

Integrated Master in Chemical Engineering

Modelling of a PSA unit by artificial neural networks

A Master's dissertation

of

Luís Miguel Cunha Oliveira

Developed within the course of Dissertation

held in

LSRE - LCM - Laboratory of Separation and Reaction Engineering - Laboratory of
Catalysis and Materials



Supervisors: Doutora Ana Mafalda Ribeiro

Doutor Idelfonso Nogueira

Professor José Miguel Loureiro



Department of Chemical Engineering

July 2019

Acknowledgements

In this section, I would like to give my thanks to everyone who was in some way involved in the making of this thesis.

Firstly, I would like to thank LSRE-LCM for providing me with the opportunity to work on a subject that interested me as soon as I read about it.

I want to thank my supervisors Ana Mafalda Ribeiro and Idelfonso Nogueira for their patience, support and valuable input for this work. Facing such a daunting task as writing this dissertation gets a lot less difficult when you have people you can count on to help you in times of need.

To my family, thank you for your unending support, which has always meant the world to me, even though it may not always seem like it.

To all my friends from FEUP, thank you for making my university experience not only bearable, but also enjoyable. These have been the best years of my life, and I have no intention of forgetting any of the great moments that I've lived here.

This work was financially supported by: Associate Laboratory LSRE-LCM - UID/EQU/50020/2019
- funded by national funds through FCT/MCTES (PIDDAC).



Abstract

Adsorption processes are considered to be a viable alternative for pretreatment of syngas, mainly due to their low energetic requirements. One of those processes, pressure swing adsorption (PSA), is employed for that purpose. However, it displays very complex dynamical behavior, which is usually simulated by phenomenological models. These models are often cumbersome, which makes them unsuitable for real-time process assessment. For that purpose, there is a need to search for alternative models; artificial neural networks (ANN) arose as a suitable alternative, due to their versatility.

This work attempts to develop different neural network models for prediction of a number of process output variables. A classical model of ANNs (FNN - Feedforward Neural Networks, based on a NARX - nonlinear autoregressive with exogenous inputs predictor), a machine learning model (RNN - Recurrent Neural Networks, based on a NOE - nonlinear output error predictor) and a deep learning model (LSTM - long short-term memory, based on a NOE - nonlinear output error predictor) were studied and their performances compared. Firstly, the sets of simulation data needed for the training and validation of each network were generated. Then, all the models were trained and validated.

It was found that LSTM networks were the only ones capable of fully representing the dynamic behavior of the PSA unit, whereas the other models were only partially capable of predicting it. Between the classical model and the machine learning, the former performed better compared to the latter, which displayed inconsistent results, putting its predictive capabilities into question. A comparison was also established between the empirical models and the phenomenological ones in terms of computational speed, and the former achieved dramatically lower simulation times than the latter.

It is therefore concluded that the LSTM models can be reliable predictors of the process' dynamical behaviour, and may be a good option for the development of control, optimization and on-line measurement strategies. On the other hand, the classical and machine learning models showed considerable limitations in handling the PSA unit's complex dynamic behaviour.

Keywords: artificial neural networks, artificial intelligence, deep learning, machine learning, pressure swing adsorption

Resumo

Os processos de adsorção são considerados alternativas viáveis para o pré-tratamento de gás de síntese, principalmente devido aos seus reduzidos custos energéticos. Um desses processos, adsorção por modulação de pressão (PSA), é usado para essa função. Contudo, o PSA demonstra um comportamento dinâmico bastante complexo, que é normalmente simulado por modelos fenomenológicos. Estes modelos são complexos e requerem elevado esforço e tempo computacional, o que os torna inapropriados para avaliação de processos em tempo real. Nesse sentido, é necessário procurar modelos alternativos; as redes neuronais artificiais (RNA) surgiram como uma alternativa adequada, devido à sua versatilidade.

Este trabalho propõe desenvolver diferentes modelos de redes neuronais com o intuito de efetuar previsões de algumas variáveis de saída do processo. Um modelo clássico de RNA's (FNN - Feedforward Neural Networks, based on a NARX - nonlinear autoregressive with exogenous inputs predictor), um modelo de machine learning (RNN - Recurrent Neural Networks, based on a NOE - nonlinear output error predictor) e um modelo de deep learning (LSTM - long short-term memory, based on a NOE - nonlinear output error predictor) foram estudados e os seus desempenhos foram comparados entre si. Em primeiro lugar, os conjuntos de dados necessários para o treino e validação de cada rede foram gerados. De seguida, todos os modelos de rede foram treinados e validados.


Foi observado que as redes LSTM foram as únicas capazes de representar o comportamento dinâmico do PSA na sua totalidade, enquanto que os restantes modelos apenas foram capazes de o prever de forma parcial. Entre o modelo clássico e o de *machine learning*, o primeiro apresentou melhor desempenho que o segundo, que obteve resultados inconsistentes, colocando as suas capacidades preditivas em causa. Os modelos fenomenológicos e empíricos foram comparados em termos de velocidade computacional e os primeiros demonstraram tempos de simulação drasticamente mais baixos.

Conclui-se então que os modelos LSTM podem ser previsores adequados do comportamento dinâmico do processo, e podem ser uma boa opção para o desenvolvimento de estratégias de controlo, otimização e medição em tempo real. Por outro lado, os modelos clássicos e de *machine learning* revelaram limitações consideráveis em lidar com o complexo comportamento dinâmico do PSA.

Palavras-chave: redes neuronais artificiais, inteligência artificial, *deep learning*, *machine learning*, adsorção por modulação de pressão

Declaração

Declara, sob compromisso de honra, que este trabalho é original e que todas as contribuições não originais foram devidamente referenciadas com identificação da fonte.



(Luís Miguel Cunha Oliveira)

Porto, 1 de Julho de 2019

Table of Contents

1	Introduction	1
1.1	Motivation and Relevance	1
1.2	Outline	2
2	State of the Art	3
2.1	Syngas	3
2.2	Pressure swing adsorption.....	3
2.3	Artificial Neural Networks.....	4
2.4	Evolution of ANN and Deep Learning.....	6
3	Materials and Methods.....	11
3.1	Mathematical model	11
3.2	Predictors.....	14
3.2.1	Nonlinear Autoregressive with Exogenous Inputs (NARX).....	14
3.2.2	Nonlinear Output Error (NOE)	15
3.3	Neural Network Models	16
3.3.1	Feedforward Neural Networks (FNN)	17
3.3.2	Recurrent Neural Networks (RNN)	18
3.3.3	Neural Network Training	19
3.3.4	Neural Network Validation	21
3.3.5	Selection of the optimal number of neurons	22
3.3.6	Neural Network Pruning	22
3.4	Long Short-Term Memory (LSTM)	23
3.5	Data acquisition procedure	24
4	Results and Discussion.....	25
4.1	Process Simulation	25
4.2	Data acquisition	27
4.3	Network training and validation - classical model	31
4.3.1	NARX - model identification	31
4.4	Network training and validation - Machine Learning strategy.....	33

4.4.1 NOE - model identification.....	33
4.5 Network training - Long Short-Term Memory model	34
4.6 Comparison and model validation	35
4.7 Phenomenological model vs Empirical model	40
5 Conclusions	41
6 References.....	43
Appendix A1. Validation results for each network model	47
A1.1 NARX	47
A1.2 NOE.....	49
A1.3 LSTM	51

List of Figures

Figure 1 - Comparison between a biological neuron and an artificial neuron	5
Figure 2 - Historical evolution of the use of AI in chemical engineering.	7
Figure 3 - Relationship between the concepts of artificial intelligence, machine learning and deep learning.....	9
Figure 4 - NARX model with $na = 1$, $nb = 1$ and $d = 1$	15
Figure 5 - NOE model with $na = 1$, $nb = 1$ and $d = 1$	16
Figure 6 - Example of Multilayer Perceptron Network with 2 inputs, 3 hidden units and a single output unit.....	18
Figure 7 - Multilayer perceptron with 2 inputs, 3 hidden units and a single output.	19
Figure 8 - Summary of the method used for developing artificial neural networks.....	22
Figure 9 - Representation of a LSTM unit.	23
Figure 10 - Scheme of the PSA process studied in this work.	25
Figure 11 - Generated inputs for a) t_{feed} , b) t_{purge} , c) t_{rinse} , d) $Phigh$, e) $Plow$, f) $Qrinse$, g) $Qpurge$ and h) $Tinlet$ to use for training the neural networks.	29
Figure 12 - Generated inputs for a) t_{feed} , b) t_{purge} , c) t_{rinse} , d) $Phigh$, e) $Plow$, f) $Qrinse$, g) $Qpurge$ and h) $Tinlet$ to use for validating the neural networks.	31
Figure 13 - Identification of the optimal number of neurons for the NARX networks: a) H_2CO , b) $PurCO_2$, c) $RecCO_2$, d) $ProdCO_2$	32
Figure 14 - Identification of the optimal number of neurons for the NOE networks: a) H_2CO , b) $PurCO_2$, c) $RecCO_2$, d) $ProdCO_2$	33
Figure 15 - Comparison of the different models for H_2/CO ratio, for the strong dynamics region (above) and the smooth dynamics region (below).....	36
Figure 16 - Comparison of the different networks for CO_2 purity, for the strong dynamics region (above) and the smooth dynamics region (below).....	37
Figure 17 - Comparison of the different models for CO_2 recovery, for the strong dynamics region (above) and the smooth dynamics region (below).....	38
Figure 18 - Comparison of the different models for CO_2 productivity, for the strong dynamics region (above) and the smooth dynamics region (below).....	40
Figure A1.1 - Validation results for the H_2/CO ratio (NARX model).	47
Figure A1.2 - Validation results for the CO_2 purity (NARX model).	47
Figure A1.3 - Validation results for the CO_2 recovery (NARX model).....	48
Figure A1.4 - Validation results for the CO_2 productivity (NARX model).....	48
Figure A1.5 - Validation results for the H_2/CO ratio (NOE model).....	49
Figure A1.6 - Validation results for the CO_2 purity (NOE model).	49

Figure A1.7 - Validation results for the CO₂ recovery (NOE model). 50

Figure A1.8 - Validation results for the CO₂ productivity (NOE model). 50

Figure A1.9 - Validation results for the H₂/CO ratio (LSTM model). 51

Figure A1.10 - Validation results for the CO₂ purity (LSTM model). 51

Figure A1.11 - Validation results for the CO₂ recovery (LSTM model)..... 52

Figure A1.12 - Validation results for the CO₂ productivity (LSTM model). 52

List of Tables

Table 1 - Boundary conditions for the PSA model.....	13
Table 2 - Parameters used for the process' adsorbent bed.	26
Table 3 - Transport parameters used in the process' mathematical model.	26
Table 4 - Operating conditions of the PSA process.	26
Table 5 - Simulation results of the PSA process.	27
Table 6 - Upper and lower bounds for the intervals of the generated training samples.	28
Table 7 - Upper and lower bounds for the intervals of the generated validation samples.	29
Table 8 - Training parameters for LSTM networks.	34
Table 9 - Number of LSTM units in the hidden layers of each network.	35

Notation and Glossary

a_p	Particle external specific area	$\text{m}^2 \text{g}^{-1}$
b_i	Neural network biases	
$C_{g,T}$	Total gas phase concentration	mol m^{-3}
$C_{g,i}$	Gas phase concentration of component i	mol m^{-3}
C_p	Heat capacity of the mixture at constant pressure	$\text{J mol}^{-1} \text{K}^{-1}$
$\hat{C}_{p,s}$	Solid specific heat (per mass unit)	$\text{J kg}^{-1} \text{K}^{-1}$
$\hat{C}_{p,w}$	Wall specific heat (per mass unit)	$\text{J kg}^{-1} \text{K}^{-1}$
$C_{s,i}$	Concentration at the solid interface of component i	mol m^{-3}
C_v	Heat capacity of the mixture at constant volume	$\text{J mol}^{-1} \text{K}^{-1}$
d	Input delay	
d_p	Particle diameter	m
d_{wi}	Wall internal diameter	m
D_{ax}	Mass axial dispersion coefficient	$\text{m}^2 \text{s}^{-1}$
$D_{p,i}$	Pore diffusivity	$\text{m}^2 \text{s}^{-1}$
f	Function	
h_f	Heat transfer coefficient between the gas and the particle	$\text{W m}^{-2} \text{K}^{-1}$
h_w	Heat transfer coefficient between the gas phase and the wall	$\text{W m}^{-2} \text{K}^{-1}$
k_f	Film mass transfer coefficient	m s^{-1}
K_i	Langmuir constant	bar^{-1}
n_a	Number of past inputs	
n_b	Number of past outputs	
$NSSE$	Normalized sum of squared errors	
P	Pressure	bar
P_{high}	High pressure	bar
P_{low}	Low pressure	bar
$Prod_{CO_2}$	CO ₂ Productivity	$\text{mol kg}^{-1} \text{h}^{-1}$
Pur_{CO_2}	CO ₂ Purity	%
Q_{purge}	Purge volumetric flow rate	$\text{m}^3 \text{s}^{-1}$
Q_{rinse}	Rinse volumetric flow rate	$\text{m}^3 \text{s}^{-1}$
R_p	Particle Radius	m
Rec_{CO_2}	CO ₂ Recovery	%
t_{feed}	Feed step time	s
t_{purge}	Purge step time	s

t_{rinse}	Rinse step time	s
T_{inlet}	Inlet temperature	K
q_i^*	Adsorbed concentration in equilibrium of component i	mol kg ⁻¹
\bar{q}_l	Particle averaged adsorbed concentration	mol kg ⁻¹
$q_{m,i}$	Maximum adsorption capacity for component i	mol kg ⁻¹
R_g	Universal gas constant	J mol ⁻¹ K ⁻¹
T_g	Temperature of the gas phase	K
T_p	Temperature of the solid phase	K
T_w	Wall temperature	K
T_∞	External temperature	K
u	Input variable	
u_0	Superficial gas velocity	m s ⁻¹
w_i	Neural network weights	
v	White noise	
y	Output variable	
y_i	Molar fraction of component i	
\hat{y}	Prediction of an output variable	
z	Axial position	m
Z^N	Training set	

Greek Letters

α_w	Ratio of the internal surface are to the volume of the column wall	m ⁻¹
α_{wl}	Ratio of the log mean surface are to the volume of the column wall	m ⁻¹
ΔH_{ads}	Heat of adsorption	kJ mol ⁻¹
ε	Bed porosity	
ε_p	Particle porosity	
θ	Set of parameters for a parametrized function	
$\hat{\theta}$	Probability distribution of ANN model predictions	
λ	Heat axial dispersion coefficient	W m ⁻¹ K ⁻¹
μ	Fluid viscosity	Pa s
ρ	Gas density	kg m ⁻³
ρ_b	Bulk density	kg m ⁻³
ρ_p	Apparent density	kg m ⁻³
ρ_w	Wall density	kg m ⁻³

List of Acronyms

AI	Artificial Intelligence
ANN	Artificial Neural Network
ASR	Automated Speech Recognition
DNN	Deep Neural Network
EBP	Error Backpropagation
FNN	Feedforward Neural Network
GPU	Graphical Processing Unit
LHS	Latin Hypercube Sampling
LSTM	Long Short-Term Memory
MIMO	Multi Input Multi Output
MISO	Multi Input Single Output
MLP	Multilayer Perceptron
NARX	Nonlinear Autoregressive with Exogenous Inputs
NOE	Nonlinear Output Error
NSSE	Normalized Sum of Squared Errors
OBS	Optimal Brain Surgeon
PSA	Pressure Swing Adsorption
RNN	Recurrent Neural Network
TSA	Temperature Swing Adsorption
VPSA	Vacuum Pressure Swing Adsorption

1 Introduction

1.1 Motivation and Relevance

Syngas is one of the main sources available for the production of pure H₂ and synthetic fuels, among others. There are numerous methods and raw materials for production of syngas, and depending on the route of production, it may get contaminated with undesirable impurities. Thus, pre-treatment might be necessary in order to remove the impurities and also adjust the H₂/CO ratio to a suitable amount to be used in fuel production through the Fischer-Tropsch process. Amine absorption is the preferred process for adjustment of H₂/CO ratio. However, adsorption processes are potentially a more viable alternative, as they are able to maintain the low energetic costs associated with the absorption while reducing the environmental impact and increasing the separation productivity.

Pressure swing adsorption (PSA) is one of the adsorptive processes that are considered a viable alternative for pre-treatment of syngas, due to the aforementioned advantages. However, this process displays very complex dynamic behaviour, which affects the development of strategies of design, control, optimization and inference of the process, making the employment of PSA in industrial scale a difficult task. The development of practical ways to assess the process dynamic behaviour is an essential step to overcome those problems. Assessment of the process' dynamic behaviour is usually accomplished by using phenomenological models, which are often complex, and their simulation requires great computational time and effort. Thus, it is difficult to make use of a phenomenological model for real-time process assessment. Therefore, it is necessary to develop ways of finding alternative models that are able to provide accurate information about the process dynamics in real-time.

Artificial Neural Networks (ANN) are empirical models that emerged as alternatives to phenomenological models because of their versatility and are part of the state-of-the-art of plenty of technologies, such as speech and facial recognition. Most of its applications are found in the field of informatics. It is necessary to construct a bridge between informatics and chemical engineering so advancements on the former can solve problems of the latter.

This work proposes to develop an empirical model based on ANNs in order to model a PSA process to adjust the H₂/CO ratio of a syngas mixture. Classical models of ANNs will be built, as well as intermediary ones, ANNs machine learning based, and finally the most recent advance in the field, deep neural networks (DNN). Their performances shall be compared, so as to find the most appropriate model for the process in question. Furthermore, the evaluation of the employment of DNN models on a chemical process is another important contribution of this work.

1.2 Outline

This work is divided into 5 main parts:

In chapter 1, an introduction explaining the motivation and context in which this work was carried out is presented.

In chapter 2, State of the Art, a brief introduction of what syngas consists of, its main routes of production is presented, as well as an explanation of why its pre-treatment may be necessary, and the main methods to achieve it. Then, a description of the functioning of pressure swing adsorption, one of the pre-treatment methods, is presented. Finally, the concept of artificial neuron networks is explained and the state-of-the art for artificial intelligence in chemical engineering is described.

In chapter 3, Materials and Methods, the phenomenological mathematical model of the PSA unit studied in this work, used to obtain experimental data, is presented. Then, the predictor models used throughout the work are defined, as well as all the network structures. Moreover, the procedures and methods used to obtain the results are explained in this section as well.

In chapter 4, Results and Discussion, the experimental data used in this work is presented, as well as the obtained results for all the network models. In the end, comparisons are established between each other, in order to make a distinction between them in terms of performance.

In chapter 5, the conclusions drawn from the results are presented, followed by proposals of future works with the potential to expand on the work done in this thesis.

2 State of the Art

2.1 Syngas

Synthetic gas, commonly known as syngas, is a gaseous mixture of hydrogen (H_2), carbon monoxide (CO) and carbon dioxide (CO_2). It is a key intermediate in the production of synthetic fuels through the Fischer-Tropsch process and it is one of the main sources for the production of pure H_2 and CO (Abraham, 2017; J. Rostrup-Nielsen & Christiansen, 2011).

In theory, syngas can be produced from several carbon sources, which include natural gas, oil, coal and biomass. These can be combined with steam and oxygen in order to generate syngas (J. Rostrup-Nielsen & Christiansen, 2011). The main processes used for syngas production are steam reforming (J. R. Rostrup-Nielsen, 1984), non-catalytic partial oxidation (Marda et al., 2009) and autothermal reforming (Christensen & Primdahl, 1994).

An essential step before the usage of syngas for Fischer-Tropsch synthesis is its purification, in order to remove impurities, such as CO_2 or acid gases, that can affect the outcome of the process and poison the catalysts used in the manufacture of syngas and in downstream processes (J. Rostrup-Nielsen & Christiansen, 2011). Furthermore, the H_2/CO ratio for syngas is a very important parameter to consider in Fischer-Tropsch synthesis and as such, prior treatment may also be necessary for adjusting the ratio to desired levels (Regufe et al., 2015). Adsorptive processes have been presented as a good choice for the pre-treatment of syngas due to their low energetic consumption. Currently, absorption using amines is the preferential method for syngas pre-treatment, but the pressure swing adsorption process is also a viable alternative and possibly a more efficient one (Regufe et al., 2015).

2.2 Pressure swing adsorption

The main principle behind adsorption processes is that an adsorbent selectively adsorbs one or more components from a fluid mixture. This selectivity might be a result of either a difference between adsorption equilibria or a difference between adsorption kinetics (Ruthven, Farooq, & Knaebel, 1994).

Adsorptive processes are comprised of two main steps: adsorption, where the adsorbent bed retains the preferentially adsorbed species in its midst, and regeneration, which is when those species are removed from the bed. The product stream obtained during the adsorption step is called raffinate and will mostly contain the least adsorbed species. The stream produced by the regeneration step is called extract and is rich in the most adsorbed species (Ruthven et al., 1994).

Large-scale adsorptive separation processes can be divided into two categories: cyclic batch systems and continuous flow systems. In cyclic batch processes, the adsorbent bed is alternately saturated and regenerated as part of a cycle, whereas in continuous flow systems, the feed stream and adsorbent are in continuous counter-current contact with each other (Ruthven, 1984).

Cyclic batch processes often feature alternative methods for the regeneration of the adsorbent bed: thermal swing consists in heating the bed to a point where the adsorbate is desorbed and removed in the fluid stream, this technique is employed in a process called temperature swing adsorption (TSA); pressure swing consists in an adsorption step with a higher pressure and then the pressure in the column is decreased in order to achieve desorption, this technique is the most distinguishable aspect of the pressure swing adsorption (PSA). Additionally, other variants might be found such as the Vacuum Pressure Swing Adsorption (VPSA).

TSA/PSA units and their variants have been applied in the industry to solve complex separation problems such as: purification of hydrogen (Sircar & Golden, 2000), removal of CO₂ from gas mixtures (Gomes & Yee, 2002) and air drying (Chihara & Suzuki, 1983).

A PSA system consists of one or more adsorbent beds that are alternately pressurized and depressurized with usually a step of adsorption, called a feed step, where the lighter components are produced, followed by a desorption step, where the most retained components are produced (Ruthven, 1984). The bed is then purged, so the species in question is thoroughly removed, so as to not contaminate the raffinate in the next step. The cycle proceeds by pressurizing the column and feeding the gas mixture to the adsorbent bed again. The more complex large-scale industrial designs usually include more than two columns and several additional steps (Ruthven et al., 1994), such as the rinse step, where after the feed step, the bed is purged with the most retained species at high pressure in order to improve the purity of the extract (Ruthven et al., 1994). This leads to a very complex dynamic behaviour, originating a series of problems in the process design and operation.

2.3 Artificial Neural Networks

Artificial neural networks (ANN) are a computational model composed of interconnected processing elements called neurons or nodes (G. Zhang, Eddy Patuwo, & Y. Hu, 1998). ANNs were first proposed by McCulloch & Pitts (1943), where the model was devised to represent the biological neuron.

McCulloch & Pitts (1943) started from the principle that a biological neural system contains cells called neurons, connected to each other by synapses. Each neuron is composed of a cell

body and functional units called dendrites and axons. The dendrites are responsible for receiving information from other neurons and transmitting them to the cell body. The axons then send the information away through the synapses to the dendrites of neighbouring neurons (Basheer & Hajmeer, 2000). This is the basic mechanism through which the nervous system transmits external signals to the entire human body.

Like biological neural systems, ANNs respond to a set of external signals, the inputs and outputs, and approximate a function of the inputs, by propagating the input values to the outputs, using the weights as intermediate parameters. Following these ideas, the ANNs components are equivalent to the parts of a biological neuron:

- Neurons, the elementary unit of an ANN, equivalent to the cell body, represent a function that sums the weighted inputs and the biases;
- The connections between the neurons are equivalent to dendrites, and transmit information between each neuron;
- Weights, which represent the strength of the synapses, scale the information input to each neuron;
- Biases are input factors of the neurons that allow them to better adjust to the input data;
- Activation function, which, in its simplest configuration, functions as an ON-OFF switch that determines whether the neuron was “activated” or not;
- Outputs, equivalent to the axons, transmit information away from the neuron.

A visual representation of the aforementioned concepts is presented in Figure 1:

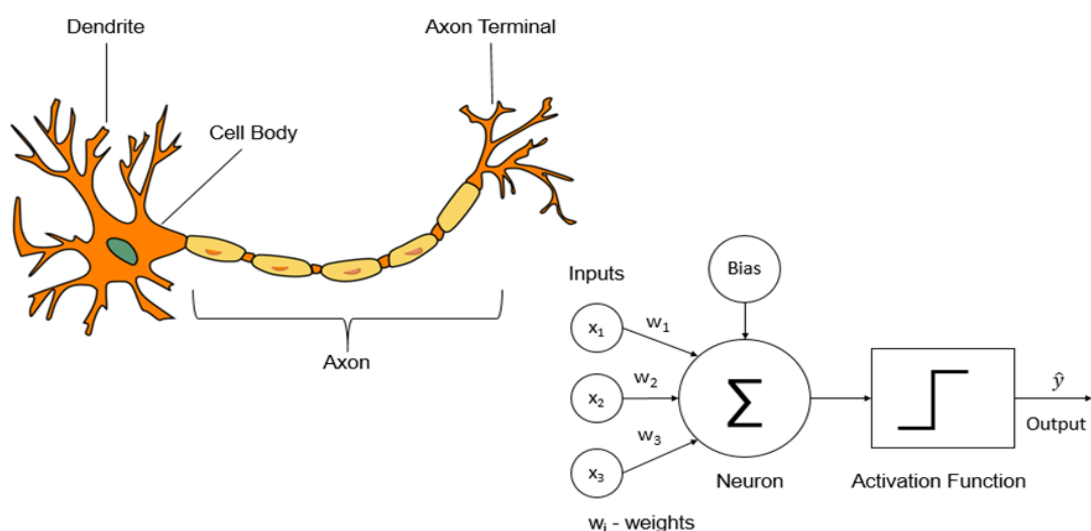


Figure 1 - Comparison between a biological neuron and an artificial neuron

The parameters of the network, weights and bias, are estimated during the training step. During this step, a set of data (training data), consisting of input-output data pairs, is fed to the

network so an optimization algorithm (training algorithm) is able to determine the set of network parameters that can provide the most accurate prediction of outputs for their corresponding inputs.

ANNs can either be single-layer or multi-layer: a single layer network will only have a set of inputs mapped to an output node by using a generalized variation of a linear function. In the multi-layer network, the neurons are arranged in layers, and the input and output layers are separated by one or more hidden layers, referred to that way because its computations are not accessible to the user (Aggarwal, 2018).

The information flux between the layers of an ANN model can be only towards the output layer, in which case the ANN is a feedforward neural network (FNN), or there can be backpropagation of information, information can flow in any direction, which is characteristic of recurrent neural networks (RNN). This complex flux of information makes RNNs more powerful tools for prediction, which makes them more suitable for modelling sequential data, thanks to their ability to store memory (Aggarwal, 2018). However, usage of RNNs requires more computational resources, and their training is more complex, often leading to problems. On the other hand, FNNs are simpler to train and implement but are less powerful, and thus, their predictive capability is much more limited. The trade-off between these two structures will be detailed further throughout this work.

2.4 Evolution of ANN and Deep Learning

Many engineering problems are very complex and are difficult or even impossible to solve through conventional methods. Some of these problems involve incomplete and noisy data, which are the type of problems that ANNs were designed to solve (Rafiq, Bugmann, & Easterbrook, 2001). As such, researchers started to take advantage of ANN's capabilities and employed them to solve problems such as wastewater treatment (Hamed, Khalafallah, & Hassanien, 2004; Gontarski, Rodrigues, Mori, & Prenem, 2000), process control (Chen & Huang, 2004) and time-series forecasting (G. P. Zhang, 2003). In the field of adsorption processes, Sant Anna, Barreto, Tavares, & de Souza (2017) proposed a Machine Learning based model to optimize a PSA unit; Ye et al. (2019) proposed a classical ANN model that also performed optimization of a PSA unit; Nogueira et al. (2018) proposed a Quasi-Virtual analyser based on FNN networks to perform real-time measurement of a Simulated Moving Bed adsorption unit. However, it was not found any work that evaluates the employment of most advanced techniques from the artificial intelligence field, such as the Deep Learning ANNs.

According to Venkatasubramanian (2019), the historical evolution of AI application in chemical engineering can be divided into three main phases and a precursor phase, which are summed up in Figure 2:

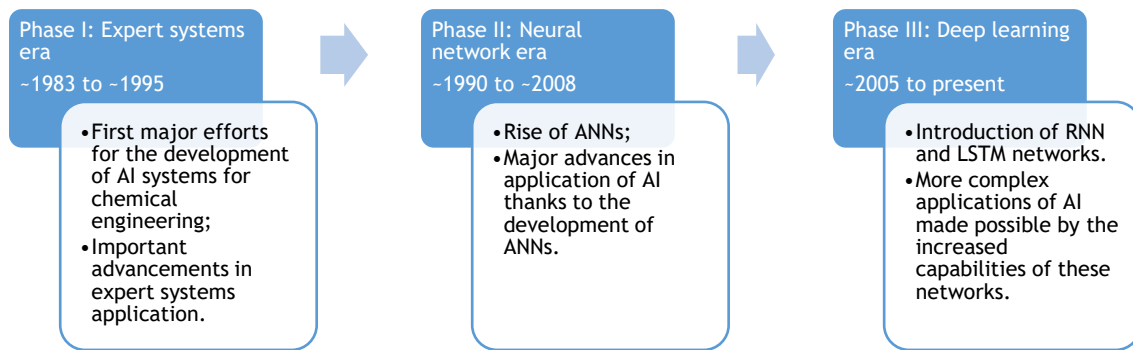


Figure 2 - Historical evolution of the use of AI in chemical engineering.

Each of the phases is described in detail in the following sections:

Phase 0: Early attempts

Even though major work in researching AI for use in chemical engineering only started in the early 1980s, some projects for the development of AI in chemical engineering started as early as the late 1960s (Venkatasubramanian, 2019). Siirola & Rudd (1971) created the first system that employed AI methods in chemical engineering.

Phase I: Expert systems era (~1983 to ~1995)

The period from the early 1980s to the mid-1990s saw some of the first major efforts for the development of AI systems for chemical engineering, where so-called expert systems, or knowledge-based systems, played a major role (Venkatasubramanian, 2019). Expert systems are computer programs that simulate the problem-solving methods of humans with expertise in a given field. This era saw impressive advancements in expert systems application, in domains such as process synthesis and design (René Bañares-Alcántara, Westerberg, & Rychener, 1985) (R. Bañares-Alcántara, Westerberg, Ko, & Rychener, 1987). However, expert systems had severe disadvantages, mainly, the fact that developing industrial applications for these systems was too costly and time-consuming, due to the extensive amounts of knowledge needed to be taught to the AI (Venkatasubramanian, 2019).

Phase II: Neural networks era (~1990 to ~2008)

Interest in expert systems waned over time due to the aforementioned drawbacks. In the early 1990s, interest in another method for developing AI started picking up. ANNs represented a good alternative to expert systems, as they are able to automatically acquire knowledge from experimental data, which required considerably less time and effort than feeding information to expert systems (Venkatasubramanian, 2019).

A key breakthrough for ANN development was the reinvention of the backpropagation algorithm for training of feedforward neural networks, which in turn introduced the possibility of approximating nonlinear functions using experimental data (Venkatasubramanian, 2019). This opened up new possibilities in the field of AI, and a tremendous amount of advancements were made thanks to ANNs, particularly in the fields of product design (Gani, 2004), fault detection (Ungar, Powell, & Kamens, 1990), control (Hernández & Arkun, 1992) and modelling (Bakshi & Stephanopoulos, 1993). In spite of the newfound success of ANNs in the 1990s, some problems remained beyond their capabilities at the time. Domains like speech recognition and natural language processing required more complex networks, with more intermediate layers, which were impossible to train at the time. Furthermore, in the domains of process control, optimization and modelling, the predictive abilities of ANNs declined over time, and required a large amount of data for their development, which was not readily available during those times (Venkatasubramanian, 2019).

Phase III: Deep learning era (~2005 to present)

The answer to the shortcomings of ANNs was revealed in the 2000s, when deep or convolutional neural networks were created. These networks had more intermediate layers in their structure, as opposed to the classical ANNs, which typically only had one (Venkatasubramanian, 2019). The training of these networks was not even possible through backpropagation and only in 2006 was a viable training method developed, a layer by layer training strategy that brought along an increase in processing speed. This allowed the networks to extract features hierarchically and learn more complex patterns (Venkatasubramanian, 2019).

Recurrent neural networks (RNN) also became common during this time. As previously mentioned, these networks are more suited for the approximation of sequences, as they are able to store memory. This opened up new possibilities for solving problems that include sequential data, like time series analysis. This was enhanced further by the development of long short-term memory units (LSTM). These structural features improve the memory saving of RNN, and are better suited for predictions based on time series data. Their structure is analysed further in section 3.3.2.

Deep learning is a group of machine learning techniques that use artificial neural networks to solve complex problems. The relationship between AI, machine learning and deep learning, as well as their definitions, is explained in Figure 3:

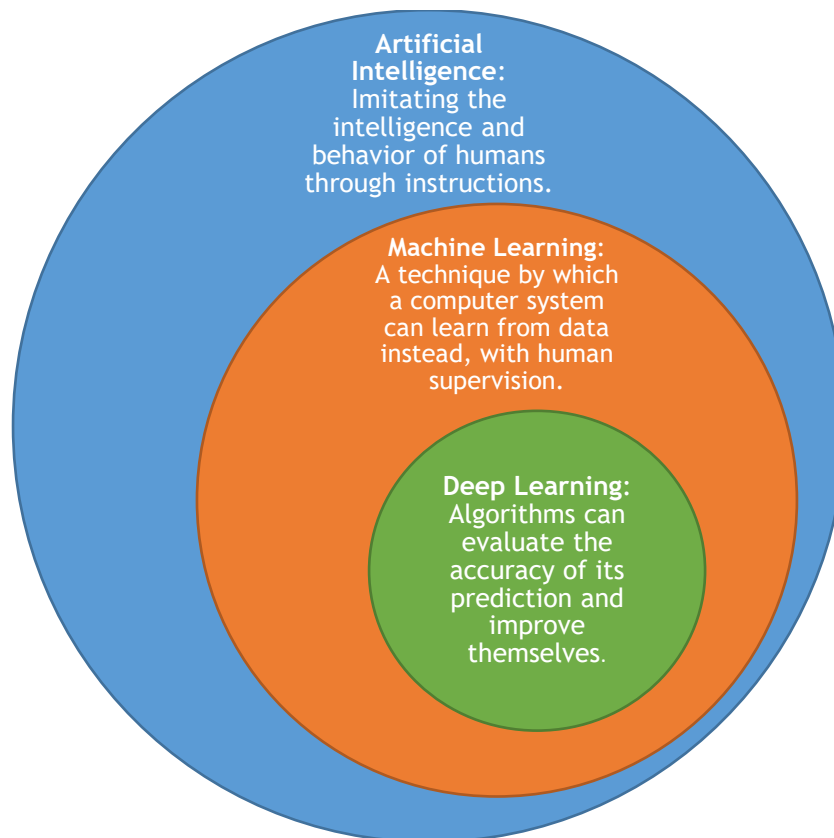


Figure 3 - Relationship between the concepts of artificial intelligence, machine learning and deep learning.

Deep learning is part of the state of the art technology for various fields, particularly in automated speech recognition (ASR), where it contributed to significant technological improvements (Hinton et al., 2012; Deng et al., 2013).

Another similar field where deep learning found several practical application is document reading. By the late 1990s, 10% of all bank cheques in the United States of America were verified by deep networks (LeCun, Bengio, & Hinton, 2015).

In the early 2010s, hardware advancements renewed interest in practical applications of deep learning. Graphical processing units (GPUs) were used to train deep learning neural networks and it was found that they speed up the training process by several orders of magnitude, causing a severe decrease of running times (Raina, Madhavan, & Ng, 2009). This is due to GPUs being well suited for the mathematical operations behind machine learning (Oh & Jung, 2004).

Hardware advancements brought about a revolution in the use of deep learning, and deep neural networks are now the predominant approach for nearly all recognition and detection tasks (LeCun et al., 2015). Most major technology companies, such as Microsoft, Google, Apple, IBM, Adobe and Facebook started research and development on deep neural networks for use in their products (LeCun et al., 2015). Currently, those companies employ LSTM networks in

their major commercial products. For instance, Google uses LSTM on Google Translate (Metz, 2016b) and Apple uses LSTM in the “Quicktype” function of their iPhones. (Metz, 2016a)

However, nowadays, the majority of the works from the process engineering field that makes use of artificial intelligence are still using tools from Phase II (Sant Anna et al., 2017; Ye et al., 2018; Nogueira et al., 2018; Nogueira et al., 2017). There is a lack of new studies in the process engineering field in order to make these new developments available to solve a series of issues from the field. Thus, deep learning has not yet found applications on the field of chemical engineering processes, and one of the goals of this work is to demonstrate the potential of this technology for the modelling of a real-life PSA process.

3 Materials and Methods

3.1 Mathematical model

A set of data, with inputs and their respective outputs, is necessary for the training of an ANN. For this work, the data is obtained by simulating a PSA process using the gPROMS software.

As such, a mathematical model that characterizes the PSA process studied in this work was developed and programmed on gPROMS, so it can be used to simulate the process.

The simulations were conducted in MATLAB by using a bridging software called goMATLAB. The inputs were introduced on MATLAB and the gPROMS model was called upon to simulate the process, obtain the results and send them to MATLAB.

The following assumptions were made in order to formulate the mathematical model:

- Ideal gas behavior;
- Axial dispersed flow;
- External mass and heat transfer resistances expressed with the film model;
- Internal mass transfer resistance expressed with the Linear Driving Force (LDF) model;
- No temperature gradients inside each particle, as the heat transfer in the solid phase is much faster than in the gas phase;
- Constant porosity along the bed;
- The Ergun equation is valid locally.

The material balance for each component in the gas phase is given by:

$$\frac{\partial}{\partial z} \left(\varepsilon D_{ax} C_{g,T} \frac{\partial y_i}{\partial z} \right) - \frac{\partial}{\partial z} (u_0 C_{g,i}) - \varepsilon \frac{\partial C_{g,i}}{\partial t} - (1 - \varepsilon) a_p k_f (C_{g,i} - C_{s,i}) = 0 \quad (1)$$

where z is the axial position, ε is the bed porosity, D_{ax} is the axial dispersion, u_0 is the superficial velocity, $C_{g,T}$ is the total concentration in the gas phase, $C_{g,i}$ is the component concentration in this gas phase (for component i), y_i is the molar fraction of component i , $C_{s,i}$ is the concentration of component i in the solid phase, k_f is the film mass transfer coefficient and a_p is the external specific area of the particle.

The LDF model is used to characterize the mass transfer rates in the solid phase. As such, the material balance in the solid phase is given by:

$$\frac{\partial \bar{q}_i}{\partial t} = \frac{15D_p}{R_p^2} (q_i^* - \bar{q}_i) \quad (2)$$

Where D_p is the pore diffusivity, R_p is the particle radius, \bar{q}_i is the particle averaged adsorbed concentration and q_i^* is the adsorbed concentration in equilibrium with $C_{s,i}$.

q_i^* is determined by the multicomponent extension of the Langmuir isotherm:

$$q_i^* = q_{m,i} \frac{K_i P_i}{1 + \sum_{j=1}^n K_j P_j}, i, j = CO_2, N_2, CH_4, CO \quad (3)$$

Equating the fluxes at the particle surface:

$$\frac{(1 - \varepsilon) a_p k_f}{\rho_b} (C_{g,i} - C_{s,i}) = k_h (q_i^* - \bar{q}_l) \quad (4)$$

where ρ_b is the bulk density of the column.

The energy balance in the gas phase is given by:

$$\begin{aligned} \frac{\partial}{\partial z} \left(\lambda \frac{\partial T_g}{\partial z} \right) - u_0 C_{g,T} C_p \frac{\partial T_g}{\partial z} + \varepsilon R_g T_g \frac{\partial C_{g,T}}{\partial t} \\ - (1 - \varepsilon) a_p h_f (T_g - T_p) - \frac{4h_w}{d_{wi}} (T_g - T_w) - \varepsilon C_{g,T} C_v \frac{\partial T_g}{\partial t} = 0 \end{aligned} \quad (5)$$

where T_g , T_p , and T_w are the gas phase, solid phase and wall temperatures, respectively, λ is the heat axial dispersion coefficient, R_g is the universal gas constant ($R=8.314 \text{ J}\cdot\text{K}^{-1}\cdot\text{mol}^{-1}$), h_f is the heat transfer coefficient between the gas and the particle, h_w is the heat transfer coefficient between the gas phase and the column wall, d_{wi} is the internal wall diameter, C_p is the heat capacity of the mixture at constant pressure and C_v is the heat capacity of the mixture at constant volume.

An overall energy balance of the gas phase, adsorbed phase and the solid phase inside a particle is described by:

$$(1 - \varepsilon) \left[\varepsilon_p \sum_{i=1}^n \bar{q}_l C_{v,ads,i} + \rho_p \hat{C}_{p,s} \right] \frac{\partial T_p}{\partial t} = \rho_b \sum_{i=1}^n (-\Delta H_{ads}) \frac{\partial \bar{q}_l}{\partial t} + (1 - \varepsilon) a_p h_f (T_g - T_p) \quad (6)$$

where $(-\Delta H_{ads})$ is the heat of adsorption of each component, and $\hat{C}_{p,s}$ is the solid specific heat per mass unit.

The energy balance to the column wall assumes energy exchange with the gas phase and the external environment, and is given by:

$$\rho_w \hat{C}_{p,w} \frac{\partial T_w}{\partial t} = \alpha_w h_w (T_g - T_w) - \alpha_{wl} U (T_w - T_\infty) \quad (7)$$

where T_∞ is the external temperature, ρ_w is the wall density, $\hat{C}_{p,w}$ is the wall specific heat per mass unit, U is the overall heat transfer coefficient, α_w is the ratio of internal surface area to the column wall volume and α_{wl} is the ratio of the log mean surface area to the column wall volume.

The momentum balance is given by the Ergun equation:

$$-\frac{\partial P}{\partial z} = \frac{150\mu(1-\varepsilon)^2}{\varepsilon^3 d_p^2} u_0 + \frac{1.75(1-\varepsilon)\rho}{\varepsilon^3 d_p} |u_0|u_0 \quad (8)$$

where P is the total pressure, μ is the gas viscosity, ρ is the gas density and d_p is the particle diameter.

The selected boundary conditions that were used with the proposed PSA model are as follows:

Table 1 - Boundary conditions for the PSA model.

Pressurization with feed	
$z = 0$, inlet	$z = L$
$u_{0inlet}C_{inlet,i} = u_0C_{g,i} - \varepsilon D_{ax}C_{g,T} \frac{\partial y_i}{\partial z}$	$\frac{\partial C_{g,i}}{\partial z} = 0$
$P = P_{inlet}$	$u_0 = 0$
$u_{0inlet}C_{inlet,T}C_pT_{inlet} = u_0C_{g,T}C_pT_g - \lambda \frac{\partial T_g}{\partial z}$	$\frac{\partial T_g}{\partial z} = 0$
Feed, Rinse	
$z = 0$, inlet	$z = L$, outlet
$u_{0inlet}C_{inlet,i} = u_0C_{g,i} - \varepsilon D_{ax}C_{g,T} \frac{\partial y_i}{\partial z}$	$\frac{\partial C_{g,i}}{\partial z} = 0$
$u_{0inlet}C_{inlet,T} = u_0C_{g,T}$	$P = P_{outlet}$
$u_{0inlet}C_{inlet,T}C_pT_{inlet} = u_0C_{g,T}C_pT_g - \lambda \frac{\partial T_g}{\partial z}$	$\frac{\partial T_g}{\partial z} = 0$
Counter-current blowdown	
$z = 0$, outlet	$z = L$
$\frac{\partial C_{g,i}}{\partial z} = 0$	$\frac{\partial C_{g,i}}{\partial z} = 0$
$P = P_{outlet}$	$u_0 = 0$
$\frac{\partial T_g}{\partial z} = 0$	$\frac{\partial T_g}{\partial z} = 0$
Purge	
$z = 0$, outlet	$z = L$, inlet
$\frac{\partial C_{g,i}}{\partial z} = 0$	$u_{0inlet}C_{inlet,i} = u_0C_{g,i} - \varepsilon D_{ax}C_{g,T} \frac{\partial y_i}{\partial z}$
$P = P_{outlet}$	$u_{0inlet}C_{inlet,T} = u_0C_{g,T}$
$\frac{\partial T_g}{\partial z} = 0$	$u_{0inlet}C_{inlet,T}C_pT_{inlet} = u_0C_{g,T}C_pT_g - \lambda \frac{\partial T_g}{\partial z}$

For this process, four performance indicators are used. These parameters will be the predicted output variables of the ANNs defined in section 4, in order to develop a simpler model of the process which requires significantly less computation effort to make the predictions. These are:

The ratio between the H_2 and CO is calculated as:

$$\frac{H_2}{CO} = \frac{\int_0^{t_{feed}} C_{H_2} u_0|_{z=L} dt + \int_0^{t_{rinse}} C_{H_2} u_0|_{z=L} dt}{\int_0^{t_{feed}} C_{CO} u_0|_{z=L} dt + \int_0^{t_{rinse}} C_{CO} u_0|_{z=L} dt} \quad (9)$$

The CO_2 purity in percentage is calculated as:

$$Pur_{CO_2} = \frac{\int_0^{t_{blow}} C_{CO_2} u_0|_{z=0} dt + \int_0^{t_{purge}} C_{CO_2} u_0|_{z=0} dt}{\sum_{i=1}^n \int_0^{t_{blow}} C_i u_0|_{z=0} dt + \int_0^{t_{purge}} C_i u_0|_{z=0} dt} \times 100 \quad (10)$$

The CO_2 recovery in percentage is calculated as:

$$Rec_{CO_2} = \frac{\int_0^{t_{blow}} C_{CO_2} u_0|_{z=0} dt + \int_0^{t_{purge}} C_{CO_2} u_0|_{z=0} dt - \int_0^{t_{rinse}} C_{CO_2} u_0|_{z=0} dt}{\int_0^{t_{press}} C_{CO_2} u_0|_{z=0} dt + \int_0^{t_{feed}} C_{CO_2} u_0|_{z=0} dt} \times 100 \quad (11)$$

The CO_2 productivity (given in: $mol_{CO_2} kg_{ads}^{-1} h^{-1}$) is calculated as:

$$Prod_{CO_2} = \frac{\int_0^{t_{blow}} C_{CO_2} u_0|_{z=0} dt + \int_0^{t_{purge}} C_{CO_2} u_0|_{z=0} dt - \int_0^{t_{rinse}} C_{CO_2} u_0|_{z=0} dt}{t_{cycle} \times mass\ dry\ adsorbent} \quad (12)$$

3.2 Predictors

Usually, the dynamic behaviour of several processes can be represented by time series or the so-called predictor. The predictors are an essential part of a model and they will define the model's intrinsic nature. Thus, they should be thoroughly defined.

A time series can be described as a series of data points ordered in time, usually at equally spaced time intervals. Considering a generic system, a time series can be represented by:

$$y(t) = G[u(t), u(t-1), \dots, u(t-k), v(t), v(t-1), \dots, v(t-k)] \quad (13)$$

where u is the model input, and v is the white noise.

Thus, based on predictors, a system can be effectively represented using a model, which can be an empirical one such as artificial neural networks. Hence, a first step in the non-linear dynamic modelling is to evaluate the correct predictor formulation in order to attain the most suitable representation of the system in study.

The most usual and important forms of predictors are Nonlinear Autoregressive with Exogenous Inputs (NARX) and Nonlinear Output Error (NOE) (I. B. R. Nogueira et al., 2018). Thus, the present work will be based on those two structures. The predictor equations for these models will be presented in the following sections and their influence on the system modelling will also be evaluated in further detail through this thesis.

3.2.1 Nonlinear Autoregressive with Exogenous Inputs (NARX)

For a dynamic system, the actual output $y(t)$ can be given by a NARX structure as:

$$y(t) = F[y(t-1), \dots, y(t-1-n_a), u(t-d), \dots, u(t-d-n_b+1)] + v(t) \quad (14)$$

where u are the process input values, n_a and n_b the number of past values (also referred to as the model orders), d is the input delay and $v(t)$ is the white noise produced by the system.

A visual representation of the NARX model is found below:

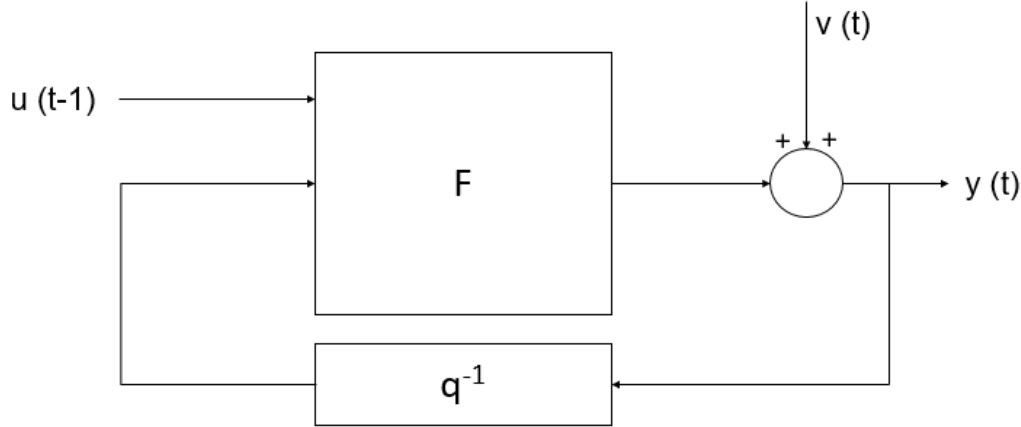


Figure 4 - NARX model with $n_a = 1$, $n_b = 1$ and $d = 1$.

This equation describes one-step-ahead predictions, but it may also be altered so it produces multi-step ahead predictions:

$$\hat{y}(t) = f[y(t-1), \dots, y(t-1-n_a), u(t-d), \dots, u(t-d-n_b+1), \theta] \quad (15)$$

where θ represents the set of parameters of a parameterized function $f(\varphi, \theta)$.

The NARX model depends upon past measurements and inputs and is the simplest predictor model available. Furthermore, it demonstrates a very high speed of convergence for function identification. For these reasons, NARX is considered to be the most reliable predictor, being widely used in literature (I. Nogueira, 2018; Koivisto, 1995).

3.2.2 Nonlinear Output Error (NOE)

The predictor equation for NOE is described below:

$$\begin{aligned} z(t) &= F[z(t-1), \dots, z(t-1-n_a), u(t-d), \dots, u(t-d-n_b+1)] \\ y(t) &= z(t) + v(t) \end{aligned} \quad (16)$$

where z is the output prediction.

Similarly to NARX, a model can also be developed for multi-step ahead predictions:

$$\begin{aligned} \hat{z}(t) &= f[\hat{z}(t-1), \dots, \hat{z}(t-1-n_a), u(t-d), \dots, u(t-d-n_b+1), \theta] \\ \hat{y}(t) &= \hat{z}(t) \end{aligned} \quad (17)$$

An example of a NOE model is found below:

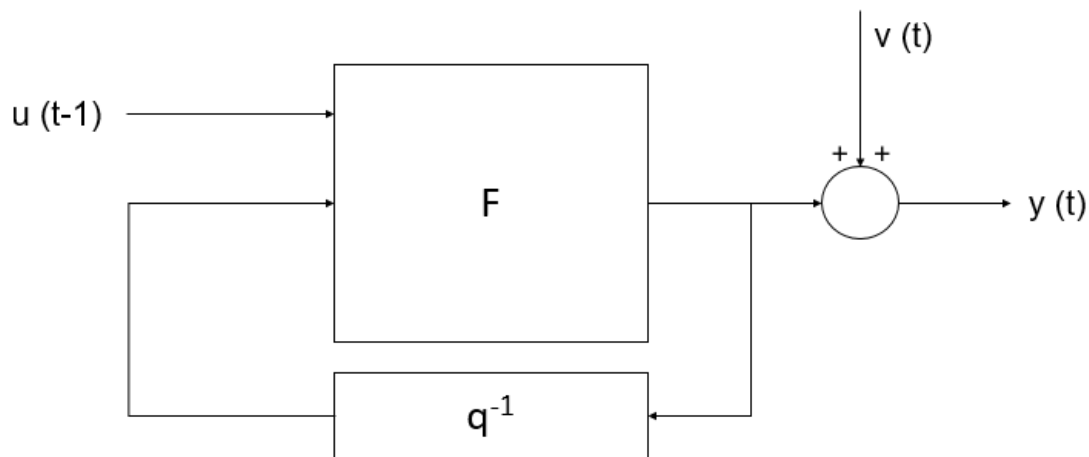


Figure 5 - NOE model with $n_a = 1$, $n_b = 1$ and $d = 1$.

As can be seen in equations 16 and 17 and Figure 5, NOE depends on past inputs and outputs, as well as its own past predictions, to evaluate the current output. This is in contrast to NARX, which only depends on past inputs and outputs, which makes the estimation of the parameters of a model based on a NOE predictor more difficult. This happens because the partial derivative of the NOE based predictions depends on the partial derivative of its own parameters. This leads to a correlation among parameters to be estimated and predictions made (Nelles, 2001).

On the other hand, it has been found that, due to their own nature, NARX models through time might lead to severe error accumulation that compromises their accuracy and efficiency, and creates model instability; this effect is diminished for NOE models, making them more suited for long-term predictions than NARX models (Nelles, 2001). This makes NOE models a preferential option for producing simulation models, whereas NARX models are a better choice for predictive purposes (Koivisto, 1995).

3.3 Neural Network Models

Once the predictors are defined it is necessary to define the modelling strategy. The present work is based on empirical models using artificial neural networks (ANN) strategies. ANNs were chosen due to their higher computation speed compared to more conventional phenomenological models (which causes less burden on computational hardware) (Cannady, 1998), the fact that it is a “black box” approach, which means that few prior knowledge of the underlying process is necessary (Karunanithi, Whitley, & Malaiya, 1992) and that they are capable of adapting their structure in response to new, unforeseen data, and thus, maintain their predictive capability (Cannady, 1998).

One of the goals of this work is to evaluate the application of cutting edge advances in ANN modelling to solve process engineering problems. Another goal is to evaluate the performance of those new tools, comparing them to the traditional ones. Thus, both deep neural networks (DNN) and classical artificial neural networks were employed in order to provide a solution to the problems of modelling and predicting unmeasurable variables of a PSA unit where syngas is purified with the main goal of adjusting its composition for future applications. Both of these neural network structures will be studied in detail and their results will be compared in the following sections.

3.3.1 Feedforward Neural Networks (FNN)

The Feedforward Neural Networks (FNN) is a classical structure of ANN where there is no backpropagation of information. The type of ANN structure selected to model the system in study is the Multilayer Perceptron (MLP). This architecture was one of the most used and extensively studied ANN structures during the second phase of ANN development, prior to the advent of deep learning (Venkatasubramanian, 2019). A considerable number of training algorithms have been developed for this type of network. MLP structures are very popular due to their versatility, as they are capable of approximating simple and complex functional relationships (Nørgaard, 2000).

The MLP consists of multiple layers of simple nodes, or neurons, that interact using weighted connections. The MLP network must have at least, and usually has, three layers: an input layer, an output layer, and a minimum of one intermediate, or hidden, layer in between. (Pal & Mitra, 1992)

An example of a MLP with 2 inputs, 3 hidden units and a single output is presented in Figure 6:

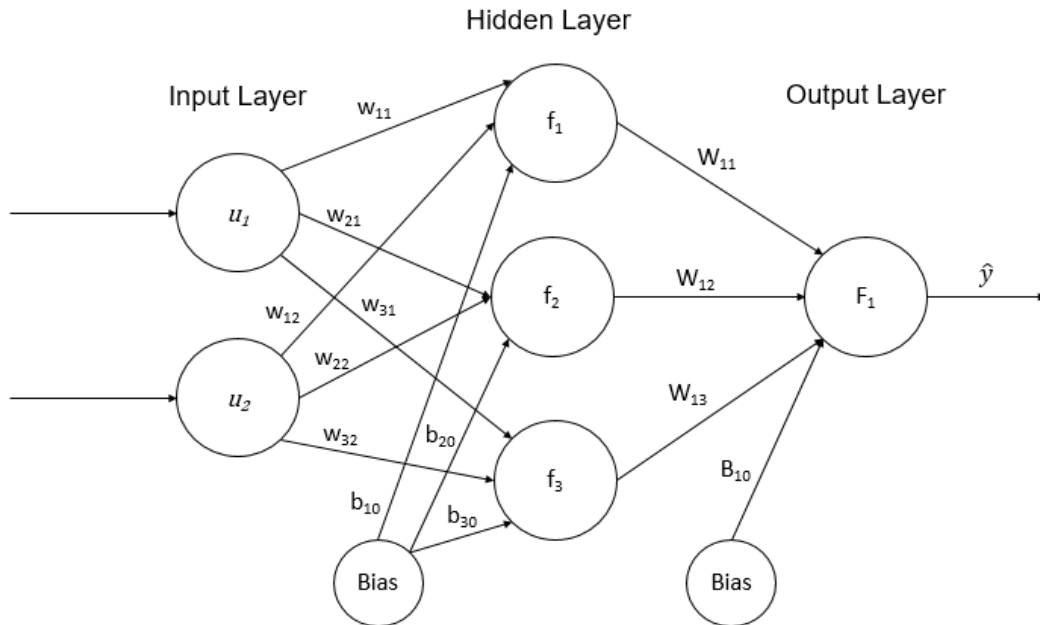


Figure 6 - Example of Multilayer Perceptron Network with 2 inputs, 3 hidden units and a single output unit.

The weights (w and W) and the biases (b) are the parameters of the network and are determined through a parameters estimation procedure called training step (Nørgaard, 2000). This ANN structure will only be used for NARX models, which are intrinsically FNN structures.

3.3.2 Recurrent Neural Networks (RNN)

On the other hand, the ANN model based on the NOE structure is intrinsically a recurrent structure, in other words, it is necessary to establish a backpropagation of information in the ANN in order to set the past predictions as a present input. Thus, the ANN structure for this case cannot be a FNN and instead, it should be a recurrent neural network (RNN). Therefore, this model is considered a Machine Learning tool, because aside from the relationship among inputs and outputs, it is capable of tracking the additional relationship among past and present states. Once again, a Multilayer Perceptron is used here as a base for the RNN, which is called recurrent multilayer perceptron network.

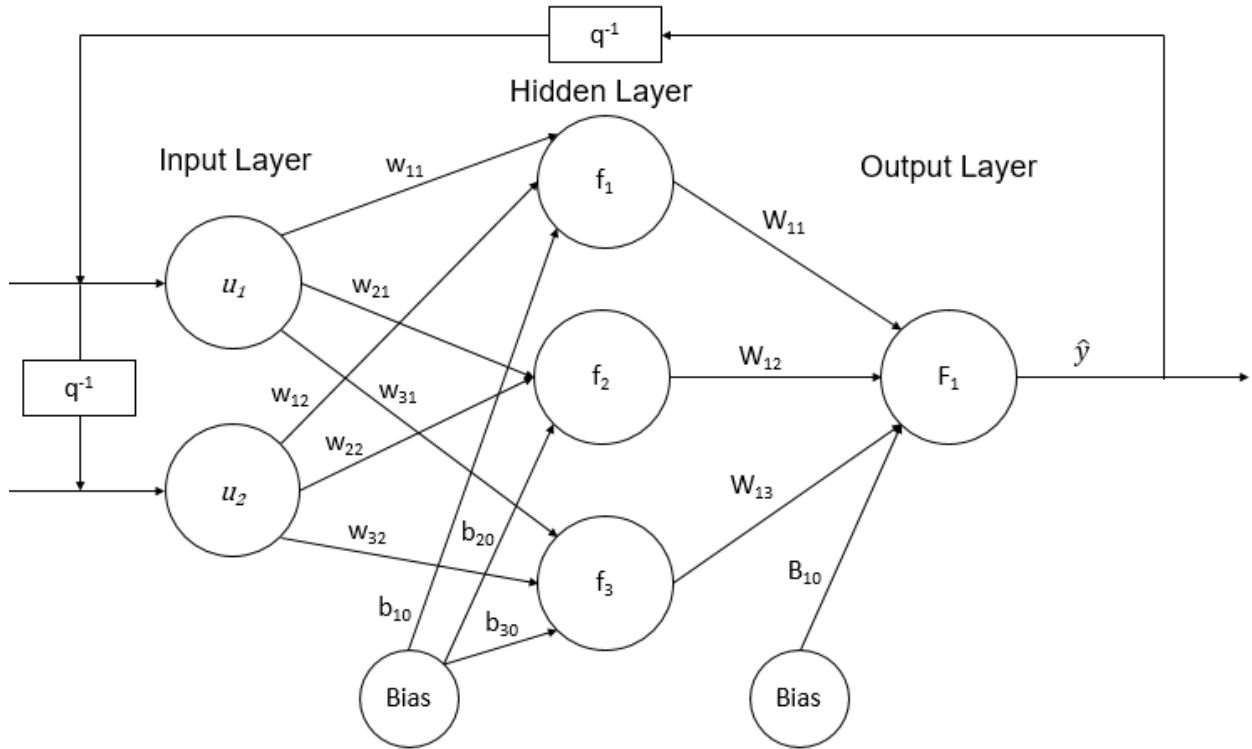


Figure 7 - Multilayer perceptron with 2 inputs, 3 hidden units and a single output.

Figure 7 presents the general structure of a RNN here used as a model for the NOE case. In the same way as in the FNN, this structure presents the weights and bias, w and b , respectively, as parameters to be estimated.

3.3.3 Neural Network Training

Considering a training set described by:

$$Z^N = \{[u(t), y(t)] | t = 1, \dots, N\} \quad (18)$$

where $u(t)$ are the inputs and $y(t)$ are the corresponding outputs.

During training, the MLP is repeatedly fed with the same training set, optimizing the ANN parameters iteratively, until a mapping that represents the input-output relation accurately is achieved (Gardner & Dorling, 1998). The main goal of the training step is to maximize the probability distribution, $\hat{\theta}$, of the ANN model predictions representing the training data, as:

$$Z^N \rightarrow \hat{\theta} \quad (19)$$

Subject to:

$$\hat{\theta} = P(z^N, z^P)$$

This way, the network should produce output predictions, \hat{y} , that are close to the true values of $y(t)$ (Nørgaard, 2000).

In order to estimate the predictions' inaccuracy, the prediction error approach is used, which uses the mean square error as a criterion:

$$V_N(\theta, Z^N) = \frac{1}{2N} \sum_{t=1}^N [y(t) - \hat{y}(t|\theta)]^T [y(t) - \hat{y}(t|\theta)] \quad (20)$$

The weights are then evaluated as the values for the θ vector that minimize the error estimated with equation 20:

$$\hat{\theta} = \operatorname{argmin} V_N(\theta, Z^N) \quad (21)$$

Then, an iterative minimization algorithm is used:

$$\theta^{(i+1)} = \theta^{(i)} + \mu^{(i)} f^{(i)} \quad (22)$$

where $\theta^{(i)}$ is the current iteration, $f^{(i)}$ is the search direction and $\mu^{(i)}$ is the step size used by the algorithm.

Several training algorithms for neural networks have been developed, and they differ from each other in their search direction and step size. The algorithm selected for the training of MLP networks in this work is the Levenberg-Marquardt method, which is implemented in the Neural Network Based System Identification Toolbox (NNSYSID). The Levenberg-Marquardt method, developed by Kenneth Levenberg in 1944 and rediscovered by Donald Marquardt in 1963, provides a numerical solution for minimizing a nonlinear function (Yu & Wilamowski, 2011).

One of the first algorithms used for training of neural networks was the steepest descent method, also known as error backpropagation (EBP) algorithm. This algorithm was continually improved throughout the years, despite being regarded as an inefficient algorithm due to its slow convergence speed (Yu & Wilamowski, 2011).

The slow convergence of the EBP algorithm can be diminished by employing the Gauss-Newton algorithm, which can find correct step sizes for every search direction and leads to faster convergence. The Levenberg-Marquardt method combines the EBP and Gauss-Newton algorithms, in order to take advantage of each method's main qualities: the robustness of the EBP algorithm, as well as the speed of convergence from Gauss-Newton. This is the reason why the Levenberg-Marquardt method acquired a reputation of being the standard method for minimization using mean squared error criteria (Yu & Wilamowski, 2011; Nørgaard, 2000). The mathematical proof of this algorithm as well as its implementation is well documented by Fletcher (1987) (Fletcher, 1987) and Nørgaard (2000).

As previously mentioned, training algorithms attempt to maximize a probability according to a criterion, so that the neural network is able to predict the desired variables, as described in equations 14 and 15. However, training the network to a global maximum is not possible, and this step should be carefully done as the problem of overfitting may occur (Sjöberg & Ljung, 1995). Overfitting means that the training of the network fails to develop generalization capabilities, or in other words, the model over learns the characteristics of the training set in detriment of its capacity to perform prediction outside of the training conditions. Due to overtraining, the final network will present too many elements, presenting a very good adaptation to the training data, but failing to make accurate predictions when presented with new data.

Thus, the regularization technique is presented in order to avoid the overfitting problem. The regularization consists in the employment of an extra criterion in order to stop the neural network training when this criterion is met. In this work, the criterion used was weight decay, which is a term introduced to the training algorithm that causes the values of the weights to exponentially decay to zero. Weight decay is implemented in the NNSYSID toolbox and is used throughout this work to avoid overtraining the classical neural networks and the machine learning models.

The identification of the ANN models was done with the NNSYSID toolbox, developed by Prof. Magnus Nørgaard from the Department of Automation of the Technical University of Denmark in 2000. The MATLAB Version 2.0 of this toolbox, released in the year 2000, was used in this work (Nørgaard, 2000). Even though NNSYSID is 20 years old, it is considered a complete and powerful tool to build classical ANNs models.

3.3.4 Neural Network Validation

The ANN validation is done using an extra data set obtained following the procedure that will be described in section 3.5. Thus, a new set of inputs and its corresponding outputs, different from the training set is used to verify the model predictions. This was done in order to verify the overfitting question aforementioned.

The validation of the neural networks is performed by determining the prediction errors that are obtained when the validation set is provided to the model. The normalized sum of squared errors (NSSE) was chosen as criterion to evaluate the validation performance of the models.

The NSSE is calculated as:

$$NSSE = \frac{1}{2N} \sum_{i=1}^N (\hat{y}(t) - y(t))^2 \quad (23)$$

3.3.5 Selection of the optimal number of neurons

A single hidden layer was selected for the network structure, as Hunt et al. (1992) and Hornik et al. (1989) showed that a single hidden layer in a MLP network is able to give a satisfactory approximation of a function (Hunt, Sbarbaro, Žbikowski, & Gawthrop, 1992; Hornik, Stinchcombe, & White, 1989).

After determining the number of hidden layers, the next step is finding out the optimal number of neurons in the hidden layers. For this purpose, the cross-validation method was proposed by Schenker et al. (1996). This method consists of training two different models using two different data sets, for a number of neurons (N_n), and validating each model with the set used to train the other network. After the validations are done, the validation error is stored in a vector. Then, the networks are trained again, with a different number of neurons. The procedure is repeated N_n times, the optimal number of neurons is the one that generated the smallest error (I. B. R. Nogueira et al., 2018).

However, this procedure was not employed in this work and instead, a simpler, yet similar method was used: the networks were simply trained with their respective training data a number of times, for different numbers of neurons, and their optimal number was the one that produced the smallest validation error.

3.3.6 Neural Network Pruning

If validation shows that there is overfitting, a good solution may be to remove the unnecessary neuron connections from the network, in order to improve its generalization capabilities, simplify the network structure, and improve training speed. This procedure is similar to a surgery where the unnecessary neurons are eliminated and it is called General Pruning. The pruning algorithm used here was the Optimal Brain Surgeon (OBS) available within NNSYSID, which is currently the main pruning strategy for ANNs (Nørgaard, 2000). After OBS removes each unnecessary weight, the network is retrained, and this goes on until all unimportant weights determined by OBS are eliminated. After the procedure, the network is retrained one final time, using the simplified structure given by the algorithm. The mathematical proof for the OBS method can be found in Hassibi et al. (1993).

A diagram summarizing the entire procedure for developing the networks is shown in Figure 8:

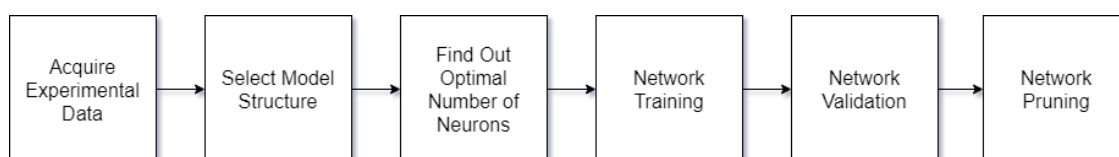


Figure 8 - Summary of the method used for developing artificial neural networks.

3.4 Long Short-Term Memory (LSTM)

Long short-term memory (LSTM) is a type of recurrent neural network (RNN) architecture used in deep learning. Unlike FNNs, RNNs have the capability of modelling sequences, by using their internal state to process sequences of inputs. LSTM is capable of keeping track of long-term data dependencies in order to model sequences (R. Zhang et al., 2019).

A schematic representation of a LSTM unit can be found in Figure 9:

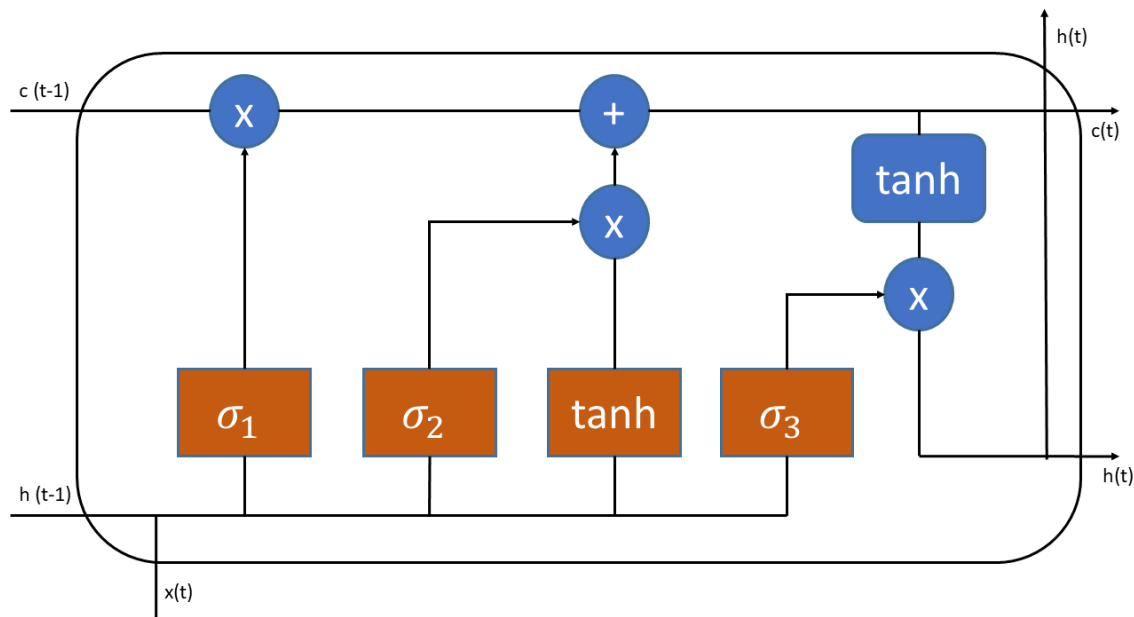


Figure 9 - Representation of a LSTM unit.

Each combination of a sigmoid unit (σ) and the pointwise multiplication (\times) it leads to is a gate. σ_1 belongs to the forget gate, σ_2 belongs to the input gate, and σ_3 belongs to the output gate.

Gradient descent methods are usually preferred for the training of RNNs. This training is more difficult than the training of FNNs because of the vanishing and exploding gradient problems - the possibility that gradients tend to zero or infinity, respectively - that occur because of the computations involved in the algorithm. The training is therefore affected by not only the variation of gradient magnitudes but also because the effect of long-term dependencies will be hidden by the effect of short-term dependencies (Chung, Gulcehre, Cho, & Bengio, 2014).

An appropriate solution for the vanishing and exploding gradient problems is to implement a type of RNN architecture that includes gating units - multiplicative units that control the flow of information in and out of cells (Chung et al., 2014). An example of such architectures is the LSTM.

Each LSTM unit contains memory blocks, called cells, and gating units. A single unit contains an input gate, a forget gate and an output gate. Input gates control the flow of information into

the cell, output gates control the output flow of information into the network and the forget gate controls the extent to which information remains in a cell (Chung et al., 2014).

3.5 Data acquisition procedure

The training data is essential information to develop a reliable model and it should be carefully chosen. This data can be obtained through practical experiments or simulations. In the present case, a phenomenological model of the process was used as a source of data. This model was previously developed and validated through experiments in a laboratorial unit by Regufe et al. (2015). Therefore, this work adopts the procedure used to generate the training data, as well as the validation data, obtaining the training sets to be used in the models' development.

The selected method for the generation of training data is called latin hypercube sampling (LHS) and was proposed by McKay et al. (1979). LHS is a method for generating near-random samples of one or more variables, and it can be seen as a blend of the known random and stratified sampling methods (Helton & Davis, 2003). LHS can be roughly divided into two different stages: firstly, samples for each variable are selected to represent that variable's cumulative probability density function, and secondly, the samples of the variables are arranged so that they can match the correlation between those variables (Huntington & Lyrintzis, 1998). LHS is a widely used sampling process because of the efficient stratification across the range of each variable (Helton & Davis, 2003). The development of LHS is detailed in McKay et al. (1979).

A subroutine for the generation of variable samples was created in MATLAB, using the *lhsdesign* function included in the software. Both training and validation data sets were generated with this procedure.

4 Results and Discussion

4.1 Process Simulation

This work's case study is a PSA process that was designed for the adjustment of H_2/CO ratio of a syngas mixture. The model of this process was previously developed by Regufe et al. (2015) and validated through experiments in a laboratorial PSA unit located in Laboratory of Separation and Reaction Engineering. The model is detailed in section 3.1, and presented below are the simulation conditions that served as basis for the generation of data for the present work was generated. The model is implemented in gPROMS (Process Systems Enterprise, 2015) and it was run in MATLAB through the go:MATLAB connection. All the remaining studies presented here were done in MATLAB.

The initial composition of the mixture is 22% CO, 30% CO_2 and 48% H_2 . As a result of the process, two product streams are obtained: a CO_2 -rich stream (the most adsorbed component of the mixture) and a H_2+CO stream with a H_2/CO ratio suitable for use in the Fischer-Tropsch process.

The PSA process is carried out with the following steps:

- Co-current pressurization with the feed mixture and feed step at $P_{High} = 4.0$ bar;
- Rinse step with a pure stream of CO_2 , in order to increase the amount of CO_2 in the adsorbent bed;
- Blowdown step, where the bed is depressurized to $P_{Low} = 1.0$ bar counter-currently;
- A purge step, carried out with a mixture of 32% CO and 68% H_2 .

A visual scheme for the full PSA cycle is presented below:

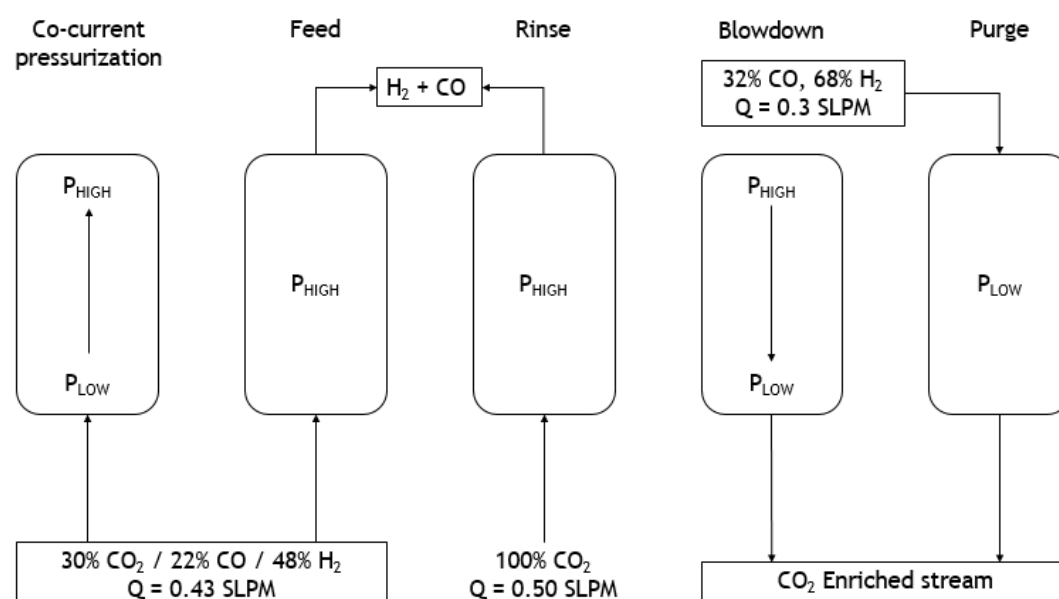


Figure 10 - Scheme of the PSA process studied in this work.

The parameters for the adsorbent bed used in the process are shown in Table 2:

Table 2 - Parameters used for the process' adsorbent bed.

Bed diameter (m)	0.0211
Bed length (m)	0.323
Bed porosity	0.353
Adsorbent	MIL-125(Ti) ₂ NH ₂
Adsorbent shape	Granulate
Particle bulk density (kg·m⁻³)	549
Particle porosity	0.56
Particle radius (m)	1.00×10 ⁻³
Particle specific heat (J·kg⁻¹·K⁻¹)	1200

The transport parameters used in the mathematical model are presented in Table 3:

Table 3 - Transport parameters used in the process' mathematical model.

D_{ax} (m²·s⁻¹)	6.9×10 ⁻⁵
λ (J·s⁻¹·m⁻¹·K⁻¹)	0.6
k_f (m·s⁻¹)	2.0×10 ⁻²
h_f (W·m⁻²·K⁻¹)	128
h_w (W·m⁻²·K⁻¹)	50
D_p (m²·s⁻¹)	CO ₂ : 7.7×10 ⁻⁷ H ₂ : 2.5×10 ⁻⁶ CO: 9.8×10 ⁻⁷
R_p (m)	1.0×10 ⁻³

The operating conditions used in the mathematical model are given in Table 4:

Table 4 - Operating conditions of the PSA process.

Temperature (K)	323
P_{High} (bar)	4.0
P_{Low} (bar)	1.0
Flow rates (SLPM)	Feed: 0.43 Rinse: 0.50 Purge: 0.30

Step times (s)	Feed: 332
	Rinse: 221
	Blowdown: 94
	Purge: 92

The results for the simulations carried out using the parameters presented in Tables 2, 3 and 4 are presented in Table 5:

Table 5 - Simulation results of the PSA process.

(H₂ + CO) product	
H₂/CO	2.37
CO₂ product	
Purity (%)	83.8
Recovery (%)	93.5
Productivity (mol·kg_{ads}⁻¹·h⁻¹)	3.15

4.2 Data acquisition

From the conditions previously presented the methodology to generate data points was applied in order to build a database large enough to train and validate the empirical models. As described in section 3.5, the input data variation was generated through the Latin hypercube sampling method, using the *lhsdesign* MATLAB function. Eight process inputs were selected as process operating variables to be correlated to the process performance parameters through the ANN models. Fixed intervals were defined for each of the eight selected input variables and the samples created were within those intervals. Those variables were selected for being the most important operating variables of the process. Furthermore, they are of interest from the control and optimization point of view, which means the ANNs models might serve as the base for those disciplines. The selected input variables are:

- t_{feed} - feed step time;
- t_{purge} - purge step time;
- t_{rinse} - rinse step time;
- P_{high} - high pressure;
- P_{low} - low pressure;
- Q_{rinse} - rinse step volumetric flow rate;
- Q_{purge} - purge step volumetric flow rate;
- T_{inlet} - inlet temperature.

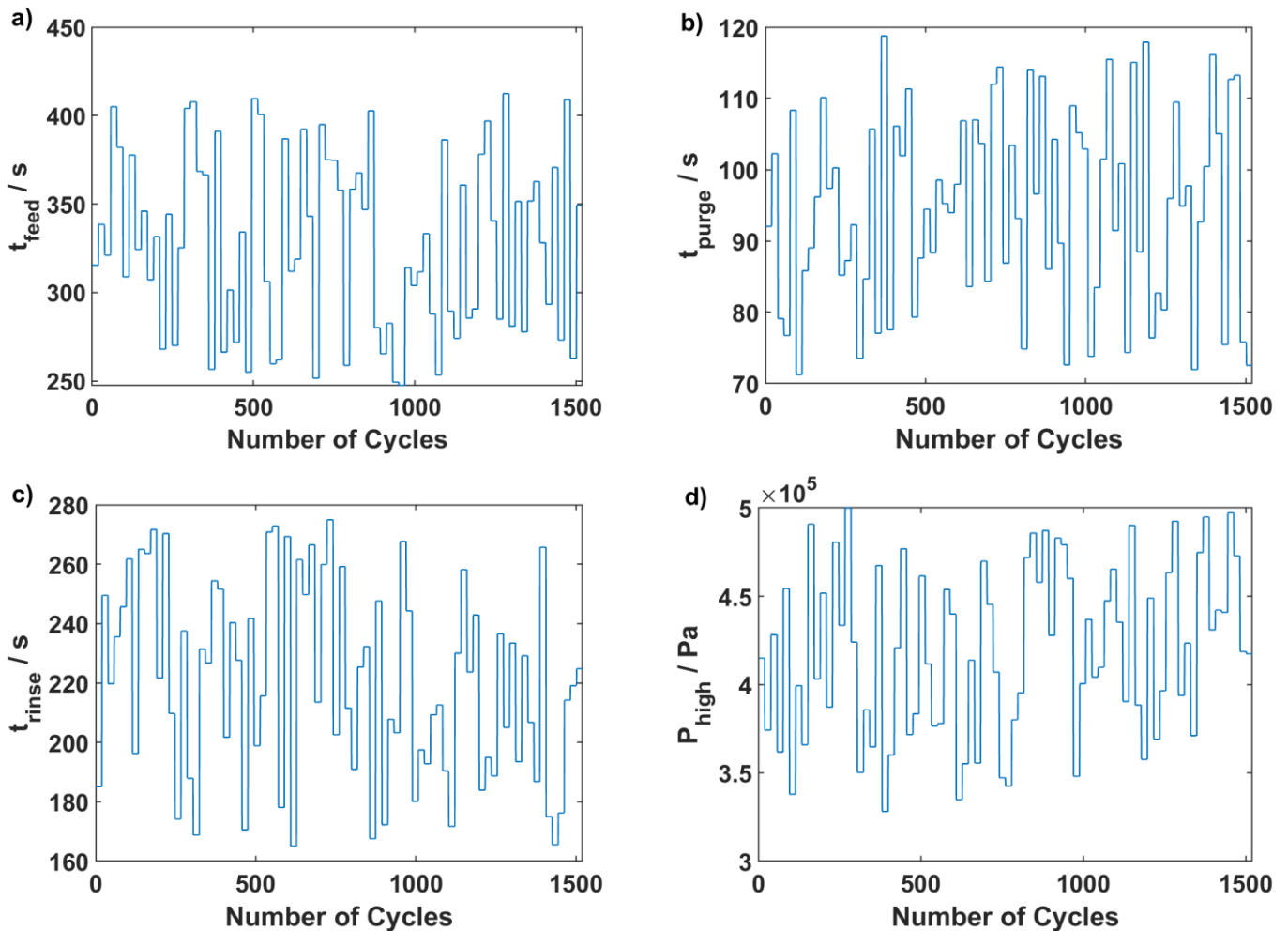
The bounds for the selected intervals are presented in Table 6 and were calculated considering $\pm 25\%$ of the values presented in Table 4:

Table 6 - Upper and lower bounds for the intervals of the generated training samples.

	t_{feed} (s)	t_{purge} (s)	t_{rinse} (s)	P_{high} (bar)	P_{low} (bar)	Q_{rinse} ($m^3 \cdot s^{-1}$)	Q_{purge} ($m^3 \cdot s^{-1}$)	T_{inlet} (K)
Upper	412.5	118.8	275	5.0	1.25	0.625	0.375	403.9
Lower	247.5	71.25	165	3.0	0.75	0.375	0.225	243.4

For the training sets, 80 different values for each input variable were generated and only one of the variables was introduced in the initial conditions vector at a time. Each one of the new operating conditions was applied to the process and then, enough time was given to reach the new steady state (19 cycles). Overall, this procedure generated a total of 12160 data points.

The inputs for each variable are represented below in graphical form in Figure 11:



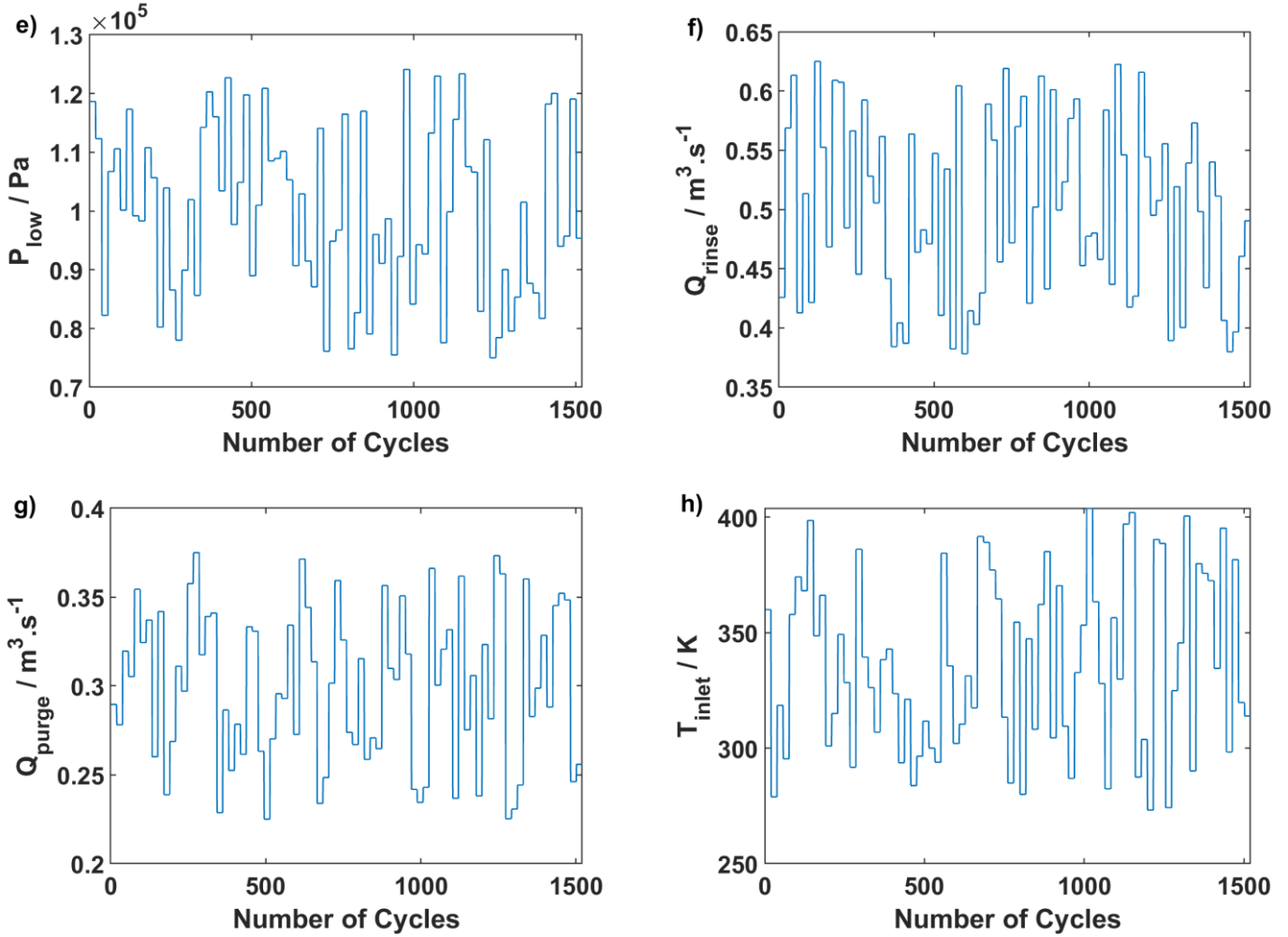


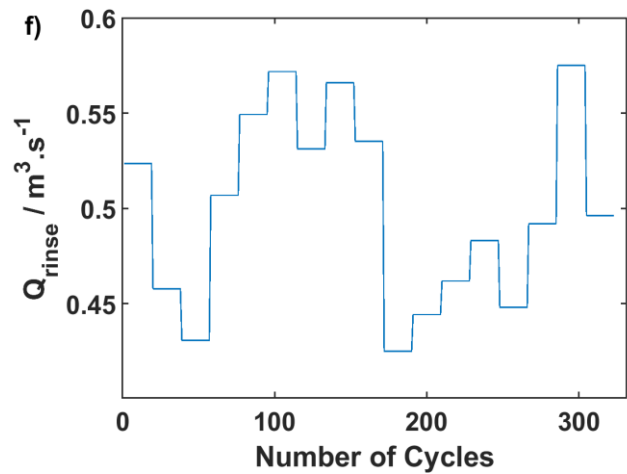
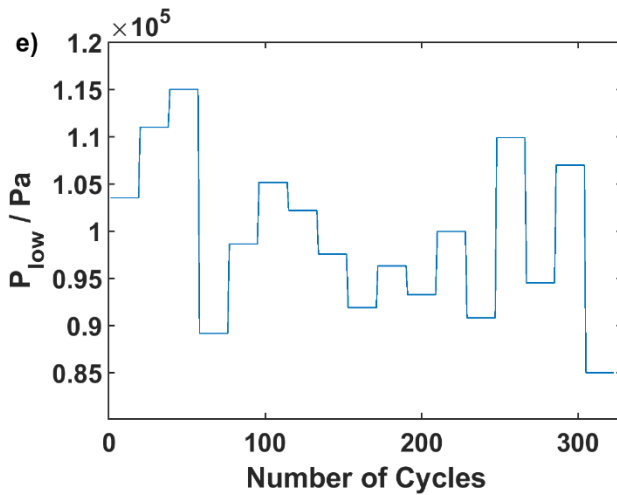
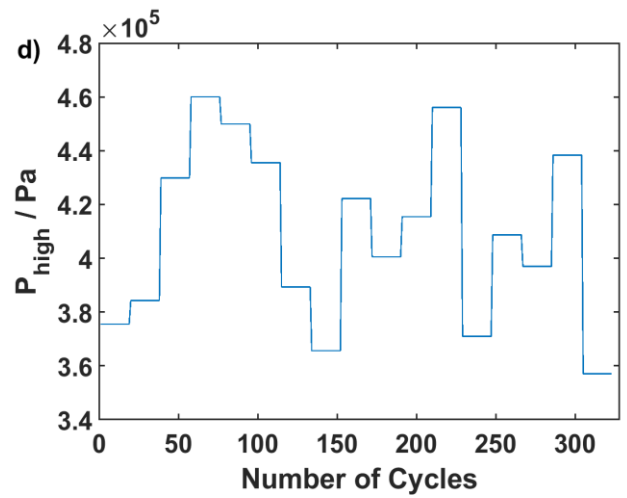
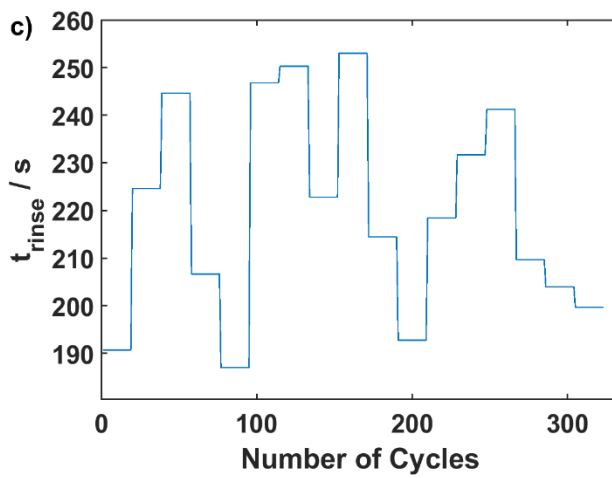
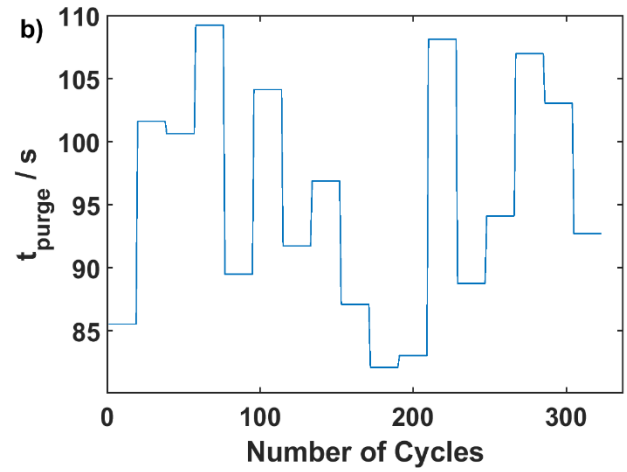
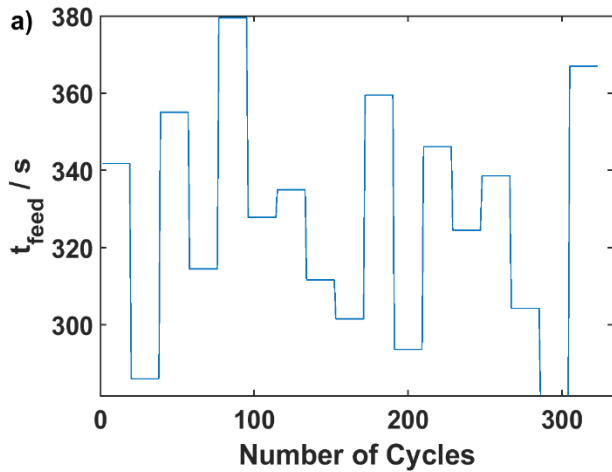
Figure 11 – Generated inputs for a) t_{feed} , b) t_{purge} , c) t_{rinse} , d) P_{high} , e) P_{low} , f) Q_{rinse} , g) Q_{purge} and h) T_{inlet} to use for training the neural networks.

The same procedure was repeated in order to generate a set for models validation. In this case, 17 different samples for each variable were generated within new operating condition intervals considering $\pm 15\%$ of the original values (Table 4) as given in Table 7, following the LHS method, and a total of 2583 data points were obtained.

Table 7 - Upper and lower bounds for the intervals of the generated validation samples.

	t_{feed} (s)	t_{purge} (s)	t_{rinse} (s)	P_{high} (bar)	P_{low} (bar)	Q_{rinse} ($m^3 \cdot s^{-1}$)	Q_{purge} ($m^3 \cdot s^{-1}$)	T_{inlet} (K)
Upper	379.5	109.25	253	4.6	1.15	0.575	0.345	371.6
Lower	280.5	80.75	187	3.4	0.85	0.425	0.255	274.7

The inputs generated for the validation set can be found in Figure 12:



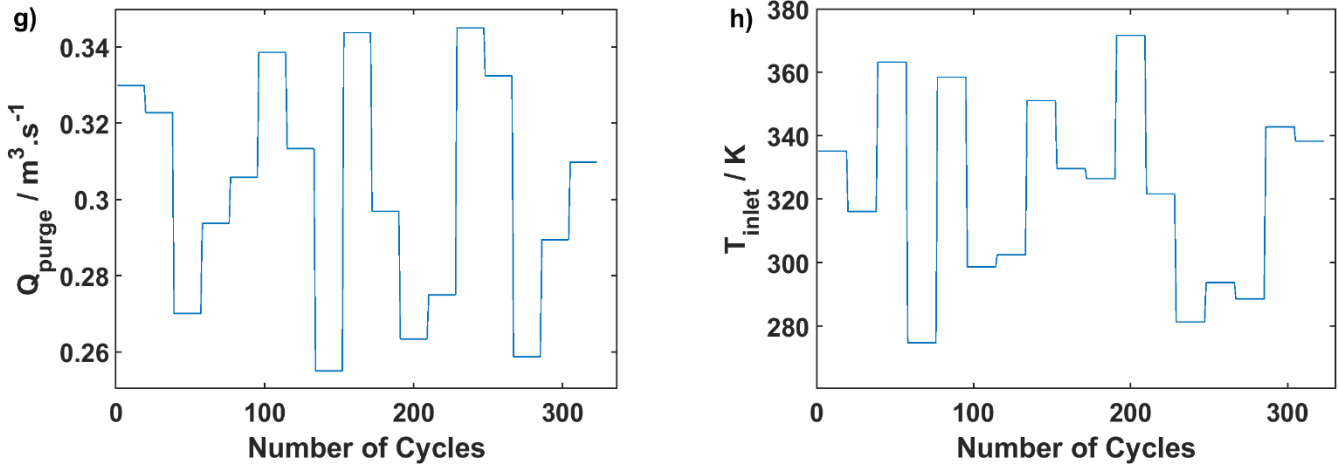


Figure 12 - Generated inputs for a) t_{feed} , b) t_{purge} , c) t_{rinse} , d) P_{high} , e) P_{low} , f) Q_{rinse} , g) Q_{purge} and h) T_{inlet} to use for validating the neural networks.

Both of the input sets were run by the model so results for the four selected output variables were obtained. These variables are the performance indicators described in section 3.1, by equations 9, 10, 11 and 12.

4.3 Network training and validation - classical model

The networks were trained with the generated training data. NARX predictor models were employed to make one-step-ahead predictions of the output variables.

A different network was trained and validated for predicting each output variable, instead of a single network that predicts every variable simultaneously, as multi-input multi-output (MIMO) networks are more complex and difficult to train than multi-input single-output (MISO) models.

The training parameters used for the NARX and NOE networks are:

Table 7 - Training parameters for the NARX and NOE models

Number of past inputs	1
Number of past outputs	1
Time delay	1
Maximum number of iterations	NARX: 500 NOE: 100
Early stop criteria (weight decay)	1

4.3.1 NARX - model identification

In order to determine the optimal number of hidden layer neurons, the networks were trained multiple times with up to 30 neurons, and the number of neurons that produced the smallest

error (NSSE, as mentioned in section 3.3.4) was selected as the optimal number. The results of this procedure shall be presented throughout this section.

The validation results (the comparison between the PSA outputs and the model's predictions) will be presented in section 4.6 and compared to the validation results of other models, in order to analyse their performances.

The validation results for NARX models may also be consulted separately in appendix A1.

The results of the identification of the optimal number of neurons for the output variables are presented in Figure 13 in which the arrows indicate the optimal number of neurons:

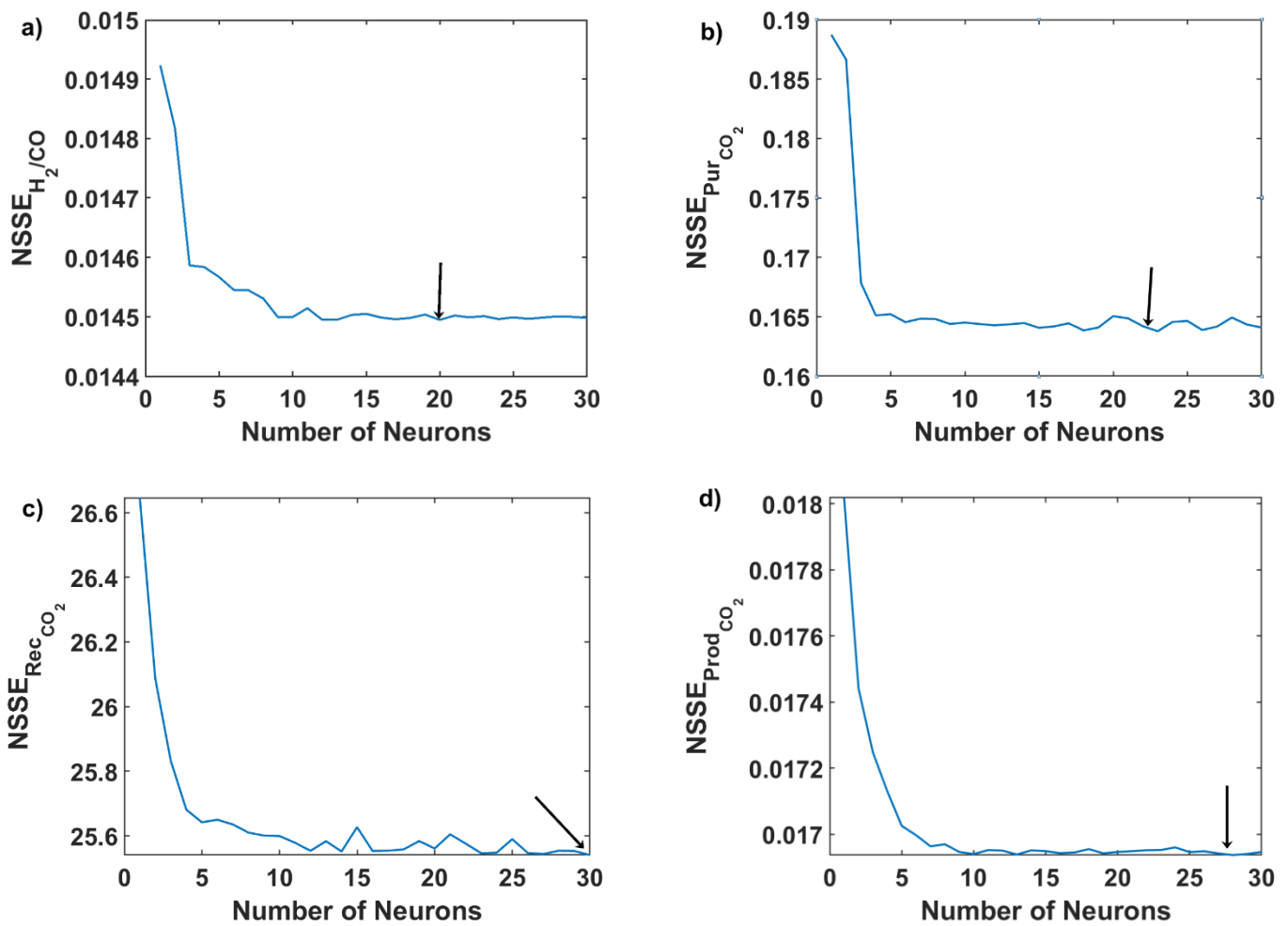


Figure 13 - Identification of the optimal number of neurons for the NARX networks: a) H_2/CO , b) Pur_{CO_2} , c) Rec_{CO_2} , d) $Prod_{CO_2}$

The results of the identification of the optimal number of neurons for the NARX model consistently showed that NSSE decreases with the number of neurons until it eventually stabilizes. The number of optimal neurons for the NARX networks tends to be quite high, which suggests that larger networks achieve better more accurate predictions until a certain point, from which overtraining might occur. However, only networks of up to 30 neurons in the hidden-

layer were trained as it was difficult to train larger networks due to the considerable computational effort required for this task.

4.4 Network training and validation - Machine Learning strategy

4.4.1 NOE - model identification

This section presents the development of the machine learning strategy, based on a NOE predictor. The same output variables were also predicted by this approach. For this case, the development methodology is very similar to the previously presented one. The validation results (the comparison between the PSA outputs and the model's predictions) will be presented in section 4.6 and compared to the validation results of other models, in order to analyse their performances.

The results of the identification of the optimal number of neurons for the output variables are presented in Figure 14:

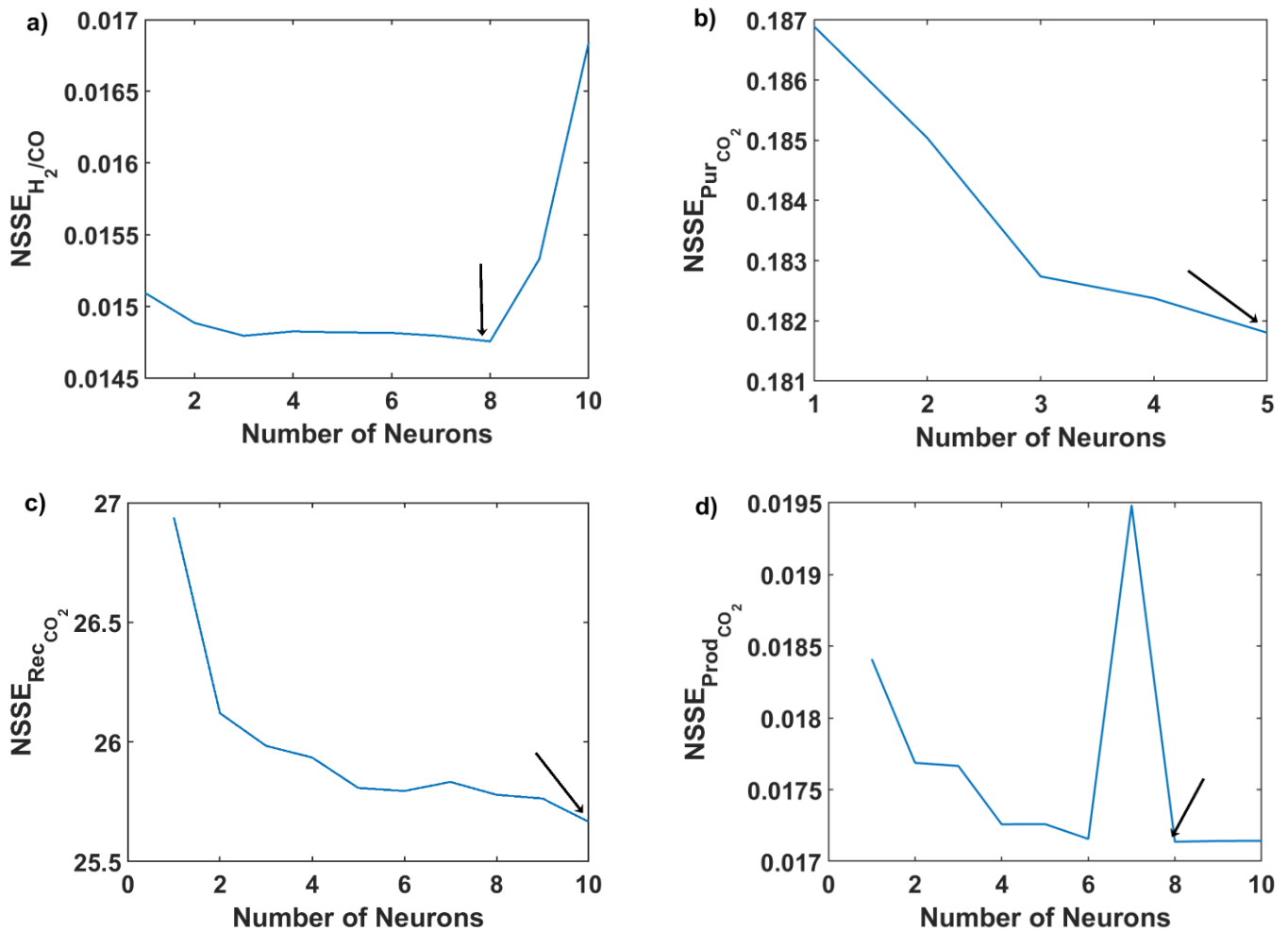


Figure 14 - Identification of the optimal number of neurons for the NOE networks: a) H_2/CO , b) Pur_{CO_2} , c) Rec_{CO_2} , d) $Prod_{CO_2}$

As can be seen in Figure 14, networks of only up to 10 and 5 hidden layer neurons were trained. This is because of the previously mentioned issues related to the vanishing/exploding gradient problems, which in this case was the vanishing gradient problem. Thus, only relatively small networks were trained, as training larger networks would be impossible due to the formation of a singular weight matrix. Furthermore, the NOE model for the CO₂ purity was only trained with up to 5 hidden layer neurons, because the same issues were verified during the training process, which rendered the training of larger networks impossible.

However, this network is an exception, and all remaining NOE networks were trained with up to 10 neurons.

From the results presented thus far, it is possible to note that unlike in NARX models, the NSSE does not always tend to decrease and eventually stabilize as the number of neurons increases. This is due to the instability of NOE models' training, which leads to the observed unpredictability of these results, such as the peak in Figure 14 d), or the sudden increase of NSSE observed in Figure 14 a).

4.5 Network training - Long Short-Term Memory model

The same output variables were also predicted by a deep learning LSTM network with the goal of comparing the performance of these network models with classical ones.

All the LSTM networks are composed of two hidden LSTM layers and each of them has a pre-selected number of hidden LSTM units. The optimization of the LSTM structure was not part of this work due to the complexity of the task, and it should be the subject of future works. Furthermore, it was noted that with two hidden layers and a number of neurons around 100, the LSTM was able to predict the process variables with good accuracy. Thus, the number of hidden LSTM units for the networks was selected through a simple trial and error approach and future works should address this issue better. However, even using this simple methodology, it was possible to obtain a better performance with the DNN models than with the classical ones, as it will be seen in section 4.6. The training parameters defined for the training of these networks and their final structure are presented in Table 8 and 9.

Table 8 - Training parameters for LSTM networks.

Maximum number of iterations (epochs)	160
Initial learning rate	0.005
Learning rate drop factor	0.2
Learning rate drop period	125
Frequency of network validation	1

Gradient treshold	1
--------------------------	---

Table 9 - Number of LSTM units in the hidden layers of each network.

H₂/CO Ratio	Number of hidden units (1 st layer): 300
	Number of hidden units (2 nd layer): 100
CO₂ Purity	Number of hidden units (1 st layer): 300
	Number of hidden units (2 nd layer): 100
CO₂ Recovery	Number of hidden units (1 st layer): 300
	Number of hidden units (2 nd layer): 100
CO₂ Productivity	Number of hidden units (1 st layer): 100
	Number of hidden units (2 nd layer): 100

The validation results for the LSTM networks can be also found in section 4.6, and the results themselves are shown in Appendix A1.

4.6 Comparison and model validation

In this section, the validation results for all of the models are compared to each other, in order to determine their relative performance.

Due to the large size of the validation data, the outputs resulting from the simulation of the PSA model had to be divided into two small regions. The division was made in order to obtain a representation of one area where there are strong dynamic oscillations, and another where the dynamics are smoother. However, full validation data and comparison between the models are shown in Appendix A1.

Two graphs are shown for each output variable: one highlighting a part of the oscillatory region, and one for the smooth region. In each of these graphs, the predictions from the three models are compared to the simulation outputs:

The results for the H₂/CO ratio are given in Figure 15:

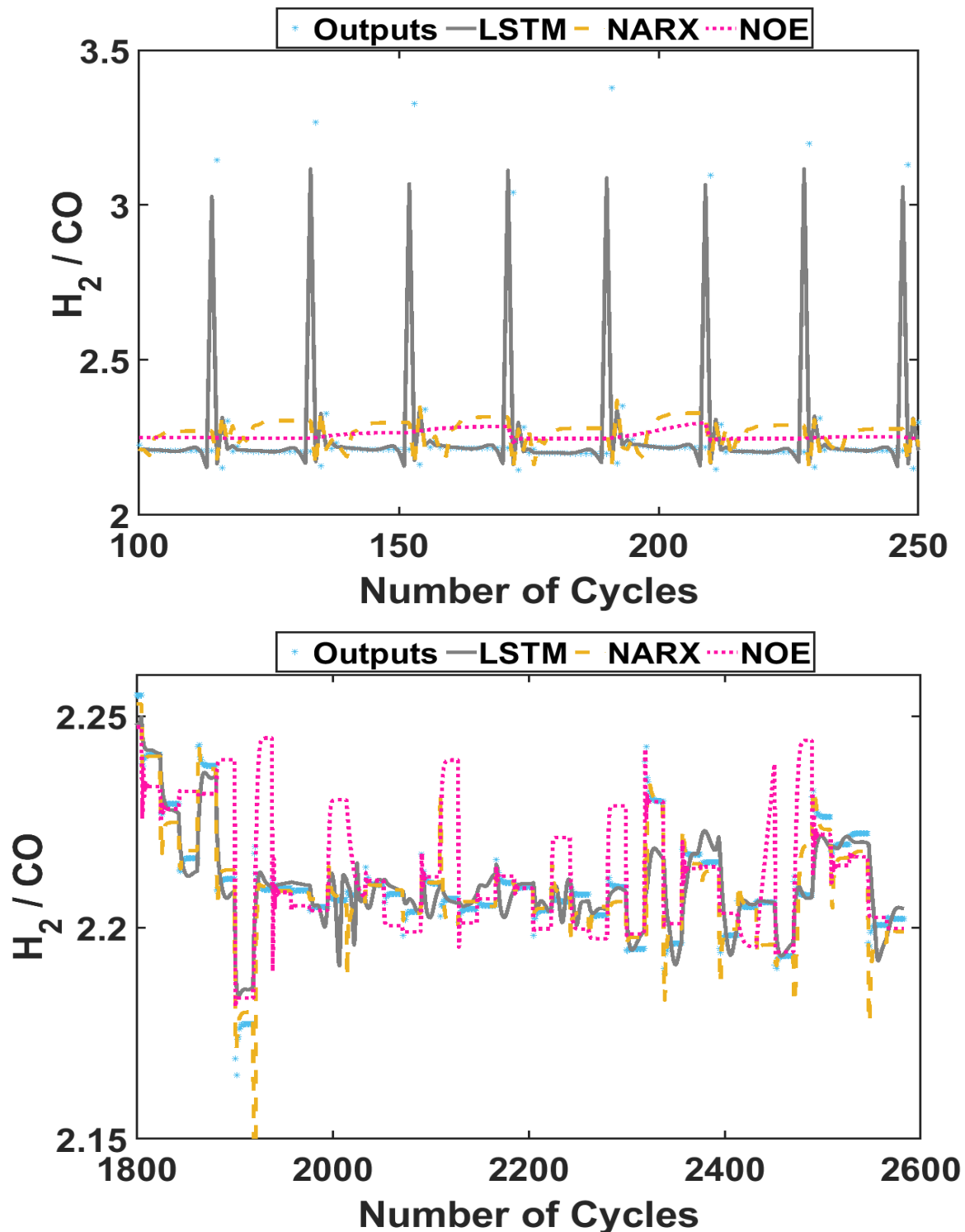


Figure 15 - Comparison of the different models for H_2/CO ratio, for the strong dynamics region (above) and the smooth dynamics region (below).

In the region with stronger dynamics, only the LSTM network is able to make precise predictions of the behaviour, whereas NARX and NOE models are not. On the other hand, the NARX model was able to do satisfactory predictions, but it presents a persistent offset. Finally, the NOE model presented a significant deviation from the process behaviour. This may be attributed to the hard training of these models, which can lead to the atrophy of networks leading to a model with poor predictive capabilities. Similar conclusions are taken from the analysis on the region

with smoother dynamics, visual inspection suggests LSTM achieved the best results, and NOE once again displayed poor performance for H₂/CO ratio predictions.

Figure 16 shows the results for CO₂ purity:

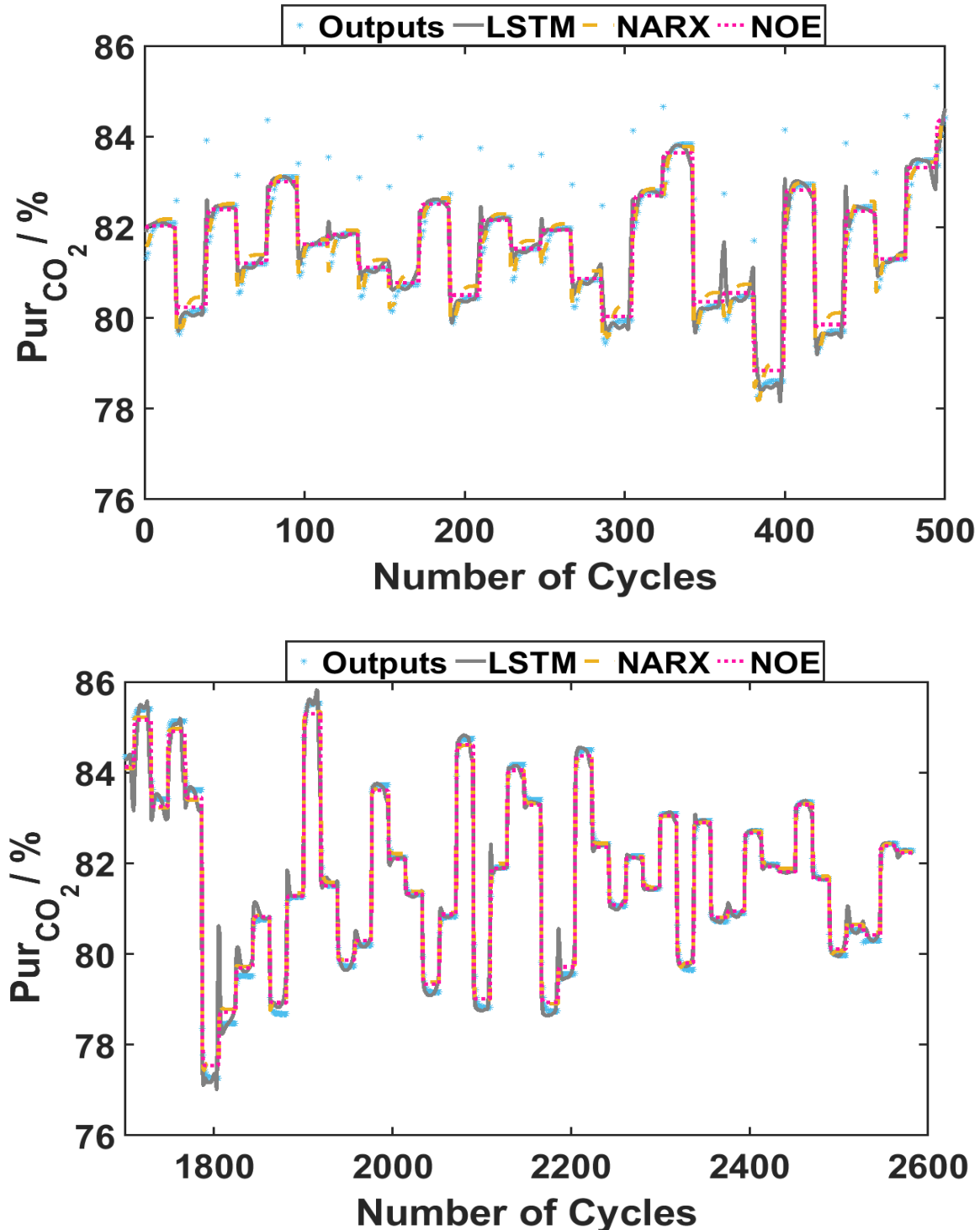


Figure 16 - Comparison of the different networks for CO₂ purity, for the strong dynamics region (above) and the smooth dynamics region (below).

In this case, through visual inspection, it is difficult to tell which model presents the best performance, since all of them produced precise predictions. However, in both NOE and NARX

a small offset is present between the steady state predicted and the actual one. On the other hand, LSTM was able to predict the steady state accurately, presenting an overall performance slight superior to the other models.

The results for the CO₂ recovery are displayed in Figure 17:

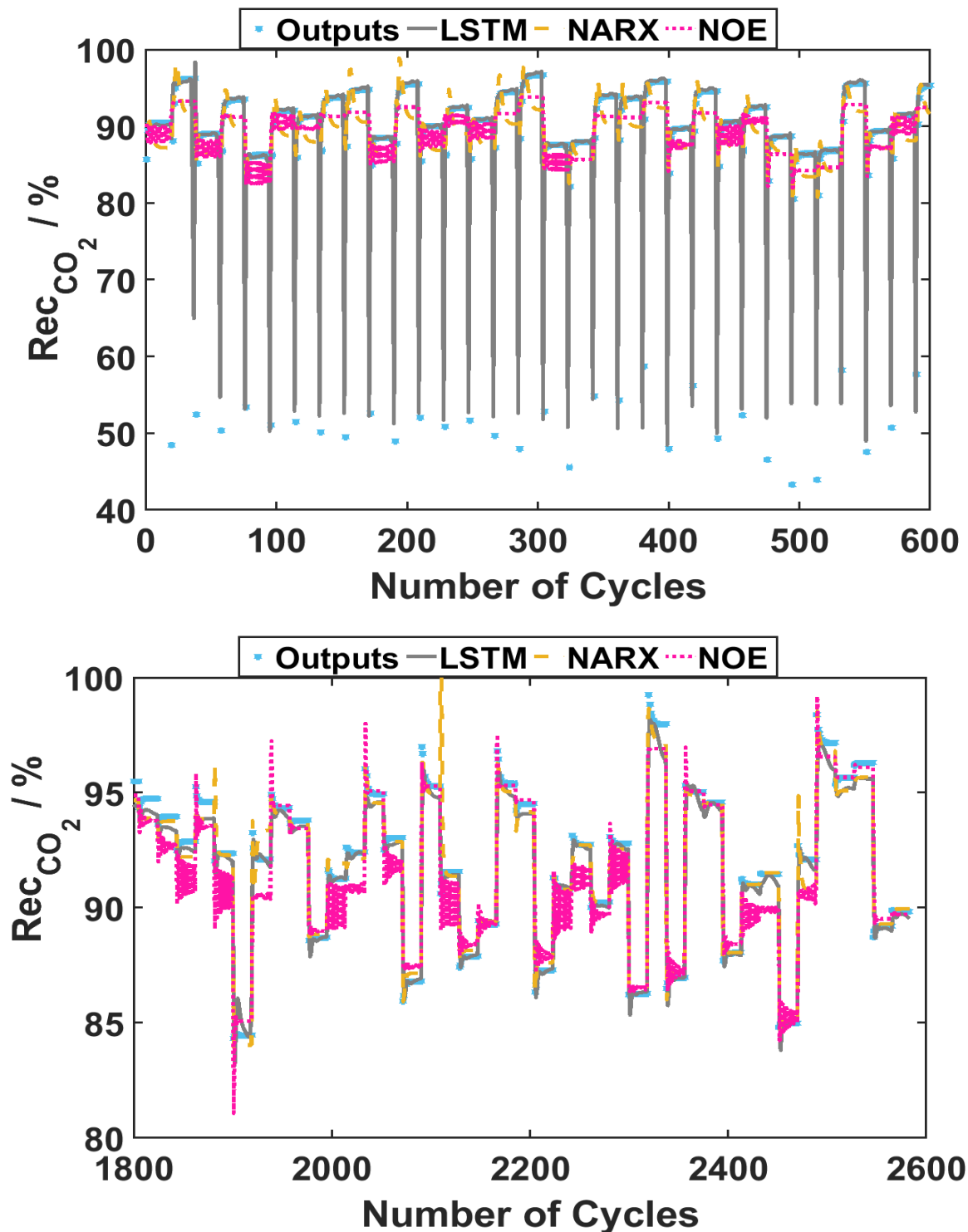


Figure 17 - Comparison of the different models for CO₂ recovery, for the strong dynamics region (above) and the smooth dynamics region (below).

The LSTM network was once again able to make acceptable predictions of the process dynamics, while the classical and machine learning networks were not able to. The dense regions on both

graphs in the NOE predictions are oscillations produced by the model, again showing how the model can often produce unstable results. LSTM networks appear to have achieved the best performance, as the NARX model did not approximate the variations in the dynamics of the CO₂ recovery as well as LSTM.

The results for the CO₂ productivity are shown in Figure 18:

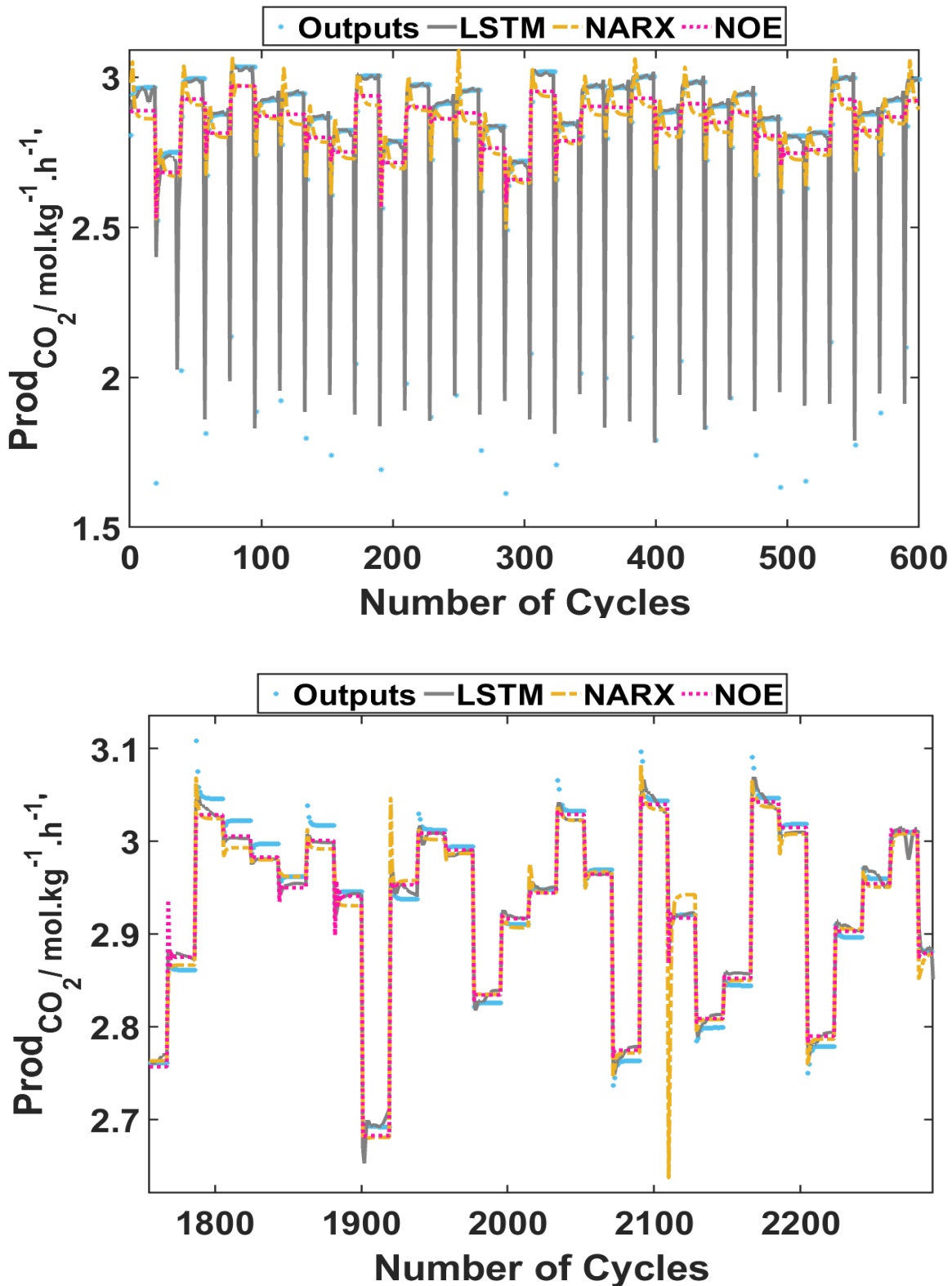


Figure 18 - Comparison of the different models for CO₂ productivity, for the strong dynamics region (above) and the smooth dynamics region (below).

The results are of a similar nature to the ones obtained for the CO₂ recovery, with LSTM appearing to achieve better results across all the conditions tested, in spite of a small offset at some instants. The most significant difference is the improvement of the NOE performance, which seems to be superior to the NARX model in this case.

4.7 Phenomenological model vs Empirical model

Finally, the simulation time of the phenomenological model was compared to the one of the empirical models, for the whole validation set. The simulations were run on the same computer with an Intel Core i5-2400 processor and 8 GB of RAM memory for the case of the phenomenological model it took 3 hours, while for the empirical models it took about 1 second. This difference is important for application in real time optimization, inference and control strategies. Furthermore, deep learning was able to represent with precision the full process dynamics when compared with phenomenological model, and this is an essential characteristic to address PSA dynamic related issues.

5 Conclusions

In this work, three sets of ANN-based empirical models were developed to model a PSA process with the goal of adjusting the H_2/CO ratio of a syngas mixture. Each set of ANNs presented a Multi-MISO system, with each of the four ANNs receiving information on eight process inputs and predicting one process output. Thus, a total of 12 Artificial Intelligence models were developed in the present work. The models were developed based on the historical advances on the Artificial Intelligence field, one model based on the classical approach, another based on machine learning and a final one based on deep learning. The performance of each model was studied and discussed, and the models were compared to each other.

The deep learning models generally showed better performances than the classical model and machine learning model, as the DNNs were able to predict precisely the entire dynamical behaviour of the PSA unit for all outputs, whereas the others were only able to achieve a close approximation of the process dynamics. Between the classical model and the machine learning, NARX performed better when compared to NOE, which displayed inconsistent results, putting its predictive capabilities into question. The overall poor behaviour of the machine learning model is due the gradient explosion/vanishing problems related to the field. Precisely to address those problems the DNNs where developed.

A comparison was also established between the empirical models and the phenomenological ones, and the former achieved dramatically lower simulation times than the latter.

It is therefore concluded that the LSTM-based DNNs can be reliable predictors of the process' dynamical behaviour, and may be a good option for the development of control, optimization and on-line measurement strategies. On the other hand, the classical networks showed considerable limitations in handling the PSA unit's complex dynamic behaviour.

Future works

Examples of possible projects that may expand on the results of this work include:

- Optimization of the LSTM networks;
- Employing empirical models on a nonlinear model predictive control system;
- Optimization of the PSA unit based on the built models;
- Development of systems with on-board empirical models, to be employed on a laboratorial installation, that can provide real-time information about the process.

6 References

- Abraham, M. A. (2017). *Encyclopedia of sustainable technologies*. In (pp. 1 online resource (4 volumes)).
- Aggarwal, C. C. (2018). *Neural networks and deep learning : a textbook*. In (pp. 1 online resource (xxiii, 497 pages)). Retrieved from <http://dx.doi.org/10.1007/978-3-319-94463-0> MIT Access Only doi:10.1007/978-3-319-94463-0
- Bakshi, B. R., & Stephanopoulos, G. (1993). Wave-net: a multiresolution, hierarchical neural network with localized learning. *AIChE Journal*, 39(1), 57-81. doi:10.1002/aic.690390108
- Bañares-Alcántara, R., Westerberg, A. W., Ko, E. I., & Rychener, M. D. (1987). Decade—A hybrid expert system for catalyst selection—I. Expert system consideration. *Computers & Chemical Engineering*, 11(3), 265-277. doi:[https://doi.org/10.1016/0098-1354\(87\)85008-1](https://doi.org/10.1016/0098-1354(87)85008-1)
- Bañares-Alcántara, R., Westerberg, A. W., & Rychener, M. D. (1985). Development of an expert system for physical property predictions. *Computers & Chemical Engineering*, 9(2), 127-142. doi:[https://doi.org/10.1016/0098-1354\(85\)85003-1](https://doi.org/10.1016/0098-1354(85)85003-1)
- Basheer, I. A., & Hajmeer, M. (2000). Artificial neural networks: fundamentals, computing, design, and application. *Journal of Microbiological Methods*, 43(1), 3-31. doi:[https://doi.org/10.1016/S0167-7012\(00\)00201-3](https://doi.org/10.1016/S0167-7012(00)00201-3)
- Cannady, J. (1998). *Artificial Neural Networks for Misuse Detection*.
- Chen, J., & Huang, T.-C. (2004). Applying neural networks to on-line updated PID controllers for nonlinear process control. *Journal of Process Control*, 14(2), 211-230. doi:[https://doi.org/10.1016/S0959-1524\(03\)00039-8](https://doi.org/10.1016/S0959-1524(03)00039-8)
- Chihara, K., & Suzuki, M. (1983). AIR DRYING BY PRESSURE SWING ADSORPTION. *Journal of Chemical Engineering of Japan*, 16(4), 293-299. doi:10.1252/jcej.16.293
- Christensen, T. S., & Primdahl, I. I. (1994). Improve syngas production using autothermal reforming.
- Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*.
- Deng, L., Li, J., Huang, J., Yao, K., Yu, D., Seide, F., . . . Acero, A. (2013, 26-31 May 2013). *Recent advances in deep learning for speech research at Microsoft*. Paper presented at the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing.
- Fletcher, R. (1987). *Practical methods of optimization* (2nd ed.). Chichester ; New York: Wiley.
- Gani, R. (2004). Chemical product design: challenges and opportunities. *Computers & Chemical Engineering*, 28(12), 2441-2457. doi:<https://doi.org/10.1016/j.compchemeng.2004.08.010>
- Gardner, M. W., & Dorling, S. R. (1998). Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric Environment*, 32(14), 2627-2636. doi:[https://doi.org/10.1016/S1352-2310\(97\)00447-0](https://doi.org/10.1016/S1352-2310(97)00447-0)
- Gomes, V. G., & Yee, K. W. K. (2002). Pressure swing adsorption for carbon dioxide sequestration from exhaust gases. *Separation and Purification Technology*, 28(2), 161-171. doi:[https://doi.org/10.1016/S1383-5866\(02\)00064-3](https://doi.org/10.1016/S1383-5866(02)00064-3)

- Gontarski, C. A., Rodrigues, P. R., Mori, M., & Prenem, L. F. (2000). Simulation of an industrial wastewater treatment plant using artificial neural networks. *Computers & Chemical Engineering*, 24(2), 1719-1723. doi:[https://doi.org/10.1016/S0098-1354\(00\)00449-X](https://doi.org/10.1016/S0098-1354(00)00449-X)
- Hamed, M. M., Khalafallah, M. G., & Hassanien, E. A. (2004). Prediction of wastewater treatment plant performance using artificial neural networks. *Environmental Modelling & Software*, 19(10), 919-928. doi:<https://doi.org/10.1016/j.envsoft.2003.10.005>
- Hassibi, B., Stork, D. G., & Wolff, G. J. (1993, 28 March-1 April 1993). *Optimal Brain Surgeon and general network pruning*. Paper presented at the IEEE International Conference on Neural Networks.
- Helton, J. C., & Davis, F. J. (2003). Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. *Reliability Engineering & System Safety*, 81(1), 23-69. doi:[https://doi.org/10.1016/S0951-8320\(03\)00058-9](https://doi.org/10.1016/S0951-8320(03)00058-9)
- Hernández, E., & Arkun, Y. (1992). Study of the control-relevant properties of backpropagation neural network models of nonlinear dynamical systems. *Computers & Chemical Engineering*, 16(4), 227-240. doi:[https://doi.org/10.1016/0098-1354\(92\)80044-A](https://doi.org/10.1016/0098-1354(92)80044-A)
- Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A., Jaitly, N., . . . Kingsbury, B. (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine*, 29(6), 82-97. doi:[10.1109/MSP.2012.2205597](https://doi.org/10.1109/MSP.2012.2205597)
- Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5), 359-366. doi:[https://doi.org/10.1016/0893-6080\(89\)90020-8](https://doi.org/10.1016/0893-6080(89)90020-8)
- Hunt, K. J., Sbarbaro, D., Żbikowski, R., & Gawthrop, P. J. (1992). Neural networks for control systems—A survey. *Automatica*, 28(6), 1083-1112. doi:[https://doi.org/10.1016/0005-1098\(92\)90053-I](https://doi.org/10.1016/0005-1098(92)90053-I)
- Huntington, D. E., & Lyrantzis, C. S. (1998). Improvements to and limitations of Latin hypercube sampling. *Probabilistic Engineering Mechanics*, 13(4), 245-253. doi:[https://doi.org/10.1016/S0266-8920\(97\)00013-1](https://doi.org/10.1016/S0266-8920(97)00013-1)
- Karunanithi, N., Whitley, D., & Malaiya, Y. K. (1992). Using neural networks in reliability prediction. *IEEE Software*, 9(4), 53-59. doi:[10.1109/52.143107](https://doi.org/10.1109/52.143107)
- Koivisto, H. (1995). *A Practical Approach to Model Based Neural Network Control* (Tese de doutoramento).
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521, 436. doi:[10.1038/nature14539](https://doi.org/10.1038/nature14539)
- Marda, J. R., DiBenedetto, J., McKibben, S., Evans, R. J., Czernik, S., French, R. J., & Dean, A. M. (2009). Non-catalytic partial oxidation of bio-oil to synthesis gas for distributed hydrogen production. *International Journal of Hydrogen Energy*, 34(20), 8519-8534. doi:<https://doi.org/10.1016/j.ijhydene.2009.07.099>
- McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4), 115-133. doi:[10.1007/BF02478259](https://doi.org/10.1007/BF02478259)

- McKay, M. D., Beckman, R. J., & Conover, W. J. (1979). A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, 21(2), 239-245. doi:10.2307/1268522
- Metz, C. (2016a). APPLE IS BRINGING THE AI REVOLUTION TO YOUR IPHONE. Retrieved from <https://www.wired.com/2016/06/apple-bringing-ai-revolution-iphone/>
- Metz, C. (2016b). AN INFUSION OF AI MAKES GOOGLE TRANSLATE MORE POWERFUL THAN EVER. Retrieved from <https://www.wired.com/2016/09/google-claims-ai-breakthrough-machine-translation/>
- Nelles, O. (2001). *Nonlinear system identification : from classical approaches to neural networks and fuzzy models*. Berlin ; New York: Springer.
- Nogueira, I. (2018). *Optimization and Control of TMB, SMB and SMBR units* (Tese de doutoramento).
- Nogueira, I. B. R., Ribeiro, A. M., Requião, R., Pontes, K. V., Koivisto, H., Rodrigues, A. E., & Loureiro, J. M. (2018). A quasi-virtual online analyser based on an artificial neural networks and offline measurements to predict purities of raffinate/extract in simulated moving bed processes. *Applied Soft Computing*, 67, 29-47. doi:<https://doi.org/10.1016/j.asoc.2018.03.001>
- Nørgaard, M. (2000). Neural Network Based System Identification Toolbox, Version 2.0. *Tech. Report. 00-E-891, Department of Automation, Technical University of Denmark, 2000.*
- Oh, K.-S., & Jung, K. (2004). GPU implementation of neural networks. *Pattern Recognition*, 37(6), 1311-1314. doi:<https://doi.org/10.1016/j.patcog.2004.01.013>
- Pal, S. K., & Mitra, S. (1992). Multilayer perceptron, fuzzy sets, and classification. *IEEE Transactions on Neural Networks*, 3(5), 683-697. doi:10.1109/72.159058
- Process Systems Enterprise. <https://www.psenderprise.com/products/gproms>. 2015.
- Rafiq, M. Y., Bugmann, G., & Easterbrook, D. J. (2001). Neural network design for engineering applications. *Computers & Structures*, 79(17), 1541-1552. doi:[https://doi.org/10.1016/S0045-7949\(01\)00039-6](https://doi.org/10.1016/S0045-7949(01)00039-6)
- Raina, R., Madhavan, A., & Ng, A. Y. (2009). *Large-scale deep unsupervised learning using graphics processors*. Paper presented at the Proceedings of the 26th Annual International Conference on Machine Learning, Montreal, Quebec, Canada.
- Regufe, M. J., Tamajon, J., Ribeiro, A. M., Ferreira, A., Lee, U. H., Hwang, Y. K., . . . Rodrigues, A. E. (2015). Syngas Purification by Porous Amino-Functionalized Titanium Terephthalate MIL-125. *Energy & Fuels*, 29(7), 4654-4664. doi:10.1021/acs.energyfuels.5b00975
- Rostrup-Nielsen, J., & Christiansen, L. J. (2011). *Concepts in syngas manufacture*. London: Imperial College Press.
- Rostrup-Nielsen, J. R. (1984). *Catalytic steam reforming*. Berlin: Springer-Verlag.
- Ruthven, D. M. (1984). *Principles of adsorption and adsorption processes*. New York: Wiley.
- Ruthven, D. M., Farooq, S., & Knaebel, K. S. (1994). *Pressure swing adsorption*. New York, N.Y.: VCH Publishers.
- Sant Anna, H. R., Barreto, A. G., Tavares, F. W., & de Souza, M. B. (2017). Machine learning model and optimization of a PSA unit for methane-nitrogen separation. *Computers & Chemical Engineering*, 104, 377-391. doi:<https://doi.org/10.1016/j.compchemeng.2017.05.006>

- Schenker, B., & Agarwal, M. (1996). Cross-validated structure selection for neural networks. *Computers & Chemical Engineering*, 20(2), 175-186. doi:[https://doi.org/10.1016/0098-1354\(95\)00013-R](https://doi.org/10.1016/0098-1354(95)00013-R)
- Sirola, J. J., & Rudd, D. F. (1971). Computer-Aided Synthesis of Chemical Process Designs. From Reaction Path Data to the Process Task Network. *Industrial & Engineering Chemistry Fundamentals*, 10(3), 353-362. doi:10.1021/i160039a003
- Sircar, S., & Golden, T. C. (2000). Purification of Hydrogen by Pressure Swing Adsorption. *Separation Science and Technology*, 35(5), 667-687. doi:10.1081/SS-100100183
- Sjöberg, J., & Ljung, L. (1995). Overtraining, regularization and searching for a minimum, with application to neural networks. *International Journal of Control*, 62(6), 1391-1407. doi:10.1080/00207179508921605
- Ungar, L. H., Powell, B. A., & Kamens, S. N. (1990). Adaptive networks for fault diagnosis and process control. *Computers & Chemical Engineering*, 14(4), 561-572. doi:[https://doi.org/10.1016/0098-1354\(90\)87027-M](https://doi.org/10.1016/0098-1354(90)87027-M)
- Venkatasubramanian, V. (2019). The promise of artificial intelligence in chemical engineering: Is it here, finally? *AIChE Journal*, 65(2), 466-478. doi:10.1002/aic.16489
- Ye, F., Ma, S., Tong, L., Xiao, J., Bénard, P., & Chahine, R. (2019). Artificial neural network based optimization for hydrogen purification performance of pressure swing adsorption. *International Journal of Hydrogen Energy*, 44(11), 5334-5344. doi:<https://doi.org/10.1016/j.ijhydene.2018.08.104>
- Yu, H., & Wilamowski, B. (2011). Hao Yu and B. M. Wilamowski, LevenbergMarquardt Training Industrial Electronics Handbook, vol. 5 Intelligent Systems, 2nd Edition, chapter 12, pp. 12-1 to 12-15, CRC Press 2011. In (pp. 12-11 to 12).
- Zhang, G., Eddy Patuwo, B., & Y. Hu, M. (1998). Forecasting with artificial neural networks:: The state of the art. *International Journal of Forecasting*, 14(1), 35-62. doi:[https://doi.org/10.1016/S0169-2070\(97\)00044-7](https://doi.org/10.1016/S0169-2070(97)00044-7)
- Zhang, G. P. (2003). Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*, 50, 159-175. doi:[https://doi.org/10.1016/S0925-2312\(01\)00702-0](https://doi.org/10.1016/S0925-2312(01)00702-0)
- Zhang, R., Chen, Z., Chen, S., Zheng, J., Büyükköztürk, O., & Sun, H. (2019). Deep long short-term memory networks for nonlinear structural seismic response prediction. *Computers & Structures*, 220, 55-68. doi:<https://doi.org/10.1016/j.compstruc.2019.05.006>

Appendix A1. Validation results for each network model

A1.1 NARX

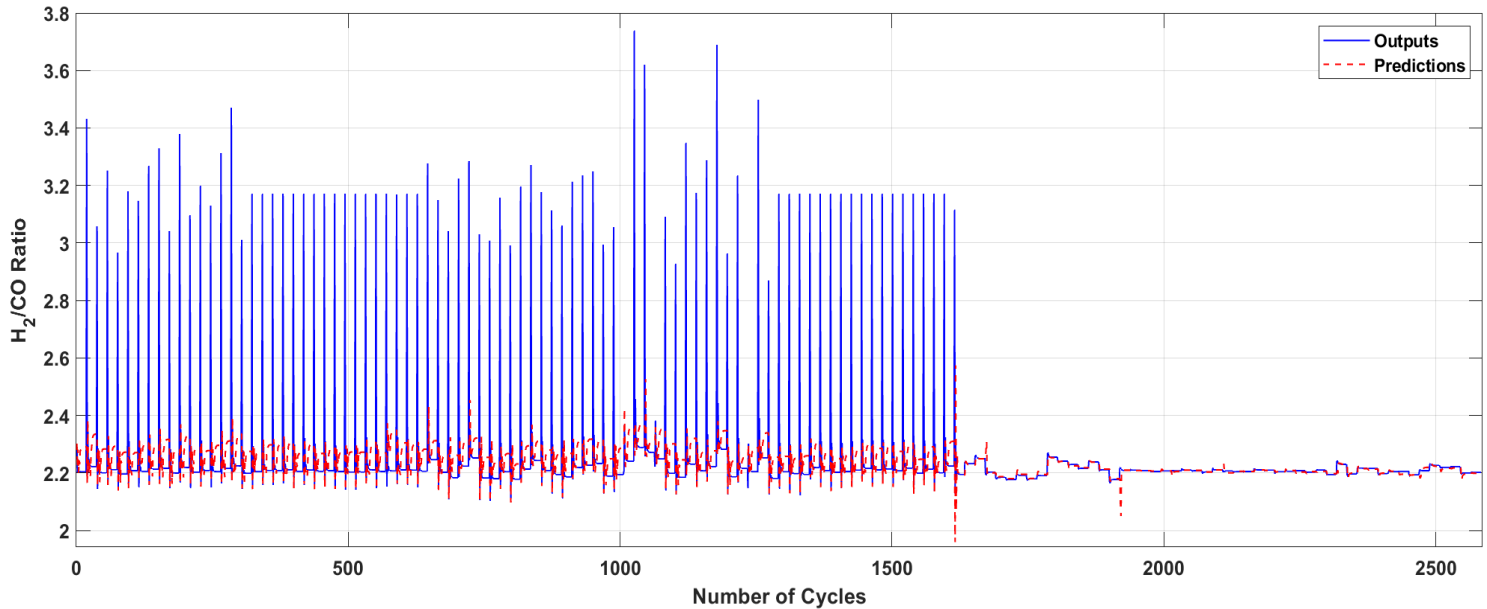


Figure A1.1 - Validation results for the H_2/CO ratio (NARX model).

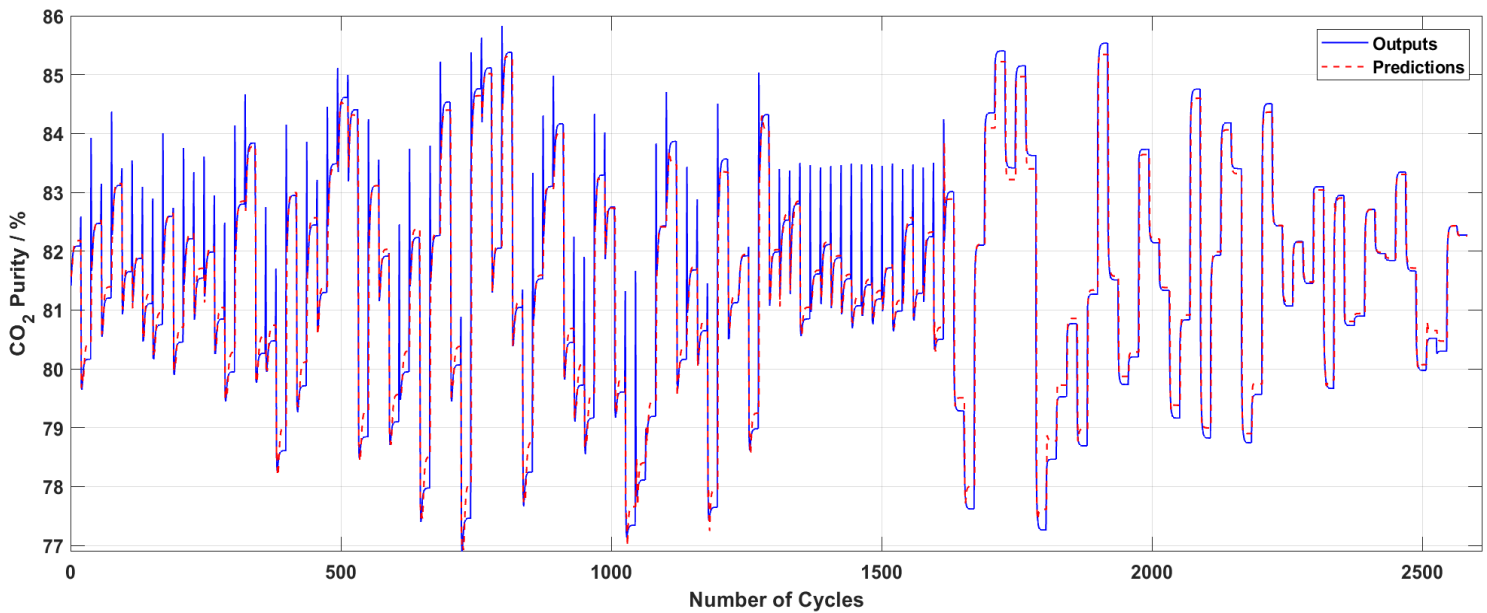


Figure A1.2 - Validation results for the CO_2 purity (NARX model).

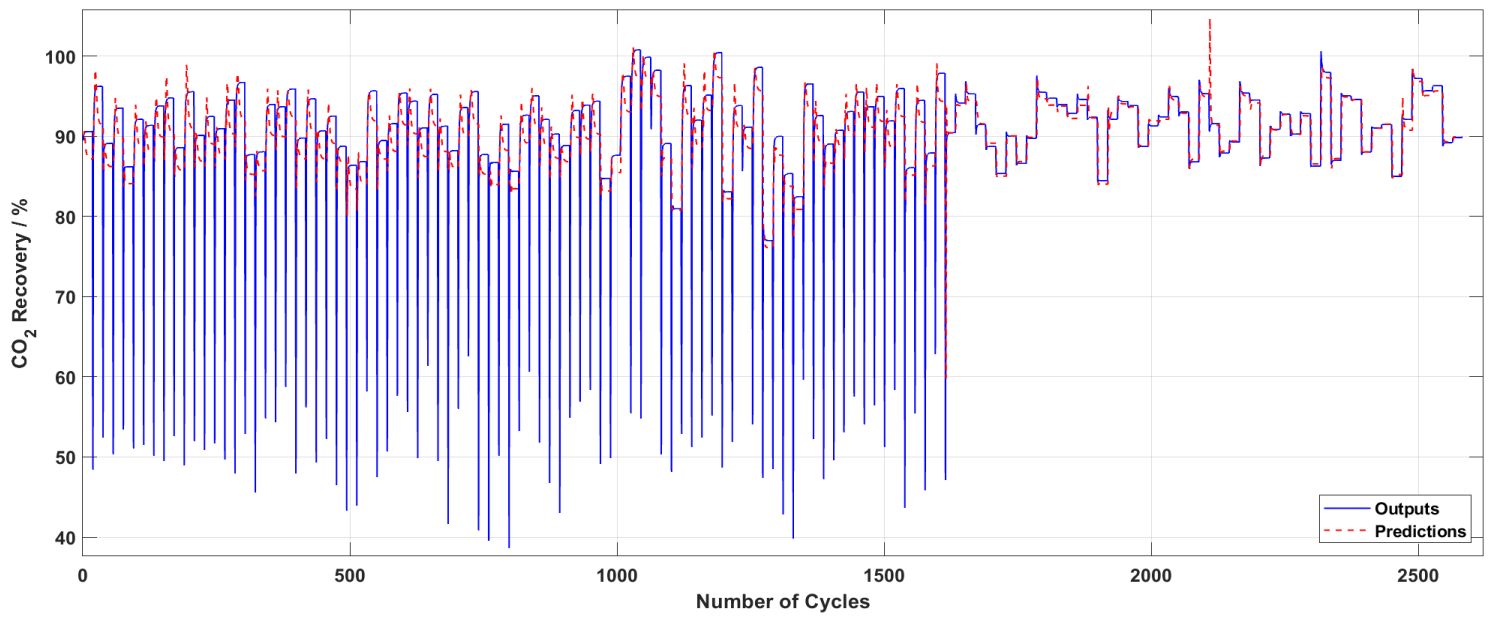


Figure A1.3 - Validation results for the CO₂ recovery (NARX model).

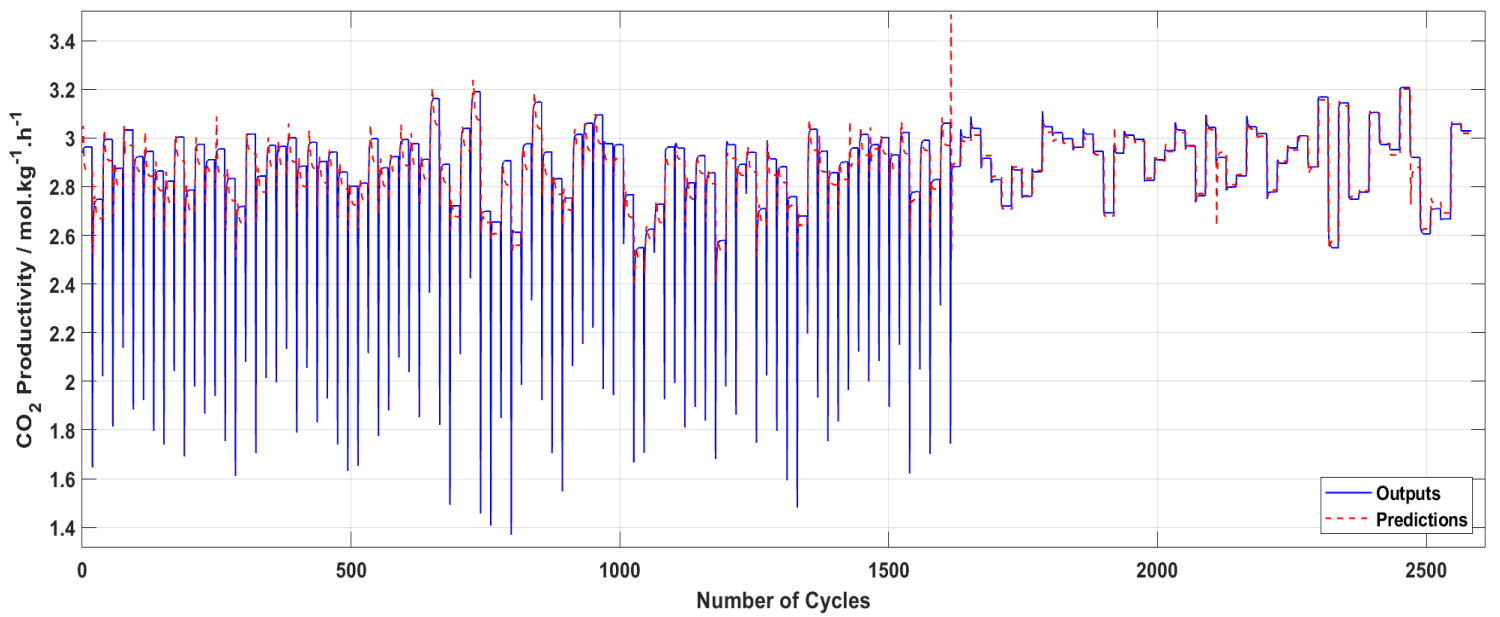


Figure A1.4 - Validation results for the CO₂ productivity (NARX model).

A1.2 NOE

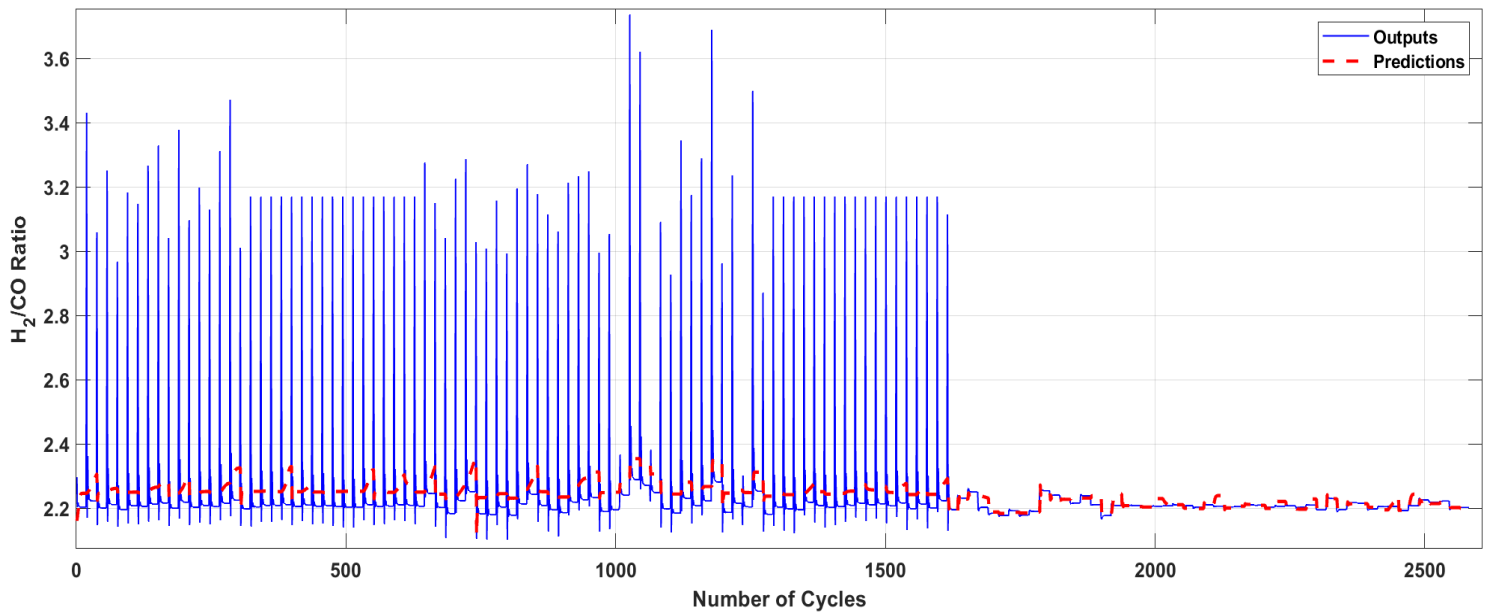


Figure A1.5 - Validation results for the H_2/CO ratio (NOE model).

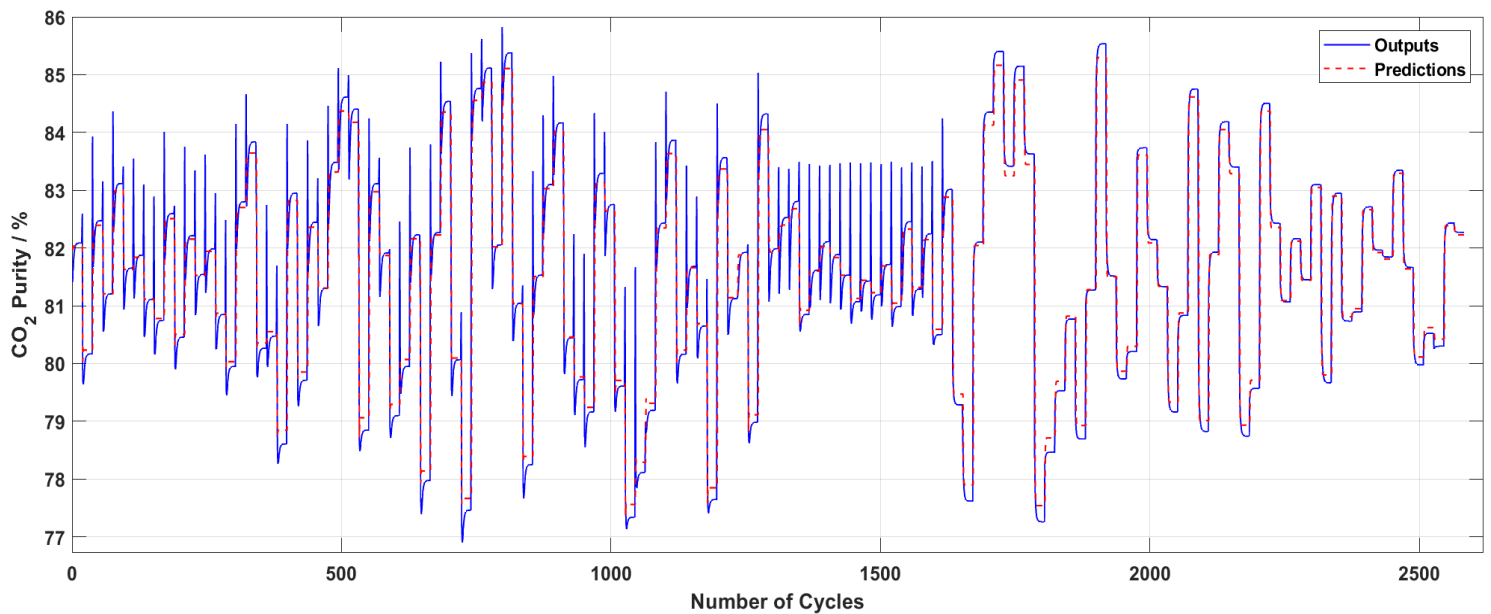


Figure A1.6 - Validation results for the CO_2 purity (NOE model).

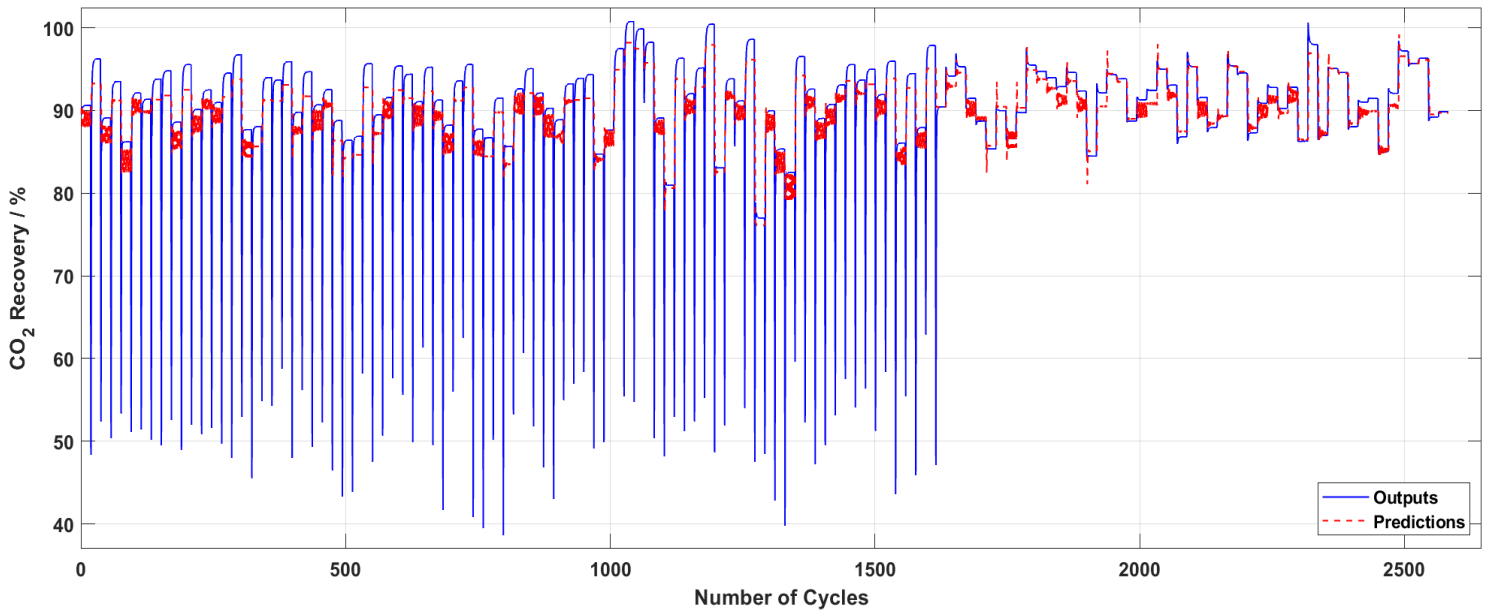


Figure A1.7 - Validation results for the CO_2 recovery (NOE model).

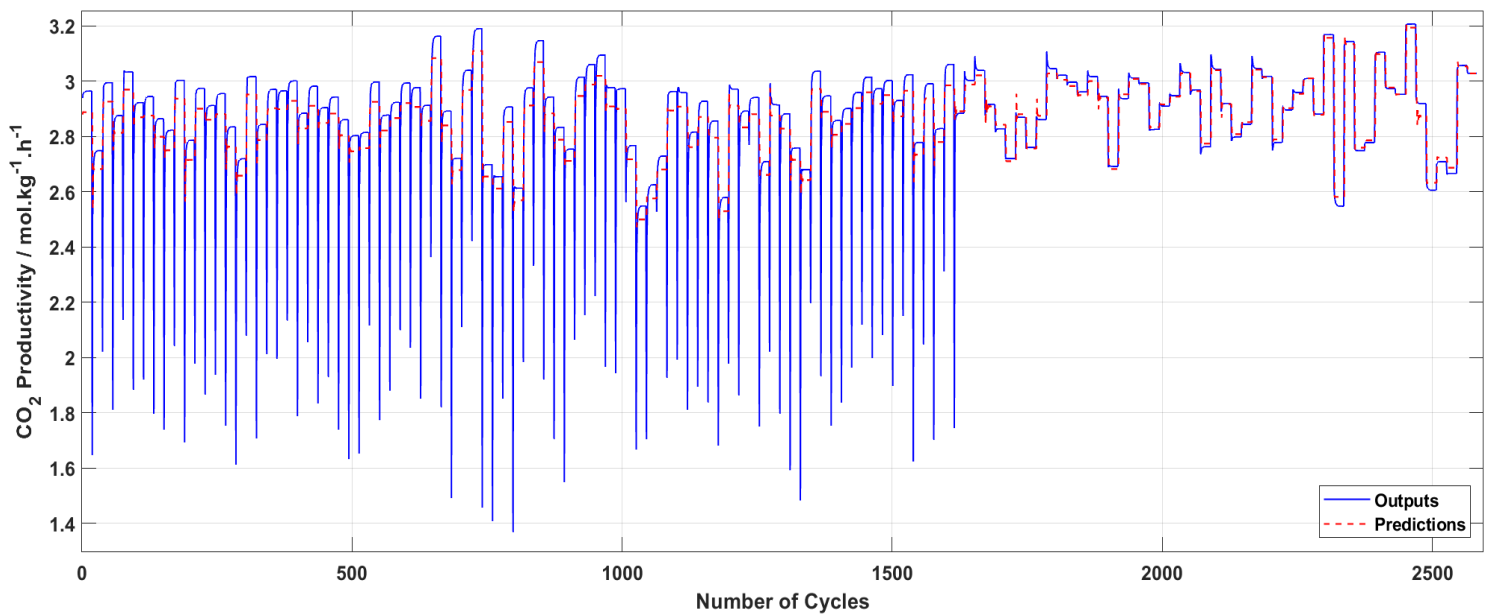


Figure A1.8 - Validation results for the CO_2 productivity (NOE model).

A1.3 LSTM

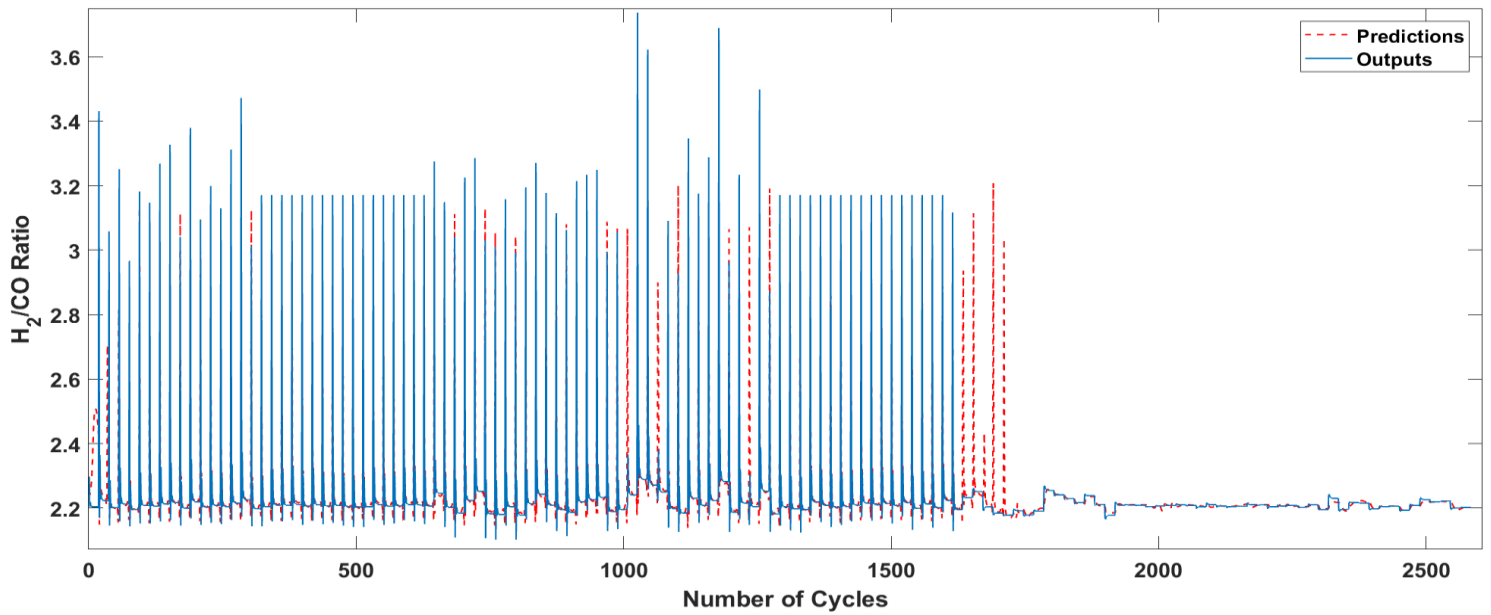


Figure A1.9 - Validation results for the H_2/CO ratio (LSTM model).

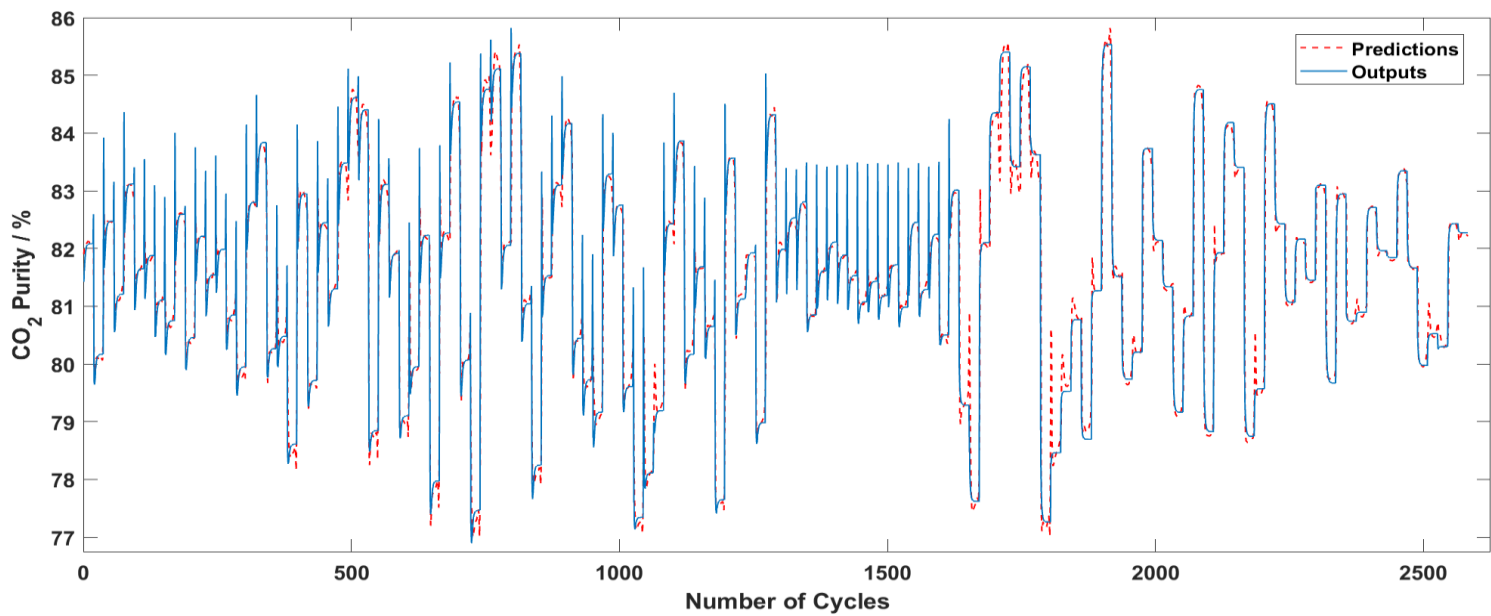


Figure A1.10 - Validation results for the CO_2 purity (LSTM model).

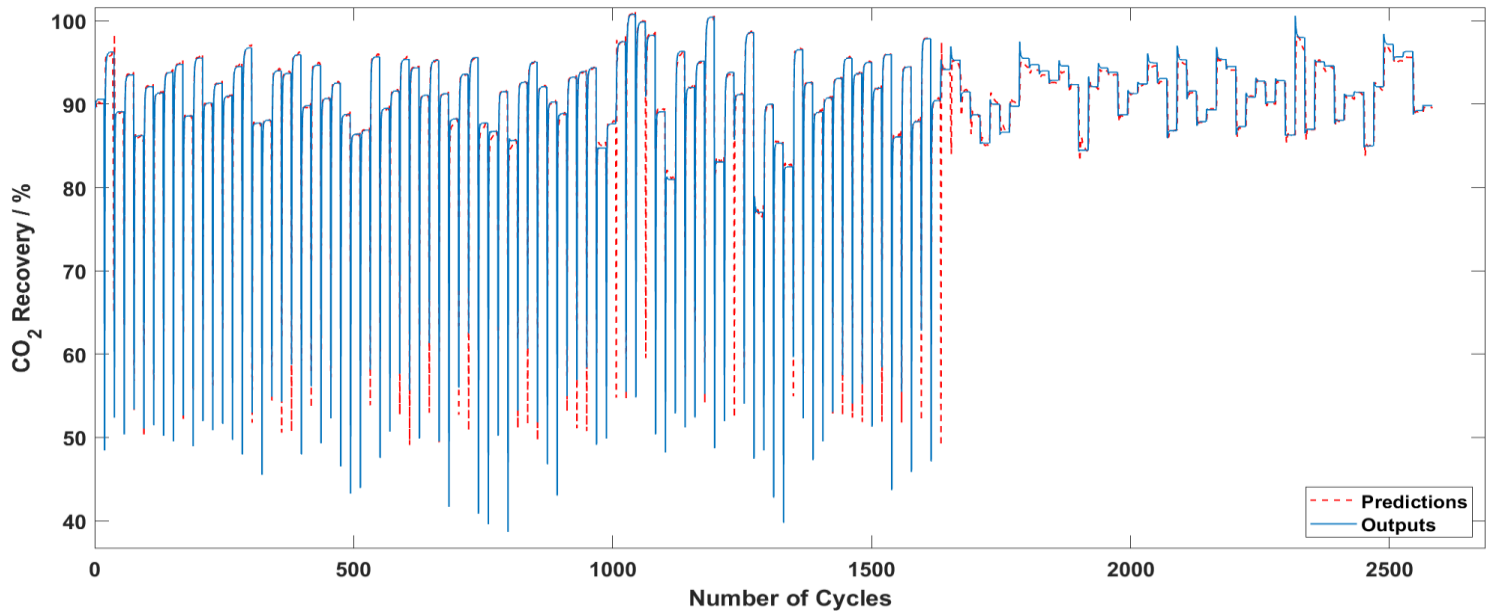


Figure A1.11 - Validation results for the CO₂ recovery (LSTM model).

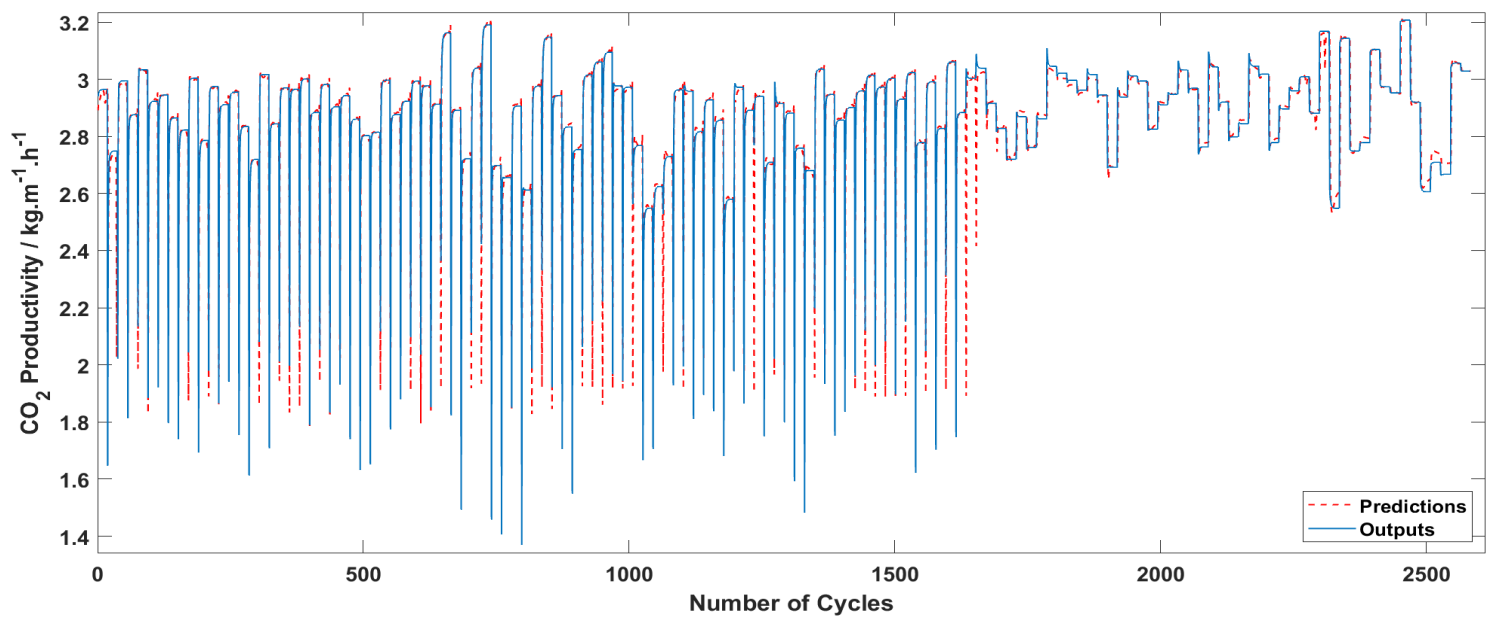


Figure A1.12 - Validation results for the CO₂ productivity (LSTM model).