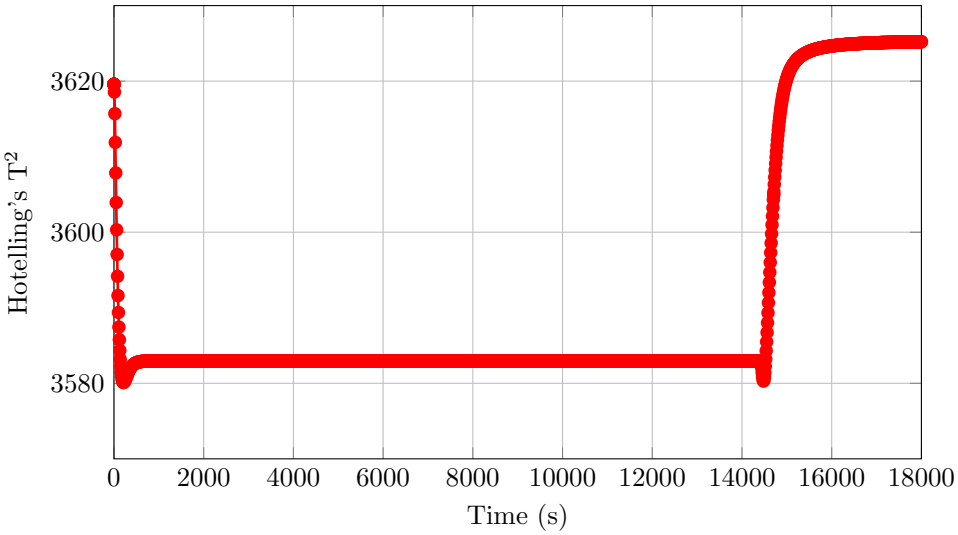


Figure 5.9: Predicted values for the compressibility of the blend in the Roller Compactor over time

The sensor was able to predict the value for the compression of the blend over time, not registering significant fluctuations. However, during the simulation a warning was triggered, evidenced by a red ring displayed in the data based sensor block. To asses the cause of this warning the statistical tests were analysed.



(a) DmodX (Critical Distance = 1.22)



(b) Hotelling's T^2 (Critical Distance = 27.14)

Figure 5.10: Statistical tests for the predictions of the compression in the Roller Compactor.

From the results of the statistical tests (Figure 5.10) it becomes evident why the warning was triggered. Both tests present a distance well above their respective critical distance. To better understand this results, the analysis of the input data of the sensor is required. By interpreting the plots in Figure B.2, present in Appendix B.2, it is possible to infer that, even though the parameters D_5, D_{10} and D_{16} never go out of bounds, the same cannot be said for the remaining parameters. In fact, all of the remaining parameters, for the exception of D_{25}, have a difference of one or more orders of magnitude from their bounds. This big difference, relative to the training data, is what triggers the statistical tests to have distances significantly larger than their respective critical distances.

The results of the statistical tests serve as evidence for the unreliability of the predictions. Even though the model was able to predict the compressibility of the blend in the roller compactor, it did so using parameters outside the range the model was intended to be used in. This once again underlines one of the biggest limitations of these types of models, namely that the quality of the predictions cannot be assured outside the range of values used in the training data. In order to consider the model's predictions reliable (assuming the model scored well during the choice of the number of components of the model), the operational conditions of the simulation must included in the range of the operational conditions used to obtain the training data.

Chapter 6

Solid Oxide Fuel Cells Case Study

The case study discussed in this chapter had a different objective than the previous ones. Instead of having the goal of creating a predictive model for soft-sensing, the main goal was to asses how the tool would behave when creating a predictive model using the partial least squares (PLS) 2 algorithm i.e creating a model with multiple outputs. In this case, training data was obtained from a global system analysis (GSA) applied to the operational conditions of a solid oxide fuel cell (SOFC) model from gPROMS AML-FC library, in order to fit a model that could predict the power and voltage of the fuel cell, the molar fractions of O_2, N_2 and H_2O and the out temperature of the cathode and the molar fractions of H_2O, H_2, CO and CO_2 and the out temperature of the anode.

6.1 Model Development

The dataset used to create the model consisted of 10000 samples points of 10 input variables: the current density; the air's flow rate, temperature and pressure; the syngas's CH_4, CO, H_2 and H_2O content and its flow rate and pressure. These data points were directly loaded to the Python tool, without any transformations, and, as it can be inferred from Table 6.1, a model with 10 components was obtained (the same model that would be obtained using a multiple linear regression (MLR)) for the prediction of the 11 outputs. For this case study no coefficients will be displayed for the same reasons given in chapter 5.

Table 6.1: Model quality results for the SOFC case study.

Nº of components	MSE CV	Q^2	R^2
1	0.7072	0.2928	0.2935
2	0.4992	0.5008	0.5019
3	0.4008	0.5992	0.6002
4	0.2946	0.7054	0.7062
5	0.2601	0.7399	0.7407
6	0.2294	0.7706	0.7713
7	0.2285	0.7715	0.7722
8	0.2278	0.7722	0.7728
9	0.2275	0.7725	0.7733
10	0.2274	0.7726	0.7733

From the model quality analysis for the models created, it is possible to see that both the goodness of

prediction (Q^2) and goodness of fit (R^2) scores present values very similar to each other. This happens due to the high amount of samples points in the data set. Even though they are separated into 10 folds during cross-validation (CV), the number of points in each fold, 1000 points each, is still enough so that the model is able to predict values with a quality close to the quality of the fit. However, as it is demonstrated by the validation curve (Figure 6.1) there is a high degree of uncertainty linked to the predictions made during CV.

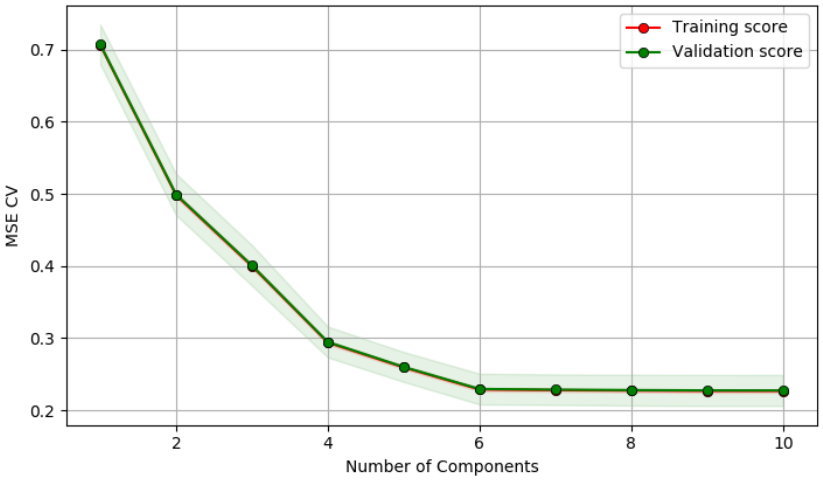
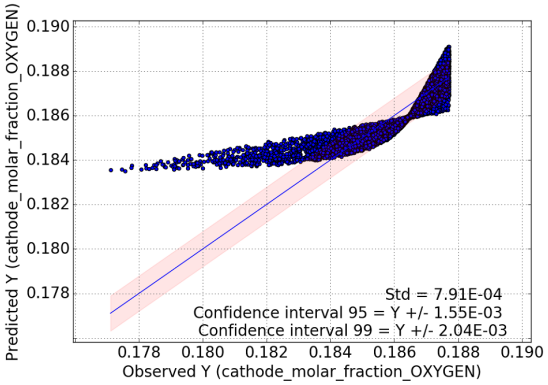
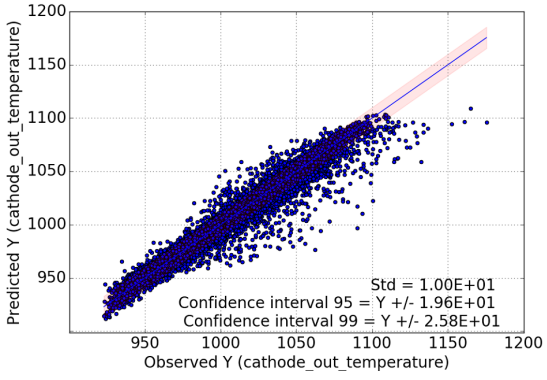


Figure 6.1: Validation curve for the SOFC case study.

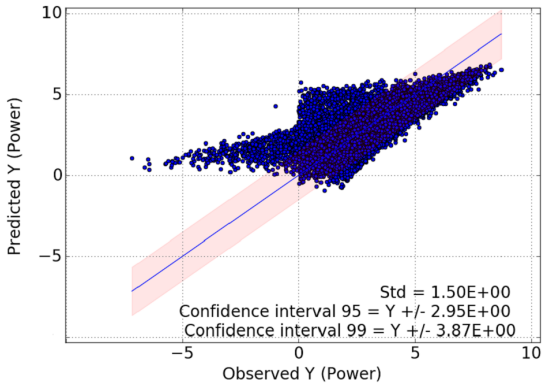
From the plots of the predicted values against the observed values (Figure 6.2) it can be seen that the model is incapable of making good predictions. This is specially true for outputs like the power and the molar fractions in the cathode where the plotted points deviate completely from the reference plot. This was somewhat expected since, by having the same number of components as the number of inputs, partial least squares regression (PLSR) turns into MLR which, for reasons explained in Chapter 2, is not recommendable to this kind of problems. It should be noted that only the plots for the power, voltage, cathode molar fraction of O_2 and the cathode out temperature are shown since they show similar behaviours between similar types of parameters. The remainder of the plots can be seen in Appendix C.1.



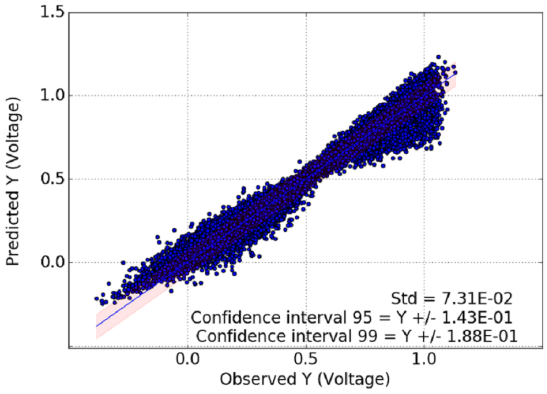
(a) Cathode molar fraction of O_2.



(b) Cathode out temperature (K).



(c) Power (kW).



(d) Voltage (V).

Figure 6.2: Predicted Y vs. Observed Y plot for the SOFC case study.

6.1.1 Polynomial Transformation

Similarly to what happened in the previous case study, it was suspected that the models inability to make good predictions was caused by its simplicity. Therefore, a polynomial transformations was applied to the input dataset, which now contained 66 input variables. The optimal result consisted of a model with 65 components (Table 6.2). Even though the models with 65 and 66 components scored similarly, the one with 65 yielded slightly better results, having scored better than the previously obtained model.

Table 6.2: Model quality for SOFC case study with polynomial transformation.

Nº of components	MSE CV	Q^2	R^2
5	0.287	0.713	0.714
10	0.190	0.810	0.811
15	0.104	0.896	0.897
20	0.089	0.911	0.912
25	0.076	0.924	0.926
30	0.066	0.934	0.935
35	0.065	0.935	0.936
40	0.064	0.936	0.937
45	0.062	0.938	0.939
50	0.062	0.938	0.939
55	0.062	0.938	0.939
65	0.061	0.939	0.940
66	0.061	0.939	0.940

As expected, the Q^2 and R^2 scores are very similar to each other, being that, in this case, the degree of uncertainty relative to the predictions score was significantly reduced (Figure 6.3).

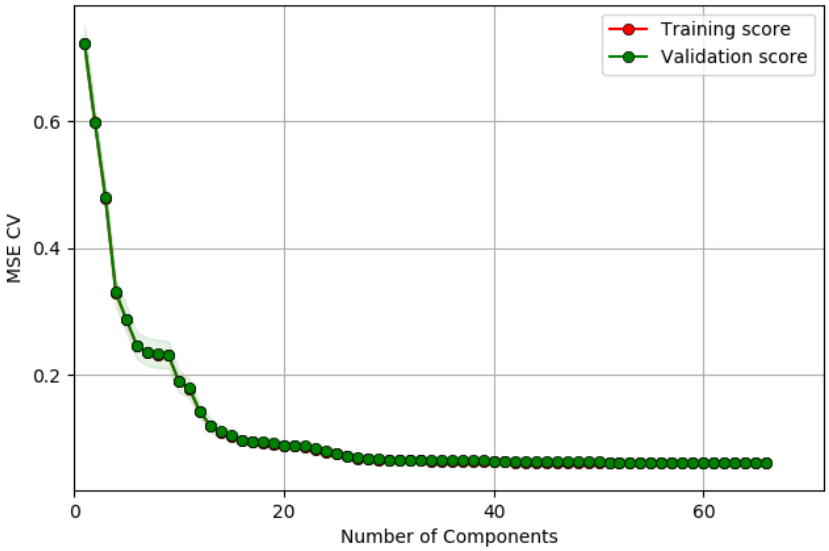
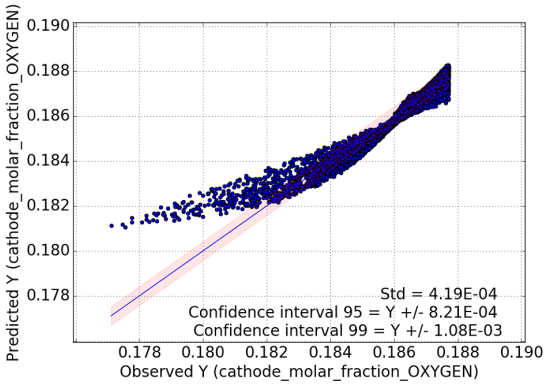
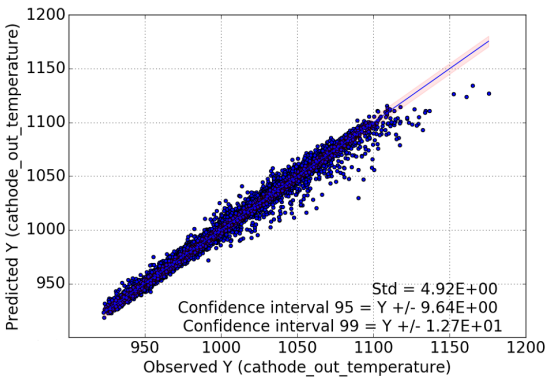


Figure 6.3: Validation curve for the SOFC case study with polynomial transformation.

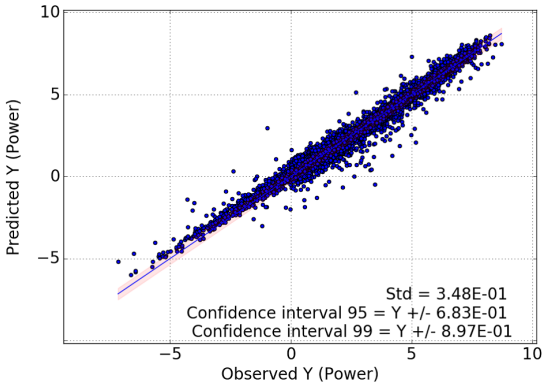
Looking at the comparison between the predicted and actual values (Figure 6.4), a significant improvement can be seen, specially regarding the power and the voltage. Nonetheless, the molar fractions for the cathode still show significant deviations from the reference. This indicates that, despite the improvement in the predictions for some of the output parameters, solely adding complexity to the model trough a polynomial transformation wasn't enough to get better results for the entirety of the response variables.



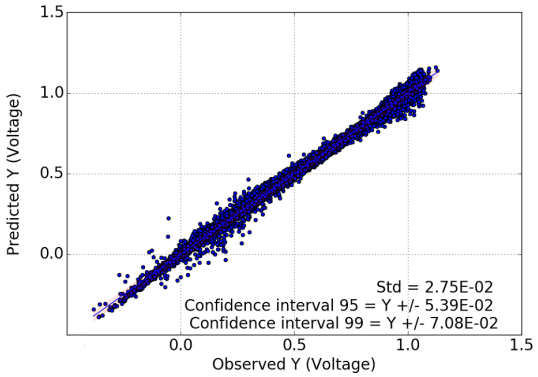
(a) Cathode molar fraction of O_2.



(b) Cathode out temperature (K).



(c) Power (kW).

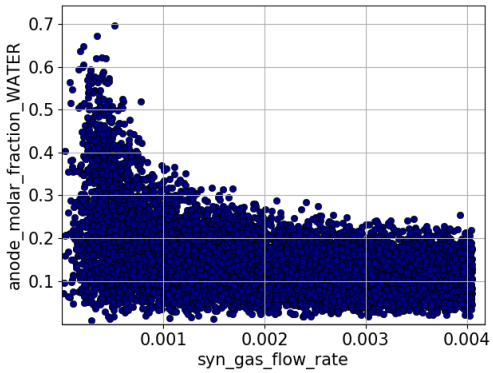


(d) Voltage (V).

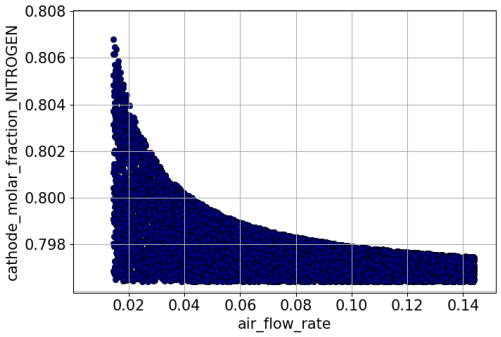
Figure 6.4: Predicted Y vs. Observed Y plot for the SOFC case study with polynomial transformation.

6.1.2 Reciprocal Transformation

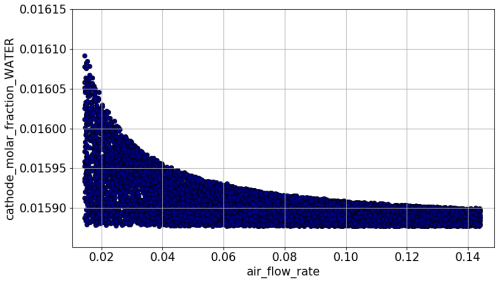
While looking at the plotting of the response variables against the inputs, it was noted that some variables display a reciprocal dependency i.e the outputs were obtained through the inverse of the inputs($\frac{1}{x}$). This is clear in the plots from Figure 6.5. Due to the high number of observations in the dataset, it is difficult do assess how some output variables vary with some inputs, however there is a suspicion that other variables, besides the ones presented in Figure 6.5, show a reciprocal dependency. The plots for those cases can be consulted in Appendix C.3.



(a) Anode molar fraction of H_2O Vs. Syngas flow rate (mol/s).



(b) Cathode molar fraction of N_2 Vs. Air flow rate (mol/s).



(c) Cathode molar fraction of H_2O Vs. Air flow rate (mol/s).

Figure 6.5: Variables with a clear dependency trough a reciprocal function.

Due to these observations, a reciprocal transformation was applied to the dataset, prior to the polynomial transformation. This transformation consists on adding the inverse of each input variable to the dataset. Upon combining the referred transformation with the polynomial one, a model was fitted to a dataset consisting of 231 inputs. The optimal case obtained consisted of a predictive model with 95 components (Table 6.3), which scored higher than the ones obtained previously.

Table 6.3: Model quality for SOFC case study with reciprocal and polynomial transformation.

Nº of components	MSE CV	Q^2	R^2
10	0.1735	0.8265	0.8463
20	0.0750	0.9250	0.9371
40	0.0243	0.9757	0.9856
60	0.0122	0.9878	0.9917
80	0.0101	0.9899	0.9936
90	0.0098	0.9902	0.9938
95	0.0096	0.9904	0.9939
100	0.0102	0.9898	0.9940
120	0.0102	0.9898	0.9942
140	0.0107	0.9893	0.9943
160	0.0108	0.9892	0.9943
180	0.0111	0.9889	0.9944
200	0.0106	0.9894	0.9944
220	0.0104	0.9897	0.9945
230	0.0104	0.9896	0.9945
231	0.0104	0.9896	0.9945

The slice of the validation curve (Figure 6.6) shows the aforementioned proximity between both scores and a low degree of uncertainty regarding the Q^2 score.

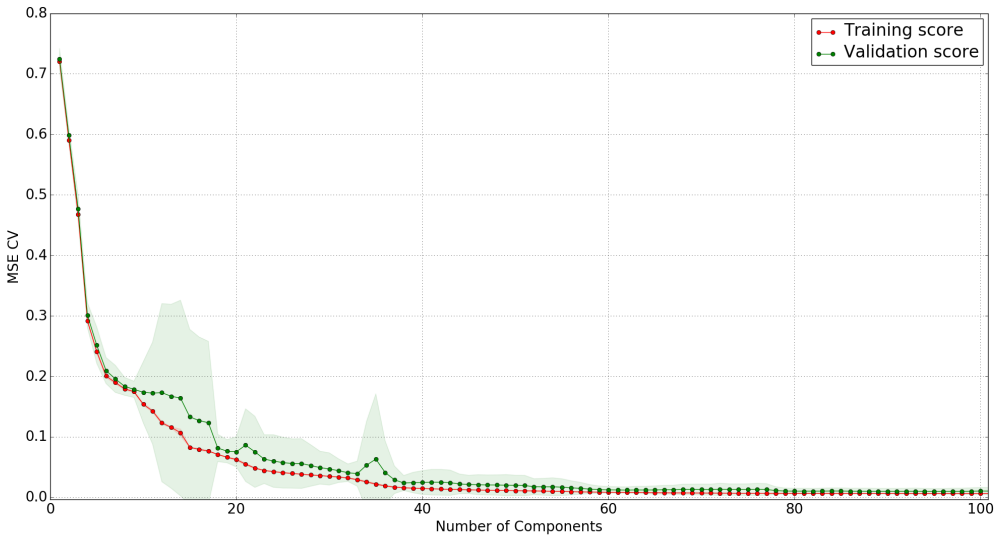
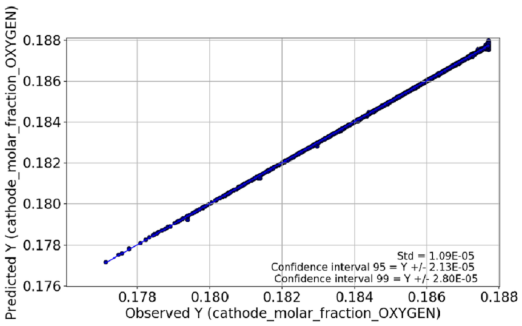


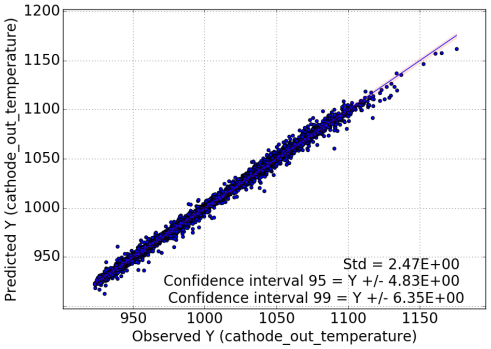
Figure 6.6: Validation curve for the SOFC case study with reciprocal and polynomial transformation.

The plots of the predicted values against the observed ones (Figure 6.7) show significant improve-

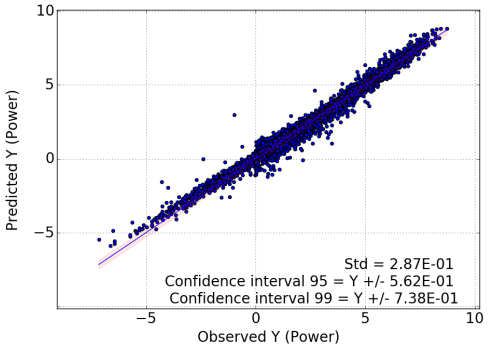
ments from the previously obtained models. This is specially true for the cathode molar fractions who were previously the ones with the worst results and now present all points extremely close to the reference. This is a direct consequence of adding the reciprocal transformation, since two out of the three cathode molar fractions show a distinct reciprocal dependency on one of the inputs. Albeit the power and voltage show a slight improvement, there are still some points with a significant distance to the reference. Regarding the anode molar fractions, the plots show that the predictive capability tends to decrease either for lower values (molar fractions of CO and H_2) or higher values (molar fractions of CO_2 H_2O). Even though these parameters show deviations, their results were considered satisfactory.



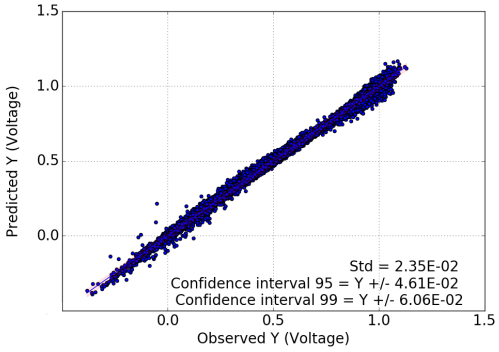
(a) Cathode molar fraction of O_2.



(b) Cathode out temperature (K).



(c) Power (kW).



(d) Voltage (V).

Figure 6.7: Predicted Y vs. Observed Y plot for the SOFC case study with reciprocal and polynomial transformation.

Having a dataset of 10000 observations there were no suspicions of overfitting, which is confirmed by the learning curve (Figure 6.8).

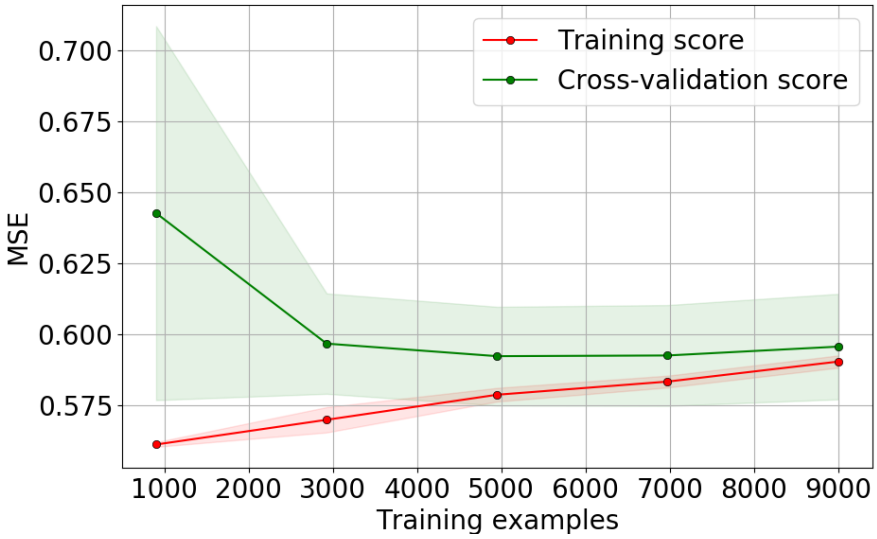
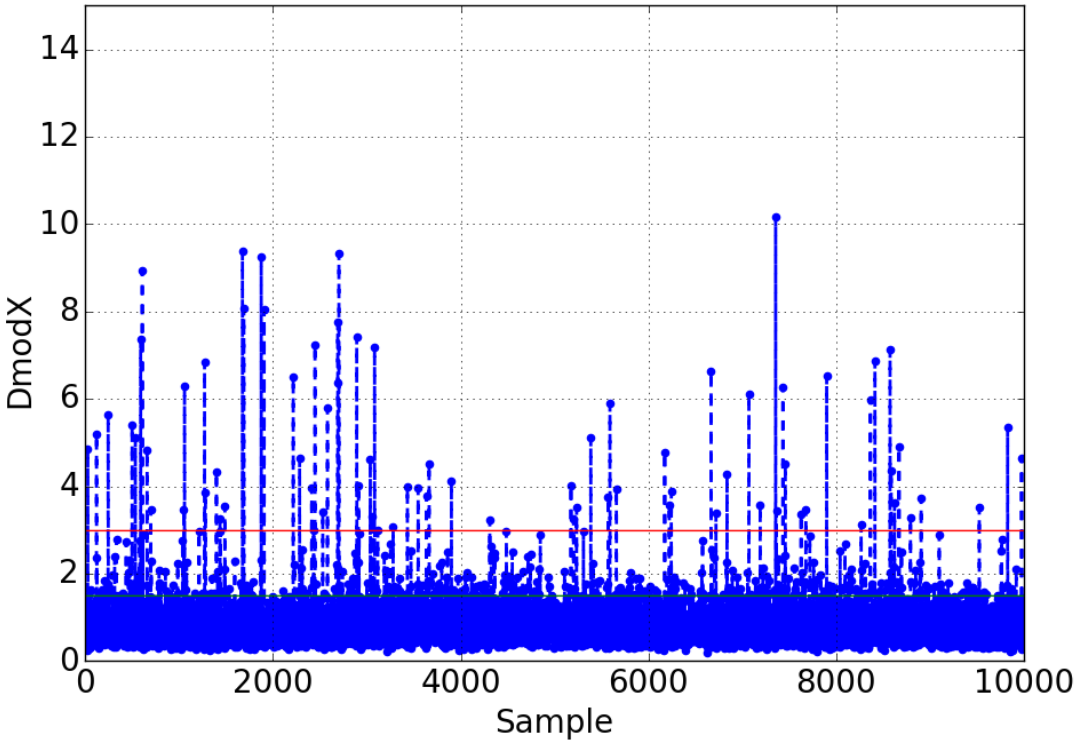


Figure 6.8: Learning curve for the SOFC case study with reciprocal and polynomial transformation.

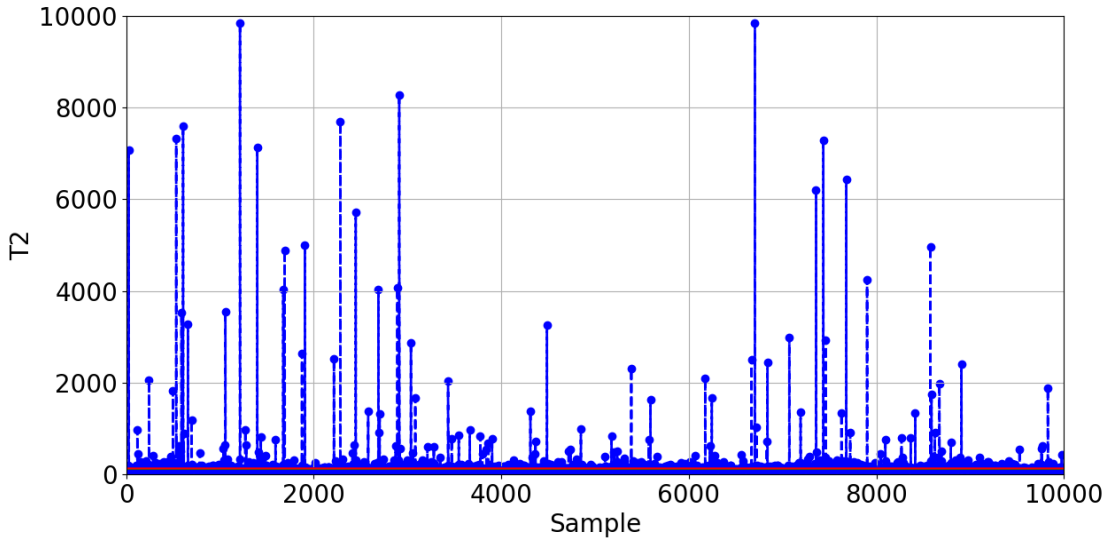
The results obtained show that the tool was able do successfully create a model for the prediction of multiple outputs. However, they also show that, even though the inputs used are known to influence the outputs, the dataset might not have the required complexity for the creation of a good predictive model. Furthermore, they also indicate that the polynomial transformations might not be sufficient to add the necessary complexity to the model, leading to the addition of the reciprocal transformation which allows for the tool to adapt to a greater number of cases.

6.2 Statistical Tests

The distance to the model of an observation (DmodX) and Hotelling's T^2 tests were applied to the dataset (Figure 6.9), evidencing both strong and moderate outliers. Nonetheless, since the data was generated trough a GSA applied to a fuel cell model, all results are explained by the process. Consequently, no changes were made to the data.



(a) DmodX



(b) Hotelling's T^2

Figure 6.9: Statistical tests for the training data of the SOFC model with reciprocal and polynomial transformations.

Chapter 7

Conclusions

This work aimed to develop a prototype tool, in Python, able to create predictive models for the purpose of soft-sensing parameters in systems that are usually not modelled. This happens either because they are poorly understood and, therefore, first principle models cannot be developed, or because they present non-linear relations between different parameters that make them extremely hard to model trough first principal models.

The literature review showed that the partial least squares regression (PLSR) was the algorithm best suited for the problems tackled in this work, in detriment of the multiple linear regression (MLR), used for simpler multivariate cases.

The tool's ability to create predictive partial least squares (PLS) models was validated against a model for a granulation case, generated by SIMCA®, a statistical software that allows for the creation of PLS models, trough plant data. The validated model was subsequently loaded into gPROMS FormulatedProducts®, trough an internally developed foreign object, being able to successfully soft-sense the intended parameter, the flow-function coefficient (FFC). The simulation's results led to the conclusion that one of the most important steps in creating a predictive model is to choose the training data carefully. Even though the model was able to predict the FFC, the simulation was made with operational conditions different from those used to generate the training data. This leads to the calculation of predictions outside the range the model was intended to be used in, generating unreliable results. Hence, when using this tool, the training data used to fit the model must be generated in a range of operational conditions that cover the ones used in the simulations, thus assuring the reliability of the predictions.

Being that the problems tackled have unknown relations between its parameters, the training data loaded into the tool might not always posses the required complexity to successfully create a good PLSR model, as evidenced by the compression case. Therefore, to widen the range of applicability of the tool, a 2^{nd} order polynomial transformation option was added. With this option, it was possible to increase the complexity of the training data through a commonly used transformation whenever the tool fails to get a good predictive model from the raw input data. The need to apply a transformation is one of the main limitations of this tool, since certain problems may require the application of transformations that aren't commonly used and, therefore, are not implemented.

The decision that the tool required training data with higher complexity was also shown in the solid oxide fuel cell (SOFC) case study. In this case, even after applying the polynomial transformation, the tool failed to get a proper predictive model. This was corrected by adding the option to apply a reciprocal transformation, which can be combined with the previously implemented polynomial transformation, further increasing the tool's applicability to different cases. Additionally, this case led to the conclusion that the PLSR was able to successfully create a single predictive model for several response variables.

7.1 Future Work

In what concerns the tool itself, it still has room for improvement, specially regarding data transformations. As it is applied to other cases, further transformations can be implemented, not only to the predictor variables, but also to the response variables.

Regarding the models integration in dynamic simulations, since it has been proved that it is able to soft-sense certain parameters, it can be integrated into the simulation itself. This would be accomplished by using the calculated predictions as inputs for other models.

Bibliography

[1] P. Kadlec, B. Gabrys, and S. Strandt, "Data-driven Soft Sensors in the process industry," *Computers and Chemical Engineering*, vol. 33, no. 4, pp. 795–814, 2009.

[2] S. Zendehboudi, N. Rezaei, and A. Lohi, "Applications of hybrid models in chemical, petroleum, and energy systems: A systematic review," *Applied Energy*, vol. 228, no. December 2017, pp. 2539–2566, 2018. [Online]. Available: https://doi.org/10.1016/j.apenergy.2018.06.051

[3] G. James, D. Witen, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning with Applications in R*, 2007, vol. 64, no. 9-12.

[4] M. N. O. Sadiku, S. M. Musa, O. M. Musa, and R. G. Perry, "Machine Learning in Chemical Industry," *International Journal of Advances In Scientific Research and Engineering (IJASRE)*, vol. 3, no. 10, pp. 12–15, 2017.

[5] Z. Ge, Z. Song, S. X. Ding, and B. Huang, "Data Mining and Analytics in the Process Industry: The Role of Machine Learning," *IEEE Access*, vol. 5, pp. 20 590–20 616, 2017.

[6] T. Wuest, D. Weimer, C. Irgens, and K.-D. Thoben, "Machine learning in manufacturing: advantages, challenges, and applications," *Production & Manufacturing Research*, vol. 4, no. 1, pp. 23–45, 2016. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/21693277.2016.1192517

[7] B. Lin, B. Recke, J. K. H. Knudsen, and S. B. Jørgensen, "A systematic approach for soft sensor development," *Computers and Chemical Engineering*, vol. 31, no. 5-6, pp. 419–425, 2007.

[8] R. D. Tobias, "An introduction to partial least squares regression," *SAS Conference Proceedings: SAS Users Group International 20 (SUGI 20)*, pp. 2–5, 1995.

[9] S. Wold, M. Sjöström, and L. Eriksson, "PLS-regression: A basic tool of chemometrics," *Chemometrics and Intelligent Laboratory Systems*, vol. 58, no. 2, pp. 109–130, 2001.

[10] F. Pedregosa, 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 É. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2078195{%}5Cnhttp://arxiv.org/abs/1201.0490

[11] J. Wegelin, "A survey of Partial Least Squares (PLS) methods, with emphasis on the two-block case," p. 371, 2000. [Online]. Available: https://www.stat.washington.edu/www/research/reports/2000/tr371.pdf

[12] T. Hasegawa, "Principal Component Regression and Partial Least Squares Modeling," 2006.

[13] V. Consonni, D. Ballabio, and R. Todeschini, "Evaluation of model predictive ability by external validation techniques," *Journal of Chemometrics*, vol. 24, no. 3-4, pp. 194–201, 2010.

[14] S. Jun, “Linear model selection by cross-validation,” *Journal of the American Statistical Association*, vol. 128, no. 1, pp. 231–240, 1993.

[15] K. Dunn, “Process Improvement Using Data,” no. July, p. 408, 2018.

[16] S. stedim Biotech, “Simca® 15 User Guide - Multivariate Data Analysis Solution.”

[17] G. K. Sofer and A. S. Rathore, *Process Validation in Manufacturing of Biopharmaceuticals*, 3rd ed.

[18] Python Software Foundation, “Python™,” accessed 2018-08-14. [Online]. Available: https://www.python.org/

[19] Process Systems Enterprise Ltd., “PSE: gPROMS - The Platform,” accessed 2018-06-12. [Online]. Available: https://www.psenterprise.com/products/gproms/platform

[20] Process Systems Enterprise, “gPROMS ModelBuilder Documentation - Release 4.2.1,” Tech. Rep., 2016. [Online]. Available: https://www.psenterprise.com/sites/default/files/documentation/il/webhelp/Help.htm

[21] “SIMCA — Umetrics,” accessed 2018-08-15. [Online]. Available: https://umetrics.com/products/simca

[22] “ADDoPT – advanced digital design transforming pharmaceutical development and manufacture,” accessed 2018-07-09. [Online]. Available: https://www.addopt.org/about{_}addopt/

[23] M. Alderliesten, “Mean Particle Diameters. Part II: Standardization of nomenclature,” *Particle & Particle Systems Characterization*, vol. 8, no. 1-4, pp. 237–241, 1991.

[24] C. Feng, H. Wang, N. Lu, T. Chen, H. He, Y. Lu, and X. M. Tu, “Log-transformation and its implications for data analysis.” *Shanghai archives of psychiatry*, vol. 26, no. 2, pp. 105–9, 2014. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=4120293{&}tool=pmcentrez{&}rendertype=abstract{%}5Cnhttp://www.ncbi.nlm.nih.gov/pubmed/25092958{%}5Cnhttp://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4120293

[25] University of California Irvine, “BlogFeedback Data Set,” 2014, accessed 2018-08-08. [Online]. Available: http://archive.ics.uci.edu/ml/datasets/BlogFeedback

[26] A. Michrafy, D. Ringenbacher, and P. Tchoreloff, “Modelling the compaction behaviour of powders: Application to pharmaceutical powders,” *Powder Technology*, vol. 127, no. 3, pp. 257–266, 2002.

Appendix A

Granulation case study data

A.1 Granulation case input data

In this particular particle size distribution, the particles of a granules powder where divided according to their size, with dimensions ranging from 7.5 μm to 2107.37 μm. The volume fraction regarding each size was recorded, and, with it, the parameters shown in the following table were calculated.

Table A.1: Particle size distribution treated data

D10	D50	D90	D43	FFC	Log(FFC)
86.00	733.67	1227.73	692.27	8.55	0.93
65.85	725.29	1237.94	679.17	5.85	0.77
81.70	860.01	1338.85	781.70	8.55	0.93
75.86	731.36	1222.98	682.92	7.10	0.85
85.64	741.29	1286.62	720.21	9.55	0.98
78.46	752.74	1237.17	704.39	6.55	0.82
82.71	857.07	1380.94	791.28	7.30	0.86
72.69	654.31	1228.08	651.40	8.10	0.91
77.56	652.12	1220.21	651.39	8.70	0.94
79.32	675.39	1230.51	663.13	9.85	0.99
64.14	688.65	1288.00	679.85	5.90	0.77
78.67	743.58	1282.95	708.26	7.70	0.89
58.67	665.33	1258.92	655.94	5.40	0.73
73.39	698.68	1256.81	684.83	7.35	0.87
55.10	476.28	1144.07	538.02	5.90	0.77
88.21	796.02	1281.39	740.10	8.95	0.95
69.99	725.83	1236.01	686.21	7.15	0.85
70.10	633.53	1200.73	631.35	8.10	0.91
81.44	772.80	1283.27	728.73	7.55	0.88

A.2 Simulation data

In order to understand why the statistical tests fail during the simulation, it's important to look at the input variables for the model and compare them with the training data of the model. Input values higher

or lower than the ones used for the fitting of the model can result in greater distances to the model plane/ to the centre of the model, triggering simulation warnings.

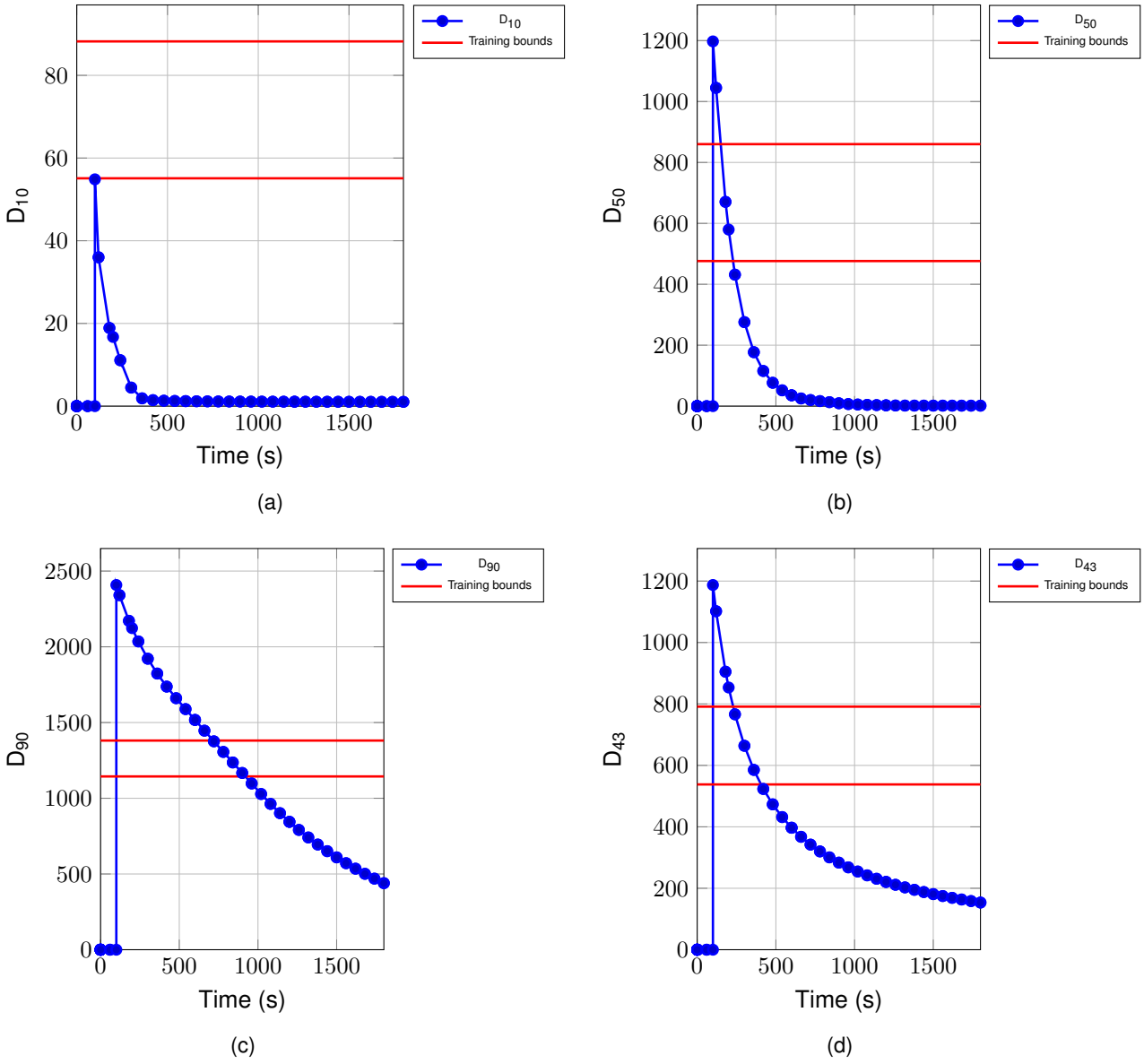


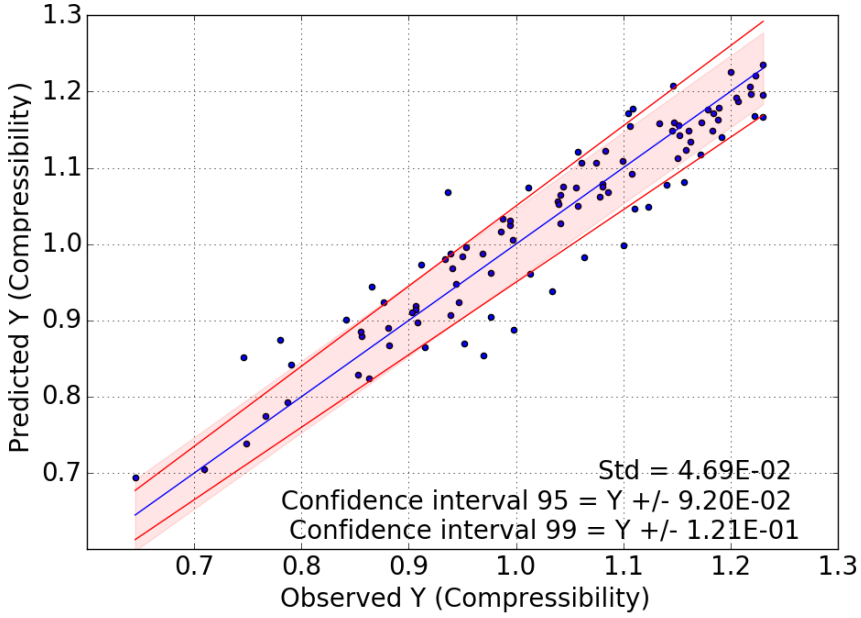
Figure A.1: Data based sensor's inputs,in μm, during simulation and respective training bounds

Appendix B

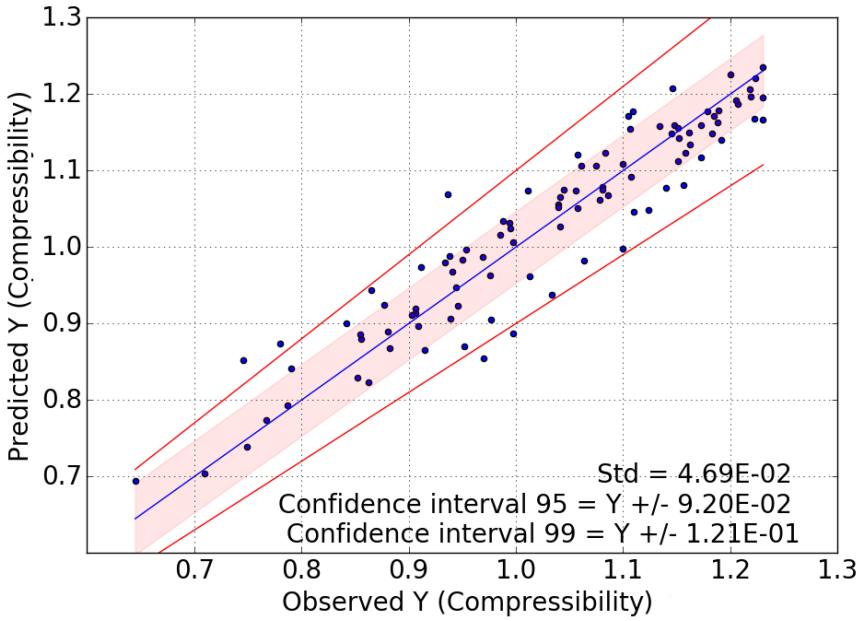
Compression Case Study Data

B.1 Compression model with polynomial transformation

Even though the model fitted to a polynomially transformed data shows improvements in the overall quality of predictions, some points in the plots of the predicted values against the observed ones still land outside the uncertainty band with a visually considerable distance to it. In order to better judge this deviation, lines referring to a deviation (positive and negative) of 5% and 10% were added to the plot.



(a)

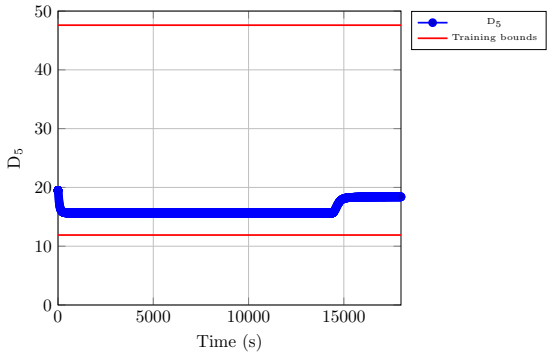


(b)

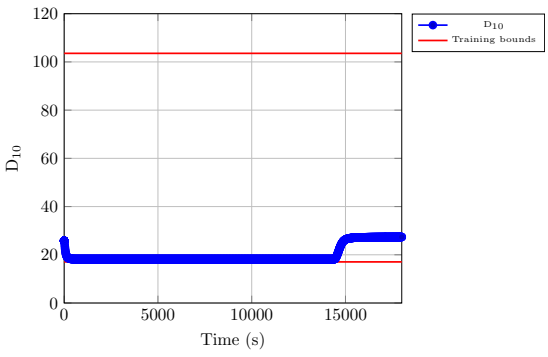
Figure B.1: Predicted Y vs. Observed Y plot for the compression case study with polynomial transformation with 5% (a) and 10% deviation reference (b).

B.2 Simulation Data

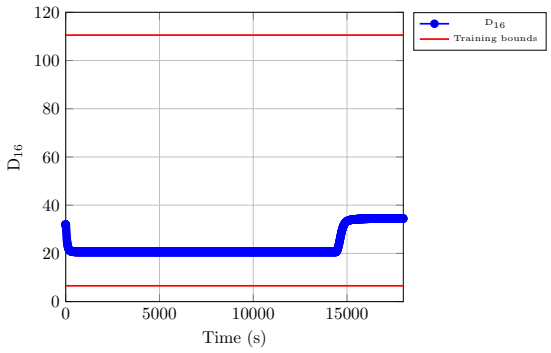
Upon running the simulation, warnings were triggered because the statistical tests present higher distances than the critical ones. To better understand why this happens, it helps to look at the values each input had during the simulation and compare them with the maximum and minimum values, of the respective parameter, in the training data set.



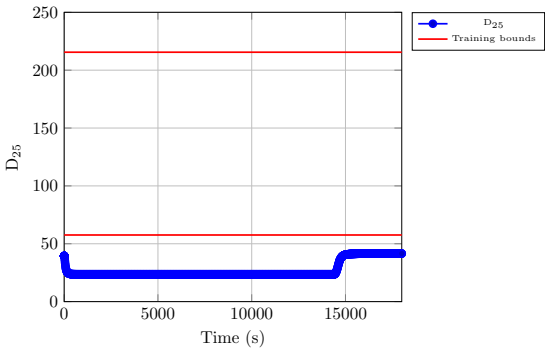
(a)



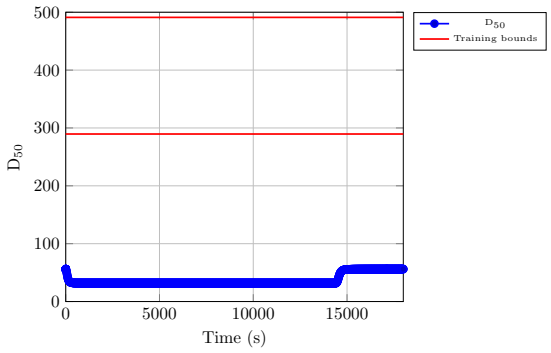
(b)


D16
Training bounds
D16
Time (s)

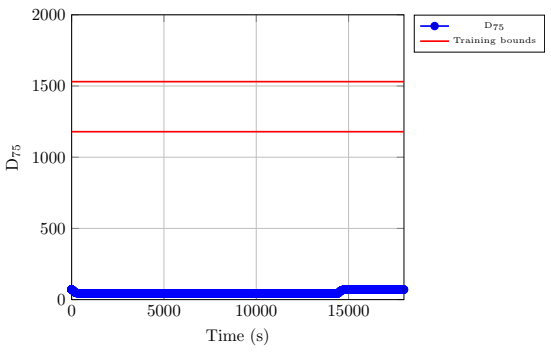
(c)


D25
Training bounds
D25
Time (s)

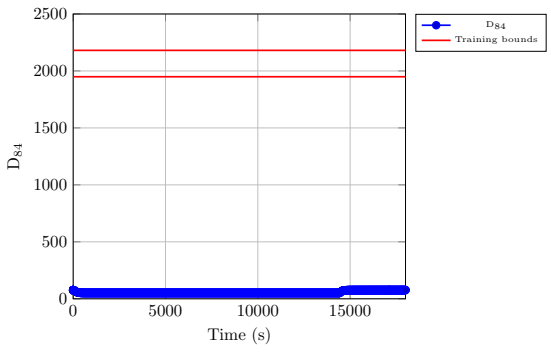
(d)


D50
Training bounds
D50
Time (s)

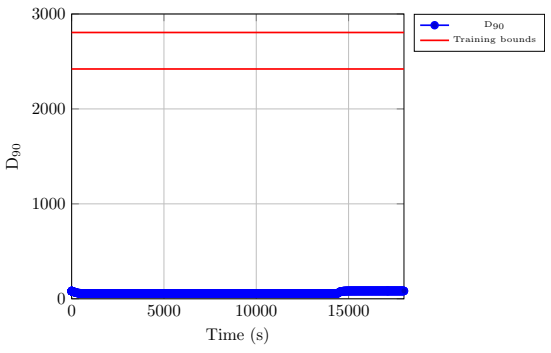
(e)


D75
Training bounds
D75
Time (s)

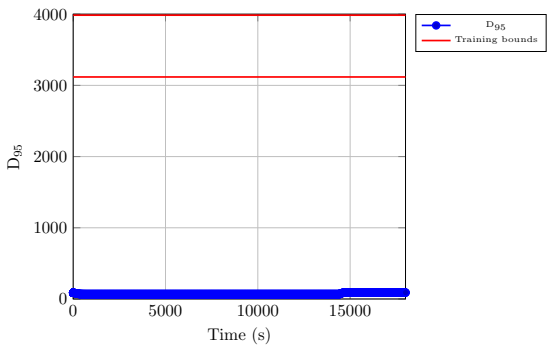
(f)


D84
Training bounds
D84
Time (s)

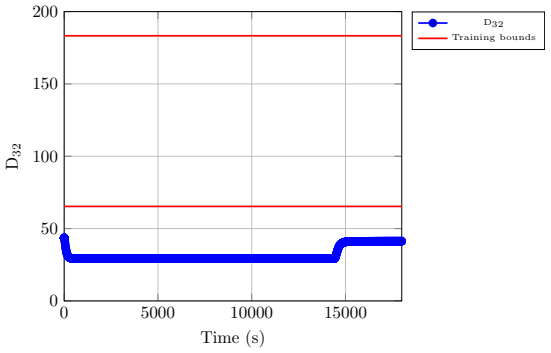
(g)


D90
Training bounds
D90
Time (s)

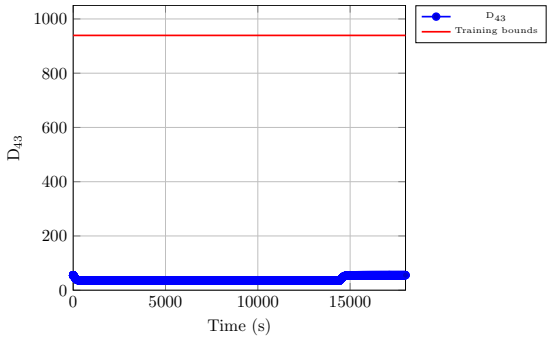
(h)


D95
Training bounds
D95
Time (s)

(i)


D32
Training bounds
D32
Time (s)

(j)



(k)

Figure B.2: Data based sensor's inputs, μm, during simulation and respective training bounds

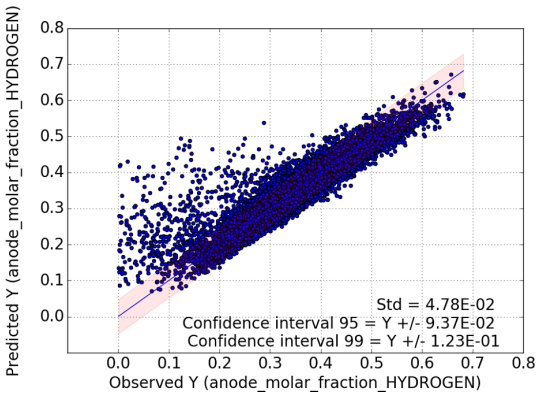
It should be noted that for D_{10} the lower bound for the training data is 17.05, which is never breached since the minimum value assumed by this parameter during the simulation is 18.22. Furthermore, for the D_{43} parameter, the upper and lower bounds for the training data are 1094.22 and 939.33, respectively. The plot only shows one bound for this parameter due to the proximity of these values and the scaling used.

Appendix C

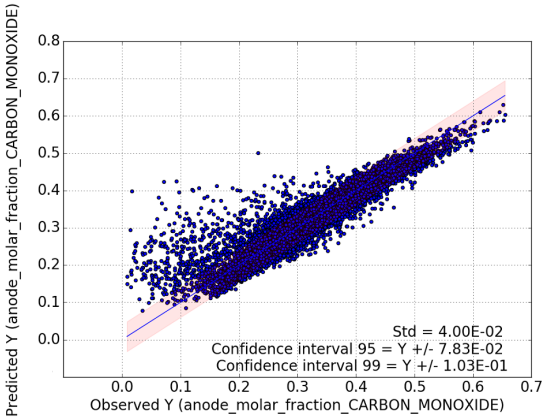
Solid Oxide Fuel Cells Case Study Data

C.1 Solid Oxide Fuel Cells model with no transformations

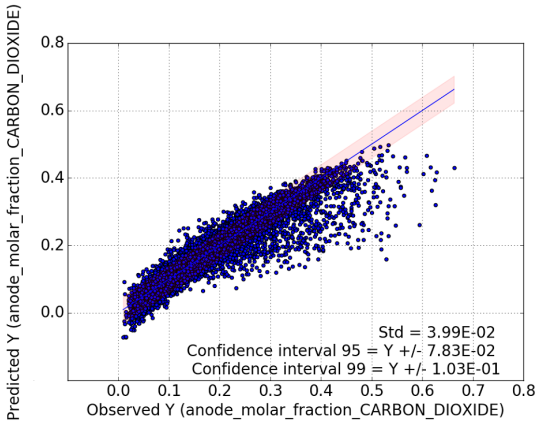
It was observed that the plots of the predictions against observations showed similar behaviours for the same type of parameters, i.e the molar fractions and the temperatures showed similar behaviours. Therefore, only the most relevant plots were shown, being that the remainder of the plots are present in Figure C.1.



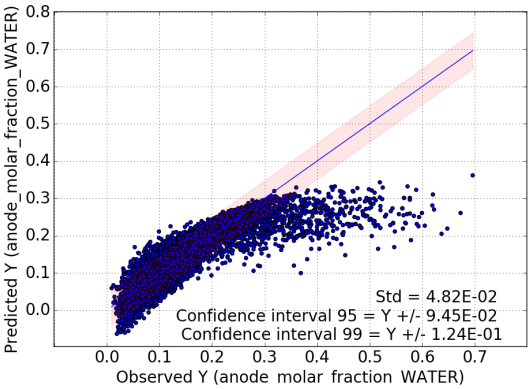
(a) Anode molar fraction of H_2.



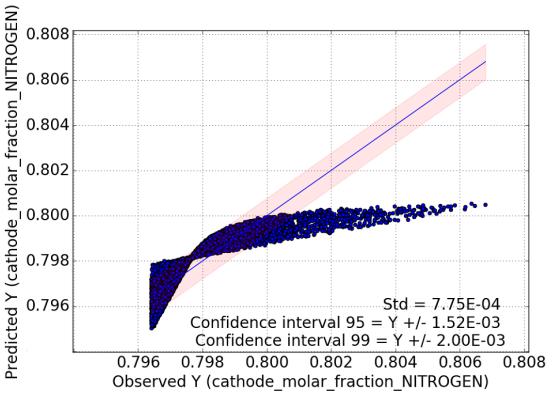
(b) Anode molar fraction of CO.



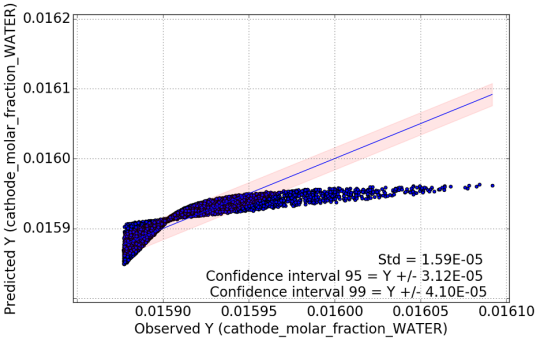
(c) Anode molar fraction of CO_2.



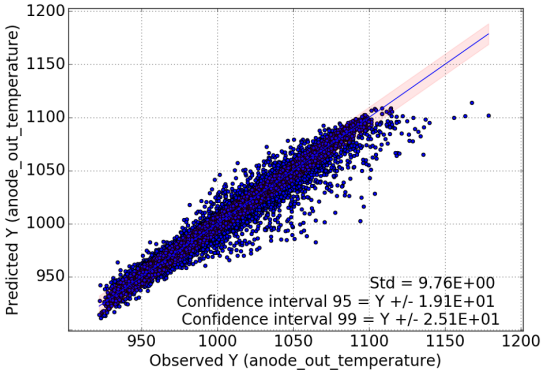
(d) Anode molar fraction of H_2O.



(e) Cathode molar fraction of N_2.



(f) Cathode molar fraction of H_2O.

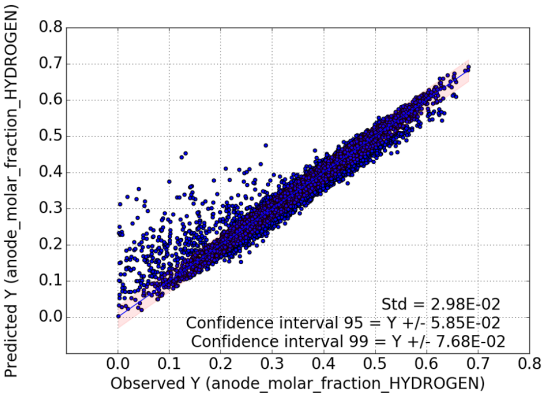


(g) Anode out temperature (K).

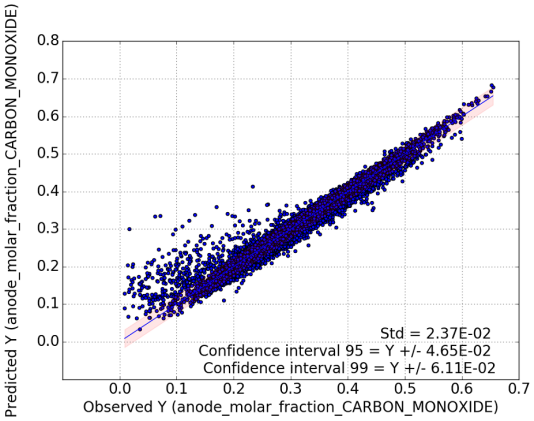
Figure C.1: Predicted Y vs. Observed Y plot for the SOFC case study.

C.2 Solid Oxide Fuel Cells model with polynomial transformation

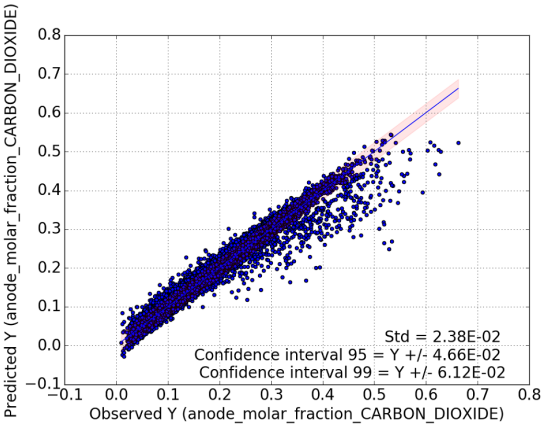
The remainder of the plots of the predicted values for the training data and its real values, for the case with the polynomial transformation are presented in Figure C.2.



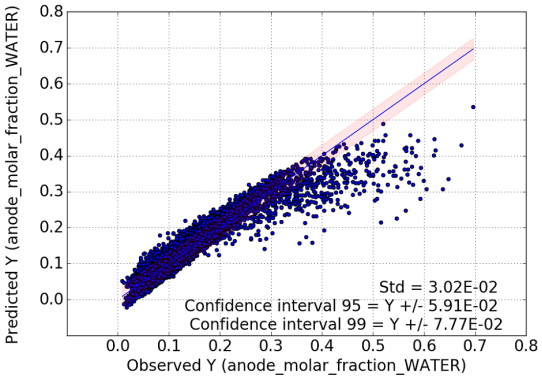
(a) Anode molar fraction of H_2.



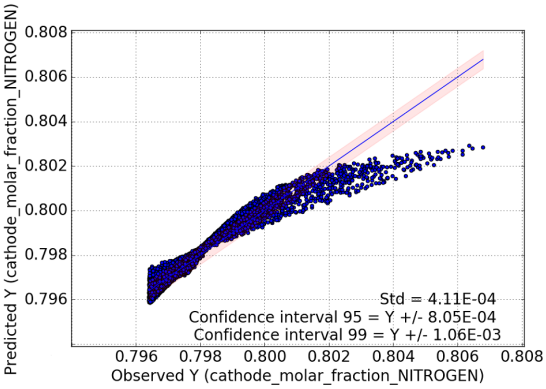
(b) Anode molar fraction of CO.



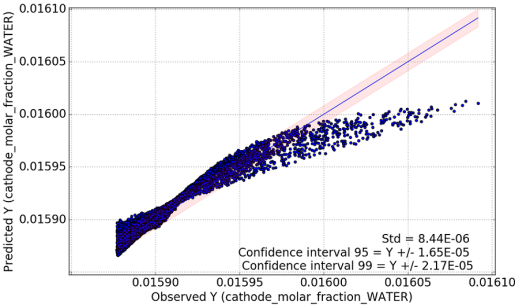
(c) Anode molar fraction of CO_2.



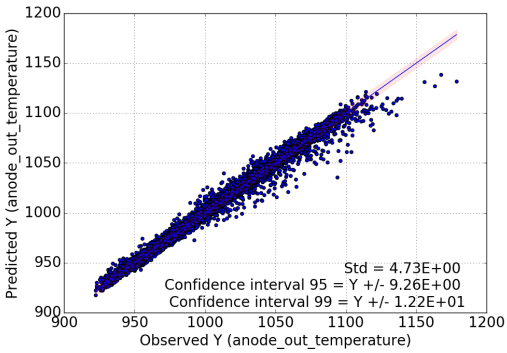
(d) Anode molar fraction of H_2O.



(e) Cathode molar fraction of N_2.



(f) Cathode molar fraction of H_2O.

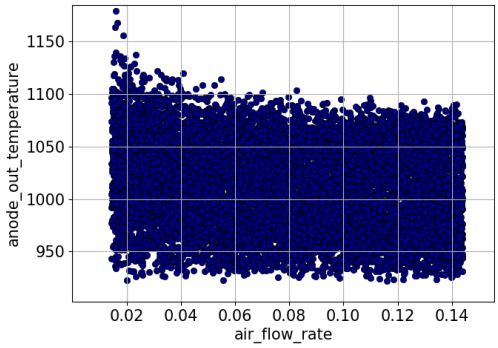


(g) Anode out temperature (K).

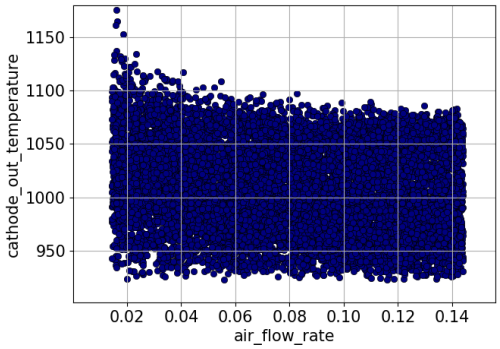
Figure C.2: Predicted Y vs. Observed Y plot for the SOFC case study with polynomial transformation.

C.3 Solid Oxide Fuel Cells model with reciprocal and polynomial transformations

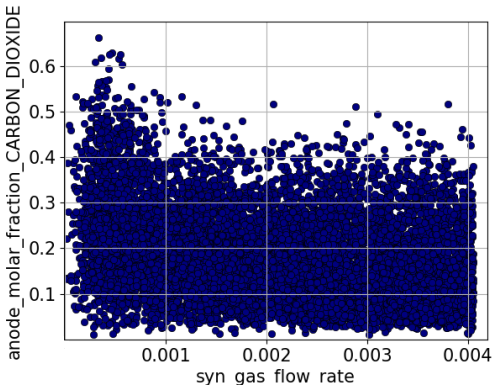
The high number of observations in the dataset makes it harder to asses, in some cases, how the ouputs vary with the inputs. The cases where the outputs are suspected to have a reciprocal dependency on the inputs are shown in Figure C.3



(a) Anode out temperature Vs.Air flow rate.



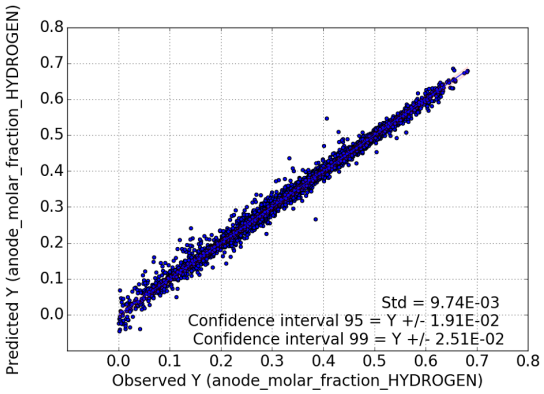
(b) Cathode out temperature Vs. Air flow rate.



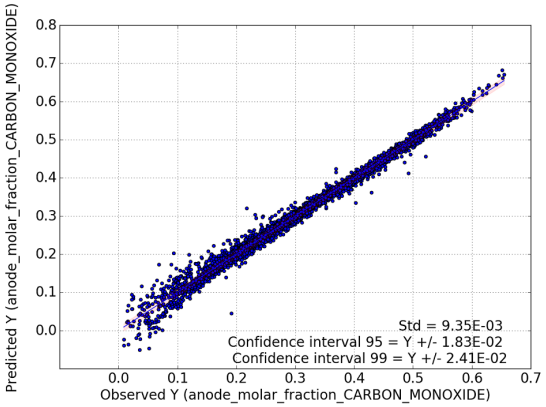
(c) Anode molar fraction of CO2 Vs. Syngas flow rate.

Figure C.3: Variables with a suspected dependency trough a reciprocal function.

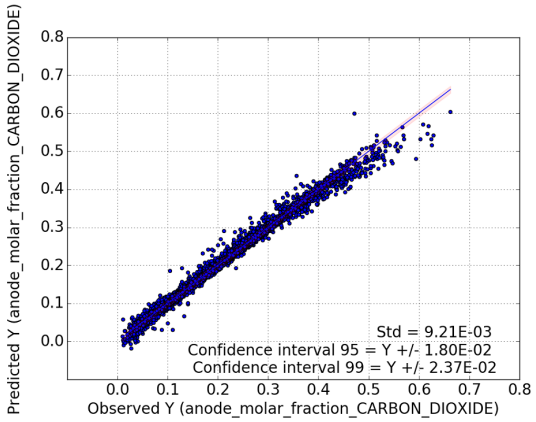
The remainder of the plots of the predicted values for the training data and its real values, for the case with the reciprocal transformation followed by the polynomial transformation are presented in Figure C.4.



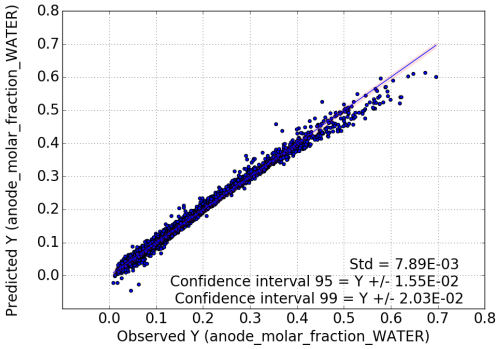
(a) Anode molar fraction of H_2.



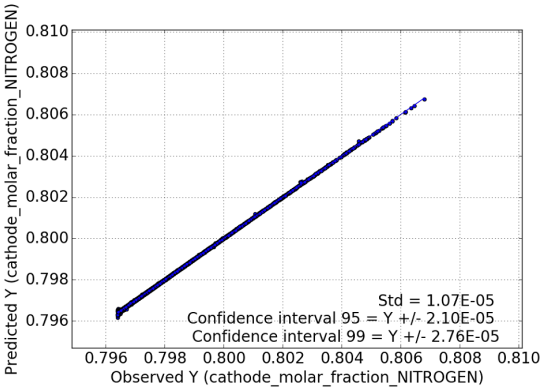
(b) Anode molar fraction of CO.



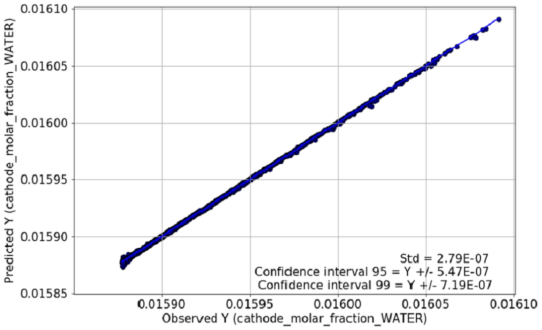
(c) Anode molar fraction of CO_2.



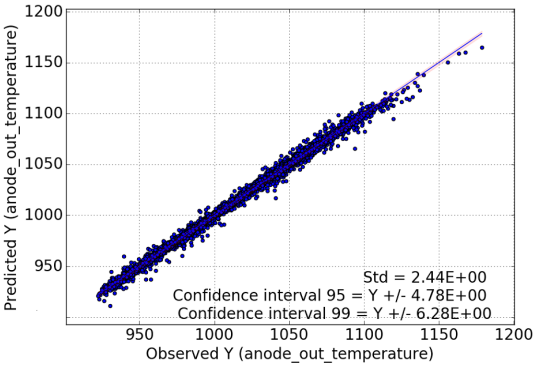
(d) Anode molar fraction of H_2O.



(e) Cathode molar fraction of N_2.



(f) Cathode molar fraction of H_2O.



(g) Anode out temperature (K).

Figure C.4: Predicted Y vs. Observed Y plot for the SOFC case study with reciprocal and polynomial transformations.