STEADY-STATE MODELLING OF A PIPELINE-RISER SYSTEM USING FEEDFORWARD NEURAL NETWORKS

PROJECT THESIS

Author:
Simen Bjorvand

Supervisor:
Sigurd Skogestad

Co-supervisor:
Dinesh Krishnamoorthy

Norwegian University of Science and Technology

December 19, 2019

Abstract

Severe slugging in the North Sea offshore oil production is a costly problem, which can cause plant trips and even shut-downs. Slugging appears in risers connected to subsea pipelines. Making a steady-state model of a riser-pipeline system using machine learning, with experimental data from a small-scale anti-slug control rig was the objective of this project. A virtual flowmeter model was made where gas and liquid flow rates were predicted from pressure measurements and the choke valve position. Multiphase flow measurements are important to optimise oil and gas production, and virtual flowmeters could be a beneficial alternative to multiphase flow meters. Feedforward neural network models were used to make the virtual flow meter model. Three approaches were tried, making a baseline model, combining first principle models with deep learning and transfer learning where a model trained on simulation data was used as the source model. The baseline model was good at predicting the liquid flow rate but failed at predicting the gas flow rate. The bad prediction of the gas flow rate was attributed to the data set having too little variation in the gas flow rate. Combining a first principle model with deep learning did not improve predictions of the gas flow rate, neither did the transfer models. To conclude, modelling the multiphase flow rates using deep learning was not successful, as none of the approaches managed to model the gas flow rate. The liquid flow rate was however convincingly estimated. To improve the performance producing more lab data, covering a larger gas flow range is recommended.

Contents

Table of Contents iii

List of Tables iv

List of Figures vii

Abbreviations viii

List of Symbols ix

1 Introduction 1

2 Theory 1
2.1 Slug flow . . . 1
2.1.1 Riser slugging . . . 2
2.2 Anti-slug control . . . 3
2.3 Jahanshahis simplified four-state model . . . 3
2.4 Surrogate Modeling . . . 4
2.5 Machine learning . . . 4
2.6 Deep learning . . . 5

3 Method 6

4 Baseline feedforward neural network model 7
4.1 Results and Discussion . . . 8

5 Feature engineering 12
5.1 Global feedforward network . . . 12
5.1.1 Results and Discussion . . . 13

6 Transfer learning 16
6.1 First approach . . . 17
6.1.1 Results . . . 17
6.2 Second approach . . . 20
6.2.1 Result . . . 21
6.3 Third approach . . . 22
6.3.1 Results . . . 22
6.4 Discussion . . . 25

7 Conclusion 26
7.1 Further work . . . 27

Bibliography 28

Appendix **i**

A Esmaeils four-state model **i**
A.1 Dynamic equations i
A.2 Tuning parameters i
A.3 Outflow conditions i
A.4 Pipeline model ii
A.5 Riser model iii
A.6 Gas flow model at riser base iii
A.7 Liquid flow model at riser base iii
A.8 Phase distribution model at outlet choke valve iv

B Simplified model **iv**

C Experimental setup **v**

D Online radial basis function network **vi**

E Training set prediction plots **viii**
E.1 Baseline model viii
E.2 Feature model xi
E.3 Transfer models xiii

F Pressure drop plots **xix**

List of Tables

4.1 Performance table for the three models 8
5.1 Performance table for the updated model, comparing with the performance of simplified FPM 13
6.1 Performance table for the two transfer model compared with the simulation model. 18
6.2 Performance table for the transfer model compared with the simulation model . . . 21
6.3 Performance table for the transfer model compared with the simulation model. . . . 23

List of Figures

2.1 Riser slug cycle[7] 3
2.2 Pipeline-riser system[6] 4
2.3 Graphical representation of a feedforward neural network 6
4.1 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a model trained on raw experimental data, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ 9
4.2 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a model trained on filtered input data, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ . . 10
4.3 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a model trained on averaged input data, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ . . 11
5.1 Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates and the simplified FPM gas and liquid flow rate predictions, for bias prediction model, for test set. Inputs, targets, FPM predictions and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ . . 14
5.2 Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for bias prediction model, for the test set. Inputs, targets, and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ 15
6.1 Three ways transfer learning can improve learning[11] 17
6.2 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for the simulation model, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ 18
6.3 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a transfer model where the last layer was retrained on experimental data, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ 19
6.4 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a transfer model where the last two layers were retrained on experimental data, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ 20
6.5 Transfer learning approach two model structure 21
6.6 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a transfer model which uses predictions from simulation network as extra input, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ 22

6.7 Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates and the simulation model gas and liquid flow rate predictions, for a bias prediction model, on the test set. Inputs, targets, simulation model predictions and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ 24
6.8 Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a bias prediction model, on the test set. Inputs, targets and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ 25
C.1 Experimental setup of Anti-slug control rig v
D.1 Performance plot of the average updated prediction for each steady-state section compared with the average first principle model predictions. vii
E.1 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for model trained on raw experimental data, on the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ viii
E.2 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for model trained on filtered input data, on the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ . . ix
E.3 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for model trained on averaged input data, on the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ . . x
E.4 Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates and the simplified FPM gas and liquid flow rate predictions, for bias prediction model, on the train set. Inputs, targets, FPM predictions and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ . . xi
E.5 Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for bias prediction model, on the train set. Inputs, targets, and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ xii
E.6 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for the simulation model, on the the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ xiii
E.7 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a transfer model where the last layer was retrained on experimental data, on the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ xiv
E.8 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a transfer model where the last two layers were retrained on experimental data, on the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ xv

E.9 Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a transfer model which uses predictions from simulation network as extra input, on the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ xvi
E.10 Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates and the simulation model gas and liquid flow rate predictions, for a bias prediction model, on the train set. Inputs, targets, simulation model predictions and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ xvii
E.11 Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a bias prediction model, on the train set. Inputs, targets and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ xviii
F.1 Pressure drop over pipeline and riser for test data set. The pressure drops were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ xix
F.2 Pressure drop over pipeline and riser for train data set. The pressure drops were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$ xx

Abbreviations

MSE	=	mean square error
FPM	=	first principle model
FFNN	=	feed forward neural network
ReLu	=	rectified linear units

List of Symbols

Z	=	Top-side choke valve opening
Q_g	=	Volumetric gas flow rate
Q_l	=	Volumetric liquid flow rate
$P1$	=	Buffer tank pressure
$P2$	=	Riser top point pressure
$P3$	=	Mixing point pressure
$P4$	=	Riser low point pressure
$m_{g,p}$	=	mass of gas in pipeline
$m_{l,p}$	=	mass of liquid in pipeline
$m_{g,r}$	=	mass of gas in riser
$m_{l,r}$	=	mass of liquid in riser
X	=	Input data set
Y	=	Output data set
x	=	Input
y	=	Output

1 Introduction

In this project, the objective was to make a steady-state model, of a slug-flow system using feedforward neural networks. Different pressure measurements in the system alongside the topside choke valve position was used to predict volumetric gas and liquid flow. The data-driven deep learning model were made using data produced in a small-scale anti slug lab at NTNU. The data used in this project was produced as a part of a summer job for SUBPRO. Three approaches to solving this problem would be attempted:

- First approach: Make a baseline model, which is a basic feedforward model, made using only experimental data.
- Second approach: Combine first principle knowledge about the system with a feedforward model.
- Third approach: Transfer learning, make simulation data, make a model using simulation data, and repurpose this model to make a model with the experimental data.

The type of model which is to be made is a steady-state virtual flow meter. A virtual flow meter is a mathematical model which estimates multiphase flow rates from available measurements like pressure and temperature[2]. In industry multiphase flow meters are commonly used to estimate multiphase flow, but as they must be physically installed in the pipelines, and maintained, it would be beneficial to model multiphase flow from more available measurements, like pressure measurements. Accurate flow measurements are important for optimal operation of oil and gas production.

2 Theory

2.1 Slug flow

Slug flow is a multiphase flow regime characterised by varying flow through the cross-sections in pipelines. In multiphase flows, the different phases can spatially distribute itself in many different ways dependent on flow conditions, like flow velocity and pipeline angles. These are called flow regimes. There are many different flow regimes in pipelines, some of these are stratified flow, annular flow, bubble flow, slug flow and churn flow. There are different kinds of slug flow.[9]

- **Hydrodynamic slugging**, this is a form of slugging which develops in horizontal pipes. These slugs are created when liquid waves in the gas-liquid interface of the pipelines grow big enough to cover the cross-section of the pipeline and close it off.
- **Riser slugging**, this is a kind of slugging which develops in the bottom of the riser, where it is connected to a down-sloping pipeline. The intersection is blocked by liquid and a slug created which grows up in the riser and back in the pipeline. The slug continuous to grow until the pressure upstream is large enough to blow the slug out of the riser.

- **Terrain slugging** is slugging which occur in pipelines going along rough seafloor. Liquid accumulates in the inclined sections and slugs are built.
- **Transient slugging** is slugging related to changing operating conditions, like changing flow rates, which happens during start-up and shut-down.

2.1.1 Riser slugging

Riser slugging is the most serious slug flow regime for oil/water systems. In the most extreme cases, the riser slugs can be several hundred meters long and fill the entire riser. The separators in the topside facilities are not large enough to handle such large slugs, so if a slug of this size enters the separator units it can cause a plant trip and stop production. Smaller slugs are also problematic as it will lead to poor separation and increased wear and tear on the equipment.[9] Frequent changes in flow cause unwanted flaring and lower the capacity of the separator and compressor units.[5]

The behaviour of riser slugging can be described by a four-step cycle as seen in figure 2.1.

- **Step 1** Liquid accumulates in the low point of the riser, due to gravity and blocks the pipe. For this to happen the gas and liquid velocities must be sufficiently low.
- **Step 2** When the hydrostatic head of the liquid increases faster then the pressure drop over the riser, the slug grows.
- **Step 3** The pressure drop over the riser overcomes the hydrostatic head and the liquid is pushed out of the riser. This is accompanied by a pressure drop, which leads to the gas expanding and increased velocities in the riser.
- **Step 4** Gas starts penetrating the slug and the driving force is too low to push out the remaining liquid. The remaining liquid accumulates in the low point of the riser and the cycle is restarted.

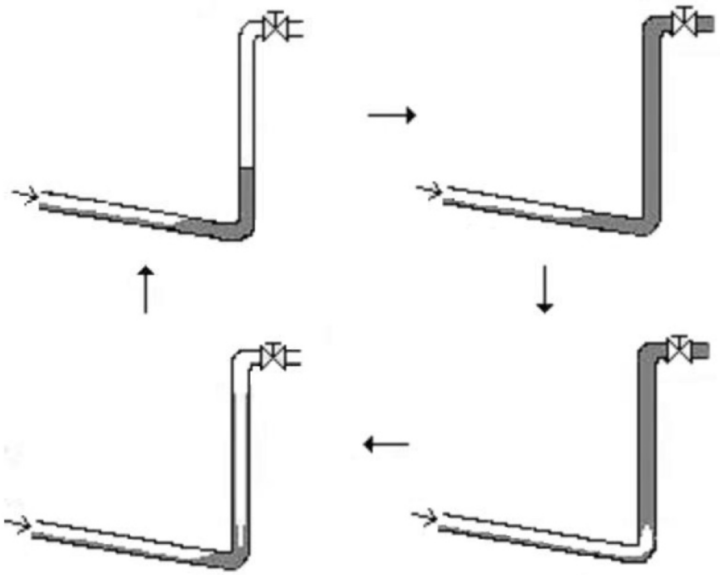

Figure 2.1: Riser slug cycle[7]

2.2 Anti-slug control

Due to the unstable flow behaviour of riser slugging it is necessary to counteract it in some capacity. The conventional ways of counteracting slugging are using slug catchers and topside Choking[6]. A slug catcher is a large tank between the riser and topside processing facility which acts as a buffer for the slugs. This solution is more of a design problem than control problem, and a slug catcher requires a lot of space on the topside facility which might not be available. By closing the topside choke valve severe slugging can be counteracted but at the cost of increasing the back pressures of the subsea oil wells. This reduces the production rate. Alternatively, automatic anti-slug control can be applied. By using anti-slug control the flow regime boundaries can be moved such that slugging regions can become non-slugging regions. This way slugging can be counteracted without significantly reducing the production rate. To design good control systems, good models are needed.

2.3 Jahanshahis simplified four-state model

A simplified first principle model describing cyclic slug flow was previously formulated by Jahanshahi and Skogestad.[6] This model describes the system using four states, the four states being:

- $m_{g,p}$: mass of gas in pipeline
- $m_{l,p}$: mass of liquid in pipeline
- $m_{g,r}$: mass of gas in riser
- $m_{l,r}$: mass of liquid in riser

The changes in the states are described using simple mass balances over each phase in each subsection. The phase distributions in the sections of the model are found using simple relations, with the gas and liquid flows at the riser base being described by an orifice equation. Four tuning parameters are used to fit the model to different riser-pipeline systems. The rest of the model can be found in Appendix A.

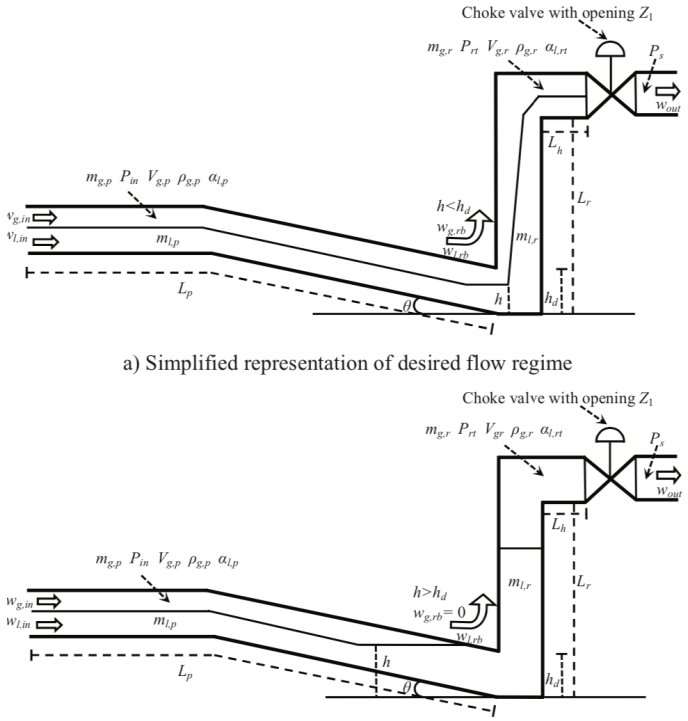

Figure 2.2: Pipeline-riser system[6]

2.4 Surrogate Modeling

An alternative to first principle models is data-driven surrogate models. A surrogate model is a simple numerical representation of a complex model[10]. Surrogate models are also called reduced order models or response surfaces as complex state-space models can be reduced to simple input-output mappings. The benefits of surrogate models are that they can reduce the computational cost, making computations faster. This can be important in applications like model predictive control or real-time optimization. Many different functions can be used to make surrogate models like artificial neural networks.

2.5 Machine learning

Machine learning is the study of algorithms which makes computers able to learn how to solve tasks without having been explicitly programmed. For a computer to solve a task it needs some algorithm, but for some tasks, algorithms are difficult to develop. Machine learning is trying to

make a model by having the computer learn from a data set called ”training data”[1]. In machine learning, a model is defined with some parameters, and during training, the optimal parameters are found. Statistics are used to build models, and the models can be evaluated by testing them on a ”test data set”, which is data the model has not seen during training.

There are different kinds of learning: supervised learning which can include classification and regression, unsupervised learning and reinforcement learning to name a few[1]. In supervised learning, the model is trained using input data **X** with accompanying output labels **Y**, to find some input-output mapping. In classification problems, the goal is to assign an input to a finite number of classes. In regression problems, the goal is to predict continuous outputs based on inputs. In unsupervised learning output labels **Y**, are not available, only input data **X**. The goal is then instead to find patterns in the input data, like clustering where the goal is to find clusters in the input data. In reinforcement learning, the output of the model is a sequence of actions. So the success of the algorithm is if the actions the model perform leads to achieving some goal. This could, for example, be to make a model which had the goal to control the level of a tank by manipulating a valve.

2.6 Deep learning

Deep learning is a class of machine learning algorithms which is based on artificial neural networks. The simplest deep learning model is the feedforward neural network also called deep feedforward network or multilayer perceptron[4]. These models are called feedforward networks because information flows forward through the network from input **x** to output **y**. A network with feedback in which information flows backwards is called recurrent network. A FFNN is composed of several functions in a chain, with each functions acting as a layer of the model, with the number of functions determining the depth of the model. The function of the FFNN model with n layers is,

$$\mathbf{y} = f(\mathbf{x};\theta) = f^n(f^{n-1}(...f^2(f^1(\mathbf{x})))) \tag{2.1}$$

$$\mathbf{z}_i = f^i(\mathbf{z}_{i-1};\theta_i) = g_i(W_i^T\mathbf{z}_{i-1} + b_i) \tag{2.2}$$

where **y** is the output of the model, **x** is the input of the model, f^i representing the ith layer of the model, $\mathbf{z}_i$ being the output of the ith layer, and the input for layer i+1, g_i is the activation function of layer i, θ being the parameters of the model, with θ_i being the parameters of layer i consisting off W_i being the matrix containing the weights from layer i-1 to layer i, and b_i being the biases of each layer. In this case $\mathbf{z}_0 = \mathbf{x}$, and $\mathbf{z}_n = \mathbf{y}$. **x** is the input layer of the model, also called the feature layer. The final layer f^n is called the output layer. The layers in between the input layer and the output layer are called hidden layers as their outputs are hidden. The units in the layers are called neurons, and the amount of neurons in each layer is the width of the layer. A graphical representation of a FFNN can be seen in figure 2.3

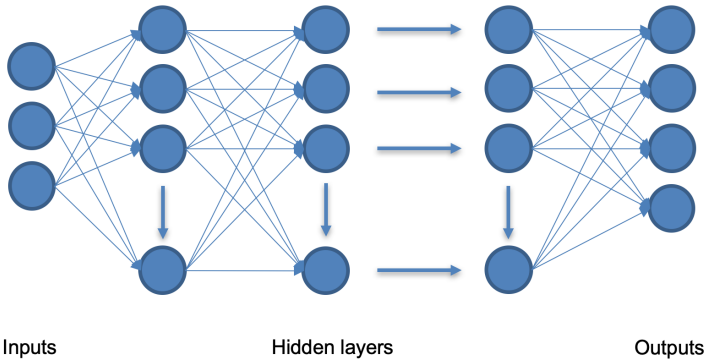

Figure 2.3: Graphical representation of a feedforward neural network

For FFNN there are many different choices for activation functions $g(z)$. For regression models a linear activation function is used for the output layer. For the hidden layers a number of different activation functions can be used. A much used activation function is the rectified linear units activation function[4]. The form of the ReLu activation function is $g(z)$ = max(0,z). Other activation functions used for hidden layers are Sigmoid $g(z) = \frac{1}{1+e^{-z}}$, or the hyperbolic tangent function $g(z)$ = $tanh(z)$. Another activation function is the radial basis function, $g_i(z) = \exp\left(-\beta \left\|x - c_i\right\|^2\right)$, with c_i being the center vector for neuron i. Parameters c_i and β must be found during training.

To train a FFNN to solve some task a performance measurement is needed so that the model can evaluate its own performance. A loss function is needed. For regression FFNN models mean square error is used. There are different kinds of training algorithms used, with stochastic gradient descent and its variants being the most used[4]. In this algorithm, a minibatch of the training data is used to calculate the gradient per step. An important parameter of this algorithm is the learning rate ϵ which affects how much the model parameters are updated per step. The gradient is calculated using an algorithm called back-propagation in which information flows backwards through the network to calculate the gradient. Adam is a variant of the stochastic gradient descent algorithm, which gets its name from adaptive moment[4]. It is an adaptive learning rate optimization algorithm. This algorithm uses the concept of momentum where gradient information from previous steps is used when updating the parameters.

3 Method

The goal of this project was to make a data-driven steady-state model of a pipeline riser system, based on experimental data. Different approaches to achieve this goal were attempted. The machine learning technique used to achieve this goal is deep learning. A virtual flowmeter model was constructed, where the steady-state gas and liquid flow rates were predicted from steady-state pressure measurements and the choke-valve opening.

The data used to make the models was experimental data produced at a small scale Anti-slug control rig at the Department of Chemical Engineering at NTNU. The data was produced in the summer

as a part of a summer internship for SUBPRO. The data produced was time-series data, where the topside choke valve opening Z, volumetric gas flow rate Q_g, volumetric liquid flow rate Q_l, buffer tank pressure $P1$, mixing point pressure $P3$, riser bottom pressure $P4$ and riser top pressure $P2$ was measured. Most of the data was measured with a sampling rate of $10\,\mathrm{Hz}$, but some data was measured with a sampling rate of $1\,\mathrm{Hz}$.

Since a steady-state model was made, only a subsection of the experimental data was used. Slugging occurs when the topside choke valve opening is greater than 15 %, so the data with the topside choke valve opening of 15 or less was used as the data set for this project. Transient data was removed from the data set, and the data set was divided into 63 different time series sections which were assumed to be at steady state. The data sections have varying length both in terms of the number of data points, and the time duration of each. The total amount of data points was 227011, with a total time of the data being 34285 seconds or around 9.5 hours. The difference between the number of data points and time duration is due to the sampling rate not being the same for all the sections when producing the experimental data.

To model the experimental pipeline-riser system, three approaches were tried. The first approach was to create a deep learning model using only experimental data. The second approach was to combine deep learning with a first principle model. The third approach was to create a deep learning model using simulation data and use this for transfer learning.

4 Baseline feedforward neural network model

The first approach was to create a baseline model of the system. A baseline model is a model made using only raw data. To create a deep FFNN the data set needs to be split into a training set used to train the model, and a test set used to evaluate the performance of the model. The data set was divided such that the training set contained the data where Z was 10% or lower, with the test set then containing the remaining data where Z was 11% or higher. The training set then consists of 43 of the 63-time series sections, 156120 data points, with a time duration of 23066 seconds or 6.4 hours. The test set consist of 20 of the 63-time series sections, 70891 data points, with a time duration of 11219 seconds or 3.1 hours. This division of the total data set into training and test set was used for all three approaches.

The experimental data contains a lot of noise and some preprocessing might be appropriate. In addition to noise, the assumption of steady-state is not necessarily true for all the data, as the system is still in the process of settling in many cases in the experimental data. While some noise can improve performance and make the model more robust[4], too much noise could make training difficult. A low pass filter could be used to reduce the amount of high-frequency noise in the data. Alternatively, since the data was divided into time series sections which are assumed to be at steady-state, the mean of one or more input or output features could be used for the entire section. Three different ways of preprocessing the data were applied to the training data set, in this project. The first was to simply not do anything and train on unaltered experimental data. The second was to filter the input features Z, $P1$, $P2$, $P3$ and $P4$ with a first-order low pass filter with a cut off

frequency of $0.2\,\mathrm{Hz}$, while leaving the output unaltered. The third way was to use the average for each time series section for the entire section, for the input features.

A deep FFNN was used to make a data-driven model. To train the model the Adam optimization algorithm was used, with a learning rate of 0.001. This was used for all deep learning models in this project. Different architectures, epochs, and batch sizes were tested for the three different ways of preprocessing the training data set. For the unaltered data set the model used had a depth of 15 hidden layers, each 10 nodes wide with ReLu activation function. For the filtered data set and the data set with the mean input features, the model used had a depth of 10 hidden layers, each 10 nodes wide with ReLu activation function. All models trained for 2 epochs with a batch size of 32. Standardising the training data before training such that it has 0 mean and a standard deviation of 1 was attempted but resulted in bad generalisation and was therefore not considered.

Mean square error was used as the loss function when training the models. To evaluate the models MSE is used as well, but since there is noise in the experimental outputs Q_g and Q_l(targets), and noise in the inputs which was expected to propagate to the model predictions, filtered input data was used to make predictions when calculating the MSE. The predictions and targets was filtered using a low pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$. The Tensorflow software in python was used to train and evaluate deep learning models in this project.

4.1 Results and Discussion

The three models were evaluated and their performance is given in table 4.1.

Table 4.1: Performance table for the three models

preprocessing	test MSE			train MSE		
	Q_g	Q_l	total	Q_g	Q_l	total
Raw data	9.6×10^{-3}	3.5×10^{-2}	4.5×10^{-2}	9.6×10^{-3}	9.7×10^{-2}	1.0×10^{-1}
Filtered	9.2×10^{-3}	2.0×10^{-2}	3.0×10^{-2}	3.7×10^{-3}	5.7×10^{-2}	6.1×10^{-2}
Averaged	1.0×10^{-2}	2.7×10^{-2}	3.8×10^{-2}	4.2×10^{-3}	5.5×10^{-2}	5.9×10^{-2}

From the performance table, the model trained on filtered input features generalized the best, having the lowest MSE for both outputs. The model trained on unfiltered experimental data generalized the worst in total but performed slightly better at predicting Q_g than the model trained on the mean input features. The model trained on unfiltered experimental data performed the worst on the train data set as well, performing the worst for both outputs. The model trained on mean input features performed overall the best on the train data set, but not a lot better than the model trained on filtered input data.

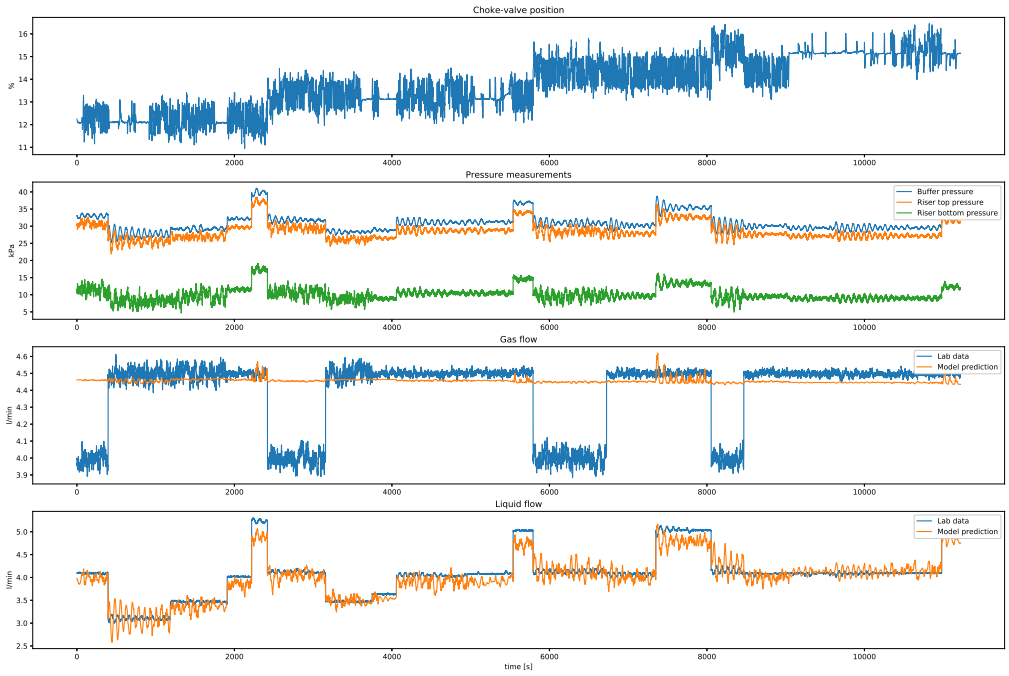

Figure 4.1: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a model trained on raw experimental data, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

In figure 4.1 the predicted Q_g and Q_l from the model trained on unfiltered input data, is plotted against the experimental data, for the test data. The prediction was made with filtered input data, and both the prediction and experimental data was filtered as well to make the plot more readable. As can be seen in the figure the model predicts Q_g to be slightly less than $4.5\,\mathrm{l/\,min}$ for all the different input combinations, which leads to a small offset when the target is $4.5\,\mathrm{l/\,min}$, but a large offset when it is $4\,\mathrm{l/\,min}$. It appears the model fits Q_g to just one value instead of finding a relation between input and output. For Q_l the prediction was much better. There were oscillatory predictions, but these probably propagate from the oscillatory pressure measurements. The network was quite good at predicting Q_l, learning the trends, but it under predicts for the larger flow rates.

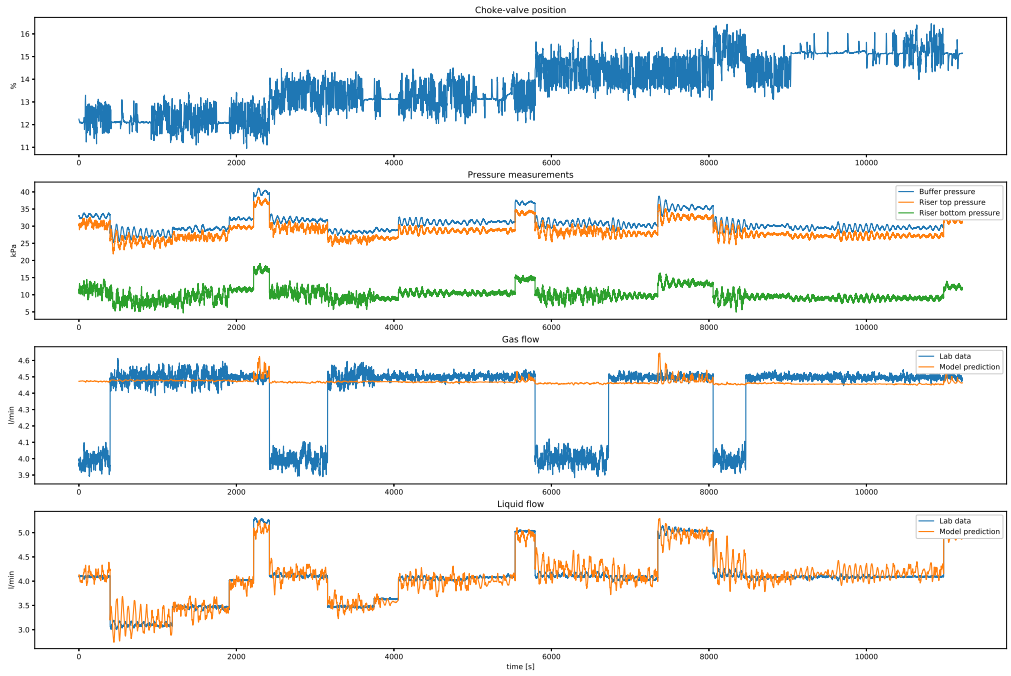

Figure 4.2: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a model trained on filtered input data, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

In figure 4.2 the predicted Q_g and Q_l from the model trained on filtered input data, is plotted against the experimental data, for the test data. The prediction was made with filtered input data, and both the prediction and experimental data was filtered as well to make the plot more readable. For this model as well Q_g was predicted to be almost $4.5\,\mathrm{l/\,min}$, but with a smaller offset. Just as with the other model, no relation between different inputs and output was found for Q_g The model is good at predicting Q_l, being quite a lot better at predicting the larger flow rates compared to the model trained on unfiltered data.

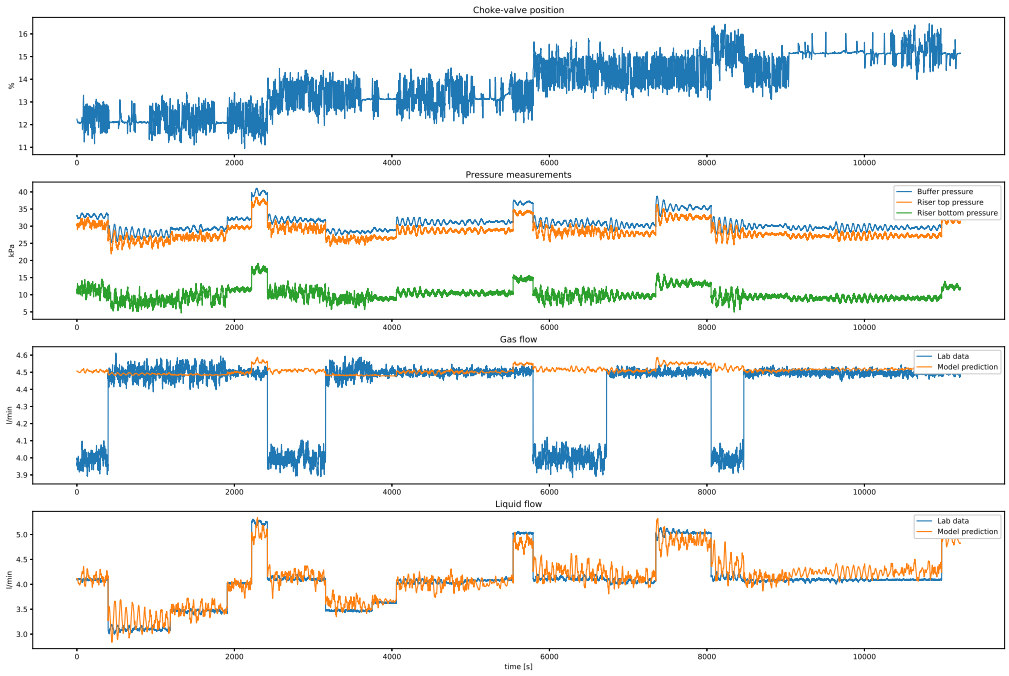

Figure 4.3: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a model trained on averaged input data, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

In figure 4.3 the predicted Q_g and Q_l from the model trained on mean input data, is plotted against the experimental data, for the test data. The prediction was made with filtered input data, and both the prediction and experimental data is filtered as well to make the plot more readable. Unlike the two previous models, a relation between the prediction of Q_g and the inputs can be seen. When $P1$ increases so do the prediction of Q_g ever so slightly. This change was, however, unrelated to target. As with the two other models it predicts Q_g to be around $4.5\mathrm{l}/\,\mathrm{min}$ no matter the input, not finding a proper relation between input and output as well. The prediction of Q_l was quite good, but not as good as the model trained on filtered input features. The model under predicts Q_l for high flow rates, and over predicts Q_l when Z is 15%.

All three models had the same problem when trying to predict Q_g, where no relation between input and output was found. This can be seen in figure 4.2 when the target value for Q_g was $4\mathrm{l}/\,\mathrm{min}$ as the prediction continued to be the same as when the target is $4.5\mathrm{l}/\,\mathrm{min}$. The reason for this lack of a proper input-output mapping for Q_g is that the train data set is not diverse enough. For most of the experimental data, the value of Q_g was $4.5\mathrm{l}/\,\mathrm{min}$, so expecting to map a relation which can barely be found in the experimental data is optimistic. The prediction of Q_l was, however, good for all three baseline models trained, but from both the performance table and the plots the form

of preprocessing which gave the best prediction on the test set was to filter the input training data. Because of this, filtering the input data was done for the rest of the models trained in this project.

5 Feature engineering

The second approach was to attempt to combine a FPM - *first principle model* with deep learning. This is similar to feature engineering, which is a process of using knowledge about the data and the system being modelled, to create inputs/features which makes generalization better[3]. This could include input/feature selection in which the most important inputs for prediction are found, or using knowledge about the system to transform raw data into more useful data.

In this project a FPM was combined with machine learning, by constructing a data-driven model which predicts the model bias of a simplified FPM:

$$\hat{\mathbf{y}} = \mathbf{y}^*(\mathbf{u}) + \phi(\mathbf{u}) \tag{5.1}$$

Where $\hat{\mathbf{y}}$ is the updated prediction, $\mathbf{y}^*$ is the FPM prediction, ϕ is the bias prediction and $\mathbf{u}$ is model inputs.

Jahanshahis simplified four-state model was used as the basis for the FPM. Jahanshahis simplified four-state model is a dynamic state-space model, where the pressures P_{in}, P_{rb} and P_{rt} are the outputs, and the mass flow rates $w_{g,in}$ and $w_{l,in}$, and the topside choke valve opening Z are the inputs. Some modifications and simplifications must be made to the model before it can be used for the intended purpose. First of all the object of this project was to predict the flow rates from pressure and choke valve opening measurements, so the modified model must have Z, P_{in}, P_{rb} and P_{rt} as inputs, and $w_{g,in}$ and $w_{l,in}$ as outputs. To achieve this the simplifying assumptions that there is no friction in the pipeline or riser was made. The dynamic model was simplified to a steady-state model by setting the differential equations to zero. This gives:

$$w_{g,in} = w_{g,rb} = w_{g,out} = w_g$$
$$w_{l,in} = w_{l,rb} = w_{l,out} = w_l$$

With this relation the model was rewritten as a set of equations in which w_g and w_l could be calculated from Z, P_{in}, P_{rb} and P_{rt}. The rest of the simplified model can be found in Appendix B

5.1 Global feedforward network

With a simplified first principle model using pressure data and choke valve opening as input, predicting flow rates, a deep FFNN model modelling the model bias can be constructed. As with the approach of making a model only using lab data, some preprocessing of the data is necessary. The first principle model requires P_{in} to be greater than P_{rb} which in turn must be greater than P_{rt}, to solve. In this model $P1$ will be used as P_{in}, $P2$ will be used as P_{rt} and $P4$ will be used as P_{rt}. Due to noise in the experimental data, $P4$ is occasionally greater than $P1$ and likewise $P2$ is sometimes greater than $P4$. This can be removed by filtering the pressure data. Two cases of

preprocessing the input data were considered. The first one was to filter it with a low pass filter with a cut off frequency of $0.2\,\mathrm{Hz}$. The second was to use the mean for each time series section as the input for the entire section. The first approach was the most successful and used to train a model.

The model was trained using the train data set with filtered input features. The input features used was Z, $P1$, $P2$ and $P4$, with the training targets being the difference between the experimental Q_g and Q_l, and the simplified first principle model predictions for the same parameters. Different architectures were tested, and the architecture used was one with 10 hidden layers, with 10 nodes in each hidden layer. The model was trained for 1 epoch, with a batch size of 32.

5.1.1 Results and Discussion

The updated model was evaluated and its performance is given alongside the performance of the simplified FPM in table 5.1

Table 5.1: Performance table for the updated model, comparing with the performance of simplified FPM

model	test MSE			train MSE		
	Q_g	Q_l	total	Q_g	Q_l	total
FPM	1.2	2.0×10^{-2}	1.2	1.7	3.0×10^{-1}	2.0
Bias model	1.2×10^{-1}	3.9×10^{-2}	1.6×10^{-1}	1.6×10^{-1}	3.7×10^{-2}	2.0×10^{-1}

From the performance table it can be seen that the simplified FPM is quite good at predicting Q_l on the test set, but bad at predicting Q_g. The bias updating model makes a lot of improvements on predicting Q_g, but it makes bad corrections on Q_l as it performs worse on the test set. On the train set, the FPM performs badly for both outputs, and the bias updating model makes improvements by improving the performance for both outputs.

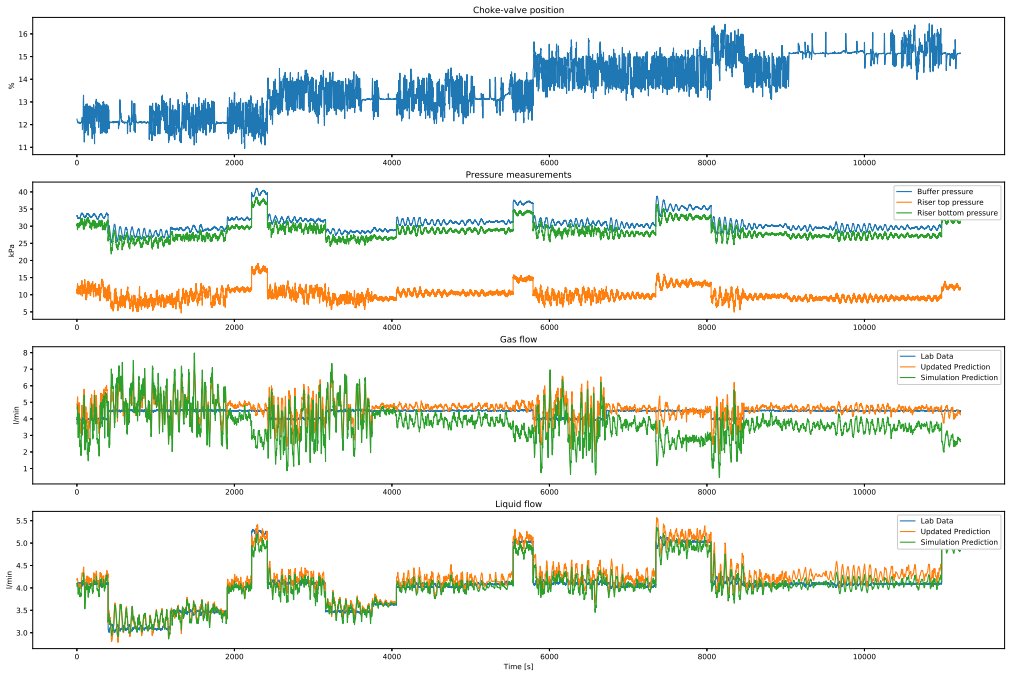

Figure 5.1: Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates and the simplified FPM gas and liquid flow rate predictions, for bias prediction model, for test set. Inputs, targets, FPM predictions and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

In figure 5.1 the simplified FPM predictions and the bias updated predictions of Q_g and Q_l are plotted against the experimental data. The predictions were made with filtered input data, and both the predictions and the experimental outputs were filtered as well. As can be seen in the figure the bias model improves in predicting Q_g compared to the simplified FPM. The FPM predictions of Q_g were varying quite a lot for certain sections, which makes it difficult to tell if the bias model improves. The FPM is good at predicting Q_l, with the bias model making bad corrections which makes the predictions worse. The exception to that is when the water flow rate is $5\,\mathrm{l/\,min}$.

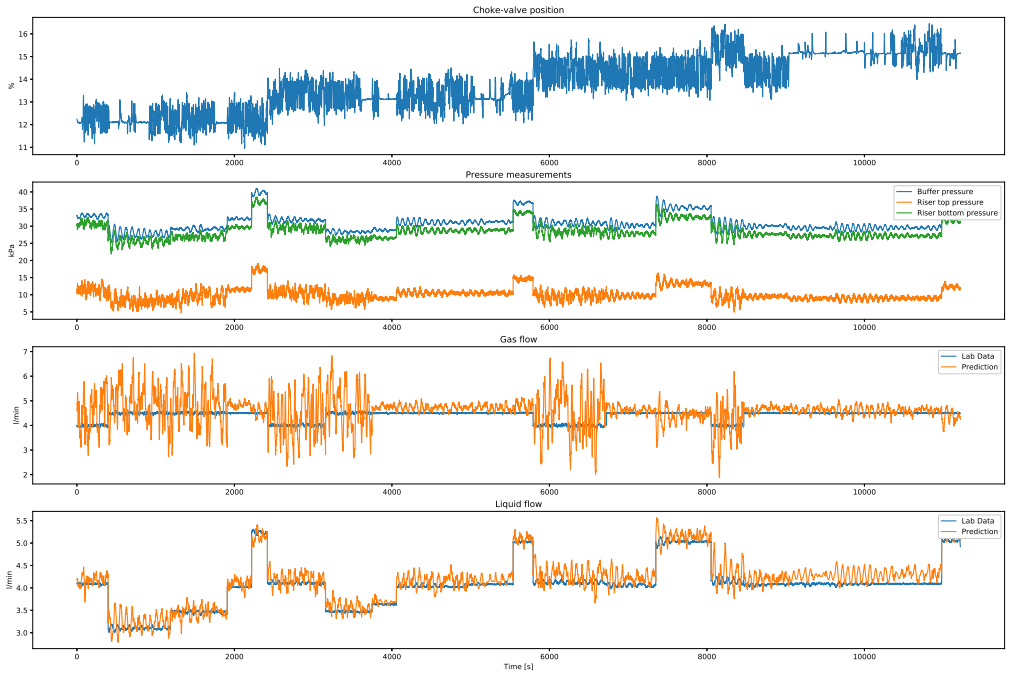

Figure 5.2: Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for bias prediction model, for the test set. Inputs, targets, and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

By using a FPM to make initial predictions and correcting the prediction with a FFNN model, predicting the model bias, it was assumed that the learning problem could be simplified. Having the main structure of the prediction being made by a FPM, and making small corrections on paper seems like an easy task. This, however, requires the FPM to make somewhat predictable predictions. In this project a steady-state model was to be constructed, so a steady-state FPM model was used to make the initial predictions. While the data used in this project is assumed to be at steady state, it was very clear from figure 5.2 that this was not the case. The topside choke valve opening Z was not constant, but instead oscillating around its setpoint. This can also be observed in the pressure measurements as the pressures are oscillating. In figures F.2 and F.1 in appendix **??** it can be observed that the pressure drop between $P1$ and $P4$, as well as the pressure drop between $P4$ and $P2$ are not constant either. Since a steady-state FPM was used to make predictions, it was sensitive to the deviations from steady-state when estimating the outputs. This was especially true when the pressure drops in the system are not constant, which leads the prediction of Q_g changing a lot, when the system should be at steady state. This makes learning the model bias difficult.

When the FPM prediction of Q_g was relatively stable, the bias model managed to make good corrections improving the prediction, however, for the sections where the predictions were varying a

lot, the bias model did not manage to make good corrections. Overall the prediction of Q_g was not improved by this approach at combining a FPM with a FFNN model.

The FPM prediction of Q_l was a lot better than the predictions of Q_g. On the test set, the predictions were so good that the corrections from the FFNN model made the predictions worse. This means the prediction of Q_l using the steady-state FPM was not affected as strongly by the inputs deviating from steady-state, compared with Q_g. The reason for this was probably because mass flow rates are estimated in the FPM and not volumetric flow rates. Water, which was the liquid used in this system has a density approximately 1000 times as large as that of air, which was the gas used in this system. This means a small change in the total mass flow rate calculation, due to the inputs not being steady state, would lead to a larger percentage-wise change in the gas flow rate compared to the change in the liquid flow rate.

The reason for the bad bias prediction on the test set was probably due to the difference in model bias for the test and train set. From figure E.5 it can be observed that for low choke valve openings Z, the bias of the FPM was quite large, as the FPM was over predicting Q_l, with the bias becoming less for increasing values. At the highest choke valve opening in the train data set, there was no model bias. In the test set, the choke valve opening was greater than for the train set, with the model bias being small or non-existing. As the trend in the train set is that the FPM overpredicts at low choke valve openings, and predicts correctly when the choke valve opening is 10%, the bias correction model believes that the FPM will start underpredicting Q_l for the test set, and makes corrections to correct this non-existing underprediction.

Compared to the baseline model this approach was not an improvement. The baseline model predicted Q_l better on the test set than this approach. The baseline model did not manage to find any meaningful relationship between the inputs and outputs, due to a limited data set. Transforming the outputs by modelling the difference between the experimental data and the predictions of a FPM, might make it possible to model the relation between the inputs and Q_g, however, the FPM model used was too sensitive to deviations from steady-state in the train and test data set to make good predictions.

6 Transfer learning

Transfer learning is a machine learning technique where the learning of a new task is improved by re-purposing an old model which solves a similar task[11][8]. The idea is that by using transfer learning, a new task can be learned with less training, less data and/or with a lower generalization error. There are three ways in which transfer learning can improve learning. The first is to give a better start for training, the second is to give a larger initial slope when training (larger improvement in performance when training), and the third is giving a higher asymptote (higher performance).

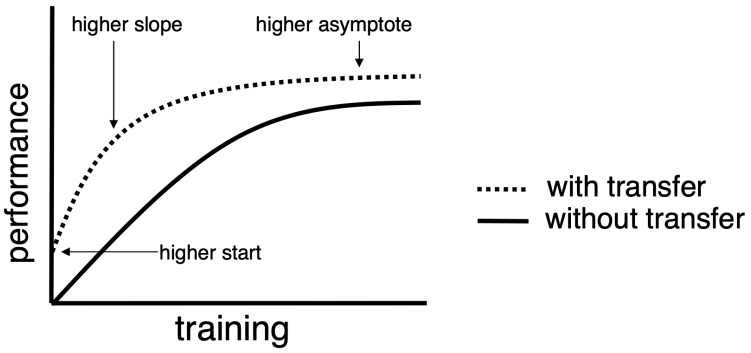

Figure 6.1: Three ways transfer learning can improve learning[11]

In this project, transfer learning would be attempted by training a model on simulation data covering a larger data range than the range of the experimental data and using that as the source model for transfer learning. By using a simulation model covering a larger range of data, the goal was to make an experimental model which can give more accurate prediction over a larger range than the baseline model. Simulation data were generated using Jahanshahis simplified four-state model. The steady-state data was generated for $Z \in [4, 15]$ %, $Q_g \in [2, 6]$ l/ min and $Q_l \in [2, 6]$ l/ min. With this data set, a FFNN model was trained, with a depth of 10 hidden layers, a width of 10 nodes in each layer, ReLu activation function in each layer, trained for 100 epochs with a batch size of 32. The input features of this model were Z, P_{in}, P_{rt} and P_rb, with the outputs being Q_g and Q_l. When repurposing this model for transfer learning $P1$ will be used for P_{in}, $P2$ will be used for P_{rt} and $P4$ will be used for P_{rb}.

6.1 First approach

Three approaches of transfer learning were attempted in this project. For all three approaches, the training data was preprocessed by having the input features filtered with a low pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$. The first approach was to retrain parts of the simulation model using experimental data. To retain some of the original model structure only the last layers were retrained. Two models were trained, one where only the last layer was retrained, and one where the two last layers were retrained. The models were retrained for 5 epochs with a batch size of 32.

6.1.1 Results

In table 6.1 below the simulation model, the transfer model which had only the last layer retrained, and the transfer model which had the two last layers retrained evaluated on the test and train data.

Table 6.1: Performance table for the two transfer model compared with the simulation model.

model	test MSE			train MSE		
	Q_g	Q_l	total	Q_g	Q_l	total
Sim model	3.7×10^{-1}	5.7×10^{-2}	4.3×10^{-1}	7.3×10^{-1}	3.6×10^{-1}	1.1
Last layer	9.3×10^{-3}	1.1×10^{-1}	1.2×10^{-1}	5.9×10^{-3}	1.9×10^{-1}	2.0×10^{-1}
2 Last layers	1.1×10^{-2}	4.5×10^{-2}	5.6×10^{-2}	4.7×10^{-3}	1.8×10^{-1}	1.9×10^{-1}

The generalization for the transfer models improved with the amount layers which were retrained, with the test MSE for the model which retrained the two layers being half of the test MSE for the model which only retrained the last layer. The train MSE for both transfer models were very similar, with the model which retrained two layers being slightly better.

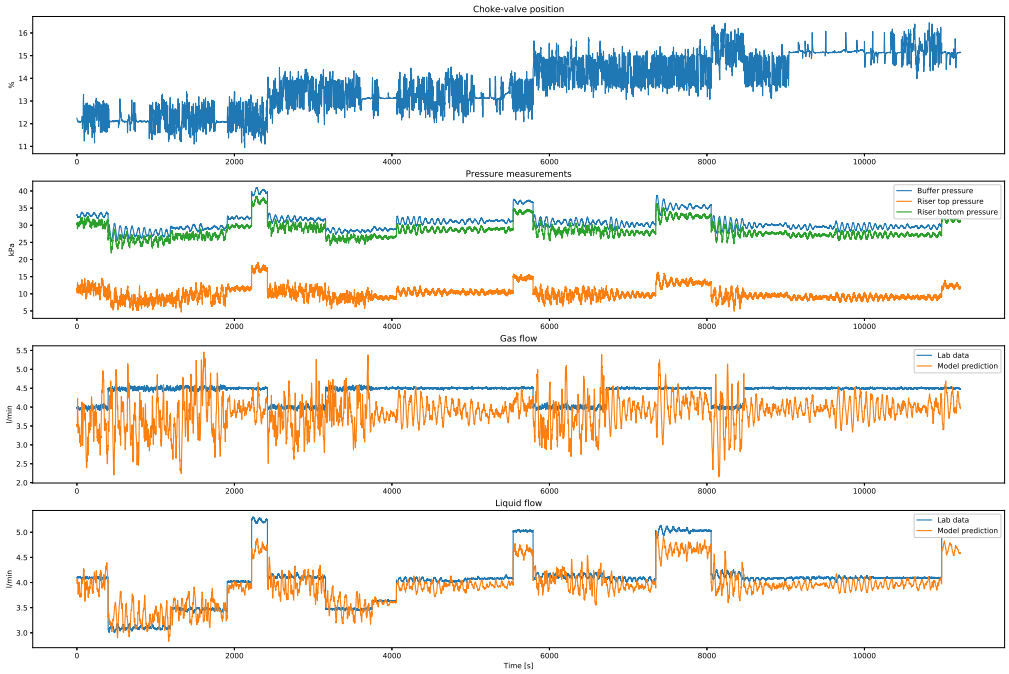

Figure 6.2: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for the simulation model, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of 0.2 Hz

In figure 6.2 the predictions of Q_g and Q_l using the simulation model, are plotted against the experimental data. The predictions were made with filtered input data, and both the predictions and the experimental outputs were filtered as well. The simulation model was somewhat good at

predicting Q_l but slightly underpredicts when the liquid flow rate is high, and when Z is larger than 13%. The simulation model underpredicts Q_g, and the predictions are varying a lot.

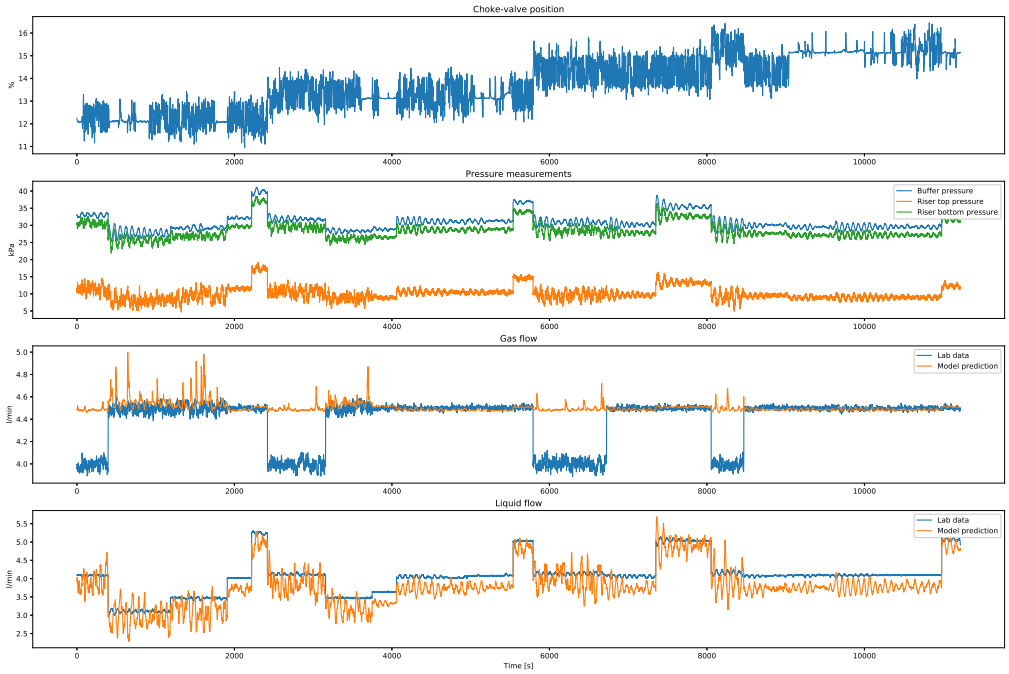

Figure 6.3: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a transfer model where the last layer was retrained on experimental data, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

In figure 6.3 the predictions of Q_g and Q_l using the model which retrained the last layer, are plotted against the experimental data. The predictions were made with filtered input data, and both the predictions and the experimental outputs were filtered as well. This model underpredicts Q_l when Z was larger than 13%, just as the simulation model but with a larger deviation from the target variable, but it does, however, makes better predictions when Q_l is around $5\,\mathrm{l/min}$. For the prediction of Q_g, this model does the same as the models trained using only experimental data and it predicts Q_g to be $4.5\,\mathrm{l/min}$ for all inputs.

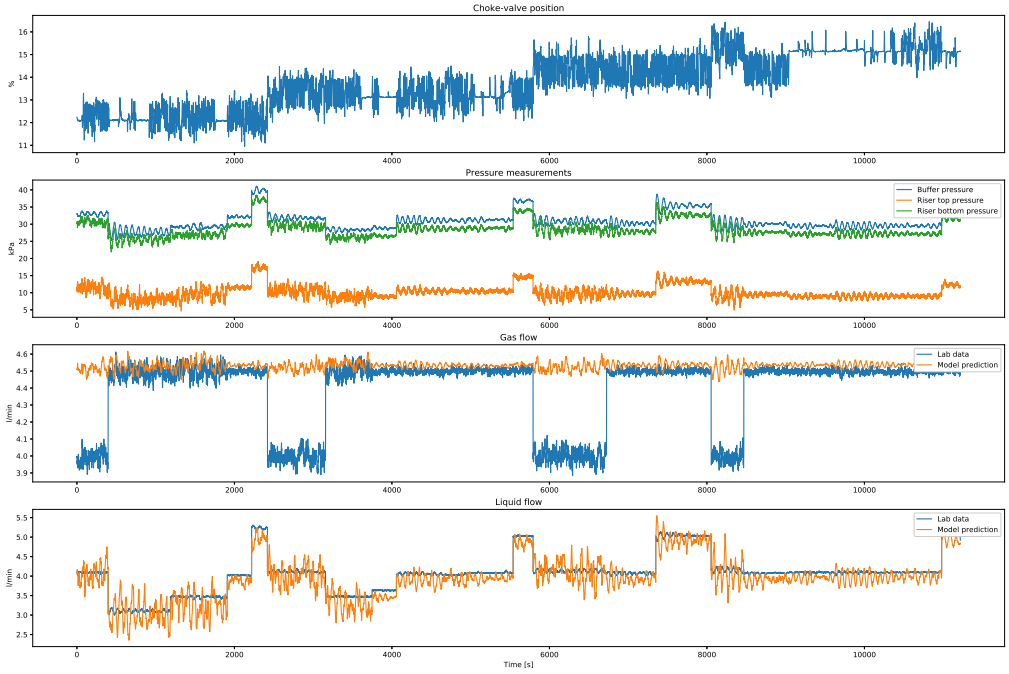

Figure 6.4: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a transfer model where the last two layers were retrained on experimental data, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

In figure 6.4 the predictions of Q_g and Q_l using the model which retrained the last two layers, are plotted against the experimental data. The predictions were made with filtered input data, and both the predictions and the experimental outputs were filtered as well. Just as the other transfer model, this predicts Q_g to be $4.5\,\mathrm{l/\,min}$ for all inputs. It is better at predicting Q_l but slightly underpredicts.

6.2 Second approach

The second approach was to use the predictions from the simulation model as extra input features for a new FFNN model alongside the other input features.

$$\hat{\mathbf{y}}_{sim} = \Phi_{sim}(\mathbf{u})$$
$$\hat{\mathbf{y}} = \Phi_{transfer}([\mathbf{u}^T, \hat{\mathbf{y}}_{sim}^T]^T)$$

Where $\hat{\mathbf{y}}_{sim}$ is the simulation predictions, Φ_{sim} is the simulation model, $\mathbf{u}$ is the input, $\hat{\mathbf{y}}$ is the transfer model predictions, and $\Phi_{transfer}$ is the transfer model. Here the total model would be two models in series as shown in figure 6.5.

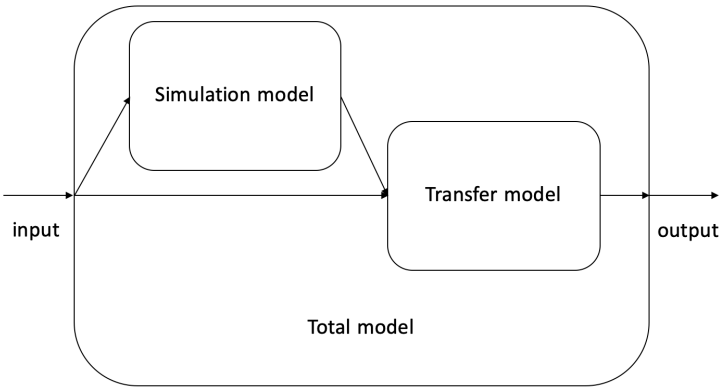

Figure 6.5: Transfer learning approach two model structure

A FFNN Transfer model was constructed and trained. The inputs into this model were Z, $P1$, $P2$, $P4$, $\hat{Q}_{g_{sim}}$ and $\hat{Q}_{l_{sim}}$. Different architectures were tested and a structure consisting of 4 hidden layers, with 10 nodes in each and ReLu activation function was used. The models were trained for 3 epochs with a batch size of 32.

6.2.1 Result

The transfer model was evaluated and its performance was listed in table 6.2 below.

Table 6.2: Performance table for the transfer model compared with the simulation model

model	test MSE			train MSE		
	Q_g	Q_l	total	Q_g	Q_l	total
Sim model	3.7×10^{-1}	5.7×10^{-2}	4.3×10^{-1}	7.3×10^{-1}	3.6×10^{-1}	1.1
Transfer model	1.1×10^{-2}	2.9×10^{-2}	4.0×10^{-2}	4.3×10^{-3}	4.8×10^{-2}	5.2×10^{-2}

The transfer model in this approach improved the generalization compared to the transfer models in the first approach, it also performed better on the train data in the first approach. In figure 6.6 below, the predictions of Q_g and Q_l using the model which retrained the last layer, were plotted against the experimental data. The predictions were made with filtered input data, and the predictions were filtered as well. The model is bad at predicting Q_g, just as the models trained using only experimental data, predicting it to be $4.5\mathrm{l}/\min$. The prediction of Q_l is mostly good,

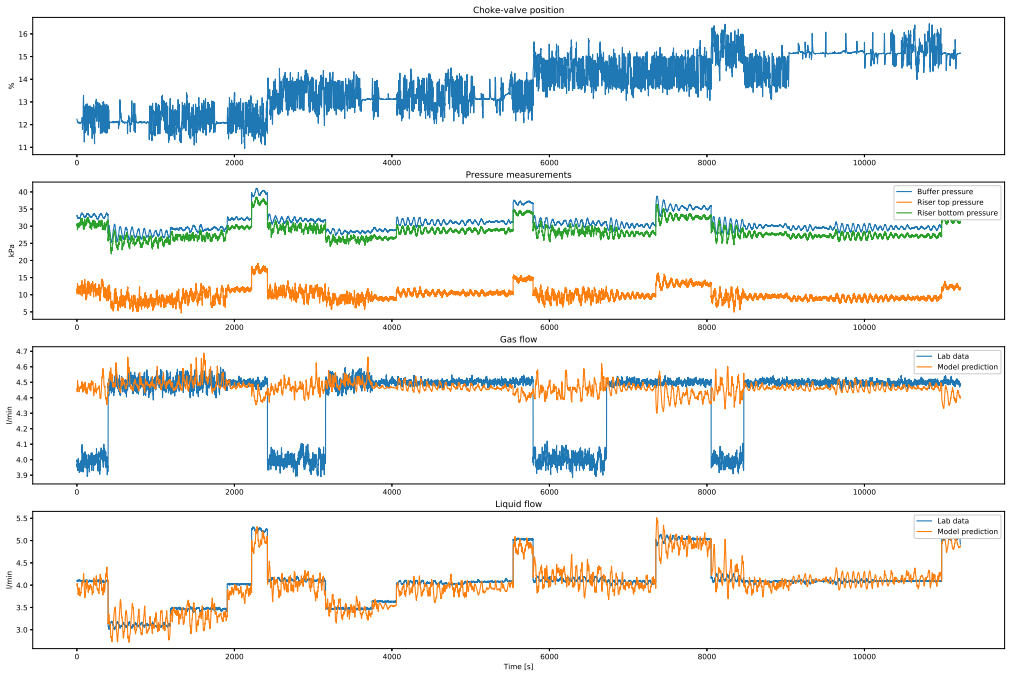

Figure 6.6: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a transfer model which uses predictions from simulation network as extra input, on the test set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

6.3 Third approach

The third approach was to train a model which predicts the bias between the simulation model and the experimental data, similar to what was attempted in the feature engineering section of the project. The depth of the transfer bias model was 10 layers, each 10 nodes wide, with ReLu activation functions. The model was trained for 5 epochs with a batch size of 32.

6.3.1 Results

The third approach at transfer learning was also evaluated and the performance is listed in table 6.3 below.

Table 6.3: Performance table for the transfer model compared with the simulation model.

model	test MSE			train MSE		
	Q_g	Q_l	total	Q_g	Q_l	total
Sim model	3.7×10^{-1}	5.7×10^{-2}	4.3×10^{-1}	7.3×10^{-1}	3.6×10^{-1}	1.1
Bias model	4.1×10^{-2}	2.4×10^{-2}	6.5×10^{-2}	1.2×10^{-2}	1.8×10^{-2}	3.0×10^{-2}

As seen in the table this third approach was better at predicting Q_l compared to the other two approaches at transfer learning, it was, however, worse at predicting Q_g having a larger test error than the two other approaches. Compared to the transfer learning approach which attempted something similar, this approach performed better. This model also improved performance compared to the simulation model. Below in figure 6.7 the simulation model predictions and the bias transfer model predictions of Q_g and Q_l are plotted against the experimental data. The predictions were made with filtered input data, and both the predictions and the experimental outputs were filtered as well. As can be seen, the transfer model predicting the simulation model bias makes great improvements at predicting Q_l. While mostly improving the prediction of Q_g, the correction is not that good as it adjusts the prediction to be close to $4.5\mathrm{l}/\min$, even when it should be corrected to $4\mathrm{l}/\min$.

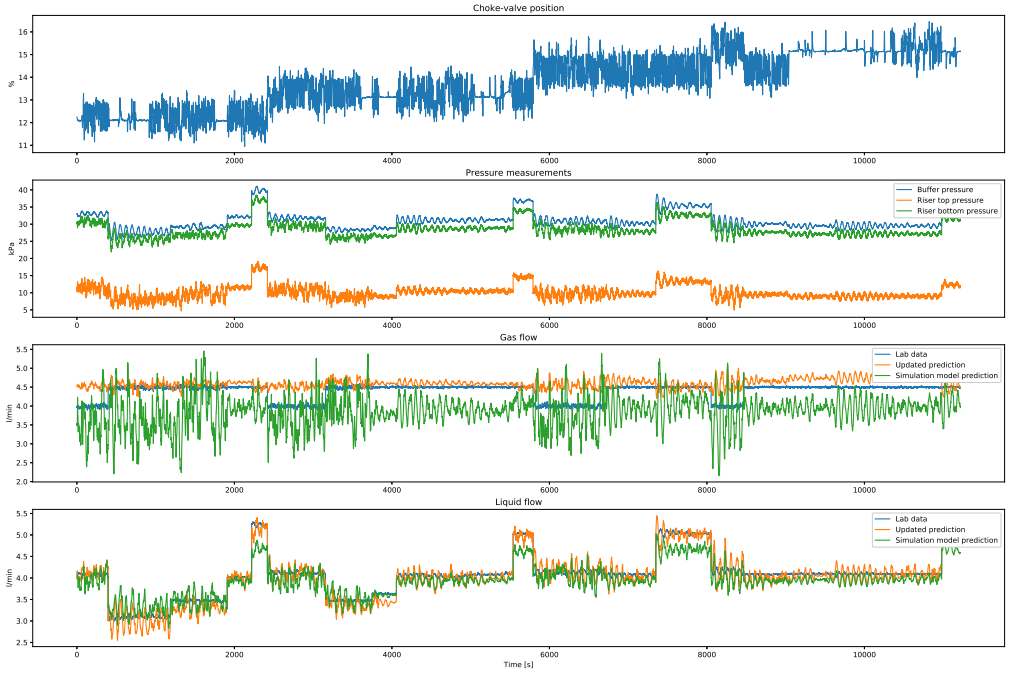

Figure 6.7: Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates and the simulation model gas and liquid flow rate predictions, for a bias prediction model, on the test set. Inputs, targets, simulation model predictions and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

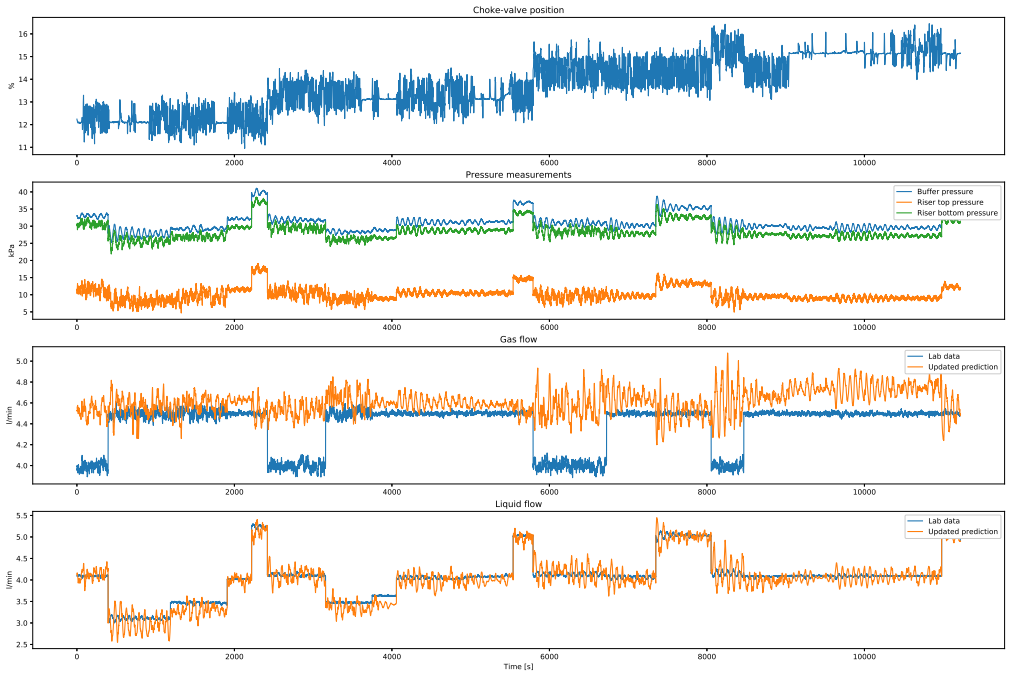

Figure 6.8: Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a bias prediction model, on the test set. Inputs, targets and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

6.4 Discussion

Three approaches of doing transfer learning were tried, to try and improve predictive performance. This was not achieved and no improvement was not made by this approach. The largest problem of the baseline model was its inability to make meaningful predictions of Q_g, so using a model which could make meaningful predictions of Q_g as a starting point for further training seemed like a good idea. But for all the three approaches tried, the same problem arose as with the baseline model, that no meaningful relationship between the inputs and the prediction of Q_g was achieved. For all three models, the prediction of Q_g was around $4.5\mathrm{l}/\,\mathrm{min}$, no matter what the input or target data was, just as was the case for the baseline model.

The model which was used as a basis for transfer learning was trained on simulation data covering a larger span of gas and liquid flow rates. This model was evaluated using experimental data in figures 6.2 and E.6. As can be seen in the figures the simulation model models the trends of Q_l quite well, and is a good starting point to make a model which predicts Q_l as small adjustments should correct the predictions. The predictions of Q_g are not that good. The simulation model underpredicts Q_g for all the data, and the predictions are varying a lot. There is little connection

between the prediction of Q_g and the target.
When retraining the last layers of the simulation model, the idea was to make small changes to the simulation model and have some of the structure from the model remain in the transfer model, making only small modifications to adjust the model to the experimental data. This did not work well, as a Q_g was predicted to be the same value for all input combinations. This was the same problem as for the baseline model. This makes sense as retraining the last layers is similar to training a new shallow model, using the output from the last hidden layer as input. While these new input features could improve performance, the problem with having a limited data set is not solved. So the predictions for Q_g was not improved. It should be noted by retraining just two layers the prediction of Q_l was improved a lot, however, the prediction was worse than that of the baseline model.

The second approach to transfer learning was similar to the first attempt, by using predictions from the simulation model as input to a new model. This resulted in the same problems, where the overall model is good at predicting Q_l but fails to find a mapping of Q_g. The extra inputs from the simulation model helped improve generalization of Q_l using a smaller model, only 4 hidden layers deep, but it did not negate the problem of limited variation in the data set for Q_g.
The third approach where a model was trained on experimental data to predict the model bias of the simulation model, did not manage to improve the prediction of Q_g either. The prediction of Q_l is improved and is quite good. This was expected as small adjustments needed to be made to the simulation model to improve the performance.

7 Conclusion

In this project the goal was to make a steady-state model of a riser pipeline system based on experimental data, using machine learning. Volumetric gas and liquid flow rates were to be predicted using pressure measurements in the system and the topside choke valve position. The system to be modelled was a small scale experimental rig, in which experimental was used. To achieve this goal three approaches were attempted. The first approach was to construct a baseline feedforward neural network model. This model was successful in predicting the liquid flow rate but did not manage to make good predictions of the gas flow rate. This was attributed to a data set with little variation in gas flow rate. The second approach was to combine first principle knowledge about the system with machine learning. A feedforward model predicting the bias of a simplified first principle model was constructed. This approach did not improve performance, as the liquid flow rate prediction was worse, and the gas flow prediction being unsatisfactory. The third approach was to train a feedforward neural network model on simulation data and repurpose this model to make a model predicting the experimental data. This was attempted in three different ways. The first was to retrain the last layers of the simulation model, the second was to use the simulation predictions as additional input and the third was to make a model predicting the bias of the simulation model. While all these three approaches managed to model the liquid flow rate quite well, they did not manage to improve the prediction of the gas flow rate compared to the baseline model.

To conclude, the volumetric liquid flow rate was possible to model using the available data set. The gas flow rate, however, was not possible to model. This was because the gas flow rate measurements

in the experimental data had too little variation. The attempts at combining deep learning with a first principle model or doing transfer learning did not negate the effect of a limited data set. To improve the predictions of the volumetric gas flow rate, more experimental data are needed, covering a larger range.

7.1 Further work

To make a model which is better at predicting the volumetric flow rate the best course of action is to produce more lab data, covering a larger range of gas flow rates. While other measures to improve the prediction might be done, expanding the training data would improve predictions the most. With a larger data set, other attempts at transfer learning could be attempted. The idea of using transfer learning was to retain the structure of the simulation model and only tweak it slightly, to better fit the lab data. In this project, this was attempted by only training parts of the model, but that resulted in the retrained parts of the model changing a lot. What could be attempted is to train the entire simulation model, but adding a penalty to changing the parameters of the model, thereby reducing the number of changes made to the model. Besides, more traditional feature engineering could be attempted by creating new features, like the pressure drop over the pipeline or the riser.

References

[1] Ethem Alpaydın. *Introduction to Machine Learning*. MIT Press, 2 edition, 2010.

[2] Nikolai Andrianov. A machine learning approach for virtual flow metering and forecasting. In *Proceedings of the 3rd IFAC Workshop on Automatic Control in Offshore Oil and Gas Production*, page 191, 2018.

[3] Jason Brownlee. Discover feature engineering, how to engineer features and how to get good at it. Blog-post, 09 2014. `https://machinelearningmastery.com/discover-feature-engineering-how-to-engineer-features-and-how-to-get-go`

[4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[5] Kjetil Havre, Karl Ole Stornes, and Henrik Stray. *Taming slug flow in pipelines*. ABB Review, 4, 2000.

[6] Esmaeil Jahanshahi. *Control Solutions for Multiphase Flow: Linear and nonlinear approaches to anti-slug control*. PhD thesis, Norwegian University of Science and Technology, Trondheim, 2013.

[7] Esmaeil Jahanshahi and Sigurd Skogestad. Closed-loop model identification and pid/pi tuning for robust anti-slug control. In *10th IFAC International Symposium on Dynamics and Control of Process Systems*, Mumbai, India, 2013.

[8] Emilio Soria Olivas, Jose David Martin Guerrero, Marcelino Martinez Sober, Jose Rafael Magdalena Benedito, and Antonio Jose Serrano Lopez. *Handbook Of Research On Machine Learning Applications and Trends: Algorithms, Methods and Techniques - 2 Volumes*. Information Science Reference - Imprint of: IGI Publishing, Hershey, PA, 2009.

[9] Espen Storkaas. *Stabilizing control and controllability: Control soultions to avoid slug flow in pipeline-riser systems*. PhD thesis, Norwegian University of Science and Technology, Trondheim, 2005.

[10] Julian Straus. *Optimal Operation of Integrated Chemical Processes: With Application to the Ammonia Synthesis*. PhD thesis, Norwegian University of Science and Technology, Trondheim, 2018.

[11] Lisa Torrey and Jude W. Shavlik. Chapter 11 transfer learning. 2009.

A Esmaeils four-state model

A.1 Dynamic equations

$$\frac{dm_{g,p}}{dt} = w_{g,in} - w_{g,rb} \tag{A.1}$$
$$\frac{dm_{l,p}}{dt} = w_{l,in} - w_{l,rb} \tag{A.2}$$
$$\frac{dm_{g,r}}{dt} = w_{g,rb} - w_{g,rt} \tag{A.3}$$
$$\frac{dm_{g,r}}{dt} = w_{g,rb} - w_{g,rt} \tag{A.4}$$

A.2 Tuning parameters

- K_h: correction factor for liquid level in pipeline
- C_{v1}: topside choke valve opening
- K_g: coefficient for gas flow through low point
- K_l: coefficient for liquid flow through low point

A.3 Outflow conditions

$$w_{out} = C_{v1} f\left(z\right) \sqrt{\rho_{rt} \max\left(P_{rt} - P_s, 0\right)} \tag{A.5}$$

$$f(z) = z, \quad 0 \leq z \leq 1$$
$$Z = 100z, \quad 0 \leq Z \leq 100$$

$$w_{l,out} = \alpha_{l,rt}^m w_{out} \tag{A.6}$$
$$w_{g,out} = \left(1 - \alpha_{l,rt}^m\right) w_{out} \tag{A.7}$$
$$\alpha_{l,rt}^m = \frac{\alpha_{l,rt}\rho_l}{\alpha_{l,rt}\rho_l + \left(1 - \alpha_{l,rt}\right)\rho_{g,r}} \tag{A.8}$$
$$\rho_{rt} = \alpha_{l,rt}\rho_l + \left(1 - \alpha_{l,rt}\right)\rho_{g,r} \tag{A.9}$$

A.4 Pipeline model

$$\bar{\alpha}_{l,p} = \frac{\bar{\rho}_{g,p} w_{l,in}}{\bar{\rho}_{g,p} w_{l,in} + \rho_l w_{g,in}} \tag{A.10}$$

$$\bar{\rho}_{g,p} = \frac{P_{in,nom} M_g}{R T_p} \tag{A.11}$$

$$\bar{m}_{l,p} = \rho_l V_p \bar{\alpha}_{l,p} \tag{A.12}$$

$$h_d = D_p / \cos(\theta) \tag{A.13}$$

$$\bar{h} = K_h h_d \bar{\alpha}_{l,p} \tag{A.14}$$

$$h = \bar{h} + \left(\frac{m_{l,p} - \bar{m}_{l,p}}{A_p \left(1 - \bar{\alpha}_{l,p}\right) \rho_l} \right) \sin(\theta) \tag{A.15}$$

$$\rho_{g,p} = \frac{m_{g,p}}{V_{g,p}} \tag{A.16}$$

$$V_{g,p} = V_p - m_{l,p} / \rho_l \tag{A.17}$$

$$P_{in} = \frac{\rho_{g,p} R T_p}{M_g} \tag{A.18}$$

$$\Delta P_{fp} = \frac{\bar{\alpha}_{l,p} \lambda_p \rho_l \bar{U}_{sl,in}^2 L_p}{2 D_p} \tag{A.19}$$

$$\frac{1}{\sqrt{\lambda_p}} = -1.8 \log_{10} \left[\left(\frac{\epsilon / D_p}{3.7} \right)^{1.11} + \frac{6.9}{Re_p} \right] \tag{A.20}$$

$$Re_p = \frac{\rho_l \bar{U}_{sl,in} D_p}{\mu} \tag{A.21}$$

$$\bar{U}_{sl,in} = \frac{4 w_{l,in}}{\pi D_p^2 \rho_l} \tag{A.22}$$

A.5 Riser model

$$V_{g,r} = V_r - m_{l,r}/\rho_l \tag{A.23}$$

$$\rho_{g,r} = \frac{m_{g,r}}{V_{g,r}} \tag{A.24}$$

$$P_{rt} = \frac{\rho_{g,r} R T_r}{M_g} \tag{A.25}$$

$$\bar{\alpha}_{l,r} = \frac{m_{l,r}}{V_r \rho_l} \tag{A.26}$$

$$\bar{\rho}_{m,r} = \frac{m_{g,r} + m_{l,r}}{V_r} \tag{A.27}$$

$$\Delta P_{fr} = \frac{\bar{\alpha}_{l,r} \lambda_r \bar{\rho}_{m,r} \bar{U}_m^2 \left(L_r + L_h\right)}{2 D_r} \tag{A.28}$$

$$\frac{1}{\sqrt{\lambda_r}} = -1.8 \log_{10} \left[\left(\frac{\epsilon / D_r}{3.7} \right)^{1.11} + \frac{6.9}{Re_r} \right] \tag{A.29}$$

$$Re_r = \frac{\bar{\rho}_{m,r} \bar{U}_m D_r}{\mu} \tag{A.30}$$

$$\bar{U}_m = \bar{U}_{sl,r} + \bar{U}_{sg,r} \tag{A.31}$$

$$\bar{U}_{sl,r} = \frac{w_{l,in}}{\rho_l A_r} \tag{A.32}$$

$$\bar{U}_{sg,r} = \frac{w_{g,in}}{\rho_{g,r} A_r} \tag{A.33}$$

A.6 Gas flow model at riser base

$$w_{g,rb} = 0, \quad h \geq h_d \tag{A.34}$$

$$w_{g,rb} = K_g A_g \sqrt{\rho_{g,p} \Delta P_g}, \quad h < h_d \tag{A.35}$$

$$\Delta P_g = P_{in} - \Delta P_{fp} - P_{rt} - \bar{\rho}_{m,r} g L_r - \Delta P_{fr} \tag{A.36}$$

$$A_g = A_p \left(\frac{h_d - h}{h_d} \right)^2, \quad h < h_d \tag{A.37}$$

$$A_g = 0, \quad h \geq h_d \tag{A.38}$$

$$\tag{A.39}$$

A.7 Liquid flow model at riser base

$$w_{l,rb} = K_l A_l \sqrt{\rho_l \Delta P_l} \tag{A.40}$$

$$\Delta P_l = P_{in} - \Delta P_{fp} + \rho_l g h - P_{rt} - \bar{\rho}_{m,r} g L_r - \Delta P_{fr} \tag{A.41}$$

$$A_l = A_p - A_g \tag{A.42}$$

A.8 Phase distribution model at outlet choke valve

$$\alpha_{l,rt} = 2\bar{\alpha}_{l,r} - \alpha_{l,rb} = \frac{2m_{l,r}}{V_r\rho_l} - \frac{A_l}{A_p} \tag{A.43}$$

B Simplified model

This is a simplified version of Jahanshahis 4-state model reshaped as a input output model, predicting w_g and w_l from Z, P_{in}, P_{rb} and P_{rt}. Q_g and Q_l are then calculated using the standard density for air and water. The assumptions made are steady-state and no friction.

$$\rho_{g,r} = \frac{P_{rt}M_g}{RT_r} \tag{B.1}$$

$$\bar{\rho}_{m,r} = \frac{P_{rb} - P_{rt}}{gL_r} \tag{B.2}$$

$$m_{l,r} = \rho_l \frac{V_r(\bar{\rho}_{m,r} - \rho_{g,r})}{\rho_l - \rho_{g,r}} \tag{B.3}$$

$$\rho_{g,p} = \frac{M_g P_{in}}{RT_p} \tag{B.4}$$

$$A_g = A_p \left(\frac{h_d - h}{h_d}\right)^2, \quad h < h_d \tag{B.5}$$

$$A_g = 0, \quad h \geq h_d \tag{B.6}$$

$$A_l = A_p - A_g \tag{B.7}$$

$$\alpha_{l,rt} = \frac{2m_{l,r}}{V_r\rho_l} - \frac{A_l}{A_p} \tag{B.8}$$

$$\alpha^m_{l,rt} = \frac{\alpha_{l,rt}\rho_l}{\alpha_{l,rt}\rho_l + (1-\alpha_{l,rt})\,\rho_{g,r}} \tag{B.9}$$

$$\rho_{rt} = \alpha_{l,rt}\rho_l + (1-\alpha_{l,rt})\,\rho_{g,r} \tag{B.10}$$

$$\Delta P_l = P_{in} - P_{rb} + \rho_l g h \tag{B.11}$$

$$\Delta P_g = P_{in} - P_{rb} \tag{B.12}$$

Due to the steady state assumption,

$$w = w_{in} = w_{rb} = w_{rt}$$
$$w_g = w_{g,in} = w_{g,rb} = w_{g,rt}$$
$$w_l = w_{l,in} = w_{l,rb} = w_{l,rt}$$

This gives a set of two non-linear equations with h as unknown. This set of equations is solved for h, and w_g and w_l is calculated.

$$w = C_{v1} f(z) \sqrt{\rho_{rt} \max(P_{rt} - P_s, 0)} \tag{B.13}$$

$$w_l = \alpha_{l,rt}^m w \tag{B.14}$$

$$w_g = \left(1 - \alpha_{l,rt}^m\right) w \tag{B.15}$$

$$w_l = K_l A_l \sqrt{\rho_l \Delta P_l} \tag{B.16}$$

$$w_g = K_g A_g \sqrt{\rho_{g,p} \Delta P_g} \tag{B.17}$$

C Experimental setup

The experimental rig consists of a reservoir tank, buffer tank and top tank connected by a pipeline and riser as seen in figure C.1. There is also a pump, pumping the liquid in the reservoir tank through the system. The pipes have an inner diameter of 2 cm. The pipeline is 4 m long with an inclination angle of 15 degrees. The riser is 3 m long. The buffer tank is used to simulate a 70 m long pipeline, with the volume of the buffer tank and pipeline being that of a 70 m long pipe.

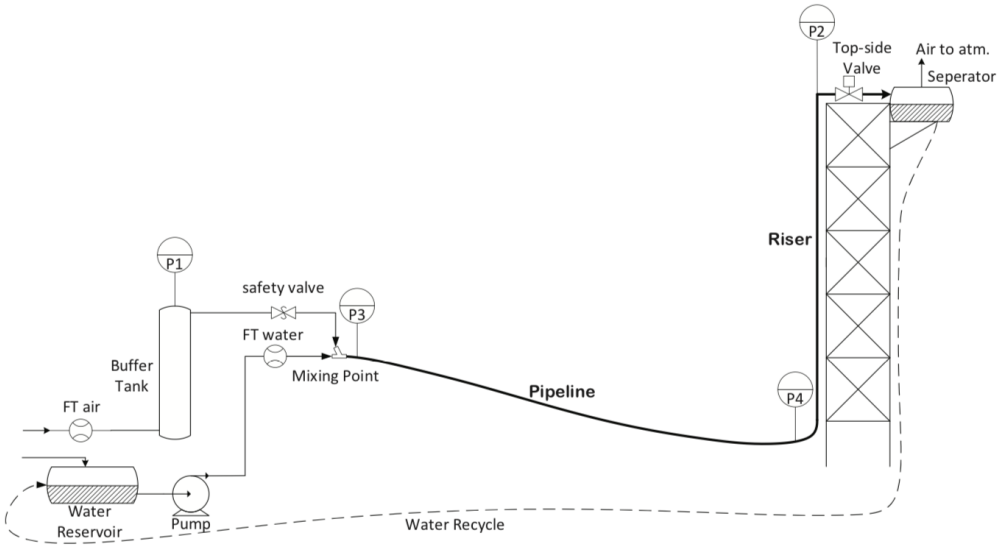

Figure C.1: Experimental setup of Anti-slug control rig

Water is used as the liquid in the system, and air is used as the gas in the system. In the experimental rig, the gas flow is set by giving a setpoint to a flow controller, while the water flow is manipulated by manually opening or closing a globe valve after the pump. The topside choke valve can also be manipulated by giving a setpoint to a controller which opens or closes the valve. The pressure is measured at four points in the model:

- P1, the pressure in the buffer tank
- P2, the pressure in the at the top of the riser

- P3, the pressure after the mixing point of the liquid and gas
- P4, the pressure in the at the bottom of the riser

D Online radial basis function network

The feature engineering model formulation, where a machine learning model was used to model the bias of a first principle model was attempted online using a radial basis function network. A radial basis function network is a shallow neural network, which is a neural network with only one hidden layer. The activation function used was a normalized radial basis function. A normalized radial basis function is $u_i(x) = \frac{g_i(x)}{\sum_{j=1}^{N} g_j(x)}$. The model was trained online, training on a given number of previously measured data points given by a backwards time horizon. The centres $\mathbf{c}_i$ in the radial basis functions were found using k-means clustering on the training data, with the cluster centroid for each cluster being the centre $\mathbf{c}_i$ for each node. The weights of the model were then found using standard regression.

With a backwards time horizon of $80\,\mathrm{s}$ and 5 nodes in the hidden layer, this was tested on the data set. Training on the previous 80 steps, model corrections were made to next time step. This was done for the entire data set. Figure D.1 is a performance plot where the average updated prediction for each steady-state section is plotted against the average target value, and compared against the average first principle model predictions. As can be seen, the average updated prediction is really good for both gas and liquid flow rate.

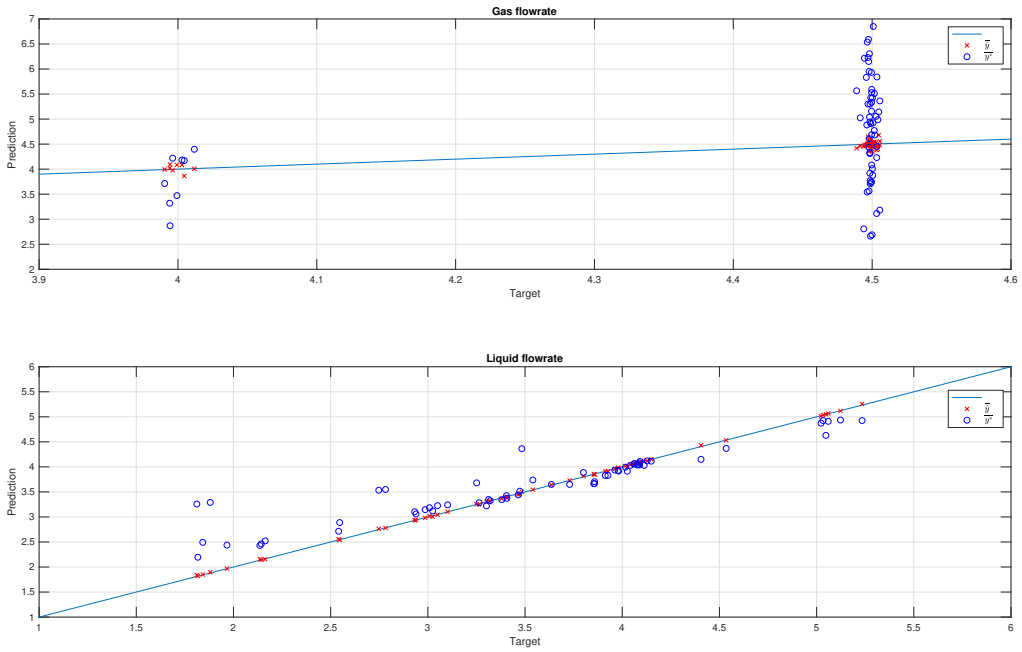

Figure D.1: Performance plot of the average updated prediction for each steady-state section compared with the average first principle model predictions.

E Training set prediction plots

E.1 Baseline model

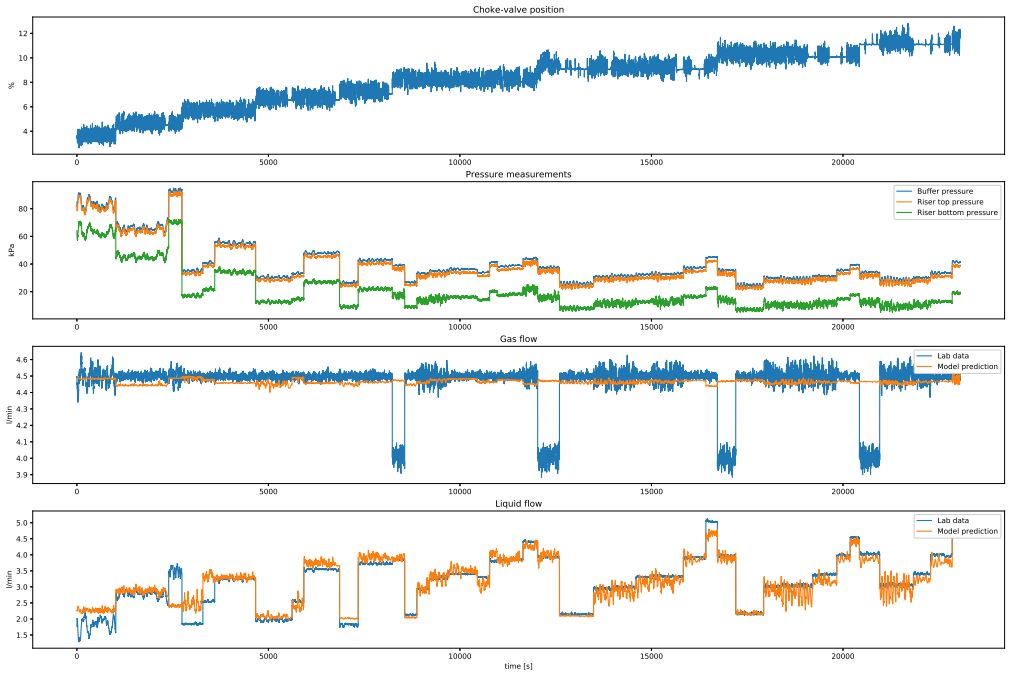

Figure E.1: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for model trained on raw experimental data, on the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

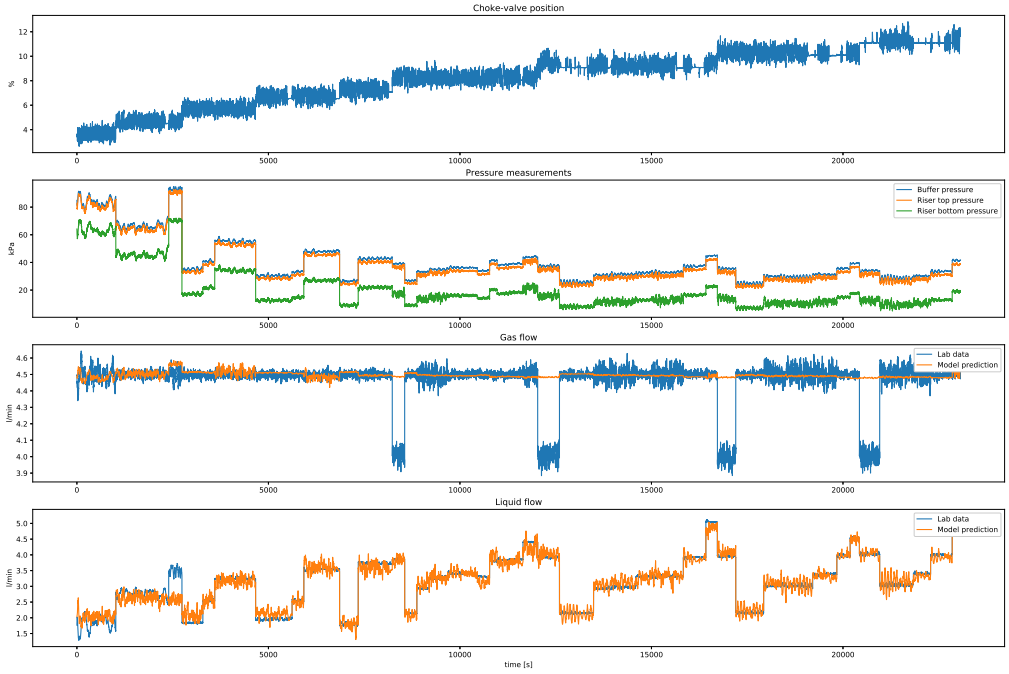

Figure E.2: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for model trained on filtered input data, on the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

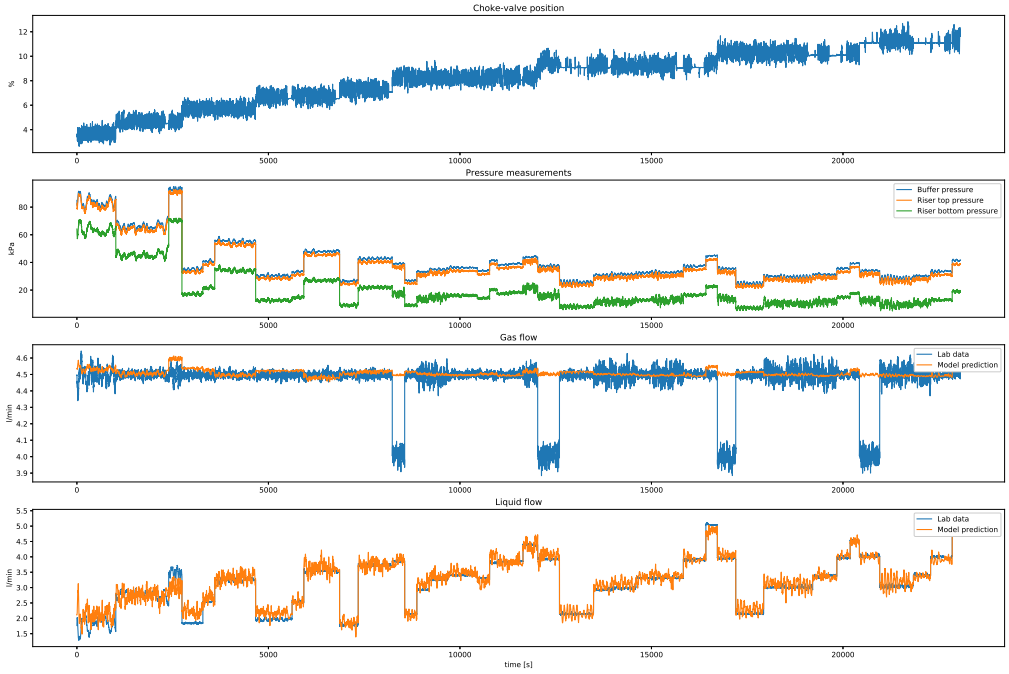

Figure E.3: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for model trained on averaged input data, on the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

E.2 Feature model

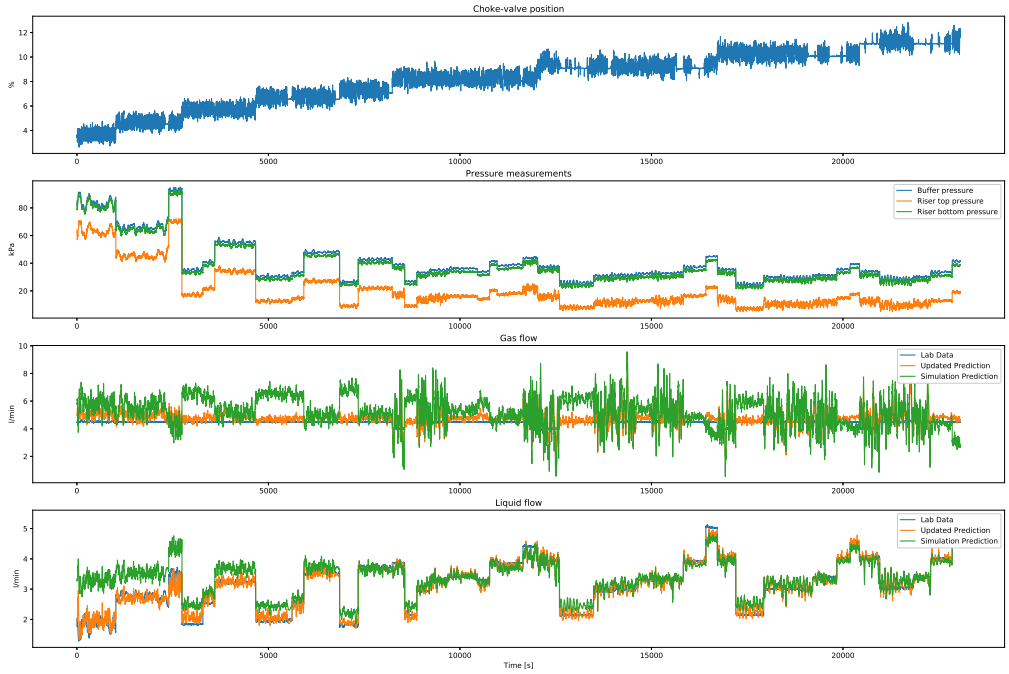

Figure E.4: Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates and the simplified FPM gas and liquid flow rate predictions, for bias prediction model, on the train set. Inputs, targets, FPM predictions and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

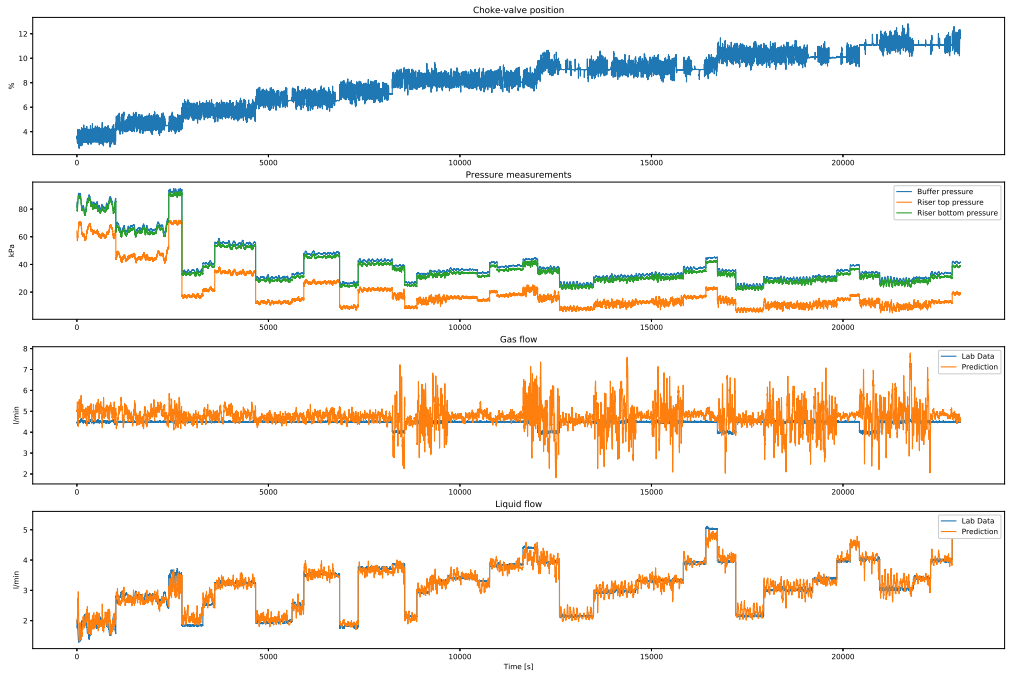

Figure E.5: Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for bias prediction model, on the train set. Inputs, targets, and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

E.3 Transfer models

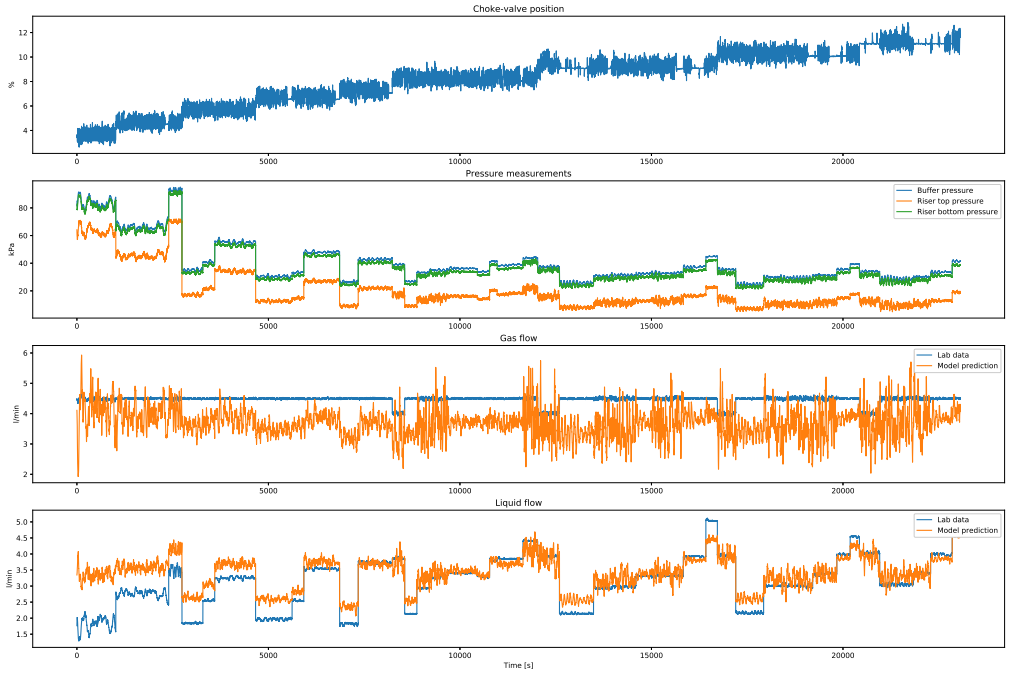

Figure E.6: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for the simulation model, on the the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

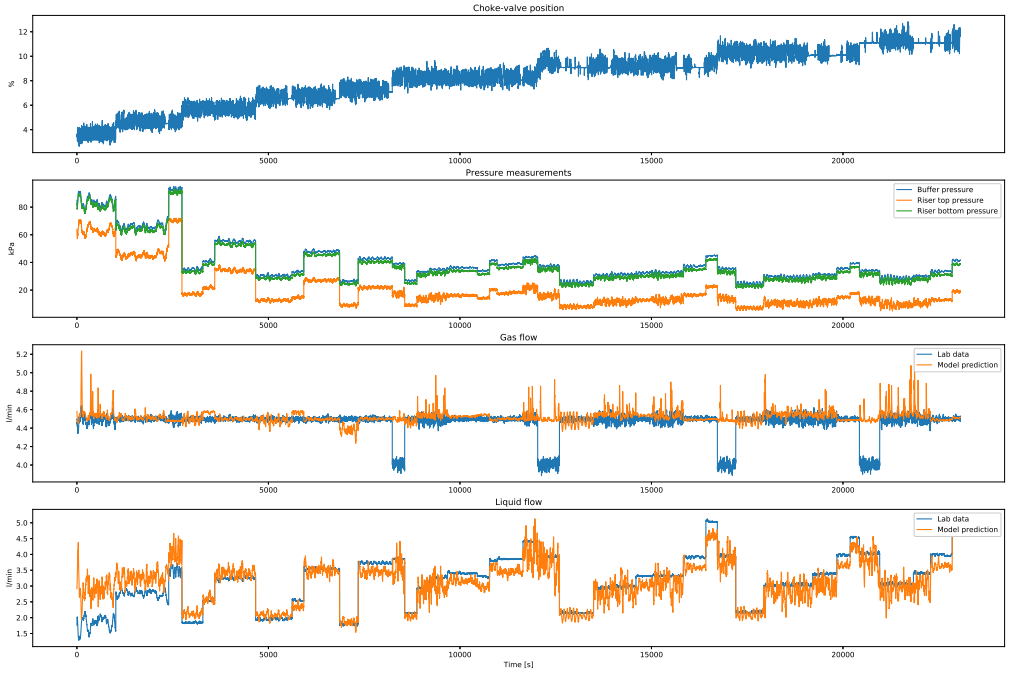

Figure E.7: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a transfer model where the last layer was retrained on experimental data, on the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

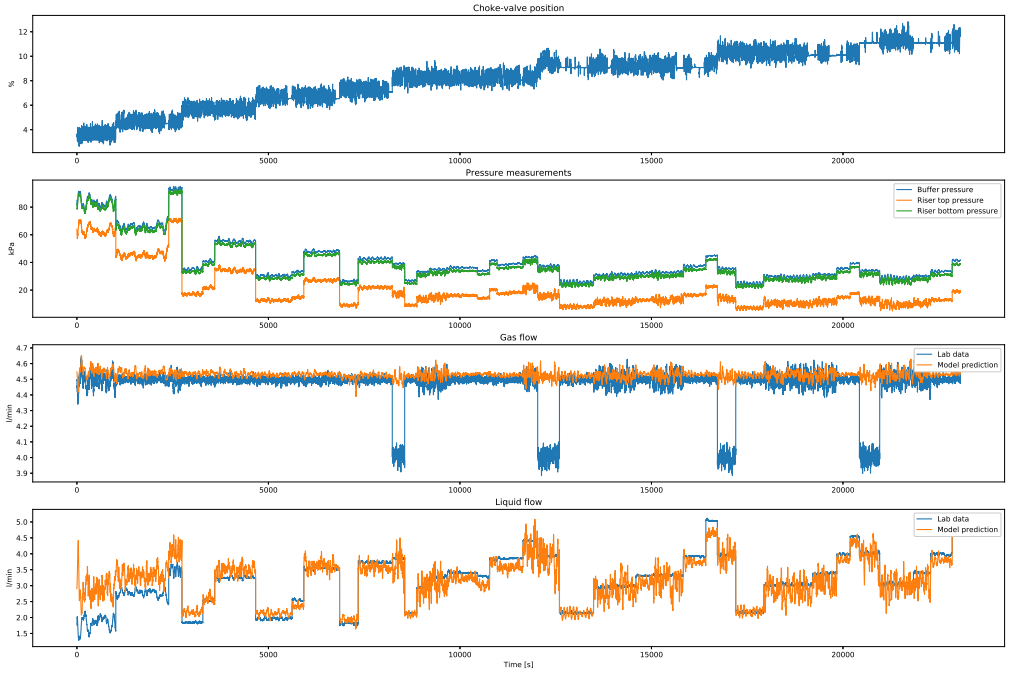

Figure E.8: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a transfer model where the last two layers were retrained on experimental data, on the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

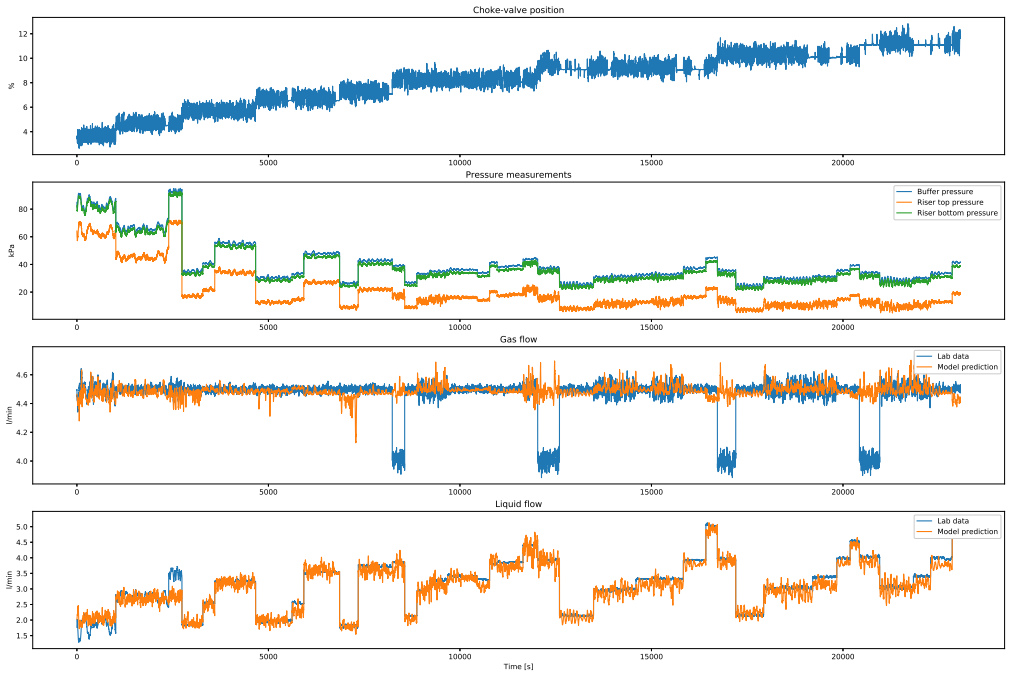

Figure E.9: Predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a transfer model which uses predictions from simulation network as extra input, on the train set. Inputs, targets and predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

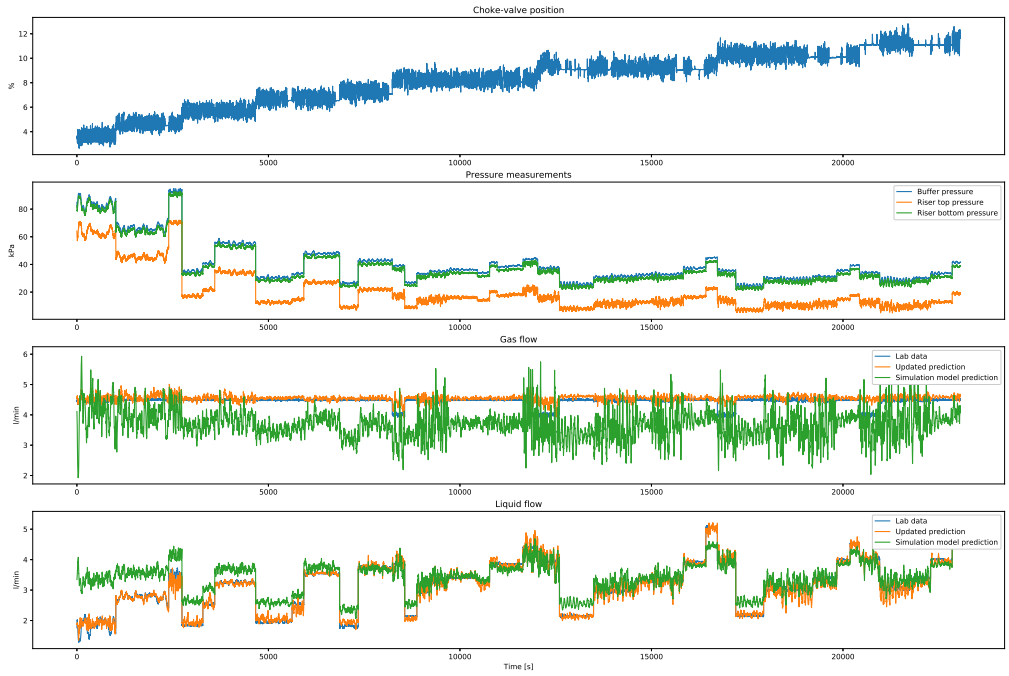

Figure E.10: Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates and the simulation model gas and liquid flow rate predictions, for a bias prediction model, on the train set. Inputs, targets, simulation model predictions and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

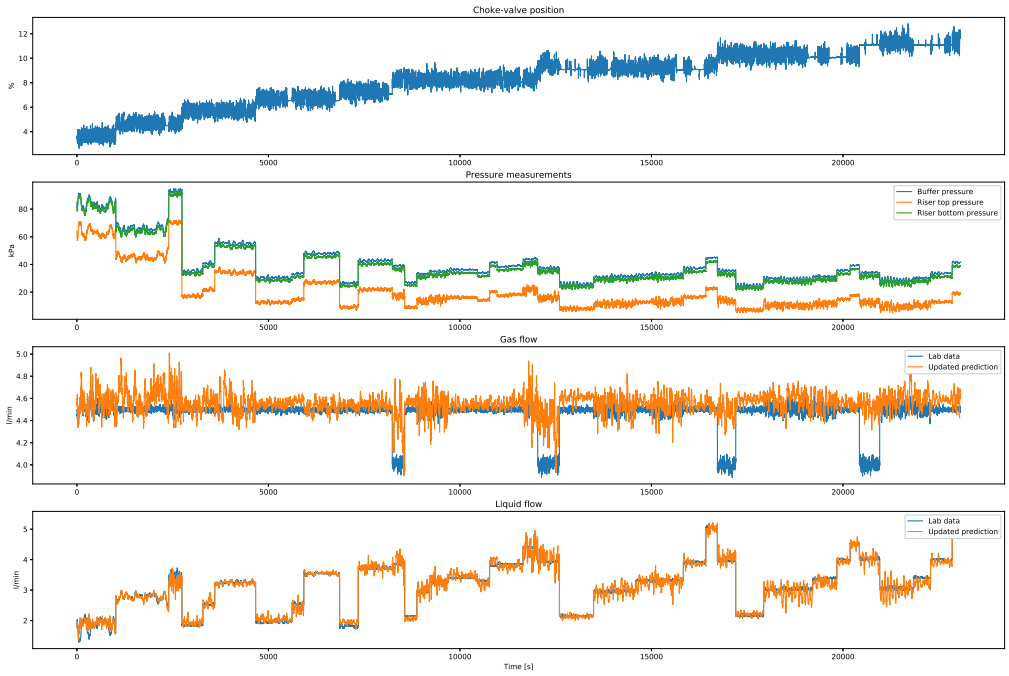

Figure E.11: Updated predicted gas and liquid flow rates plotted against experimental gas and liquid flow rates, for a bias prediction model, on the train set. Inputs, targets and updated predictions were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

F Pressure drop plots

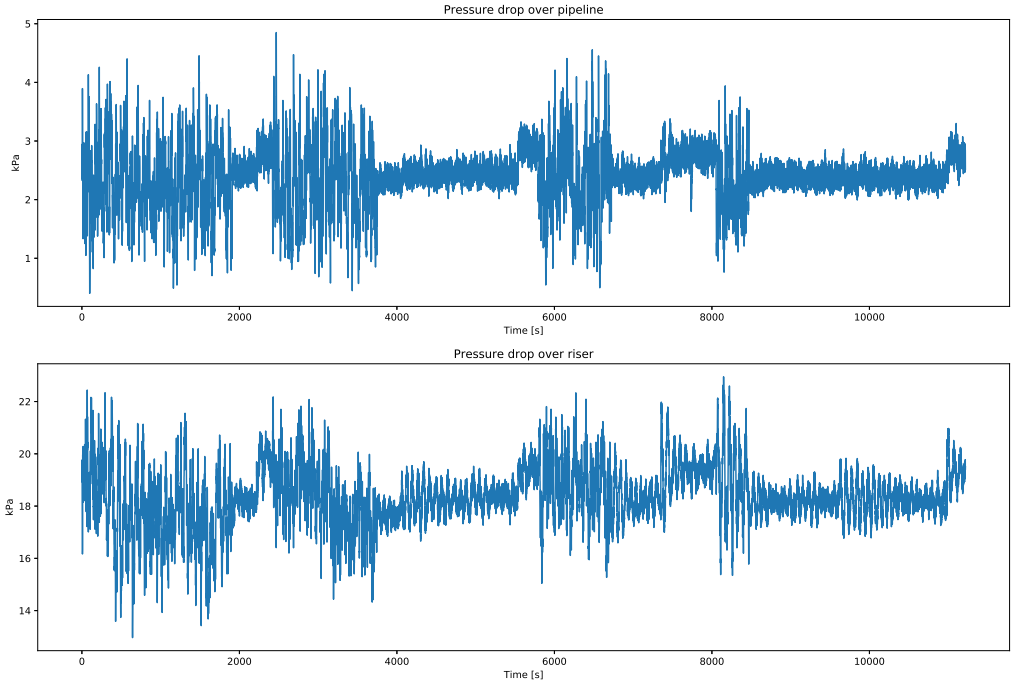

Figure F.1: Pressure drop over pipeline and riser for test data set. The pressure drops were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$

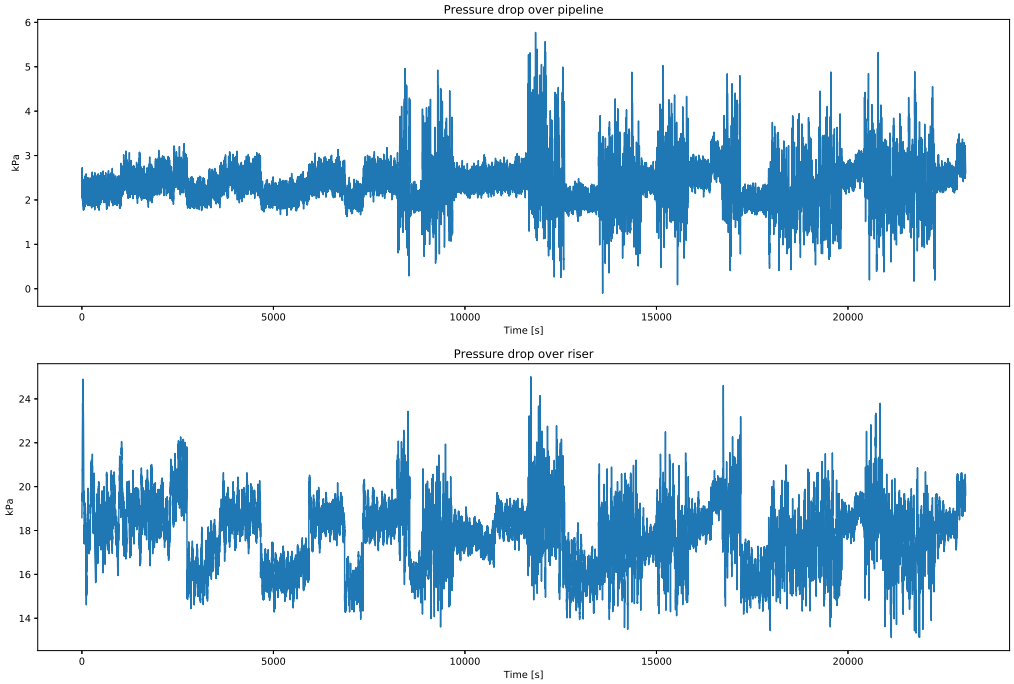

Figure F.2: Pressure drop over pipeline and riser for train data set. The pressure drops were filtered with a low-pass filter with a cut-off frequency of $0.2\,\mathrm{Hz}$